ANSYS LS-DYNA®
KEYWORD USER'S MANUAL

VOLUME I

master@12c487c75 (10/30/26)
LS-DYNA Dev

ANSYS
PART OF SYNOPSYS®

Websites

https://lsdyna.ansys.com
https://support.ansys.com

Address

ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Copyright and Trademark Information

Copyright © 2026 Synopsys, Inc. and ANSYS, Inc. This Synopsys product and all associated documentation are proprietary to ANSYS, Inc. and Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement. All other use, reproduction, modification or distribution of the Synopsys product or the associated documentation is strictly prohibited.

ANSYS, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. ICEM CFD is a trademark used by ANSYS, Inc. under license. CFX is a trademark of Sony Corporation in Japan. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXlm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the legal information in the product help files for the complete Legal Notice for ANSYS proprietary software and third-party software. If you are unable to access the Legal Notice, contact ANSYS, Inc.

Published in the U.S.A.

Patents

Ansys products acquired from LSTC are protected under the following patents as well as pending patent applications.

US Patents:

7167816, 7286972, 7308387, 7382367, 7386425, 7386428, 7392163, 7395128, 7415400, 7428713, 7472602, 7499050, 7516053, 7533577, 7590514, 7613585, 7657394, 7660480, 7664623, 7702490, 7702494, 7945432, 7953578, 7987143, 7996344, 8050897, 8069017, 8126684, 8150668, 8165856, 8180605, 8190408, 8200458, 8200464, 8209157, 8271237, 8296109, 8306793, 8374833, 8423327, 8467997, 8489372, 8494819, 8515714, 8521484, 8577656, 8612186, 8666719, 8744825, 8768660, 8798973, 8855976, 8855977, 8898042, 9020784, 9098657, 9117042, 9135377, 9286422, 9292632, 9405868, 9430594, 9507892, 9607115, 9639638, 9798841, 9817926, 9852235, 9910939, 9910942, 10303822, 10311181, 10330645, 10423737, 10452796, 10467359, 10474773, 10579739.

Japan Patents:

5090426, 5281057, 5330300, 5373689, 5404516, 5411013, 5411057, 5431133, 5520553, 5530552, 5589198, 5601961, 5775708, 5792995, 5823170, 6043146, 6043198, 6253391, 6253429, 6267573, 6285793, 6548532, 6560577, 6580979, 6653569, 6665016.

China Patents:

ZL200910207380.5, ZL200910165817.3, ZL200910221325.1, ZL200910246429.8, ZL201010128222.3, ZL201010132510.6, ZL201010155066.X, ZL201010171603.X, ZL201010174074.9, ZL201010287263.7, ZL201110037461.2, ZL201010533046.1, ZL201110035253.9, ZL201110065789.5, ZL201110132394.2, ZL201110140142.4, ZL201210039535.0, ZL201210286459.3, ZL201210275406.1, ZL201210514668.9, ZL201210422902.5, ZL201210424131.3, ZL201310021716.5, ZL201210475617.X, ZL201310081855.7, ZL201310081815.2, ZL201310167211.X, ZL201310561338.X, ZL201310586645.3, ZL201410151809.4, ZL201410137756.7, ZL201410132003.0, ZL201410298399.6, ZL201410503337.4, ZL201410526080.4, ZL201510388334.5, ZL201510120888.7.

EPO Patents:

EP219192881, EP2180416, EP2219124.

Korea Patents:

10-2556445.

AES

AES. Copyright © 2001, Dr. Brian Gladman < brg@gladman.uk.net>, Worcester, UK. All rights reserved.

LICENSE TERMS

The free distribution and use of this software in both source and binary form is allowed (with or without changes) provided that:

1. distributions of this source code include the above copyright notice, this list of conditions and the following disclaimer;

2. distributions in binary form include the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other associated materials;

3. the copyright holder's name is not used to endorse products built using this software without specific written permission.

DISCLAIMER

This software is provided 'as is' with no explicit or implied warranties in respect of any properties, including, but not limited to, correctness and fitness for purpose.

Issue Date: 21/01/2002

This file contains the code for implementing the key schedule for AES (Rijndael) for block and key sizes of 16, 24, and 32 bytes.

TABLE OF CONTENTS

TABLE OF CONTENTS 0-1

INTRODUCTION 1-1

CHRONOLOGICAL HISTORY 1-1

1989-1990 1-3

1991-1992 1-4

1993-1994 1-5

1995 1-6

950 1-9

960 1-13

970 1-17

971 1-23

971R6 1-31

971R6.1 1-37

R7.0 1-42

R7.1 1-57

R8.0 1-73

R9.0 1-96

R10 1-151

R11 1-200

R12 1-256

R13 1-316

R14 1-316

R15 1-316

R16 1-316

R17 1-316

MATERIAL MODELS 1-316

SPATIAL DISCRETIZATION 1-318

CONTACT-IMPACT INTERFACES 1-322

INTERFACE DEFINITIONS FOR COMPONENT ANALYSIS 1-323

PRECISION 1-324

GETTING STARTED 2-1

DESCRIPTION OF KEYWORD INPUT 2-1

SUMMARY OF COMMONLY USED OPTIONS 2-12

TABLE OF CONTENTS

EXECUTION SYNTAX 2-14
SENSE SWITCH CONTROLS 2-19
Procedure for LS-DYNA/MPP 2-20
FILES 2-21
RESTART ANALYSIS 2-23
VDA/IGES DATABASES 2-25
LS-PrePost® 2-26
EXECUTION SPEEDS 2-27
UNITS 2-28
GENERAL CARD FORMAT 2-29

*AIRBAG 3-1
*AIRBAG 3-2
Core Cards 3-3
*AIRBAG_SIMPLE_PRESSURE_VOLUME 3-9
*AIRBAG_SIMPLE_AIRBAG_MODEL 3-12
*AIRBAG_ADIABATIC_GAS_MODEL 3-17
*AIRBAG_WANG_NEFSKE 3-20
JETTING models 3-30
CM option 3-35
*AIRBAG_LOAD_CURVE 3-38
*AIRBAG_LINEAR_FLUID 3-40
*AIRBAG_HYBRID 3-44
*AIRBAG_HYBRID_JETTING 3-44
*AIRBAG_HYBRID_CHEMKIN 3-54
*AIRBAG_FLUID_AND_GAS 3-60
*AIRBAG_ALE 3-65
*AIRBAG_CPG 3-81
*AIRBAG_INTERACTION 3-92
*AIRBAG_PARTICLE 3-94
*AIRBAG_REFERENCE_GEOMETRY 3-118
*AIRBAG_SALE 3-121
*AIRBAG_SHELL_REFERENCE_GEOMETRY 3-130

*ALE 4-1
*ALE_AMBIENT_HYDROSTATIC 4-5
*ALE_BURN_SWITCH_MMG 4-9
*ALE_COUPLING_NODAL_CONSTRAINT 4-15

*ALE_COUPLING_NODAL_DRAG 4-18
*ALE_COUPLING_NODAL_PENALTY 4-21
*ALE_COUPLING_RIGID_BODY 4-24
*ALE_ESSENTIAL_BOUNDARY 4-26
*ALE_FAIL_SWITCH_MMG 4-28
*ALE_FRAGMENTATION 4-30
*ALE_FSI_PROJECTION 4-32
*ALE_FSI_SWITCH_MMG 4-35
*ALE_FSI_TO_LOAD_NODE 4-40
*ALE_INJECTION 4-42
*ALE_MAPPING 4-52
*ALE_MAPPING_FROM_LAGRANGIAN 4-64
*ALE_MESH_INTERFACE 4-68
*ALE_MULTI-MATERIAL_GROUP 4-71
*ALE_PRESCRIBED_MOTION 4-75
*ALE_REFERENCE_SYSTEM_CURVE 4-78
*ALE_REFERENCE_SYSTEM_GROUP 4-81
*ALE_REFERENCE_SYSTEM_NODE 4-89
*ALE_REFERENCE_SYSTEM_SWITCH 4-91
*ALE_REFINE 4-93
*ALE_SMOOTHING 4-94
*ALE_STRUCTURED_FSI 4-97
*ALE_STRUCTURED_MESH 4-103
*ALE_STRUCTURED_MESH_CONTROL_POINTS 4-107
*ALE_STRUCTURED_MESH_JOIN 4-114
*ALE_STRUCTURED_MESH_MOTION 4-115
*ALE_STRUCTURED_MESH_REFINE 4-118
*ALE_STRUCTURED_MESH_TRIM 4-121
*ALE_STRUCTURED_MESH_VOLUME_FILLING 4-126
*ALE_STRUCTURED_MULTI_DOMAIN_CONTACT 4-133
*ALE_STRUCTURED_MULTI-MATERIAL_GROUP 4-136
*ALE_STRUCTURED_POINT_SOURCE 4-139
*ALE_SWITCH_MMG 4-141
*ALE_TANK_TEST 4-148
*ALE_UP_SWITCH 4-152

*BOUNDARY 5-1

TABLE OF CONTENTS

*BOUNDARY_ACOUSTIC_COUPLING 5-4
*BOUNDARY_ACOUSTIC_COUPLING_SPECTRAL 5-7
*BOUNDARY_ACOUSTIC_FREE_SURFACE 5-8
*BOUNDARY_ACOUSTIC_IMPEDANCE 5-9
*BOUNDARY_ACOUSTIC_IMPEDANCE_COMPLEX 5-10
*BOUNDARY_ACOUSTIC_IMPEDANCE_MECHANICAL 5-11
*BOUNDARY_ACOUSTIC_INTERFACE 5-12
*BOUNDARY_ACOUSTIC_MAPPING 5-13
*BOUNDARY_ACOUSTIC_NON_REFLECTING 5-14
*BOUNDARY_ACOUSTIC_PRESCRIBED_MOTION 5-16
*BOUNDARY_ACOUSTIC_PRESSURE_SPECTRAL 5-17
*BOUNDARY_ALE_MAPPING 5-18
*BOUNDARY_AMBIENT 5-25
*BOUNDARY_AMBIENT_EOS 5-28
*BOUNDARY_CONVECTION 5-30
*BOUNDARY_COUPLED 5-33
*BOUNDARY_CYCLIC 5-35
*BOUNDARY_DE_NON_REFLECTING 5-38
*BOUNDARY_FLUIDM 5-39
*BOUNDARY_FLUIDM_BOTTOM 5-41
*BOUNDARY_FLUIDM_FREE_SURFACE 5-42
*BOUNDARY_FLUIDM_INTERIOR 5-43
*BOUNDARY_FLUX 5-46
*BOUNDARY_FLUX_TRAJECTORY 5-50
*BOUNDARY_MCOL 5-58
*BOUNDARY_NON_REFLECTING 5-60
*BOUNDARY_NON_REFLECTING_2D 5-62
*BOUNDARY_PAP 5-64
*BOUNDARY_PORE_FLUID 5-66
*BOUNDARY_PRECRACK 5-68
*BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID 5-69
*BOUNDARY_PRESCRIBED_FINAL_GEOMETRY 5-73
*BOUNDARY_PRESCRIBED_MOTION 5-76
*BOUNDARY_PRESCRIBED_ORIENTATION_RIGID 5-90
*BOUNDARY_PRESSURE_OUTFLOW 5-99
*BOUNDARY_PWP 5-100
*BOUNDARY_PZEPOT 5-108

*BOUNDARY_RADIATION 5-109
*BOUNDARY_RADIATION_ENCLOSURE 5-111
*BOUNDARY_RADIATION_SEGMENT 5-117
*BOUNDARY_RADIATION_SEGMENT_VF 5-120
*BOUNDARY_RADIATION_SET 5-122
*BOUNDARY_RADIATION_SET_VF 5-125
*BOUNDARY_SALE_MESH_FACE 5-128
*BOUNDARY_SLIDING_PLANE 5-131
*BOUNDARY_SPC 5-132
*BOUNDARY_SPC_SYMMETRY_PLANE 5-135
*BOUNDARY_SPH_FLOW 5-138
*BOUNDARY_SPH_HEAT_CONVECTION 5-141
*BOUNDARY_SPH_NON_REFLECTING 5-143
*BOUNDARY_SPH_NOSLIP 5-144
*BOUNDARY_SPH_PERIODIC 5-146
*BOUNDARY_SPH_SYMMETRY_PLANE 5-148
*BOUNDARY_SPH_TRACK_SURFACE 5-149
*BOUNDARY_SYMMETRY_FAILURE 5-151
*BOUNDARY_TEMPERATURE 5-152
*BOUNDARY_TEMPERATURE_PERIODIC_SET 5-154
*BOUNDARY_TEMPERATURE_RSW 5-156
*BOUNDARY_TEMPERATURE_TRAJECTORY 5-161
*BOUNDARY_THERMAL_BULKFLOW 5-166
*BOUNDARY_THERMAL_BULKNODE 5-167
*BOUNDARY_THERMAL_WELD 5-169
*BOUNDARY_THERMAL_WELD_TRAJECTORY 5-173
*BOUNDARY_USA_SURFACE 5-181
*BOUNDARY_ELEMENT_METHOD 5-1
*BOUNDARY_ELEMENT_METHOD_CONTROL 6-2
*BOUNDARY_ELEMENT_METHOD_FLOW 6-4
*BOUNDARY_ELEMENT_METHOD_NEIGHBOR 6-6
*BOUNDARY_ELEMENT_METHOD_SYMMETRY 6-10
*BOUNDARY_ELEMENT_METHOD_WAKE 6-11
*CASE 7-1
*COMMENT 7-1

TABLE OF CONTENTS

*COMPONENT 9-1
*COMPONENT_GEBOD 9-2
*COMPONENT_GEBOD_JOINT 9-4
*COMPONENT_HYBRIDIII 9-7
*COMPONENT_HYBRIDIII_JOINT 9-10

*CONSTRAINED 10-1
*CONSTRAINED_ADAPTIVITY 10-3
*CONSTRAINED_BEAM_IN_SOLID 10-5
*CONSTRAINED_BUTT_WELD 10-13
*CONSTRAINED_COORDINATE 10-16
*CONSTRAINED_EULER_IN_EULER 10-20
*CONSTRAINED_EXTRA_NODES 10-22
*CONSTRAINED_GENERALIZED_WELD 10-24
Spot Weld Card 10-27
Fillet Weld Card 10-29
Butt Weld Card 10-31
Cross Fillet Weld Card 10-34
Combined Weld Cards 10-36
*CONSTRAINED_GLOBAL 10-38
*CONSTRAINED_IMMERSED_IN_SPG 10-40
*CONSTRAINED_INTERPOLATION 10-41
*CONSTRAINED_INTERPOLATION_SPOTWELD 10-47
*CONSTRAINED_JOINT 10-61
*CONSTRAINED_JOINT_COOR 10-79
*CONSTRAINED_JOINT_STIFFNESS 10-89
Flexion-Torsion Joint Stiffness Cards 10-93
Generalized Joint Stiffness Cards 10-98
Translational Joint Stiffness Cards 10-103
Cylindrical Joint Stiffness Cards 10-107
*CONSTRAINED_JOINT_USER_FORCE 10-112
*CONSTRAINED_LAGRANGE_IN_SOLID 10-113
*CONSTRAINED_LINEAR_GLOBAL 10-135
*CONSTRAINED_LINEAR_LOCAL 10-138
*CONSTRAINED_LOCAL 10-141
*CONSTRAINED_MULTIPLE_GLOBAL 10-143
*CONSTRAINED_NODAL_RIGID_BODY 10-146

*CONSTRAINED_NODE_INTERPOLATION....10-160
*CONSTRAINED_NODE_SET....10-162
*CONSTRAINED_NODE_TO_NURBS_PATCH....10-165
*CONSTRAINED_POINTS....10-167
*CONSTRAINED_RIGID_BODIES....10-169
*CONSTRAINED_RIGID_BODY_INSERT....10-171
*CONSTRAINED_RIGID_BODY_STOPPERS....10-173
*CONSTRAINED_RIVET....10-177
*CONSTRAINED_SHELL_IN_SOLID....10-179
*CONSTRAINED_SHELL_TO_SOLID....10-182
*CONSTRAINED_SOIL_PILE....10-184
*CONSTRAINED_SOLID_IN_SOLID....10-206
*CONSTRAINED_SPLINE....10-209
*CONSTRAINED_SPOTWELD....10-211
*CONSTRAINED_SPR2....10-215
*CONSTRAINED_TIE-BREAK....10-224
*CONSTRAINED_TIED_NODES_FAILURE....10-226

*CONTACT....11-1
*CONTACT....11-3
Introduction....11-3
Data cards for *CONTACT keyword....11-6
Options for *CONTACT keyword....11-8
ID Card....11-18
MPP Cards....11-19
Mandatory Card 1....11-24
Mandatory Card 2....11-27
Mandatory Card 3....11-33
Card 4: AUTOMATIC_…_TIEBREAK....11-35
Card 4: AUTOMATIC_…_SURFACE_TO_SURFACE_TIEBREAK_USER_....11-43
Card 4: AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE/LUBRICATION....11-46
Card 4: AUTOMATIC_SINGLE_SURFACE_TIED....11-48
Card 4: AUTOMATIC_SURFACE_TO_SURFACE_..._TIED_WELD....11-49
Card 4: CONSTRAINT_…_TO_SURFACE....11-52
Card 4: DRAWBEAD....11-53
Card 4: ERODING_..._SURFACE....11-64
Card 4: SURFACE_INTERFERENCE....11-66

TABLE OF CONTENTS

Card 4: RIGID_TO_RIGID 11-68
Card 4: TIEBREAK_NODES 11-70
Card 4: TIEBREAK_SURFACE 11-72
Card 4: CONTRACTION_JOINT 11-74
THERMAL 11-76
THERMAL_FRICTION 11-79
ORTHO_FRICTION 11-83
Optional Card A 11-89
Optional Card B 11-101
Optional Card C 11-105
Optional Card D 11-110
Optional Card E 11-114
Optional Card F 11-118
Optional Card G 11-121
General Remarks 11-123
Contact Examples 11-140
*CONTACT_ADD_WEAR 11-143
*CONTACT_AUTO_MOVE 11-147
*CONTACT_COUPLING 11-153
*CONTACT_ENTITY 11-155
*CONTACT_EXCLUDE_INTERACTION 11-165
*CONTACT_FORCE_TRANSDUCER 11-167
*CONTACT_GEBOD 11-173
*CONTACT_GUIDED_CABLE 11-176
*CONTACT_INTERIOR 11-178
*CONTACT_RIGID_SURFACE 11-180
*CONTACT_SPG 11-184
*CONTACT_1D 11-186
*CONTACT_2D 11-188
*CONTACT_2D_[SLIDING, TIED, & PENALTY] 11-191
*CONTACT_2D_[AUTOMATIC & FORCE_TRANSDUCER] 11-197
*CONTACT_2D_NODE_TO_SOLID 11-207

*CONTROL 12-1
*CONTROL_ACCURACY 12-8
*CONTROL_ACOUSTIC 12-14
*CONTROL_ACOUSTIC_COUPLING 12-15

*CONTROL_ACOUSTIC_SPECTRAL 12-16
*CONTROL_ADAPSTEP 12-18
*CONTROL_ADAPTIVE 12-19
*CONTROL_ADAPTIVE_CURVE 12-42
*CONTROL_AIRBAG 12-49
*CONTROL_ALE 12-50
*CONTROL_BULK_VISCOSITY 12-60
*CONTROL_CHECK_SHELL 12-62
*CONTROL_CHECK_SOLID 12-64
*CONTROL_COARSEN 12-65
*CONTROL_CONSTRAINED 12-67
*CONTROL_CONTACT 12-69
*CONTROL_COUPLING 12-87
*CONTROL_CPG 12-89
*CONTROL_CPG_OUTPUT 12-92
*CONTROL_CPM 12-94
*CONTROL_CPU 12-98
*CONTROL_DEBUG 12-99
*CONTROL_DISCRETE_ELEMENT 12-100
*CONTROL_DYNAMIC_RELAXATION 12-107
*CONTROL_EFG 12-113
*CONTROL_ENERGY 12-115
*CONTROL_EOS_USER_LIBRARY 12-117
*CONTROL_EXPLICIT_THERMAL 12-119
    *CONTROL_EXPLICIT_THERMAL_ALE_COUPLING 12-120
    *CONTROL_EXPLICIT_THERMAL_BOUNDARY 12-121
    *CONTROL_EXPLICIT_THERMAL_CONTACT 12-122
    *CONTROL_EXPLICIT_THERMAL_INITIAL 12-123
    *CONTROL_EXPLICIT_THERMAL_OUTPUT 12-124
    *CONTROL_EXPLICIT_THERMAL_PROPERTIES 12-126
    *CONTROL_EXPLICIT_THERMAL_SOLVER 12-129
*CONTROL_EXPLOSIVE_SHADOW 12-130
*CONTROL_FORMING 12-132
    *CONTROL_FORMING_AUTO_NET 12-134
    *CONTROL_FORMING_AUTOCHECK 12-138
    *CONTROL_FORMING_AUTOPOSITION_PARAMETER 12-147
    *CONTROL_FORMING_BESTFIT 12-155

TABLE OF CONTENTS

*CONTROL_FORMING_DRAWBEAD_TRANSITION 12-161
*CONTROL_FORMING_HOME_GAP 12-163
*CONTROL_FORMING_INITIAL_THICKNESS 12-164
*CONTROL_FORMING_MAXID 12-167
*CONTROL_FORMING_ONESTEP 12-169
*CONTROL_FORMING_OUTPUT 12-186
*CONTROL_FORMING_PARAMETER_READ 12-195
*CONTROL_FORMING_POSITION 12-198
*CONTROL_FORMING_PRE_BENDING 12-200
*CONTROL_FORMING_PROJECTION 12-205
*CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS 12-207
*CONTROL_FORMING_SCALE 12-209
*CONTROL_FORMING_SCRAP_FALL 12-211
*CONTROL_FORMING_SHELL_TO_TSHELL 12-224
*CONTROL_FORMING_STONING 12-229
*CONTROL_FORMING_STRAIN_RATIO_SMOOTH 12-236
*CONTROL_FORMING_TEMPLATE 12-238
*CONTROL_FORMING_TIPPING 12-244
*CONTROL_FORMING_TRAVEL 12-250
*CONTROL_FORMING_TRIM_MERGE 12-252
*CONTROL_FORMING_TRIM_SOLID_REFINEMENT 12-255
*CONTROL_FORMING_TRIMMING 12-257
*CONTROL_FORMING_UNFLANGING 12-259
*CONTROL_FORMING_USER 12-268
*CONTROL_FREQUENCY_DOMAIN 12-272
*CONTROL_HOURGLASS 12-274
*CONTROL_IMPLICIT 12-278
*CONTROL_IMPLICIT_AUTO 12-280
*CONTROL_IMPLICIT_BUCKLE 12-288
*CONTROL_IMPLICIT_CONSISTENT_MASS 12-290
*CONTROL_IMPLICIT_DYNAMICS 12-291
*CONTROL_IMPLICIT_EIGENVALUE 12-297
*CONTROL_IMPLICIT_FORMING 12-310
*CONTROL_IMPLICIT_GENERAL 12-324
*CONTROL_IMPLICIT_INERTIA_RELIEF 12-329
*CONTROL_IMPLICIT_JOINTS 12-331
*CONTROL_IMPLICIT_MODAL_DYNAMIC 12-332

*CONTROL_IMPLICIT_MODAL_DYNAMIC_DAMPING 12-336
*CONTROL_IMPLICIT_MODAL_DYNAMIC_MODE 12-339
*CONTROL_IMPLICIT_MODES 12-341
*CONTROL_IMPLICIT_ORDERING 12-346
*CONTROL_IMPLICIT_RESIDUAL_VECTOR 12-348
*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS 12-352
*CONTROL_IMPLICIT_SOLUTION 12-357
*CONTROL_IMPLICIT_SOLVER 12-370
*CONTROL_IMPLICIT_SSD_DIRECT 12-382
*CONTROL_IMPLICIT_STABILIZATION 12-384
*CONTROL_IMPLICIT_STATIC_CONDENSATION 12-386
*CONTROL_IMPLICIT_TERMINATION 12-390
*CONTROL_LSDA 12-392
*CONTROL_MAPPING_SHIFT_BY_DT 12-393
*CONTROL_MAT 12-394
*CONTROL_MPP 12-396
*CONTROL_MPP_CONTACT_GROUPABLE 12-398
*CONTROL_MPP_DECOMPOSITION_ADJUST_MTYPE_COST 12-400
*CONTROL_MPP_DECOMPOSITION_ADJUST_PART_COST 12-401
*CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS 12-402
*CONTROL_MPP_DECOMPOSITION_AUTOMATIC 12-404
*CONTROL_MPP_DECOMPOSITION_BAGREF 12-405
*CONTROL_MPP_DECOMPOSITION_CHECK_SPEED 12-406
*CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE 12-407
*CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE 12-409
*CONTROL_MPP_DECOMPOSITION_DEFORMED_GEOMETRY 12-410
*CONTROL_MPP_DECOMPOSITION_DISABLE_UNREF_CURVES 12-411
*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS 12-412
*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_SPH_ELEMENTS 12-413
*CONTROL_MPP_DECOMPOSITION_ELCOST 12-414
*CONTROL_MPP_DECOMPOSITION_FILE 12-415
*CONTROL_MPP_DECOMPOSITION_FLAG_STRESS_STRAIN_CURVE 12-417
*CONTROL_MPP_DECOMPOSITION_METHOD 12-418
*CONTROL_MPP_DECOMPOSITION_NODISTRIBUTE_DES_ELEMENTS 12-419
*CONTROL_MPP_DECOMPOSITION_NUMPROC 12-420
*CONTROL_MPP_DECOMPOSITION_OUTDECOMP 12-421
*CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE 12-423

TABLE OF CONTENTS

*CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE 12-424
*CONTROL_MPP_DECOMPOSITION_RCBLOG 12-425
*CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION 12-426
*CONTROL_MPP_DECOMPOSITION_SCALE_CONTACT_COST 12-428
*CONTROL_MPP_DECOMPOSITION_SCALE_FACTOR_SPH 12-429
*CONTROL_MPP_DECOMPOSITION_SHOW 12-430
*CONTROL_MPP_DECOMPOSITION_TRANSFORM_KEYWORD 12-431
*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION 12-432
*CONTROL_MPP_IO_LSTC_REDUCE 12-434
*CONTROL_MPP_IO_NOBEAMOUT 12-435
*CONTROL_MPP_IO_NOD3DUMP 12-436
*CONTROL_MPP_IO_NODUMP 12-437
*CONTROL_MPP_IO_NOFULL 12-438
*CONTROL_MPP_IO_NOTIEDIO 12-439
*CONTROL_MPP_IO_SWAPBYTES 12-440
*CONTROL_MPP_MATERIAL_MODEL_DRIVER 12-441
*CONTROL_MPP_PFILE 12-442
*CONTROL_MPP_REBALANCE 12-443
*CONTROL_NONLOCAL 12-445
*CONTROL_OUTPUT 12-446
*CONTROL_PARALLEL 12-459
*CONTROL_PORE_AIR 12-462
*CONTROL_PORE_FLUID 12-464
*CONTROL_PZELECTRIC 12-471
*CONTROL_REFERENCE_CONFIGURATION 12-473
*CONTROL_REFINE_ALE 12-477
*CONTROL_REFINE_ALE2D 12-483
*CONTROL_REFINE_MPP_DISTRIBUTION 12-489
*CONTROL_REFINE_SHELL 12-491
*CONTROL_REFINE_SOLID 12-497
*CONTROL_REMESHING 12-502
*CONTROL_REQUIRE_REVISION 12-507
*CONTROL_RIGID 12-509
*CONTROL_SEGMENTS_IN_ALE_COUPLING 12-514
*CONTROL_SHELL 12-517
*CONTROL_SOLID 12-533
*CONTROL_SOLUTION 12-540

*CONTROL_SPH 12-543
*CONTROL_SPH_INCOMPRESSIBLE 12-551
*CONTROL_SPOTWELD_BEAM 12-553
*CONTROL_STAGED_CONSTRUCTION 12-556
*CONTROL_START 12-560
*CONTROL_STEADY_STATE_ROLLING 12-561
*CONTROL_STRUCTURED 12-563
*CONTROL_SUBCYCLE 12-564
*CONTROL_TERMINATION 12-566
*CONTROL_THERMAL_EIGENVALUE 12-568
*CONTROL_THERMAL_FORMING 12-569
*CONTROL_THERMAL_NONLINEAR 12-579
*CONTROL_THERMAL_SOLVER 12-581
*CONTROL_THERMAL_TIMESTEP 12-588
*CONTROL_TIMESTEP 12-591
*CONTROL_UNITS 12-600
*CONTROL_2D_REMESHING_REGION 12-602

*CONTROLLER 13-1
*CONTROLLER_PLANT 13-2

*COSIM 14-1
*COSIM_FMI_CONTROL 14-2
*COSIM_FMI_INTERFACE 14-9
*COSIM_SYC_CONTROL 14-15
*COSIM_SYC_INTERFACE 14-16

*DAMPING 15-1
*DAMPING_FREQUENCY_RANGE 15-2
*DAMPING_GLOBAL 15-8
*DAMPING_PART_MASS 15-10
*DAMPING_PART_STIFFNESS 15-13
*DAMPING_PART_STRUCTURAL 15-15
*DAMPING_RELATIVE 15-16
*DAMPING_STRUCTURAL 15-18

*DATABASE 16-1
*DATABASE 16-4
*DATABASE_ACEOUT 16-20

TABLE OF CONTENTS

*DATABASE_ADAMS 16-21
*DATABASE_ALE 16-23
*DATABASE_ALE_MAT 16-26
*DATABASE_ALE_OPERATION 16-27
*DATABASE_BINARY 16-32
*DATABASE_BINARY_D3PROP 16-42
*DATABASE_CPM_SENSOR 16-43
*DATABASE_CROSS_SECTION 16-47
*DATABASE_DEFRAGMENT 16-53
*DATABASE_D3FTG 16-54
*DATABASE_D3MAX 16-56
*DATABASE_ENVELOPE_BEAM 16-60
*DATABASE_EXTENT 16-62
    *DATABASE_EXTENT_AVS 16-63
    *DATABASE_EXTENT_BINARY 16-67
    *DATABASE_EXTENT_D3PART 16-82
    *DATABASE_EXTENT_INTFOR 16-85
    *DATABASE_EXTENT_MOVIE 16-89
    *DATABASE_EXTENT_MPGS 16-90
    *DATABASE_EXTENT_SSSTAT 16-91
*DATABASE_FATIGUE_RAINFLOW_MATRIX 16-92
*DATABASE_FATIGUE_STRESS_CYCLE 16-93
*DATABASE_FATXML 16-94
*DATABASE_FORCE 16-95
*DATABASE_FORMAT 16-98
*DATABASE_FREQUENCY_ASCII 16-100
*DATABASE_FREQUENCY_BINARY 16-103
*DATABASE_FSI 16-111
*DATABASE_FSI_SENSOR 16-116
*DATABASE_HISTORY 16-119
*DATABASE_HISTORY_ACOUSTIC 16-125
*DATABASE_ISPHHTC 16-126
*DATABASE_MASSOUT 16-127
*DATABASE_MAX 16-128
*DATABASE_NODAL_FORCE_GROUP 16-130
*DATABASE_PAP_OUTPUT 16-132
*DATABASE_PBLAST_SENSOR 16-133

*DATABASE_PROFILE 16-135
*DATABASE_PWP_FLOW 16-138
*DATABASE_PWP_OUTPUT 16-139
*DATABASE_RCFORC_MOMENT 16-141
*DATABASE_RECOVER_NODE 16-142
*DATABASE_RVE 16-144
*DATABASE_SALE 16-145
*DATABASE_SPRING_FORWARD 16-148
*DATABASE_SUPERPLASTIC_FORMING 16-149
*DATABASE_TRACER 16-150
*DATABASE_TRACER_ALE 16-153
*DATABASE_TRACER_GENERAL 16-156
*DATABASE_TRACER_GENERATE 16-160

*DEFINE 17-1
*DEFINE_ADAPTIVE_SOLID_TO_DES 17-8
*DEFINE_ADAPTIVE_SOLID_TO_SPH 17-12
*DEFINE_BEAM_SOLID_COUPLING 17-16
*DEFINE_BIRTH 17-17
*DEFINE_BOLTED_JOINTS 17-18
*DEFINE_BOX 17-23
*DEFINE_BOX_ADAPTIVE 17-26
*DEFINE_BOX_COARSEN 17-32
*DEFINE_BOX_DRAWBEAD 17-35
*DEFINE_BOX_NODES_ADAPTIVE 17-37
*DEFINE_BOX_SPH 17-42
*DEFINE_CABLE 17-46
*DEFINE_CONNECTION_PROPERTIES 17-49
*DEFINE_CONSTRUCTION_STAGES 17-59
*DEFINE_CONTACT_EXCLUSION 17-61
*DEFINE_CONTACT_VOLUME 17-63
*DEFINE_CONTROL_VOLUME 17-66
*DEFINE_CONTROL_VOLUME_FLOW_AREA 17-70
*DEFINE_CONTROL_VOLUME_INTERACTION 17-72
*DEFINE_COORDINATE_NODES 17-74
*DEFINE_COORDINATE_SYSTEM 17-77
*DEFINE_COORDINATE_VECTOR 17-83

TABLE OF CONTENTS

*DEFINE_CPG_ADAPTIVE 17-85
*DEFINE_CPG_GAS_PROPERTIES 17-86
*DEFINE_CPG_REGION 17-88
*DEFINE_CPG_VENT 17-90
*DEFINE_CPM_BAG_INTERACTION 17-91
*DEFINE_CPM_CHAMBER 17-93
*DEFINE_CPM_GAS_PROPERTIES 17-96
*DEFINE_CPM_NPDATA 17-99
*DEFINE_CPM_SWITCH_REGION 17-101
*DEFINE_CPM_VENT 17-102
*DEFINE_CURVE 17-107
*DEFINE_CURVE_BOX_ADAPTIVITY 17-112
*DEFINE_CURVE_COMPENSATION_CONSTRAINT 17-116
*DEFINE_CURVE_DRAWBEAD 17-122
*DEFINE_CURVE_DUPLICATE 17-125
*DEFINE_CURVE_ENTITY 17-126
*DEFINE_CURVE_FEEDBACK 17-128
*DEFINE_CURVE_FLC 17-131
*DEFINE_CURVE_FLD_FROM_TRIAXIAL_LIMIT 17-134
*DEFINE_CURVE_FUNCTION 17-137
*DEFINE_CURVE_SMOOTH 17-157
*DEFINE_CURVE_STRESS 17-159
*DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_FLD 17-162
*DEFINE_CURVE_TRIM 17-165
*DEFINE_DE_ACTIVE_REGION 17-184
*DEFINE_DE_BOND 17-186
*DEFINE_DE_BOND_OVERRIDE 17-194
*DEFINE_DE_BY_PART 17-197
*DEFINE_DE_COHESIVE 17-200
*DEFINE_DE_FLOW_DRAG 17-202
*DEFINE_DE_HBOND 17-208
*DEFINE_DE_INJECT_BONDED 17-212
*DEFINE_DE_INJECT_SHAPE 17-216
*DEFINE_DE_INJECTION 17-219
*DEFINE_DE_INTERNAL_SKIP 17-224
*DEFINE_DE_MASSFLOW_PLANE 17-225
*DEFINE_DE_MESH_BEAM 17-226

*DEFINE_DE_MESH_SURFACE 17-228
*DEFINE_DE_MESH_VOLUME 17-230
*DEFINE_DE_PARTS_INTERACTION 17-232
*DEFINE_DE_PATTERN_OUTPUT 17-233
*DEFINE_DE_TEMP 17-235
*DEFINE_DE_TO_BEAM_COUPLING 17-236
*DEFINE_DE_TO_CPG_COUPLING 17-238
*DEFINE_DE_TO_CPM_COUPLING 17-240
*DEFINE_DE_TO_SURFACE_COUPLING 17-241
*DEFINE_DE_TO_SURFACE_TIED 17-248
*DEFINE_DEATH_TIMES 17-250
*DEFINE_DRIFT_REMOVE 17-253
*DEFINE_ELEMENT_DEATH 17-255
*DEFINE_ELEMENT_EROSION 17-257
*DEFINE_ELEMENT_GENERALIZED_SHELL 17-259
*DEFINE_ELEMENT_GENERALIZED_SOLID 17-264
*DEFINE_FABRIC_ASSEMBLIES 17-267
*DEFINE_FIBERS 17-269
*DEFINE_FIELD 17-273
*DEFINE_FILTER 17-277
*DEFINE_FORMING_ADAPTIVE_PART_INITIAL 17-279
*DEFINE_FORMING_BLANKMESH 17-280
*DEFINE_FORMING_CLAMP 17-286
*DEFINE_FORMING_ONESTEP_PRIMARY 17-290
*DEFINE_FP_TO_SURFACE_COUPLING 17-292
*DEFINE_FRICTION 17-294
*DEFINE_FRICTION_ORIENTATION 17-297
*DEFINE_FRICTION_SCALING 17-303
*DEFINE_FUNCTION 17-305
*DEFINE_FUNCTION_TABULATED 17-310
*DEFINE_GROUND_MOTION 17-312
*DEFINE_HAZ_PROPERTIES 17-313
*DEFINE_HAZ_TAILOR_WELDED_BLANK 17-316
*DEFINE_HEX_SPOTWELD_ASSEMBLY 17-317
*DEFINE_ISPG_TO_SURFACE_COUPLING 17-319
*DEFINE_LANCE_SEED_POINT_COORDINATES 17-323
*DEFINE_MATERIAL_HISTORIES 17-324

TABLE OF CONTENTS

*DEFINE_MULTI_DRAWBEADS_IGES 17-339
*DEFINE_MULTI_SHEET_CONNECTORS 17-341
*DEFINE_MULTISCALE 17-344
*DEFINE_NURBS_CURVE 17-345
*DEFINE_PART_FROM_LAYER 17-348
*DEFINE_PARTICLE_BLAST 17-351
*DEFINE_PBLAST_AIRGEO 17-357
*DEFINE_PBLAST_GEOMETRY 17-360
*DEFINE_PLANE 17-363
*DEFINE_POINT_CLOUD 17-365
*DEFINE_POROUS 17-368
*DEFINE_PRESSURE_TUBE 17-371
*DEFINE_QUASAR_COUPLING 17-382
*DEFINE_REGION 17-385
*DEFINE_SALEAB_PARAMETERS 17-390
*DEFINE_SD_ORIENTATION 17-392
*DEFINE_SET_ADAPTIVE 17-394
*DEFINE_SHELL_SWITCH 17-395
*DEFINE_SPG_TO_SPG_COUPLING 17-397
*DEFINE_SPG_TO_SURFACE_COUPLING 17-399
*DEFINE_SPH_ACTIVE_REGION 17-401
*DEFINE_SPH_AMBIENT_DRAG 17-406
*DEFINE_SPH_DE_COUPLING 17-408
*DEFINE_SPH_INJECTION 17-410
*DEFINE_SPH_INJECTION_SIMPLIFIED 17-412
*DEFINE_SPH_MASSFLOW_PLANE 17-415
*DEFINE_SPH_MESH_BOX 17-416
*DEFINE_SPH_MESH_OBJ 17-418
*DEFINE_SPH_MESH_SURFACE 17-420
*DEFINE_SPH_TO_SPH_COUPLING 17-421
*DEFINE_SPH_VICINITY_SENSOR 17-424
*DEFINE_SPOTWELD_FAILURE 17-425
*DEFINE_SPOTWELD_FAILURE_RESULTANTS 17-430
*DEFINE_SPOTWELD_MULTISCALE 17-432
*DEFINE_SPOTWELD_RUPTURE_PARAMETER 17-433
*DEFINE_SPOTWELD_RUPTURE_STRESS 17-436
*DEFINE_STAGED_CONSTRUCTION_PART 17-438

*DEFINE_STOCHASTIC_ELEMENT....17-441
*DEFINE_STOCHASTIC_VARIATION....17-442
*DEFINE_STOCHASTIC_VARIATION_PROPERTIES....17-449
*DEFINE_TABLE....17-454
*DEFINE_TABLE_2D....17-457
*DEFINE_TABLE_3D....17-459
*DEFINE_TABLE_{X}D....17-461
*DEFINE_TABLE_COMPACT....17-463
*DEFINE_TABLE_MATRIX....17-470
*DEFINE_TARGET_BOUNDARY....17-474
*DEFINE_TRACER_PARTICLES_2D....17-476
*DEFINE_TRANSFORMATION....17-477
*DEFINE_TRIM_SEED_POINT_COORDINATES....17-482
*DEFINE_TWINBUILDER_COUPLING....17-484
*DEFINE_VECTOR....17-488
*DEFINE_VECTOR_NODES....17-489
EXAMPLES....17-490

*DEFORMABLE_TO_RIGID....18-1
*DEFORMABLE_TO_RIGID....18-2
*DEFORMABLE_TO_RIGID_AUTOMATIC....18-3
*DEFORMABLE_TO_RIGID_INERTIA....18-9

*ELEMENT....19-1
*ELEMENT_BEAM....19-3
*ELEMENT_BEAM_PULLEY....19-18
*ELEMENT_BEAM_SOURCE....19-20
*ELEMENT_BEARING....19-22
*ELEMENT_BLANKING....19-27
*ELEMENT_DIRECT_MATRIX_INPUT....19-28
*ELEMENT_DISCRETE....19-31
*ELEMENT_DISCRETE_SPHERE....19-34
*ELEMENT_GENERALIZED_SHELL....19-40
*ELEMENT_GENERALIZED_SOLID....19-42
*ELEMENT_INERTIA....19-44
*ELEMENT_INTERPOLATION_SHELL....19-47
*ELEMENT_INTERPOLATION_SOLID....19-50
*ELEMENT_LANCING....19-53

TABLE OF CONTENTS

*ELEMENT_MASS 19-64
*ELEMENT_MASS_MATRIX 19-65
*ELEMENT_MASS_PART 19-67
*ELEMENT_PLOTEL 19-69
*ELEMENT_SEATBELT 19-70
*ELEMENT_SEATBELT_ACCELEROMETER 19-73
*ELEMENT_SEATBELT_PRETENSIONER 19-75
*ELEMENT_SEATBELT_RETRACTOR 19-81
*ELEMENT_SEATBELT_SENSOR 19-88
*ELEMENT_SEATBELT_SLIPRING 19-93
*ELEMENT_SHELL 19-99
*ELEMENT_SHELL_NURBS_PATCH 19-108
*ELEMENT_SHELL_SOURCE_SINK 19-118
*ELEMENT_SOLID 19-119
*ELEMENT_SOLID_NURBS_PATCH 19-127
*ELEMENT_SOLID_PERI 19-134
*ELEMENT_SOLID_REFERENCE 19-136
*ELEMENT_SPH 19-137
*ELEMENT_TRIM 19-139
*ELEMENT_TSHELL 19-140

*END 20-145

*EOS 21-1

*FATIGUE 22-2
*FATIGUE 22-3
*FATIGUE_FAILURE 22-9
*FATIGUE_LOADSTEP 22-10
*FATIGUE_MEAN_STRESS_CORRECTION 22-11
*FATIGUE_MULTIAXIAL 22-13
*FATIGUE_SUMMATION 22-15

*FREQUENCY_DOMAIN 23-1
*FREQUENCY_DOMAIN_ACCELERATION_UNIT 23-2
*FREQUENCY_DOMAIN_ACOUSTIC_BEM 23-4
*FREQUENCY_DOMAIN_ACOUSTIC_DIRECTIVITY 23-18
*FREQUENCY_DOMAIN_ACOUSTIC_FEM 23-20
*FREQUENCY_DOMAIN_ACOUSTIC_FREQUENCY 23-28

*FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT 23-31
*FREQUENCY_DOMAIN_ACOUSTIC_INCIDENT_WAVE 23-37
*FREQUENCY_DOMAIN_ACOUSTIC_SOUND_SPEED 23-39
*FREQUENCY_DOMAIN_FRF 23-41
*FREQUENCY_DOMAIN_LOCAL 23-56
*FREQUENCY_DOMAIN_MODE 23-57
*FREQUENCY_DOMAIN_PATH 23-61
*FREQUENCY_DOMAIN_RANDOM_VIBRATION 23-64
*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM 23-81
*FREQUENCY_DOMAIN_SEA_CONNECTION 23-94
*FREQUENCY_DOMAIN_SEA_INPUT 23-97
*FREQUENCY_DOMAIN_SEA_SUBSYSTEM 23-99
*FREQUENCY_DOMAIN_SSD 23-105

*HASH 24-119
*HASH 24-120
*HASH_END 24-121

*HOURGLASS 25-1

*IGA 26-1
*IGA_1D_BREP 26-3
*IGA_1D_NURBS_UVW 26-4
*IGA_1D_NURBS_XYZ 26-8
*IGA_2D_BASIS_TRANSFORM_XYZ 26-11
*IGA_2D_BEZIER_XYZ 26-12
*IGA_2D_BREP 26-13
*IGA_2D_NURBS_UVW 26-14
*IGA_2D_NURBS_XYZ 26-19
*IGA_3D_BASIS_TRANSFORM_XYZ 26-24
*IGA_3D_BEZIER_XYZ 26-25
*IGA_3D_NURBS_XYZ 26-26
*IGA_EDGE_UVW 26-32
*IGA_EDGE_XYZ 26-36
*IGA_FACE_UVW 26-39
*IGA_FACE_XYZ 26-43
*IGA_INCLUDE_BEZIER 26-47
*IGA_INTEGRATION_SHELL_REDUCE 26-49
*IGA_INTEGRATION_SOLID_REDUCE 26-50

TABLE OF CONTENTS

*IGA_MASS 26-51
*IGA_POINT_UVW 26-53
*IGA_REFINE_SHELL 26-55
*IGA_REFINE_SOLID 26-58
*IGA_SHELL 26-61
*IGA_SOLID 26-63
*IGA_TIED_EDGE_TO_EDGE 26-65
*IGA_VOLUME_XYZ 26-66

*INCLUDE 27-1
*INCLUDE 27-3
*INCLUDE_AUTO_OFFSET 27-9
*INCLUDE_COMPENSATION 27-12
*INCLUDE_COMPENSATION_BEFORE_SPRINGBACK 27-15
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK 27-16
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK 27-17
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE 27-18
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP 27-19
*INCLUDE_COMPENSATION_CURRENT_TOOLS 27-20
*INCLUDE_COMPENSATION_CURVE 27-21
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE 27-22
*INCLUDE_COMPENSATION_NEW_RIGID_TOOL 27-23
*INCLUDE_COMPENSATION_NODE_SET 27-24
*INCLUDE_COMPENSATION_ORIGINAL_DYNAIN 27-25
*INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL 27-26
*INCLUDE_COMPENSATION_ORIGINAL_TOOL 27-27
*INCLUDE_COMPENSATION_SPRINGBACK_INPUT 27-28
*INCLUDE_COMPENSATION_SYMMETRIC_LINES 27-29
*INCLUDE_COMPENSATION_TANGENT_CONSTRAINT 27-31
*INCLUDE_COMPENSATION_TRIM_CURVE 27-32
*INCLUDE_COMPENSATION_TRIM_NODE 27-33
*INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE 27-34
*INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL 27-35
*INCLUDE_COSIM 27-36
*INCLUDE_ISPG 27-43
*INCLUDE_MULTISCALE 27-45
*INCLUDE_MULTISCALE_SPOTWELD 27-48

*INCLUDE_PATH 27-51
*INCLUDE_STAMPED 27-53
*INCLUDE_STAMPED_PART_SOLID_TO_SOLID 27-63
*INCLUDE_TRIM 27-70
*INCLUDE_UNITCELL 27-72
*INCLUDE_WD 27-76
*INCLUDE_WD_FINAL_PART 27-77
*INCLUDE_WD_INITIAL_BLANK 27-78
*INCLUDE_WD_WELDING_CURVE 27-79

*INITIAL 28-1
*INITIAL_AIRBAG_PARTICLE_POSITION 28-4
*INITIAL_ALE_MAPPING 28-5
*INITIAL_AXIAL_FORCE_BEAM 28-10
*INITIAL_CONTACT_WEAR 28-13
*INITIAL_CPG 28-15
*INITIAL_CRASHFRONT 28-17
*INITIAL_DETONATION 28-18
*INITIAL_DETONATION_GEOMETRY 28-22
*INITIAL_EOS_ALE 28-29
*INITIAL_FATIGUE_DAMAGE_RATIO 28-31
*INITIAL_FIELD_SOLID 28-36
*INITIAL_FOAM_REFERENCE_GEOMETRY 28-38
*INITIAL_GAS_MIXTURE 28-40
*INITIAL_HISTORY_NODE 28-42
*INITIAL_HYDROSTATIC_ALE 28-45
*INITIAL_IMPULSE_MINE 28-49
*INITIAL_INTERNAL_DOF_SOLID 28-53
*INITIAL_LAG_MAPPING 28-55
*INITIAL_MOMENTUM 28-59
*INITIAL_PWP_DEPTH 28-60
*INITIAL_PWP_NODAL_DATA 28-61
*INITIAL_SOLID_VOLUME 28-62
*INITIAL_STRAIN_BEAM 28-63
*INITIAL_STRAIN_IGA_SHELL 28-64
*INITIAL_STRAIN_SHELL 28-67
*INITIAL_STRAIN_SHELL_NURBS_PATCH 28-70

TABLE OF CONTENTS

*INITIAL_STRAIN_SOLID 28-73
*INITIAL_STRAIN_SOLID_NURBS_PATCH 28-75
*INITIAL_STRAIN_TSHELL 28-77
*INITIAL_STRESS_BEAM 28-79
*INITIAL_STRESS_DEPTH 28-85
*INITIAL_STRESS_DES 28-87
*INITIAL_STRESS_IGA_SHELL 28-88
*INITIAL_STRESS_SECTION 28-91
*INITIAL_STRESS_SHELL 28-95
*INITIAL_STRESS_SHELL_NURBS_PATCH 28-100
*INITIAL_STRESS_SOLID 28-103
*INITIAL_STRESS_SOLID_NURBS_PATCH 28-108
*INITIAL_STRESS_SPH 28-111
*INITIAL_STRESS_TSHELL 28-112
*INITIAL_TEMPERATURE 28-115
*INITIAL_TIED_CONTACT_DATA 28-117
*INITIAL_VAPOR_PART 28-119
*INITIAL_VEHICLE_KINEMATICS 28-120
*INITIAL_VELOCITY 28-123
*INITIAL_VELOCITY_GENERATION 28-127
*INITIAL_VELOCITY_GENERATION_START_TIME 28-131
*INITIAL_VELOCITY_NODE 28-132
*INITIAL_VELOCITY_RIGID_BODY 28-133
*INITIAL_VOID 28-134
*INITIAL_VOLUME_FRACTION 28-135
*INITIAL_VOLUME_FRACTION_GEOMETRY 28-139

*INTEGRATION 29-1
*INTEGRATION_BEAM 29-2
*INTEGRATION_SHELL 29-16

*INTERFACE 30-1
*INTERFACE_ACOUSTIC 30-3
*INTERFACE_BLANKSIZE 30-4
Card Set for *INTERFACE_BLANKSIZE_DEVELOPMENT 30-5
Card Set for *INTERFACE_BLANKSIZE_INITIAL_TRIM 30-10
Card Set for *INTERFACE_BLANKSIZE_INITIAL_ADAPTIVE 30-13
Card Set for *INTERFACE_BLANKSIZE_SCALE_FACTOR 30-15

Card Set for *INTERFACE_BLANKSIZE_SYMMETRIC_PLANE 30-16
Remarks and Examples 30-17
*INTERFACE_COMPENSATION_3D 30-34
*INTERFACE_COMPENSATION_TRIMMING_CURVE 30-59
*INTERFACE_COMPONENT_FILE 30-61
*INTERFACE_COMPONENT 30-63
*INTERFACE_DE_HBOND 30-66
*INTERFACE_LINKING_DISCRETE_NODE 30-69
*INTERFACE_LINKING_EDGE 30-70
*INTERFACE_LINKING_FILE 30-71
*INTERFACE_LINKING_FORCES 30-72
*INTERFACE_LINKING_NODE 30-73
*INTERFACE_LINKING_SEGMENT 30-77
*INTERFACE_SPG_1 30-78
*INTERFACE_SPG_2 30-79
*INTERFACE_SPRINGBACK 30-80
*INTERFACE_SSI 30-90
*INTERFACE_SSI_AUX 30-95
*INTERFACE_SSI_AUX_EMBEDDED 30-96
*INTERFACE_SSI_STATIC 30-98
*INTERFACE_THICKNESS_CHANGE_COMPENSATION 30-100
*INTERFACE_WELDLINE_DEVELOPMENT 30-102

*ISPG 31-1
*ISPG_ASCII_CPL 31-3
*ISPG_ASCII_SUMFORC 31-4
*ISPG_BOUNDARY_CONVECTION_SET 31-5
*ISPG_BOUNDARY_SYMMETRY 31-7
*ISPG_CONTROL_ADAPTIVITY 31-8
*ISPG_CONTROL_D3ISPG 31-12
*ISPG_CONTROL_IMPLICIT 31-14
*ISPG_CONTROL_MPP 31-16
*ISPG_CONTROL_SOLUTION 31-17
*ISPG_DAMPING_SURF 31-18
*ISPG_DEFINE_MERGER_MPP 31-19
*ISPG_INITIAL_TEMPERATURE_SET 31-20
*ISPG_LOAD_GRAVITY 31-21

TABLE OF CONTENTS

*ISPG_TO_ISPG_CONTACT 31-22
*ISPG_TRANS_CPL 31-23

*KEYWORD 32-1

*LOAD 33-1
*LOAD_ACOUSTIC_SOURCE 33-4
*LOAD_ALE_CONVECTION 33-7
*LOAD_BEAM 33-9
*LOAD_BLAST 33-11
*LOAD_BLAST_CLEARING 33-14
*LOAD_BLAST_ENHANCED 33-16
*LOAD_BLAST_SEGMENT 33-23
*LOAD_BLAST_SEGMENT_SET 33-24
*LOAD_BODY 33-25
*LOAD_BODY_GENERALIZED 33-31
*LOAD_BODY_POROUS 33-35
*LOAD_BRODE 33-38
*LOAD_DENSITY_DEPTH 33-40
*LOAD_EDGE_UVW 33-42
*LOAD_ERODING_PART_SET 33-43
*LOAD_EXPANSION_PRESSURE 33-46
*LOAD_EXTERNAL_VARIABLE 33-49
*LOAD_FACE_UVW 33-56
*LOAD_FACE_XYZ 33-57
*LOAD_GRAVITY_PART 33-58
*LOAD_HEAT_CONTROLLER 33-60
*LOAD_HEAT_EXOTHERMIC_REACTION 33-61
*LOAD_HEAT_GENERATION 33-68
*LOAD_HEAT_LASER_ABSORPTION 33-70
*LOAD_MASK 33-76
*LOAD_MOTION_NODE 33-78
*LOAD_MOVING_PRESSURE 33-80
*LOAD_NODE 33-85
*LOAD_NURBS_SHELL 33-90
*LOAD_POINT_UVW 33-97
*LOAD_PYRO_ACTUATOR 33-98
*LOAD_PZE 33-101

*LOAD_REMOVE_PART 33-102
*LOAD_RIGID_BODY 33-104
*LOAD_SEGMENT 33-107
*LOAD_SEGMENT_ALE_INI 33-112
*LOAD_SEGMENT_CONTACT_MASK 33-113
*LOAD_SEGMENT_FILE 33-115
*LOAD_SEGMENT_FSILNK 33-117
*LOAD_SEGMENT_NONUNIFORM 33-120
*LOAD_SEGMENT_SET 33-123
*LOAD_SEGMENT_SET_ANGLE 33-125
*LOAD_SEGMENT_SET_NONUNIFORM 33-127
*LOAD_SEISMIC_FREE_FIELD 33-130
*LOAD_SEISMIC_SSI 33-134
*LOAD_SEISMIC_SSI_AUX 33-139
*LOAD_SHELL 33-141
*LOAD_SPCFORC 33-144
*LOAD_SSA 33-145
*LOAD_STEADY_STATE_ROLLING 33-149
*LOAD_STIFFEN_PART 33-153
*LOAD_SUPERPLASTIC_FORMING 33-155
*LOAD_SURFACE_STRESS 33-161
*LOAD_THERMAL 33-163
*LOAD_THERMAL_BINOUT 33-165
*LOAD_THERMAL_CONSTANT 33-167
*LOAD_THERMAL_CONSTANT_ELEMENT 33-169
*LOAD_THERMAL_CONSTANT_NODE 33-170
*LOAD_THERMAL_D3PLOT 33-171
*LOAD_THERMAL_LOAD_CURVE 33-172
*LOAD_THERMAL_RSW 33-173
Node defining the tail of the orientation vector (axis of rotation of the ellipsoidal region) and the base for positioning of the nugget. See Remarks 1 and 2. 33-174
*LOAD_THERMAL_TOPAZ 33-179
*LOAD_THERMAL_VARIABLE 33-180
*LOAD_THERMAL_VARIABLE_BEAM 33-182
*LOAD_THERMAL_VARIABLE_ELEMENT 33-185
*LOAD_THERMAL_VARIABLE_NODE 33-186
*LOAD_THERMAL_VARIABLE_SHELL 33-187

TABLE OF CONTENTS

*LOAD_VOLUME_LOSS 33-189

*MODULE 34-191

*MODULE_LOAD 34-192

*MODULE_PATH 34-194

*MODULE_USE 34-195

*NODE 35-1

*NODE 35-2

*NODE_MERGE_SET 35-4

*NODE_MERGE_TOLERANCE 35-5

*NODE_REFERENCE 35-6

*NODE_RIGID_SURFACE 35-7

*NODE_SCALAR 35-8

*NODE_THICKNESS 35-10

*NODE_TO_TARGET_VECTOR 35-12

*NODE_TRANSFORM 35-13

*PARAMETER 36-1

*PARAMETER 36-2

*PARAMETER_DUPLICATION 36-6

*PARAMETER_EXPRESSION 36-7

*PARAMETER_TYPE 36-11

*PART 37-1

*PART 37-2

*PART_ADAPTIVE_FAILURE 37-18

*PART_ANNEAL 37-19

*PART_COMPOSITE 37-20

*PART_DUPLICATE 37-32

*PART_MODES 37-35

*PART_MOVE 37-40

*PART_SENSOR 37-45

*PART_STACKED_ELEMENTS 37-46

*PERTURBATION 38-1

*PERTURBATION 38-2

*RAIL 39-1

*RAIL_TRACK 39-2

*RAIL_TRAIN 39-8

*RIGIDWALL 40-1
*RIGIDWALL_FORCE_TRANSDUCER 40-2
*RIGIDWALL_GEOMETRIC 40-4
*RIGIDWALL_PLANAR 40-16
*SECTION 41-1
*SECTION_ALE1D 41-3
*SECTION_ALE2D 41-7
*SECTION_BEAM 41-9
*SECTION_BEAM_AISC 41-34
*SECTION_DISCRETE 41-38
*SECTION_FPD 41-41
*SECTION_IGA_SHELL 41-44
*SECTION_IGA_SOLID 41-48
*SECTION_ISPG 41-49
*SECTION_POINT_SOURCE 41-51
*SECTION_POINT_SOURCE_MIXTURE 41-54
*SECTION_SEATBELT 41-60
*SECTION_SHELL 41-62
*SECTION_SOLID 41-85
*SECTION_SOLID_PERI 41-108
*SECTION_SPH 41-110
*SECTION_TSHELL 41-114
*SENSOR 42-1
*SENSOR_CONTROL 42-4
*SENSOR_CPM_AIRBAG 42-11
*SENSOR_DEFINE_CALC-MATH 42-13
*SENSOR_DEFINE_ELEMENT 42-16
*SENSOR_DEFINE_FORCE 42-22
*SENSOR_DEFINE_FUNCTION 42-26
*SENSOR_DEFINE_MISC 42-28
*SENSOR_DEFINE_NODE 42-32
*SENSOR_SWITCH 42-36
*SENSOR_SWITCH_CALC-LOGIC 42-37
*SENSOR_SWITCH_SHELL_TO_VENT 42-39
*SET 43-1

TABLE OF CONTENTS

*SET_BEAM 43-3
*SET_BEAM_ADD 43-8
*SET_BEAM_INTERSECT 43-10
*SET_BOX 43-12
*SET_DISCRETE 43-14
*SET_DISCRETE_ADD 43-18
*SET_IGA_EDGE 43-20
*SET_IGA_FACE 43-24
*SET_IGA_POINT_UVW 43-28
*SET_MODE 43-32
*SET_MULTI 43-35
*SET_MULTI-MATERIAL_GROUP_LIST 43-35
*SET_NODE 43-37
*SET_NODE_ADD 43-45
*SET_NODE_INTERSECT 43-48
*SET_PART 43-50
*SET_PART_ADD 43-57
*SET_PART_TREE 43-59
*SET_PERI_LAMINATE 43-62
*SET_SEGMENT 43-63
*SET_SEGMENT_ADD 43-72
*SET_SEGMENT_INTERSECT 43-74
*SET_SEGMENT_N8 43-76
*SET_SHELL 43-78
*SET_SHELL_ADD 43-86
*SET_SHELL_INTERSECT 43-88
*SET_SOLID 43-90
*SET_SOLID_ADD 43-97
*SET_SOLID_INTERSECT 43-99
*SET_TSHELL 43-101
*SET_2D_SEGMENT 43-105

*RVE 44-1
*RVE_ANALYSIS_FEM 44-2

*TERMINATION 45-1
*TERMINATION_BODY 45-2
*TERMINATION_CONTACT 45-3

*TERMINATION_CURVE 45-4
*TERMINATION_DELETED_SHELLS 45-5
*TERMINATION_DELETED_SOLIDS 45-6
*TERMINATION_FATIGUE 45-7
*TERMINATION_NODE 45-8
*TERMINATION_SENSOR 45-9

*TITLE 46-1

*UNIT 47-1
*UNIT_DEFAULTS 47-2
*UNIT_DERIVED 47-5
*UNIT_AMOUNT 47-7
*UNIT_ANGLE 47-8
*UNIT_ELECTRIC_CURRENT 47-9
*UNIT_LENGTH 47-10
*UNIT_LUMINOUS_INTENSITY 47-11
*UNIT_MASS 47-12
*UNIT_SYSTEM 47-13
*UNIT_TEMPERATURE 47-14
*UNIT_TIME 47-15

*USER 48-1
*USER_GET_HISTORY 48-2
*USER_INTERFACE 48-5
*USER_LOADING 48-9
*USER_LOADING_SET 48-10
*USER_NONLOCAL_SEARCH 48-13

Restart Input Data 49-1
*CHANGE 49-4
*BOUNDARY_CONDITION 49-4
*CONTACT_SMALL_PENETRATION 49-5
*CURVE_DEFINITION 49-5
*OUTPUT 49-6
*RIGIDWALL_GEOMETRIC 49-6
*RIGIDWALL_PLANAR 49-6
*RIGID_BODY_CONSTRAINT 49-6
*RIGID_BODY_INERTIA 49-7

TABLE OF CONTENTS

RIGID_BODY_STOPPERS 49-8
STATUS_REPORT_FREQUENCY 49-10
THERMAL_PARAMETERS 49-11
VELOCITY 49-12
VELOCITY_GENERATION 49-13
VELOCITY_NODE 49-13
VELOCITY_RIGID_BODY 49-14
VELOCITY_ZERO 49-15
*CONTROL_DYNAMIC_RELAXATION 49-16
*CONTROL_SHELL 49-18
*CONTROL_TERMINATION 49-20
*CONTROL_TIMESTEP 49-21
*DAMPING_GLOBAL 49-23
*DATABASE 49-24
*DATABASE_BINARY 49-26
*DELETE 49-27
*INTERFACE_SPRINGBACK_LSDYNA 49-29
*RIGID_DEFORMABLE 49-31
*RIGID_DEFORMABLE_CONTROL 49-32
*RIGID_DEFORMABLE_D2R 49-33
*RIGID_DEFORMABLE_R2D 49-34
*STRESS_INITIALIZATION 49-35
*TERMINATION 49-37
*TITLE 49-40

REFERENCES 50-1

APPENDIX A: User-Defined Materials 51-1
Getting Started with User-Defined Features 51-1
Download 51-1
Static or dynamic linking 51-1
Compiler and compiling 51-2
Execution 51-3
Coding 51-4
Module concept 51-4
General Overview of User-Defined Materials 51-6
Additional Features 51-11
Load curves and tables 51-11

Local coordinate system....51-12
Temperature....51-13
Failure....51-13
Deformation gradient....51-13
Implicit analysis....51-18
User-defined materials with equations-of-state....51-21
Post-processing a user-defined material....51-23

APPENDIX B: User-Defined Equation-of-State....52-1
General overview....52-1
Example and Discussion....52-2

APPENDIX C: User-Defined Element Interface for Solids and Shells....53-1
Algorithm Outline....53-1
Invoking User-Defined Elements....53-2
Integrated Elements....53-2
Resultant/Discrete Elements....53-5
Additional Features of User-Defined Elements....53-6
Nodal Fiber Vectors....53-6
Extra Degrees of Freedom....53-6
Section Keywords....53-8
The *SECTION_SHELL Card....53-8
The *SECTION_SOLID Card....53-10
Sample User Shell Element 101 (Belytschko-Tsay shell)....53-11
Sample User Solid Element 101 (constant stress solid)....53-14
Examples....53-18

APPENDIX D: User Defined Airbag Sensor....54-1

APPENDIX E: User Defined Solution Control....55-1

APPENDIX F: User Defined Interface Control....56-1

APPENDIX G: User-Defined Interface Friction and Conductivity....57-1

APPENDIX H: User-Defined Thermal Material Model....58-1

APPENDIX I: Occupant Simulation Including the Coupling to the CAL3D and MADYMO programs....59-1
INTRODUCTION....59-1
THE LS-DYNA/OCCUPANT SIMULATION PROGRAM LINK....59-1
AIRBAG MODELING....59-2
COMMON ERRORS....59-3

TABLE OF CONTENTS

APPENDIX J: Interactive Graphics Commands 60-1

APPENDIX K: Interactive Material Model Driver 61-1

INTRODUCTION 61-1

INPUT DEFINITION 61-1

INTERACTIVE DRIVER COMMANDS 61-2

APPENDIX L: VDA Database 62-1

APPENDIX M: Commands for Two-Dimensional Rezoning 63-1

APPENDIX N: Rigid-Body Dummies 64-1

APPENDIX O: LS-DYNA MPP User Guide 65-1

Supported Features 65-1

Contact Interfaces 65-1

Output Files and Post-Processing 65-2

Parallel Specific Options 65-4

PFILE 65-4

Execution of MPP/LS-DYNA 65-15

APPENDIX P: Implicit Solver 66-1

Introduction 66-1

Nonlinear Implicit Analysis 66-2

Linear Equation Solver 66-12

Elements and Materials 66-16

Contacts 66-17

Troubleshooting Convergence Problems 66-22

Checklist 66-25

APPENDIX Q: User Defined Weld Failure 67-1

APPENDIX R: User Defined Cohesive Model 68-1

APPENDIX S: User Defined Boundary Flux 69-1

APPENDIX T: Metal Forming Glossary 70-1

A Typical DRAW DIE engineering PROCESS 70-1

Types of draw dies 70-3

Types of flanging dies 70-4

Types of hemming dies 70-4

APPENDIX U: Reserved Variable Names 71-1

Appendix V: How to Read Card Summaries 73-1
Motivation for Introducing the Card Summary Section. 73-2
“Optional” Cards (How LS-DYNA Parses Blank Lines). 73-3
The Structure of Manual Entries. 73-4
Summary Table Format 73-5
Example 1. 73-6
Example 2. 73-7
Example 3. 73-8
Default Values for Data Cards. 73-9

Appendix W: Acoustic Volume Element and Spectral Element Solvers 74-1
Explicit Transient Acoustic Spectral Element Solutions 74-2
Implicit Direct Steady State Acoustic FE Solutions 74-3

Appendix X: Multistage Analysis 75-1
Why Use Stages 75-1
Example of Splitting into Stages 75-2
The “dynain” approach 75-3
Formats 75-4
LSDA format 75-4
Exporting Data 75-7
Outputting Data with *INTERFACE_SPRINGBACK_LSDYNA 75-7
PSID = 999 75-7
NHSV = 999 75-8
FTYPE = 3 75-9
NDFLAG = 1 75-9
CFLAG = 1 75-9
HFLAG = 1 75-11
Boundary Conditions 75-12
Importing data 75-12
Excluding Parts in Subsequent Stages 75-12
Changing Element and Material Types 75-14
Inertia Properties of Rigid Bodies 75-15
Deformation Gradient and Stress Update 75-17
Contact 75-17
Stabilization Forces 75-20
Cases 75-20

TABLE OF CONTENTS

APPENDIX Y: LS-DYNA HYBRID User Guide 76-1
- INTRODUCTION 76-1
- LS-DYNA HYBRID Version 76-2
- Performance and Scalability 76-2
- Output Consistency 76-3
- Supported Features 76-4
- Contact Interfaces 76-4
- Output Files and Post-Processing 76-4
- Parallel Specific Options 76-4
- PFILE 76-4
- Execution of LS-DYNA HYBRID 76-4

APPENDIX Z: Using the System Coupling Software with LS-DYNA 77-6
- About the System Coupling Software 77-6
- Coupling with LS-DYNA 77-6
  - Supported Couplings 77-6
  - Versions 77-7
  - Fluid-Structure Interaction (FSI) using Fluent and LS-DYNA 77-7
- Coupled Simulation Setup and Post-Processing 77-9
  - LS-DYNA Input 77-9
  - Fluent Input 77-9
  - Setting up the Solution in the System Coupling GUI 77-10
  - Post-Processing Interface Results 77-11
- Restarts 77-11
- Examples 77-12
  - Reed Valve Example 77-12
  - Turek-Hron FSI Benchmark 77-17
- Limitations 77-18

INTRODUCTION

CHRONOLOGICAL HISTORY

DYNA3D originated at the Lawrence Livermore National Laboratory [Hallquist 1976]. The early applications were primarily for the stress analysis of structures subjected to a variety of impact loading. These applications required what was then significant computer resources, and the need for a much faster version was immediately obvious. Part of the speed problem was related to the inefficient implementation of the element technology which was further aggravated by the fact that supercomputers in 1976 were much slower than today's PC. Furthermore, the primitive sliding interface treatment could only treat logically regular interfaces that are uncommon in most finite element discretizations of complicated three-dimensional geometries; consequently, defining a suitable mesh for handling contact was often very difficult. The first version contained trusses, membranes, and a choice of solid elements. The solid elements ranged from a one-point quadrature eight-noded element with hourglass control to a twenty-noded element with eight integration points. Due to the high cost of the twenty node solid, the zero energy modes related to the reduced 8-point integration, and the high frequency content which drove the time step size down, higher order elements were all but abandoned in later versions of DYNA3D. A two-dimensional version, DYNA2D, was developed concurrently.

A new version of DYNA3D was released in 1979 that was programmed to provide near optimal speed on the CRAY-1 supercomputers, contained an improved sliding interface treatment that permitted triangular segments and was an order of magnitude faster than the previous contact treatment. The 1979 version eliminated structural and higher order solid elements and some of the material models of the first version. This version also included an optional element-wise implementation of the integral difference method developed by Wilkins et al. [1974].

The 1981 version [Hallquist 1981a] evolved from the 1979 version. Nine additional material models were added to allow a much broader range of problems to be modeled including explosive-structure and soil-structure interactions. Body force loads were implemented for angular velocities and base accelerations. A link was also established from the 3D Eulerian code, JOY [Couch, et. al., 1983] for studying the structural response to impacts by penetrating projectiles. An option was provided for storing element data on disk thereby doubling the capacity of DYNA3D.

The 1982 version of DYNA3D [Hallquist 1982] accepted DYNA2D [Hallquist 1980] material input directly. The new organization was such that equations of state and constitutive models of any complexity could be easily added. Complete vectorization of the

material models had been nearly achieved with about a 10 percent increase in execution speed over the 1981 version.

In the 1986 version of DYNA3D [Hallquist and Benson 1986], many new features were added, including beams, shells, rigid bodies, single surface contact, interface friction, discrete springs and dampers, optional hourglass treatments, optional exact volume integration, and VAX/ VMS, IBM, UNIX, COS operating systems compatibility, that greatly expanded its range of applications. DYNA3D thus became the first code to have a general single surface contact algorithm.

In the 1987 version of DYNA3D [Hallquist and Benson 1987] metal forming simulations and composite analysis became a reality. This version included shell thickness changes, the Belytschko-Tsay shell element [Belytschko and Tsay, 1981], and dynamic relaxation. Also included were non-reflecting boundaries, user-specified integration rules for shell and beam elements, a layered composite damage model, and single point constraints.

New capabilities added in the 1988 DYNA3D [Hallquist 1988] version included a cost-effective resultant beam element, a truss element, a C^0 triangular shell, the BCIZ triangular shell [Bazeley et al. 1965], mixing of element formulations in calculations, composite failure modeling for solids, noniterative plane stress plasticity, contact surfaces with spot welds, tie break sliding surfaces, beam surface contact, finite stonewalls, stonewall reaction forces, energy calculations for all elements, a crushable foam constitutive model, comment cards in the input, and one-dimensional slidelines.

By the end of 1988 it was obvious that a much more concentrated effort would be required in the development of this software if problems in crashworthiness were to be properly solved; therefore, Livermore Software Technology Corporation (LSTC) was founded to continue the development of DYNA3D as a commercial version called LS-DYNA3D which was later shortened to LS-DYNA. The 1989 release introduced many enhanced capabilities including a one-way treatment of slide surfaces with voids and friction; cross-sectional forces for structural elements; an optional user-specified minimum time step size for shell elements using elastic and elastoplastic material models; nodal accelerations in the time history database; a compressible Mooney-Rivlin material model; a closed-form update shell plasticity model; a general rubber material model; unique penalty specifications for each slide surface; external work tracking; optional time step criterion for 4-node shell elements; and internal element sorting to allow full vectorization of right-hand-side force assembly.

Over the years, considerable progress has been made as may be seen in the chronology of the developments discussed below. In 2019, LSTC was acquired by Ansys, and this software was renamed Ansys LS-DYNA®.

Capabilities added in 1989-1990:

- arbitrary node and element numbers,
- fabric model for seat belts and airbags,
- composite glass model,
- vectorized type 3 contact and single surface contact,
- many more I/O options,
- all shell materials available for 8 node thick shell,
- strain rate dependent plasticity for beams,
- fully vectorized iterative plasticity,
- interactive graphics on some computers,
- nodal damping,
- shell thickness taken into account in shell type 3 contact,
- shell thinning accounted for in type 3 and type 4 contact,
- soft stonewalls,
- print suppression option for node and element data,
- massless truss elements, rivets – based on equations of rigid body dynamics,
- massless beam elements, spot welds – based on equations of rigid body dynamics,
- expanded databases with more history variables and integration points,
- force limited resultant beam,
- rotational spring and dampers, local coordinate systems for discrete elements,
- resultant plasticity for C^0 triangular element,
- energy dissipation calculations for stonewalls,
- hourglass energy calculations for solid and shell elements,
- viscous and Coulomb friction with arbitrary variation over surface,
- distributed loads on beam elements,
- Cowper and Symonds strain rate model,
- segmented stonewalls,
- stonewall Coulomb friction,
- stonewall energy dissipation,
- airbags (1990),
- nodal rigid bodies,

- automatic sorting of triangular shells into C^0 groups,
- mass scaling for quasi static analyses,
- user defined subroutines,
- warpage checks on shell elements,
- thickness consideration in all contact types,
- automatic orientation of contact segments,
- sliding interface energy dissipation calculations,
- nodal force and energy database for applied boundary conditions,
- defined stonewall velocity with input energy calculations,

Capabilities added in 1991-1992:

- rigid/deformable material switching,
- rigid bodies impacting rigid walls,
- strain-rate effects in metallic honeycomb model 26,
- shells and beams interfaces included for subsequent component analyses,
- external work computed for prescribed displacement/velocity/accelerations,
- linear constraint equations,
- MPGS database,
- MOVIE database,
- Slideline interface file,
- automated contact input for all input types,
- automatic single surface contact without element orientation,
- constraint technique for contact,
- cut planes for resultant forces,
- crushable cellular foams,
- urethane foam model with hysteresis,
- subcycling,
- friction in the contact entities,
- strains computed and written for the 8 node thick shells,
- "good" 4 node tetrahedron solid element with nodal rotations,
- 8 node solid element with nodal rotations,
- 2 × 2 integration for the membrane element,

- Belytschko-Schwer integrated beam,
- thin-walled Belytschko-Schwer integrated beam,
- improved TAURUS database control,
- null material for beams to display springs and seatbelts in TAURUS,
- parallel implementation on Crays and SGI computers,
- coupling to rigid body codes,
- seat belt capability.

Capabilities added in 1993-1994:

- Arbitrary Lagrangian Eulerian brick elements,
- Belytschko-Wong-Chiang quadrilateral shell element,
- Warping stiffness in the Belytschko-Tsay shell element,
- Fast Hughes-Liu shell element,
- Fully integrated thick shell element,
- Discrete 3D beam element,
- Generalized dampers,
- Cable modeling,
- Airbag reference geometry,
- Multiple jet model,
- Generalized joint stiffnesses,
- Enhanced rigid body to rigid body contact,
- Orthotropic rigid walls,
- Time zero mass scaling,
- Coupling with USA (Underwater Shock Analysis),
- Layered spot welds with failure based on resultants or plastic strain,
- Fillet welds with failure,
- Butt welds with failure,
- Automatic eroding contact,
- Edge-to-edge contact,
- Automatic mesh generation with contact entities,
- Drawbead modeling,
- Shells constrained inside brick elements,

- NIKE3D coupling for springback,
- Barlat's anisotropic plasticity,
- Superplastic forming option,
- Rigid body stoppers,
- Keyword input,
- Adaptivity,
- First MPP (Massively Parallel) version with limited capabilities.
- Built in least squares fit for rubber model constitutive constants,
- Large hysteresis in hyperelastic foam,
- Bilhku/Dubois foam model,
- Generalized rubber model,

Capabilities added in 1995:

- Belytschko - Leviathan Shell
- Automatic switching between rigid and deformable bodies.
- Accuracy on SMP machines to give identical answers on one, two or more processors.
- Local coordinate systems for cross-section output can be specified.
- Null material for shell elements.
- Global body force loads now may be applied to a subset of materials.
- User defined loading subroutine.
- Improved interactive graphics.
- New initial velocity options for specifying rotational velocities.
- Geometry changes after dynamic relaxation can be considered for initial velocities..
- Velocities may also be specified by using material or part ID's.
- Improved speed of brick element hourglass force and energy calculations.
- Pressure outflow boundary conditions have been added for the ALE options.
- More user control for hourglass control constants for shell elements.
- Full vectorization in constitutive models for foam, models 57 and 63.
- Damage mechanics plasticity model, material 81,
- General linear viscoelasticity with 6 term prony series.

- Least squares fit for viscoelastic material constants.
- Table definitions for strain rate effects in material type 24.
- Improved treatment of free flying nodes after element failure.
- Automatic projection of nodes in CONTACT_TIED to eliminate gaps in the surface.
- More user control over contact defaults.
- Improved interpenetration warnings printed in automatic contact.
- Flag for using actual shell thickness in single surface contact logic rather than the default.
- Definition by exempted part ID's.
- Airbag to Airbag venting/segmented airbags are now supported.
- Airbag reference geometry speed improvements by using the reference geometry for the time step size calculation.
- Isotropic airbag material may now be directly for cost efficiency.
- Airbag fabric material damping is specified as the ratio of critical damping.
- Ability to attach jets to the structure so the airbag, jets, and structure to move together.
- PVM 5.1 Madymo coupling is available.
- Meshes are generated within LS-DYNA3D for all standard contact entities.
- Joint damping for translational motion.
- Angular displacements, rates of displacements, damping forces, etc. in JNTFORC file.
- Link between LS-NIKE3D to LS-DYNA3D via *INITIAL_STRESS keywords.
- Trim curves for metal forming springback.
- Sparse equation solver for springback.
- Improved mesh generation for IGES and VDA provides a mesh that can directly be used to model tooling in metal stamping analyses.
- Capabilities added in 1996-1997 in Version 940:
- Part/Material ID's may be specified with 8 digits.
- Rigid body motion can be prescribed in a local system fixed to the rigid body.
- Nonlinear least squares fit available for the Ogden rubber model.
- Least squares fit to the relaxation curves for the viscoelasticity in rubber.
- Fu-Chang rate sensitive foam.
- 6 term Prony series expansion for rate effects in model 57-now 73

- Viscoelastic material model 76 implemented for shell elements.
- Mechanical threshold stress (MTS) plasticity model for rate effects.
- Thermoelastic-plastic material model for Hughes-Liu beam element.
- Ramberg-Osgood soil model
- Invariant local coordinate systems for shell elements are optional.
- Second order accurate stress updates.
- Four noded, linear, tetrahedron element.
- Co-rotational solid element for foam that can invert without stability problems.
- Improved speed in rigid body to rigid body contacts.
- Improved searching for the a_3, a_5 and a10 contact types.
- Invariant results on shared memory parallel machines with the *a_n* contact types.
- Thickness offsets in type 8 and 9 tie break contact algorithms.
- Bucket sort frequency can be controlled by a load curve for airbag applications.
- In automatic contact each part ID in the definition may have unique:
    - Static coefficient of friction
    - Dynamic coefficient of friction
    - Exponential decay coefficient
    - Viscous friction coefficient
    - Optional contact thickness
    - Optional thickness scale factor
    - Local penalty scale factor
- Automatic beam-to-beam, shell edge-to-beam, shell edge-to-shell edge and single surface contact algorithm.
- Release criteria may be a multiple of the shell thickness in types a_3, a_5, a10, 13, and 26 contact.
- Force transducers to obtain reaction forces in automatic contact definitions. Defined manually via segments, or automatically via part ID's.
- Searching depth can be defined as a function of time.
- Bucket sort frequency can be defined as a function of time.
- Interior contact for solid (foam) elements to prevent "negative volumes."
- Locking joint
- Temperature dependent heat capacity added to Wang-Nefske inflator models.
- Wang Hybrid inflator model [Wang, 1996] with jetting options and bag-to-bag venting.

- Aspiration included in Wang's hybrid model [Nusholtz, Wang, Wylie, 1996].
- Extended Wang's hybrid inflator with a quadratic temperature variation for heat capacities [Nusholtz, 1996].
- Fabric porosity added as part of the airbag constitutive model.
- Blockage of vent holes and fabric in contact with structure or itself considered in venting with leakage of gas.
- Option to delay airbag liner with using the reference geometry until the reference area is reached.
- Birth time for the reference geometry.
- Multi-material Euler/ALE fluids,
    - 2nd order accurate formulations.
    - Automatic coupling to shell, brick, or beam elements
    - Coupling using LS-DYNA contact options.
    - Element with fluid + void and void material
    - Element with multi-materials and pressure equilibrium
- Nodal inertia tensors.
- 2D plane stress, plane strain, rigid, and axisymmetric elements
- 2D plane strain shell element
- 2D axisymmetric shell element.
- Full contact support in 2D, tied, sliding only, penalty and constraint techniques.
- Most material types supported for 2D elements.
- Interactive remeshing and graphics options available for 2D.
- Subsystem definitions for energy and momentum output.
- Boundary element method for incompressible fluid dynamics and fluid-structure interaction problems.

Capabilities added during 1997-1998 in Version 950:

- Adaptive refinement can be based on tooling curvature with FORMING contact.
- The display of drawbeads is now possible since the drawbead data is output into the D3PLOT database.
- An adaptive box option, *DEFINE_BOX_ADAPTIVE, allows control over the refinement level and location of elements to be adapted.
- A root identification file, ADAPT.RID, gives the parent element ID for adapted elements.

- Draw bead box option, *DEFINE_BOX_DRAWBEAD, simplifies drawbead input.
- The new control option, CONTROL_IMPLICIT, activates an implicit solution scheme.
- 2D Arbitrary-Lagrangian-Eulerian elements are available.
- 2D automatic contact is defined by listing part ID's.
- 2D r-adaptivity for plane strain and axisymmetric forging simulations is available.
- 2D automatic non-interactive rezoning as in LS-DYNA2D.
- 2D plane strain and axisymmetric element with 2x2 selective-reduced integration are implemented.
- Implicit 2D solid and plane strain elements are available.
- Implicit 2D contact is available.
- The new keyword, *DELETE_CONTACT_2DAUTO, allows the deletion of 2D automatic contact definitions.
- The keyword, *LOAD_BEAM is added for pressure boundary conditions on 2D elements.
- A viscoplastic strain rate option is available for materials:
    - *MAT_PLASTIC_KINEMATIC
    - *MAT_JOHNSON_COOK
    - *MAT_POWER_LAW_PLASTICITY
    - *MAT_STRAIN_RATE_DEPENDENT_PLASTICITY
    - *MAT_PIECEWISE_LINEAR_PLASTICITY
    - *MAT_RATE_SENSITIVE_POWERLAW_PLASTICITY
    - *MAT_ZERILLI-ARMSTRONG
    - *MAT_PLASTICITY_WITH_DAMAGE
    - *MAT_PLASTICITY_COMPRESSION_TENSION
- Material model, *MAT_PLASTICITY_WITH_DAMAGE, has a piecewise linear damage curve given by a load curve ID.
- The Arruda-Boyce hyper-viscoelastic rubber model is available, see *MAT_-ARRUDA_BOYCE.
- Transverse-anisotropic-viscoelastic material for heart tissue, see *MAT_-HEART_-TISSUE.
- Lung hyper-viscoelastic material, see *MAT_LUNG_TISSUE.
- Compression/tension plasticity model, see *MAT_PLASTICITY_COMP-RESSION_TENSION.
- The Lund strain rate model, *MAT_STEINBERG_LUND, is added to Steinberg-Guinan plasticity model.

- Rate sensitive foam model, *MAT_FU_CHANG_FOAM, has been extended to include engineering strain rates, etc.
- Model, *MAT_MODIFIED_PIECEWISE_LINEAR_PLASTICITY, is added for modeling the failure of aluminum.
- Material model, *MAT_SPECIAL_ORTHOTROPIC, added for television shadow mask problems.
- Erosion strain is implemented for material type, *MAT_BAMMAN_DAMAGE.
- The equation of state, *EOS_JWLB, is available for modeling the expansion of explosive gases.
- The reference geometry option is extended for foam and rubber materials and can be used for stress initialization, see *INITIAL_FOAM_REFERENCE_GEOMETRY.
- A vehicle positioning option is available for setting the initial orientation and velocities, see *INITIAL_VEHICLE_KINEMATICS.
- A boundary element method is available for incompressible fluid dynamics problems.
- The thermal materials work with instantaneous coefficients of thermal expansion:
    - *MAT_ELASTIC_PLASTIC_THERMAL
    - *MAT_ORTHOTROPIC_THERMAL
    - *MAT_TEMPERATURE_DEPENDENT_ORTHOTROPIC
    - *MAT_ELASTIC_WITH_VISCOSITY
- Airbag interaction flow rate versus pressure differences.
- Contact segment search option, [bricks first optional]
- A through thickness Gauss integration rule with 1-10 points is available for shell elements. Previously, 5 were available.
- Shell element formulations can be changed in a full deck restart.
- The tied interface which is based on constraint equations, TIED_SURFACE_TO_SURFACE, can now fail if FAILURE, is appended.
- A general failure criterion for solid elements is independent of the material type, see *MAT_ADD_EROSION
- Load curve control can be based on thinning and a flow limit diagram, see *DEFINE_CURVE_FEEDBACK.
- An option to filter the spotweld resultant forces prior to checking for failure has been added the option, *CONSTRAINED_SPOTWELD, by appending FILTERED_FORCE, to the keyword.
- Bulk viscosity is available for shell types 1, 2, 10, and 16.

- When defining the local coordinate system for the rigid body inertia tensor a local coordinate system ID can be used. This simplifies dummy positioning.
- Prescribing displacements, velocities, and accelerations is now possible for rigid body nodes.
- One way flow is optional for segmented airbag interactions.
- Pressure time history input for airbag type, LINEAR_FLUID, can be used.
- An option is available to independently scale system damping by part ID in each of the global directions.
- An option is available to independently scale global system damping in each of the global directions.
- Added option to constrain global DOF along lines parallel with the global axes. The keyword is *CONSTRAINED_GLOBAL. This option is useful for adaptive remeshing.
- Beam end code releases are available, see *ELEMENT_BEAM.
- An initial force can be directly defined for the cable material, *MAT_CABLE_DISCRETE_BEAM. The specification of slack is not required if this option is used.
- Airbag pop pressure can be activated by accelerometers.
- Termination may now be controlled by contact, via *TERMINATION_CONTACT.
- Modified shell elements types 8, 10 and the warping stiffness option in the Belytschko-Tsay shell to ensure orthogonality with rigid body motions in the event that the shell is badly warped. This is optional in the Belytschko-Tsay shell and the type 10 shell.
- A one point quadrature brick element with an exact hourglass stiffness matrix has been implemented for implicit and explicit calculations.
- Automatic file length determination for D3PLOT binary database is now implemented. This insures that at least a single state is contained in each D3PLOT file and eliminates the problem with the states being split between files.
- The dump files, which can be very large, can be placed in another directory by specifying

  d=**/home/user /test/d3dump**

  on the execution line.
- A print flag controls the output of data into the MATSUM and RBDOUT files by part ID's. The option, PRINT, has been added as an option to the *PART keyword.
- Flag has been added to delete material data from the D3THDT file. See *DATABASE_EXTENT_BINARY and column 25 of the 19th control card in the structured input.

- After dynamic relaxation completes, a file is written giving the displaced state which can be used for stress initialization in later runs.

Capabilities added during 1998-2000 in Version 960:

Most new capabilities work on both the MPP and SMP versions; however, the capabilities that are implemented for the SMP version only, which were not considered critical for this release, are flagged below. These SMP unique capabilities are being extended for MPP calculations and will be available in the near future. The implicit capabilities for MPP require the development of a scalable eigenvalue solver, which is under development for a later release of LS-DYNA.

- Incompressible flow solver is available. Structural coupling is not yet implemented.
- Adaptive mesh coarsening can be done before the implicit springback calculation in metal forming applications.
- Two-dimensional adaptivity can be activated in both implicit and explicit calculations. (SMP version only)
- An internally generated smooth load curve for metal forming tool motion can be activated with the keyword: *DEFINE_CURVE_SMOOTH.
- Torsional forces can be carried through the deformable spot welds by using the contact type: *CONTACT_SPOTWELD_WITH_TORSION (SMP version only with a high priority for the MPP version if this option proves to be stable.)
- Tie break automatic contact is now available via the *CONTACT_AUTOMATIC_…_TIEBREAK options. This option can be used for glued panels. (SMP only)
- *CONTACT_RIGID_SURFACE option is now available for modeling road surfaces (SMP version only).
- Fixed rigid walls PLANAR and PLANAR_FINITE are represented in the binary output file by a single shell element.
- Interference fits can be modeled with the INTERFERENCE option in contact.
- A layered shell theory is implemented for several constitutive models including the composite models to more accurately represent the shear stiffness of laminated shells.
- Damage mechanics is available to smooth the post-failure reduction of the resultant forces in the constitutive model *MAT_SPOTWELD_DAMAGE.
- Finite elastic strain isotropic plasticity model is available for solid elements. *MAT_FINITE_ELASTIC_STRAIN_PLASTICITY.
- A shape memory alloy material is available: *MAT_SHAPE_MEMORY.

- Reference geometry for material, *MAT_MODIFIED_HONEYCOMB, can be set at arbitrary relative volumes or when the time step size reaches a limiting value. This option is now available for all element types including the fully integrated solid element.
- Non orthogonal material axes are available in the airbag fabric model. See *MAT_-FABRIC.
- Other new constitutive models include for the beam elements:
    - *MAT_MODIFIED_FORCE_LIMITED
    - *MAT_SEISMIC_BEAM
    - *MAT_CONCRETE_BEAM
- for shell and solid elements:
    - *MAT_ELASTIC_VISCOPLASTIC_THERMAL
- for the shell elements:
    - *MAT_GURSON
    - *MAT_GEPLASTIC_SRATE2000
    - *MAT_ELASTIC_VISCOPLASTIC_THERMAL
    - *MAT_COMPOSITE_LAYUP
    - *MAT_COMPOSITE_LAYUP
    - *MAT_COMPOSITE_DIRECT
- for the solid elements:
    - *MAT_JOHNSON_HOLMQUIST_CERAMICS
    - *MAT_JOHNSON_HOLMQUIST_CONCRETE
    - *MAT_INV_HYPERBOLIC_SIN
    - *MAT_UNIFIED_CREEP
    - *MAT_SOIL_BRICK
    - *MAT_DRUCKER_PRAGER
    - *MAT_RC_SHEAR_WALL
- and for all element options a very fast and efficient version of the Johnson-Cook plasticity model is available:
- *MAT_SIMPLIFIED_JOHNSON_COOK
- A fully integrated version of the type 16 shell element is available for the resultant constitutive models.
- A nonlocal failure theory is implemented for predicting failure in metallic materials. The keyword *MAT_NONLOCAL activates this option for a subset of elasto-plastic constitutive models.

- A discrete Kirchhoff triangular shell element (DKT) for explicit analysis with three in plane integration points is flagged as a type 17 shell element. This element has much better bending behavior than the C0 triangular element.
- A discrete Kirchhoff linear triangular and quadrilateral shell element is available as a type 18 shell. This shell is for extracting normal modes and static analysis.
- A C0 linear 4-node quadrilateral shell element is implemented as element type 20 with drilling stiffness for normal modes and static analysis.
- An assumed strain linear brick element is available for normal modes and statics.
- The fully integrated thick shell element has been extended for use in implicit calculations.
- A fully integrated thick shell element based on an assumed strain formulation is now available. This element uses a full 3D constitutive model which includes the normal stress component and, therefore, does not use the plane stress assumption.
- The 4-node constant strain tetrahedron element has been extended for use in implicit calculations.
- Relative damping between parts is available, see *DAMPING_RELATIVE (SMP only).
- Preload forces are can be input for the discrete beam elements.
- Objective stress updates are implemented for the fully integrated brick shell element.
- Acceleration time histories can be prescribed for rigid bodies.
- Prescribed motion for nodal rigid bodies is now possible.
- Generalized set definitions, i.e., SET_SHELL_GENERAL etc. provide much flexibility in the set definitions.
- The command “sw4.” will write a state into the dynamic relaxation file, D3DRLF, during the dynamic relaxation phase if the D3DRLF file is requested in the input.
- Added mass by PART ID is written into the MATSUM file when mass scaling is used to maintain the time step size, (SMP version only).
- Upon termination due to a large mass increase during a mass scaled calculation a print summary of 20 nodes with the maximum added mass is printed.
- Eigenvalue analysis of models containing rigid bodies is now available using BCSLIB-EXT solvers from Boeing. (SMP version only).
- Second order stress updates can be activated by part ID instead of globally on the *CONTROL_ACCURACY input.
- Interface frictional energy is optionally computed for heat generation and is output into the interface force file (SMP version only).

- The interface force binary database now includes the distance from the contact surface for the FORMING contact options. This distance is given after the nodes are detected as possible contact candidates. (SMP version only).
- Type 14 acoustic brick element is implemented. This element is a fully integrated version of type 8, the acoustic element (SMP version only).
- A flooded surface option for acoustic applications is available (SMP version only).
- Attachment nodes can be defined for rigid bodies. This option is useful for NVH applications.
- CONSTRAINED_POINTS tie any two points together. These points must lie on a shell element.
- Soft constraint is available for edge to edge contact in type 26 contact.
- CONSTAINED_INTERPOLATION option for beam to solid interfaces and for spreading the mass and loads. (SMP version only).
- A database option has been added that allows the output of added mass for shell elements instead of the time step size.
- A new contact option allows the inclusion of all internal shell edges in contact type *CONTACT_GENERAL, type 26. This option is activated by adding "_INTERIOR" after the GENERAL keyword.
- A new option allows the use deviatoric strain rates rather than total rates in material model 24 for the Cowper-Symonds rate model.
- The CADFEM option for ASCII databases is now the default. Their option includes more significant figures in the output files.
- When using deformable spot welds, the added mass for spot welds is now printed for the case where global mass scaling is activated. This output is in the log file, d3hsp file, and the messag file.
- Initial penetration warnings for edge-to-edge contact are now written into the MESSAG file and the d3hsp file.
- Each compilation of LS-DYNA is given a unique version number.
- Finite length discrete beams with various local axes options are now available for material types 66, 67, 68, 93, and 95. In this implementation the absolute value of SCOOR must be set to 2 or 3 in the *SECTION_BEAM input.
- New discrete element constitutive models are available:
    - *MAT_ELASTIC_SPRING_DISCRETE_BEAM
    - *MAT_INELASTIC_SPRING_DISCRETE_BEAM
    - *MAT_ELASTIC_6DOF_SPRING_DISCRETE_BEAM
    - *MAT_INELASTIC_6DOF_SPRING_DISCRETE_BEAM
- The latter two can be used as finite length beams with local coordinate systems.

- Moving SPC's are optional in that the constraints are applied in a local system that rotates with the 3 defining nodes.
- A moving local coordinate system, CID, can be used to determine orientation of discrete beam elements.
- Modal superposition analysis can be performed after an eigenvalue analysis. Stress recovery is based on type 18 shell and brick (SMP only).
- Rayleigh damping input factor is now input as a fraction of critical damping, i.e. 0.10. The old method required the frequency of interest and could be highly unstable for large input values.
- Airbag option "SIMPLE_PRESSURE_VOLUME" allows for the constant CN to be replaced by a load curve for initialization. Also, another load curve can be defined which allows CN to vary as a function of time during dynamic relaxation. After dynamic relaxation CN can be used as a fixed constant or load curve.
- Hybrid inflator model utilizing CHEMKIN and NIST databases is now available. Up to ten gases can be mixed.
- Option to track initial penetrations has been added in the automatic SMP contact types rather than moving the nodes back to the surface. This option has been available in the MPP contact for some time. This input can be defined on the fourth card of the *CONTROL_CONTACT input and on each contact definition on the third optional card in the *CONTACT definitions.
- If the average acceleration flag is active, the average acceleration for rigid body nodes is now written into the D3THDT and NODOUT files. In previous versions of LS-DYNA, the accelerations on rigid nodes were not averaged.
- A capability to initialize the thickness and plastic strain in the crash model is available through the option *INCLUDE_STAMPED_PART, which takes the results from the LS-DYNA stamping simulation and maps the thickness and strain distribution onto the same part with a different mesh pattern.
- A capability to include finite element data from other models is available through the option, *INCLUDE_TRANSFORM. This option will take the model defined in an INCLUDE file: offset all ID's; translate, rotate, and scale the coordinates; and transform the constitutive constants to another set of units.

Features added during 2001-2002 for the 970 release of LS-DYNA:

Some of the new features, which are also listed below, were also added to later releases of version 960. Most new explicit capabilities work for both the MPP and SMP versions; however, the implicit capabilities for MPP require the development of a scalable eigenvalue solver and a parallel implementation of the constraint equations into the global matrices. This work is underway. A later release of version 970 is planned in 2003 that will be scalable for implicit solutions.

Below is list of new capabilities and features:

- MPP decomposition can be controlled using *CONTROL_MPP_DECOMPOSITION commands in the input deck.
- The MPP arbitrary Lagrangian-Eulerian fluid capability now works for airbag deployment in both SMP and MPP calculations.
- Euler-to-Euler coupling is now available through the keyword *CONSTRAINED_EULER_TO_EULER.
- Up to ten ALE multi-material groups may now be defined. The previous limit was three groups.
- Volume fractions can be automatically assigned during initialization of multi-material cells. See the GEOMETRY option of *INITIAL_VOLUME_FRACTION.
- A new ALE smoothing option is available to accurately predict shock fronts.
- DATABASE_FSI activates output of fluid-structure interaction data to ASCII file DBFSI.
- Point sources for airbag inflators are available. The origin and mass flow vector of these inflators are permitted to vary with time.
- A majority of the material models for solid materials are available for calculations using the SPH (Smooth Particle Hydrodynamics) option.
- The Element Free Galerkin method (EFG or meshfree) is available for two-dimensional and three-dimensional solids. This new capability is not yet implemented for MPP applications.
- A binary option for the ASCII files is now available. This option applies to all ASCII files and results in one binary file that contains all the information normally spread between a large number of separate ASCII files.
- Material models can now be defined by numbers rather than long names in the keyword input. For example the keyword *MAT_PIECEWISE_LINEAR_PLASTICITY can be replaced by the keyword: *MAT_024.
- An embedded NASTRAN reader for direct reading of NASTRAN input files is available. This option allows a typical input file for NASTRAN to be read directly and used without additional input. See the *INCLUDE_NASTRAN keyword.
- Names in the keyword input can represent numbers if the *PARAMETER option is used to relate the names and the corresponding numbers.
- Model documentation for the major ASCII output files is now optional. This option allows descriptors to be included within the ASCII files that document the contents of the file.
- ID's have been added to the following keywords:

- *BOUNDARY_PRESCRIBED_MOTION
- *BOUNDARY_PRESCRIBED_SPC
- *CONSTRAINED_GENERALIZED_WELD
- *CONSTRAINED_JOINT
- *CONSTRAINED_NODE_SET
- *CONSTRAINED_RIVET
- *CONSTRAINED_SPOTWELD
- *DATABASE_CROSS_SECTION
- *ELEMENT_MASS

- Penetration warnings for the contact option, ignore initial penetration, î are added as an option. Previously, no penetration warnings were written when this contact option was activated.
- Penetration warnings for nodes in-plane with shell mid-surface are printed for the AUTOMATIC contact options. Previously, these nodes were ignored since it was assumed that they belonged to a tied interface where an offset was not used; consequently, they should not be treated in contact.
- For the arbitrary spot weld option, the spot welded nodes and their contact segments are optionally written into the d3hsp file. See *CONTROL_CONTACT.
- For the arbitrary spot weld option, if a segment cannot be found for the spot welded node, an option now exists to error terminate. See *CONTROL_CONTACT.
- Spot weld resultant forces are written into the SWFORC file for solid elements used as spot welds.
- Solid materials have now been added to the failed element report.
- A new option for terminating a calculation is available, *TERMINATION_CURVE.
- A 10-noded tetrahedron solid element is available with either a 4 or 5 point integration rule. This element can also be used for implicit solutions.
- A new 4 node linear shell element is available that is based on Wilson's plate element combined with a Pian-Sumihara membrane element. This is shell type 21.
- A shear panel element has been added for linear applications. This is shell type 22. This element can also be used for implicit solutions.
- A null beam element for visualization is available. The keyword to define this null beam is *ELEMENT_PLOTEL. This element is necessary for compatibility with NASTRAN.
- A scalar node can be defined for spring-mass systems. The keyword to define this node is *NODE_SCALAR. This node can have from 1 to 6 scalar degrees-of-freedom.

- A thermal shell has been added for through-thickness heat conduction. Internally, 8 additional nodes are created, four above and four below the mid-surface of the shell element. A quadratic temperature field is modeled through the shell thickness. Internally, the thermal shell is a 12 node solid element.
- A beam OFFSET option is available for the *ELEMENT_BEAM definition to permit the beam to be offset from its defining nodal points. This has the advantage that all beam formulations can now be used as shell stiffeners.
- A beam ORIENTATION option for orienting the beams by a vector instead of the third node is available in the *ELEMENT_BEAM definition for NASTRAN compatibility.
- Non-structural mass has been added to beam elements for modeling trim mass and for NASTRAN compatibility.
- An optional checking of shell elements to avoid abnormal terminations is available. See *CONTROL_SHELL. If this option is active, every shell is checked each time step to see if the distortion is so large that the element will invert, which will result in an abnormal termination. If a bad shell is detected, either the shell will be deleted or the calculation will terminate. The latter is controlled by the input.
- An offset option is added to the inertia definition. See *ELEMENT_INERTIA_-OFFSET keyword. This allows the inertia tensor to be offset from the nodal point.
- Plastic strain and thickness initialization is added to the draw bead contact option. See *CONTACT_DRAWBEAD_INITIALIZE.
- Tied contact with offsets based on both constraint equations and beam elements for solid elements and shell elements that have 3 and 6 degrees-of-freedom per node, respectively. See BEAM_OFFSET and CONSTRAINED_OFFSET contact options. These options will not cause problems for rigid body motions.
- The segment-based (SOFT = 2) contact is implemented for MPP calculations. This enables airbags to be easily deployed on the MPP version.
- Improvements are made to segment-based contact for edge-to-edge and sliding conditions, and for contact conditions involving warped segments.
- An improved interior contact has been implemented to handle large shear deformations in the solid elements. A special interior contact algorithm is available for tetrahedron elements.
- Coupling with MADYMO 6.0 uses an extended coupling that allows users to link most MADYMO geometric entities with LS-DYNA FEM simulations. In this coupling MADYMO contact algorithms are used to calculate interface forces between the two models.
- Release flags for degrees-of-freedom for nodal points within nodal rigid bodies are available. This makes the nodal rigid body option nearly compatible with the RBE2 option in NASTRAN.

- Fast updates of rigid bodies for metalforming applications can now be accomplished by ignoring the rotational degrees-of-freedom in the rigid bodies that are typically inactive during sheet metal stamping simulations. See the keyword: *CONTROL_RIGID.
- Center of mass constraints can be imposed on nodal rigid bodies with the SPC option in either a local or a global coordinate system.
- Joint failure based on resultant forces and moments can now be used to simulate the failure of joints.
- CONSTRAINED_JOINT_STIFFNESS now has a TRANSLATIONAL option for the translational and cylindrical joints.
- Joint friction has been added using table look-up so that the frictional moment can now be a function of the resultant translational force.
- The nodal constraint options *CONSTRAINED_INTERPOLATION and *CONSTRAINED_LINEAR now have a local option to allow these constraints to be applied in a local coordinate system.
- Mesh coarsening can now be applied to automotive crash models at the beginning of an analysis to reduce computation times. See the new keyword: *CONTROL_COARSEN.
- Force versus time seatbelt pretensioner option has been added.
- Both static and dynamic coefficients of friction are available for seat belt slip rings. Previously, only one friction constant could be defined.
- *MAT_SPOTWELD now includes a new failure model with rate effects as well as additional failure options.
- Constitutive models added for the discrete beam elements:
    - *MAT_1DOF_GENERALIZED_SPRING
    - *MAT_GENERAL_NONLINEAR_6DOF_DISCRETE_BEAM
    - *MAT_GENERAL_NONLINEAR_1DOF_DISCRETE_BEAM
    - *MAT_GENERAL_SPRING_DISCRETE_BEAM
    - *MAT_GENERAL_JOINT_DISCRETE_BEAM
    - *MAT_SEISMIC_ISOLATOR
- for shell and solid elements:
    - *MAT_PLASTICITY_WITH_DAMAGE_ORTHO
    - *MAT_SIMPLIFIED_JOHNSON_COOK_ORTHOTROPIC_DAMAGE
    - *MAT_HILL_3R
    - *MAT_GURSON_RCDC
- for the solid elements:

- *MAT_SPOTWELD
- *MAT_HILL_FOAM
- *MAT_WOOD
- *MAT_VISCOELASTIC_HILL_FOAM
- *MAT_LOW_DENSITY_SYNTHETIC_FOAM
- *MAT_RATE_SENSITIVE_POLYMER
- *MAT_QUASILINEAR VISCOELASTIC
- *MAT_TRANSVERSELY_ANISOTROPIC_CRUSHABLE_FOAM
- *MAT_VACUUM
- *MAT_MODIFIED_CRUSHABLE_FOAM
- *MAT_PITZER_CRUSHABLE FOAM
- *MAT_JOINTED_ROCK
- *MAT_SIMPLIFIED_RUBBER
- *MAT_FHWA_SOIL
- *MAT_SCHWER_MURRAY_CAP_MODEL

- Failure time added to MAT_EROSION for solid elements.
- Damping in the material models *MAT_LOW_DENSITY_FOAM and *MAT_-LOW_DENSITY_VISCOUS_FOAM can now be a tabulated function of the smallest stretch ratio.
- The material model *MAT_PLASTICITY_WITH_DAMAGE allows the table definitions for strain rate.
- Improvements in the option *INCLUDE_STAMPED_PART now allow all history data to be mapped to the crash part from the stamped part. Also, symmetry planes can be used to allow the use of a single stamping to initialize symmetric parts.
- Extensive improvements in trimming result in much better elements after the trimming is completed. Also, trimming can be defined in either a local or global coordinate system. This is a new option in *DEFINE_CURVE_TRIM.
- An option to move parts close before solving the contact problem is available, see *CONTACT_AUTO_MOVE.
- An option to add or remove discrete beams during a calculation is available with the new keyword: *PART_SENSOR.
- Multiple jetting is now available for the Hybrid and Chemkin airbag inflator models.
- Nearly all constraint types are now handled for implicit solutions.
- Calculation of constraint and attachment modes can be easily done by using the option: *CONTROL_IMPLICIT_MODES.
- Penalty option, see *CONTROL_CONTACT, now applies to all *RIGIDWALL options and is always used when solving implicit problems.

- Solid elements types 3 and 4, the 4 and 8 node elements with 6 degrees-of-freedom per node are available for implicit solutions.
- The warping stiffness option for the Belytschko-Tsay shell is implemented for implicit solutions. The Belytschko-Wong-Chang shell element is now available for implicit applications. The full projection method is implemented due to it accuracy over the drill projection.
- Rigid to deformable switching is implemented for implicit solutions.
- Automatic switching can be used to switch between implicit and explicit calculations. See the keyword: *CONTROL_IMPLICIT_GENERAL.
- Implicit dynamics rigid bodies are now implemented. See the keyword *CONTROL_IMPLICIT_DYNAMIC.
- Eigenvalue solutions can be intermittently calculated during a transient analysis.
- A linear buckling option is implemented. See the new control input: *CONTROL_IMPLICIT_BUCKLE
- Implicit initialization can be used instead of dynamic relaxation. See the keyword *CONTROL_DYNAMIC_RELAXATION where the parameter, IDFLG, is set to 5.
- Superelements, i.e., *ELEMENT_DIRECT_MATRIX_INPUT, are now available for implicit applications.
- There is an extension of the option, *BOUNDARY_CYCLIC, to symmetry planes in the global Cartesian system. Also, automatic sorting of nodes on symmetry planes is now done by LS-DYNA.
- Modeling of wheel-rail contact for railway applications is now available, see *RAIL_TRACK and *RAIL_TRAIN.
- A new, reduced CPU, element formulation is available for vibration studies when elements are aligned with the global coordinate system. See *SECTION_SOLID and *SECTION_SHELL formulation 98.
- An option to provide approximately constant damping over a range of frequencies is implemented, see *DAMPING_FREQUENCY_RANGE.

Features added during 2003-2005 for the 971 release of LS-DYNA:

Initially, the intent was to quickly release version 971 after 970 with the implicit capabilities fully functional for distributed memory processing using MPI. Unfortunately, the effort required for parallel implicit was grossly underestimated, and, as a result, the release has been delayed. Because of the delay, version 971 has turned into a major release. Some of the new features, listed below, were also added to later releases of version 970. The new explicit capabilities are implemented in the MPP version and except for one case, in the SMP version as well.

Below is list of new capabilities and features:

- A simplified method for using the ALE capability with airbags is now available with the keyword *AIRBAG_ALE.
- Case control using the *CASE keyword, which provides a way of running multiple load cases sequentially within a single run
- New option to forming contact: *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_SMOOTH, which use fitted surface in contact calculation.
- Butt weld definition by using the *CONSTRAINED_BUTT_WELD option which makes the definition of butt welds simple relative to the option: *CONSTRAINED_GENERALIZED_WELD_BUTT.
- H-adaptive fusion is now possible as an option with the control input, *CONTROL_ADAPTIVE.
- Added a parameter on, *CONTROL_ADAPTIVE, to specify the number of elements generated around a 90 degree radius. A new option to better calculate the curvature was also implemented.
- Added a new keyword: *CONTROL_ADAPTIVE_CURVE, to refine the element along trimming curves
- Birth and death times for implicit dynamics on the keyword *CONTROL_IMPLICIT_DYNAMICS.
- Added an option to scale the spot weld failure resultants to account for the location of the weld on the segment surface, see *CONTROL_SPOTWELD_BEAM.
- Added an option which automatically replaces a single beam spot weld by an assembly of solid elements using the same ID as the beam that was replaced, see *CONTROL_SPOTWELD_BEAM.
- Boundary constraint in a local coordinate system using *CONSTRAINED_LOCAL keyword.
- A cubic spline interpolation element is now available, *CONSTRAINED_SPLINE.
- Static implicit analyses in of a structure with rigid body modes is possible using the option, *CONTROL_IMPLICIT_INERTIA_RELIEF.
- Shell element thickness updates can now be limited to part ID's within a specified set ID, see the *CONTROL_SHELL keyword. The thickness update for shells can now be optionally limited to the plastic part of the strain tensor for better stability in crash analysis.
- Solid element stresses in spot welds are optionally output in the local system using the SWLOCL parameter on the *CONTROL_SOLID keyword.
- SPOTHIN option on the *CONTROL_CONTACT keyword cards locally thins the spot welded parts to prevent premature breakage of the weld by the contact treatments.

- New function: *CONTROL_FORMING_PROJECT, which can initial move the penetrating slave nodes to the master surface
- New function *CONTROL_FORMING_TEMPLATE, which allows user to easily set up input deck. Its function includes auto-position, define travel curve, termination time, and most of the forming parameters for most of the typical forming process.
- New function *CONTROL_FORMING_USER, *CONTROL_FORMING_POSITION, and *CONTROL_FORMING_TRAVEL, when used together, can allow the user to define atypical forming process.
- Added new contact type *CONTACT_GUIDED_CABLE.
- Circular cut planes are available for *DATABASE_CROSS_SECTION definitions.
- New binary database FSIFOR for fluid structure coupling.
- Added *DATABASE_BINARY_D3PROP for writing the material and property data to the first D3PLOT file or to a new database D3PROP.
- DATABASE_EXTENT_BINARY has new flags to output peak pressure, surface energy density, nodal mass increase from mass scaling, thermal fluxes, and temperatures at the outer surfaces of the thermal shell.
- Eight-character alphanumeric labels can now be used for the parameters SECID, MID, EOSID, HGID, and TMID on the *PART keyword.
- Two NODOUT files are now written: one for high frequency output and a second for low frequency output.
- Nodal mass scaling information can now be optionally written to the D3PLOT file.
- Added option, MASS_PROPERTIES, to include the mass and inertial properties in the GLSTAT and SSSTAT files.
- Added option in *CONTROL_CPU to output the cpu and elapsed time into the GLSTAT file.
- Added an option, IERODE, on the *CONTROL_OUTPUT keyword to include eroded energies by part ID into the MATSUM file. Lumped mass kinetic energy is also in the MATSUM file as part ID 0.
- Added an option, TET10, on the *CONTROL_OUTPUT keyword to output ten connectivity nodes into D3PLOT database rather than 4.
- New keyword, *ELEMENT_SOLID_T4TOT10 to convert 4 node tetrahedron elements to 10 node tetrahedron elements.
- New keyword, *ELEMENT_MASS_PART defines the total additional non-structural mass to be distributed by an area weighted distribution to all nodes of a given part ID.
- New keyword option, SET, for *INTIAL_STRESS_SHELL_SET allows a set of shells to be initialized with the state of stress.

- New option allows the number of cpu's to be specified on the *KEYWORD input.
- Tubular drawbead box option for defining the elements that are included in the drawbead contact, see *DEFINE_BOX_DRAWBEAD.
- New function: *DEFINE_CURVE_DRAWBEAD, allow user to conveniently define drawbead by using curves (in x, y format or iges format)
- New function: *DEFINE_DRAWBEAD_BEAM, which allows user to conveniently define drawbead by using beam part ID, and specify the drawbead force.
- Analytic function can be used in place of load curves with the option *DEFINE_CURVE_FUNCTION.
- Friction can now be defined between part pair using the *DEFINE_FRICTION input.
- New keyword: *DEFINE_CURVE_TRIM_3D, to allow trimming happens based on blank element normal, rather than use pre-defined direction
- A new trimming algorithm was added: *DEFINE_CURVE_TRIM_NEW, which allow seed node to be input and is much faster then the original algorithm.
- A new keyword, *DEFINE_HEX_SPOTWELD_ASSEMBLY, is available to define a cluster of solid elements that comprise a single spot weld.
- The definition of a vector, see *DEFINE_VECTOR, can be done by defining coordinates in a local coordinate system.
- The definition of a failure criteria between part pairs is possible with a table defined using the keyword, *DEFINE_SPOTWELD_FAILURE_RESULTANTS.
- A new keyword, *DEFINE_CONNECTION_PROPERTIES is available for defining failure properties of spot welds.
- Added *DEFINE_SET_ADAPTIVE to allow the adaptive level and element size to be specified by part ID or element set ID.
- Static rupture stresses for beam type spot welds can be defined in the keyword input, *DEFINE_SPOTWELD_RUPTURE_STRESS.
- Section properties can be define in the *ELEMENT_BEAM definitions for resultant beam elements using the SECTION option.
- Physical offsets of the shell reference surface can be specified on the shell element cards, see the OFFSET option on *ELEMENT_SHELL.
- File names can be located in remote directories and accessed through the *INCLUDE_PART keyword.
- New features to *INCLUDE_STAMPED_PART: two different mirror options, user-defined searching radius.
- *INTIAL_STRESS_SECTION allows for stress initialization across a cross-section, which consists of solid elements.

- An option, IVATN, is available for setting the velocities of slaved nodes and parts for keyword, *INITIAL_VELOCITY_GENERATION.
- Twenty-two built-in cross-section are now available in the definition of beam integration rules, see *INTEGRATION_BEAM.
- The possibility of changing material types is now available for shells using the user defined integration rule, see *INTEGRATION_SHELL.
- The interface springback file created by using the keyword, *INTERFACE_SPRINGBACK is now optionally written as a binary file.
- An optional input line for *KEYWORD allows the definition of a prefix for all file names created during a simulation. This allows multiple jobs to be executed in the same directory.
- Body force loads can now be applied in a local coordinate system for *LOAD_BODY.
- A pressure loading feature allows moving pressures to be applied to a surface to simulate spraying a surface with stream of fluid through a nozzle. See keyword *LOAD_MOVING_PRESSURE.
- Thermal expansion can be added to any material by the keyword, *MAT_ADD_THERMAL_EXPANSION.
- Curves can now be used instead of eight digitized data points in the material model *MAT_ELASTIC_WITH_VISCOSITY_CURVE
- New options for spot weld failure in *MAT_SPOTWELD, which apply to beam and solid elements.
- Failure criteria based on plastic strain to failure is added to material *MAT_ANISOTROPIC_VISCOPLASTIC.
- Strain rate failure criterion is added to material *MAT_MODIFIED_PIECEWISE_LINEAR_PLASTICITY.
- Strain rate scaling of the yield stress can now be done differently in tension and compression in material with separate pressure cut-offs in tension and compression in material model *MAT_PLASTICITY_TENSION_COMPRESSION.
- The RCDC model is now available to predict failure in material *MAT_PLASTICITY_WITH_DAMAGE.
- Two additional yield surfaces have been added to material *MAT_MODIFIED_HONEYCOMB to provide more accurate predictions of the behavior of honeycomb barrier models.
- Unique coordinate systems can be assigned to the two nodal points of material *MAT_1DOF_GENERALIZED_SPRING.
- Poisson's ratio effects are available in foam defined by load curves in the material *MAT_SIMPLIFIED_RUBBER/FOAM

- Failure effects are available in the rubber/foam material defined by load curves in the *MAT_SIMPLIFIED_RUBBER/FOAM_WITH_FAILURE.
- The material option *MAT_ADD_EROSION now allows the maximum pressure at failure and the minimum principal strain at failure to be specified.
- Strains rather than displacements can now be used with the material model for discrete beams, *MAT_GENERAL_NONLINEAR_6DOF_DISCRETE_BEAM.
- New option for *MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC_(ECHANGE), which allow two ways to change the Young's modulus during forming simulation.
- New Material model: *MAT_HILL_3R: includes the shear term in the yield surface calculation by using Hill's 1948 an-isotropic material model.
- New Material model: *MAT_KINEMATIC_HARDENING_TRANSVERSELY_ANISOTROPIC: which integrates Mat #37 with Yoshida's two-surface kinematic hardening model.
- Improved formulation for the fabric material, *MAT_FABRIC for formulations 2, 3, and 4. The improved formulations are types 12, 13, and 14.
- Constitutive models added for truss elements:
    - *MAT_MUSCLE
- For beam elements
    - *MAT_MOMENT_CURVATURE
- For shell elements
    - *MAT_RESULTANT_ANISOTROPIC
    - *MAT_RATE_SENSITIVE_COMPOSITE_FABRIC.
    - *MAT_SAMP-1
    - *MAT_SHAPE_MEMORY is now implemented for shells.
- for shell and solid elements:
    - *MAT_BARLAT_YLD2000 for anisotropic aluminum alloys.
    - *MAT_SIMPLIFIED_RUBBER_WITH_DAMAGE
    - *MAT_VISCOELASTIC_THERMAL
    - *MAT_THERMO_ELASTO_VISCOPLASTIC_CREEP
- for the solid elements:
    - *MAT_ARUP_ADHESIVE
    - *MAT_BRAIN_LINEAR_VISCOELASTIC.
    - *MAT_CSCM for modeling concrete.

  - *MAT_PLASTICITY_COMPRESSION_TENSION_EOS for modeling ice.
  - *MAT_COHESIVE_ELASTIC
  - *MAT_COHESIVE_TH
  - *MAT_COHESIVE_GENERAL
  - *MAT_EOS_GASKET
  - *MAT_SIMPLIFIED_JOHNSON_COOK is now implemented for solids.
  - *MAT_PLASTICITY_WITH_DAMAGE is now implemented for solids.
  - *MAT_SPOTWELD_DAIMLERCHRYSLER

- User defined equations-of-state are now available.
- There is now an interface with the MOLDFLOW code.
- Damping defined in *DAMPING_PART_STIFFNESS now works for the Belytschko –Schwer beam element.
- The option *NODE_TRANSFORMATION allows a node set to be transformed based on a transformation defined in *DEFINE_TRANSFORMATION.
- Parameters can be defined in FORTRAN like expressions using *PARAMETER_EXPRESSION.
- A part can be moved in a local coordinate system in *PART_MOVE.
- A simplified method for defining composite layups is available with *PART_COMPOSITE
- The rigid body inertia can be changed in restart via *CHANGE_RIGID_BODY_INERTIA.
- A part set can now be defined by combining other part sets in *SET_PART_ADD.
- Termination of the calculation is now possible if a specified number of shell elements are deleted in a give part ID. See *TERMINATION_DELETED_SHELLS.
- Added hourglass control type 7 for solid elements for use when modeling hyperelastic materials.
- Shell formulations 4, 11, 16, and 17 can now model rubber materials.
- Added a new seatbelt pretensioner type 7 in which the pretensioner and retractor forces are calculated independently and added.
- A new composite tetrahedron element made up from 12 tetrahedron is now available as solid element type 17.
- Shell thickness offsets for *SECTION_SHELL now works for most shell elements, not just the Hughes-Liu shell.
- The Hughes-Liu beam has been extended to include warpage for open cross-sections.
- A resultant beam formulation with warpage is available as beam type 12.

- Two nonlinear shell elements are available with 8 degrees-of-freedom per node to include thickness stretch.
- Tetrahedron type 13, which uses nodal pressures, is now implemented for implicit applications.
- Cohesive solid elements are now available for treating failure.
- Seatbelt shell elements are available for use with the all seatbelt capabilities.
- Superelements can now share degrees-of-freedom and are implemented for implicit applications under MPI.
- A user defined element interface is available for solid and shell elements.
- Thermal shells are available for treating heat flow through shell elements.
- EFG shell formulations 41 and 42 are implemented for explicit analysis.
- EFGPACK is implemented in addition to BCSLIB-EXT solver on the keyword *CONTROL_EFG.
- EFG MPP version is available for explicit analysis.
- EFG fast transformation method is implemented in the EFG solid formulation.
- EFG Semi-Lagrangian kernel and Eulerian kernel options are added for the foam materials.
- EFG 3D adaptivity is implemented for the metal materials.
- EFG E.O.S. and *MAT_ELASTIC_FLUID materials are included in the 4-noded background element formulation.
- Airbag simulations by using ALE method can be switched to control volume method by *ALE_CV_SWITCH.
- *MAT_ALE_VISCOUS now supports Non-Newtonian viscosity by power law or load curve.
- *DATABASE_BINARY_FSIFOR outputs fluid-structure interaction data to binary file.
- *DATABASE_FSI_SENSOR outputs ALE element pressure to ASCII file dbsor.
- *MAT_GAS_MIXTURE supports nonlinear heat capacities.
- *INITIAL_VOLUME_FRACTION_GEOMETRY uses an enhanced algorithm to handle both concave and convex geometries and substantially reduce run time.
- A new keyword *DELETE_FSI allows the deletion of coupling definitions.
- Convection heat transfer activates by *LOAD_ALE_CONVECTION in ALE FSI analysis.
- *ALE_FSI_SWITCH_MMG is implemented to switch between ALE multi-material groups to treat immersed FSI problems.

- Type 9 option is added in *ALE_REFERENCE_SYSTEM_GROUP to deal complex ALE mesh motions including translation, rotation, expansion and contraction, etc.

    - New options in *CONSTRAINED_LAGRANGE_IN_SOLID
    - Shell thickness option for coupling type 4.
    - Bulk modulus based coupling stiffness.
    - Shell erosion treatment.
    - Enable/disable interface force file.

- New coupling method for fluid flowing through porous media are implemented as type 11 (shell) and type 12 (solid) in *CONSTRAINED_LAGRANGE_IN_SOLID.
- *ALE_MODIFIED_STRAIN allows multiple strain fields in certain ALE elements to solve sticking behavior in FSI. (MPP underdevelopment)
- *ALE_FSI_PROJECTION is added as a new constraint coupling method to solve small pressure variation problem. (MPP underdevelopment)
- *BOUNDARY_PRESCRIBED_ORIENTATION_RIGID is added as a means to prescribe as a function of time the general orientation of a rigid body using a variety of methods. This feature is available in release R3 and higher of Version 971.
- *BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID is added as a means to prescribe the motion of a rigid body based un experimental data gathered from accelerometers affixed to the rigid body. This feature is available in release R3 and higher of Version 971.

Capabilities added during 2008-2011 for Version 971R6 of LS-DYNA:

During the last four years the implicit capabilities are now scalable to a large number of cores; therefore, LS-DYNA has achieved a major goal over 15 years of embedding a scalable implicit solver. Also, in addition to the progress made for implicit solutions many other new and useful capabilities are now available.

- The keyword *ALE_AMBIENT_HYDROSTATIC initializes the hydrostatic pressure field in the ambient ALE domain due to an acceleration like gravity.
- The keyword *ALE_FAIL_SWITCH_MMG allows switching an ALE multi-material-group ID (AMMGID) if the material failure criteria occurs.
- The keyword *ALE_FRAGMENTATION allow switching from the ALE multi-material-group ID, AMMGID, (FR_MMG) of this failed material to another AMMGID (TO_MMG). This feature may typically be used in simulating fragmentation of materials.
- The keyword *ALE_REFINE refines ALE hexahedral solid elements automatically.

- The keyword *BOUNDARY_ALE_MAPPING maps ALE data histories from a previous run to a region of elements. Data are read from or written to a mapping file with a file name given by the prompt "map=" on the command line starting the execution.
- The keyword *BOUNDARY_PORE_FLUID is used to define parts that contain pore fluid where defaults are given on *CONTROL_PORE_FLUID input.
- With the keyword, *BOUNDARY_PRESCRIBED_FINAL_GEOMETRY, the final displaced geometry for a subset of nodal points is defined. The nodes of this subset are displaced from their initial positions specified in the *NODE input to the final geometry along a straight line trajectory. A load curve defines a scale factor as a function of time that is bounded between zero and unity corresponding to the initial and final geometry, respectively. A unique load curve can be specified for each node, or a default load curve can apply to all nodes.
- The keyword, *BOUNDARY_PWP defines pressure boundary conditions for pore water at the surface of the software.
- The keyword, *CONSTRAINED_JOINT_COOR, defines a joint between two rigid bodies. The connection coordinates are given instead of the nodal point IDs used in *CONSTRAINED_JOINT.
- The keyword, *CONSTRAINED_SPR2, defines a self-piercing rivet with failure. This model for a self-piercing rivet (SPR2) includes a plastic-like damage model that reduces the force and moment resultants to zero as the rivet fails. The domain of influence is specified by a diameter, which should be approximately equal to the rivet's diameter. The location of the rivet is defined by a single node at the center of two riveted sheets.
- Through the keyword, *CONTROL_BULK_VISCOSITY, bulk viscosity is optional for the Hughes-Liu beam and beam type 11 with warpage. This option often provides better stability, especially in elastic response problems.
- The display of nodal rigid bodies is activated by the parameter, PLOTEL, on the *CONTROL_RIGID keyword.
- The mortar contact, invoked by appending the suffix MORTAR to either FORMING_SURFACE_TO_SURFACE, AUTOMATIC_SURFACE_TO_SURFACE or AUTOMATIC_SINGLE_SURFACE, is a segment to segment penalty based contact. For two segments on each side of the contact interface that are overlapping and penetrating, a consistent nodal force assembly taking into account the individual shape functions of the segments is performed. In this respect the results with this contact may be more accurate, especially when considering contact with elements of higher order. By appending the suffix TIED to the CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR keyword or the suffix MORTAR to the CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK keyword, this is treated as a tied contact interface with tiebreak failure in the latter case. Only OPTION = 9 is supported for the mortar tiebreak contact. The mortar

contact is intended for implicit analysis in particular but is nevertheless supported for explicit analysis as well.

- In the database, ELOUT, the number of history variables can be specified for output each integration point in the solid, shell, thick shell, and beam elements. The number of variables is given on the *DATABASE_ELOUT keyword definition.
- A new option is available in *DATABASE_EXTENT_BINARY. Until now only one set of integration points were output through the shell thickness. The lamina stresses and history variables were averaged for fully integrated shell elements, which results in less disk space for the D3PLOT family of files, but makes it difficult to verify the accuracy of the stress calculation after averaging. An option is now available to output all integration point stresses in fully integrated shell elements: 4 x # of through thickness integration points in shell types 6, 7, 16, 18-21, and 3 x # of through thickness integration points in triangular shell types 3, and 17.
- The keyword *DATABASE_PROFILE allows plotting the distribution or profile of data along x, y, or z-direction.
- The purpose of the keyword, *DEFINE_ADAPTIVE_SOLID_TO_SPH, is to adaptively transform a Lagrangian solid Part or Part Set to SPH particles when the Lagrange solid elements comprising those parts fail. One or more SPH particles (elements) will be generated for each failed element to. The SPH particles replacing the failed element inherit all of the properties of failed solid element, e.g. mass, kinematic variables, and constitutive properties.
- With the keywords beginning with, *DEFINE_BOX, a LOCAL option is now available. With this option the diagonal corner coordinates are given in a local coordinate system defined by an origin and vector pair.
- The keyword, *DEFINE_CURVE_DUPLICATE, defines a curve by optionally scaling and offsetting the abscissa and ordinates of another curve defined by the *DEFINE_CURVE keyword.
- The keyword, *DEFINE_ELEMENT_DEATH, is available to delete a single element or an element set at a specified time during the calculation.
- The purpose of the keyword, *DEFINE_FRICTION_ORIENTATION, is to allow for the definition of different coefficients of friction (COF) in specific directions, specified using a vector and angles in degrees. In addition, COF can be scaled according to the amount of pressure generated in the contact interface.
- With the new keyword, *DEFINE_FUNCTION, an arithmetic expression involving a combination of independent variables and other functions, i.e.,

  f(a,b,c) = a*2 + b*c + sqrt(a*c)

  is defined where a, b, and c are the independent variables. This option is implemented for a subset of keywords.

- *ELEMENT_SEATBELT_SLIPRING
- *LOAD_BEAM
- *LOAD_MOTION_NODE
- *LOAD_MOVING_PRESSURE
- *LOAD_NODE
- *LOAD_SEGMENT
- *LOAD_SEGMENT_NONUNIFORM
- *LOAD_SETMENT_SET_NONUNIFORM
- *BOUNDARY_PRESCRIBED_MOTION

- If a curve ID is not found, then the function ID's are checked.
- The keyword, *DEFINE_SPH_TO_SPH_COUPLING, defines a penalty based contact to be used for the node to node contacts between SPH parts.
- The keyword, *DEFINE_TABLE_2D, permits the same curve ID to be referenced by multiple tables, and the curves may be defined anywhere in the input.
- The keyword, *DEFINE_TABLE_3D, provides a way of defining a three-dimensional table. A 2D table ID is specified for each abscissa value defined for the 3D table.
- The keyword, *ELEMENT_BEAM_PULLEY, allows the definition of a pulley for truss beam elements (see *SECTION_BEAM, ELFORM = 3). Currently, the beam pulley is implemented for *MAT_001 and *MAT_156. Pulleys allow continuous sliding of a string of truss beam element through a sharp change of angle.
- The purpose of the keyword, *ELEMENT_MASS_MATRIX, is to define a 6x6 symmetric nodal mass matrix assigned to a nodal point or each node within a node set.
- The keyword, *ELEMENT_DISCRETE_SPHERE, allows the definition of a discrete spherical element for discrete element calculations. Each particle consists of a single node with its mass, mass moment of inertia, and radius. Initial coordinates and velocities are specified via the nodal data.
- The two keywords, *ELEMENT_SHELL_COMPOSITE and *ELEMENT_TSHELL_COMPOSITE, are used to define elements for a general composite shell part where the shells within the part can have an arbitrary number of layers. The material ID, thickness, and material angle are specified for the thickness integration points for each shell in the part
- The keyword, *EOS_USER_DEFINED, allows a user to supply their own equation-of-state subroutine.
- The new keyword *FREQUENCY_DOMAIN provides a way of defining and solving frequency domain vibration and acoustic problems. The related keyword cards given in alphabetical order are:

  - *FREQUENCY_DOMAIN_ACOUSTIC_BEM_{*OPTION*}

- *FREQUENCY_DOMAIN_ACOUSTIC_FEM
- *FREQUENCY_DOMAIN_FRF
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION
- *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM
- *FREQUENCY_DOMAIN_SSD

- The keyword, *INITIAL_AIRBAG_PARTICLE, initializes pressure in a closed airbag volume, door cavities for pressure sensing studies, and tires.
- The keyword *INITIAL_ALE_HYDROSTATIC initializes the hydrostatic pressure field in an ALE domain due to an acceleration like gravity.
- The keyword *INITIAL_ALE_MAPPING maps ALE data histories from a previous run. Data are read from a mapping file with a file name given by the prompt "map=" on the command line starting the execution.
- The keyword, *INITIAL_AXIAL_FORCE_BEAM, provides a simplified method to model initial tensile forces in bolts.
- The keyword, *INITIAL_FIELD_SOLID, is a simplified version of the *INITIAL_STRESS_SOLID keyword which can be used with hyperelastic materials. This keyword is used for history variable input. Data is usually in the form of the eigenvalues of diffusion tensor data. These are expressed in the global coordinate system.
- The equation-of-state, *EOS_MIE_GRUNEISEN, type 16, is a Mie-Gruneisen form with a p-α compaction model.
- The keyword, *LOAD_BLAST_ENHANCED, defines an air blast function for the application of pressure loads due the explosion of conventional charge. While similar to *LOAD_BLAST this feature includes enhancements for treating reflected waves, moving warheads and multiple blast sources. The loads are applied to facets defined with the keyword *LOAD_BLAST_SEGMENT. A database containing blast pressure history is also available (see *DATABASE_BINARY_BLSTFOR).
- The keyword, *LOAD_ERODING_PART_SET, creates pressure loads on the exposed surface composed of solid elements that erode, i.e., pressure loads are added to newly exposed surface segments as solid elements erode.
- The keyword, *LOAD_SEGMENT_SET_ANGLE, applies traction loads over a segment set that is dependent on the orientation of a vector. An example application is applying a pressure to a cylinder as a function of the crank angle in an automobile engine
- The keyword, *LOAD_STEADY_STATE_ROLLING, is a generalization of *LOAD_BODY, allowing the user to apply body loads to part sets due to translational and rotational accelerations in a manner that is more general than the *LOAD_BODY capability. The *LOAD_STEADY_STATE_ROLLING keyword may be invoked an arbitrary number of times in the problem as long as no part

has the option applied more than once and they can be applied to arbitrary meshes. This option is frequently used to initialize stresses in tire.

- The keywords INTERFACE_SSI, INTERFACE_SSI_AUX, INTERFACE_SSI_-AUX_EMBEDDED and INTERFACE_SSI_STATIC are used to define the soil-structure interface appropriately in various stages of soil-structure interaction analysis under earthquake ground motion.
- The keyword, *LOAD_SEISMIC_SSI, is used to apply earthquake loads due to free-field earthquake ground motion at certain locations — defined by either nodes or coordinates — on a soil-structure interface. This loading is used in earthquake soil-structure interaction analysis. The specified motions are used to compute a set of effective forces in the soil elements adjacent to the soil-structure interface, according to the effective seismic input–domain reduction method.
- The keyword *DEFINE_GROUND_MOTION is used to specify a ground motion to be used in conjunction with *LOAD_SEISMIC_SSI.
- Material types *MAT_005 and *MAT_057 now accept table input to allow the stress quantity versus the strain measure to be defined as a function of temperature.
- The material option *MAT_ADD_EROSION, can now be applied to all nonlinear shell, thick shell, fully integrated solids, and 2D solids. New failure criteria are available.
- The GISSMO damage model, now available as an option in *MAT_ADD_EROSION, is a phenomenological formulation that allows for an incremental description of damage accumulation, including softening and failure. It is intended to provide a maximum in variability for the description of damage for a variety of metallic materials (e.g. *MAT_024, *MAT_036, …). The input of parameters is based on tabulated data, allowing the user to directly convert test data to numerical input.
- The keyword, *MAT_RIGID_DISCRETE or MAT_220, eliminates the need to define a unique rigid body for each particle when modeling a large number of rigid particles. This gives a large reduction in memory and wall clock time over separate rigid bodies. A single rigid material is defined which contains multiple disjoint pieces. Input is simple and unchanged, since all disjoint rigid pieces are identified automatically during initialization.
- The keyword, *NODE_MERGE, causes nodes with identical coordinates to be replaced during the input phase by the node encountered that has the smallest ID.
- The keyword, *PART_ANNEAL, is used to initialize the stress states at integration points within a specified part to zero at a given time during the calculation. This option is valid for parts that use constitutive models where the stress is incrementally updated. This option also applies to the Hughes-Liu beam elements, the integrated shell elements, thick shell elements, and solid elements.

- The keyword, *PART_DUPLICATE, provides a method of duplicating parts or part sets without the need to use the *INCLUDE_TRANSFORM option.
- To automatically generate elements to visualize rigid walls the DISPLAY option is now available for *RIGIDWALL_PLANAR and *RIGIDWALL_GEOMETRIC.
- A one point integrated pentahedron solid element with hourglass control is implemented as element type 115 and can be referenced in *SECTION_SOLID. Also, the 2 point pentahedron solid, type 15, no longer has a singular mode.
- The keyword *SECTION_ALE1D defines section properties for 1D ALE elements.
- The keyword *SECTION_ALE2D defines section properties for 2D ALE elements.
- The keywords *SET_BEAM_INTERSECT, *SET_SHELL_INTERSECT, *SET_SOLID_INTERSECT, *SET_NODE_INTERSECT, and *SET_SEGMENT_INTER-SECT, allows the definition of a set as the intersection, ∩, of a series of sets. The new set, SID, contains all common members.
- The keyword, *SET_SEGMENT_ADD, is now available for defining a new segment set by combining other segment sets.
- The two keywords, *DEFINE_ELEMENT_GENERALIZED_SHELL and *DEFINE_ELEMENT_GENERALIZED_SOLID, are used to define general shell and solid element formulations to allow the rapid prototyping of new element formulations. They are used in combination with the new keywords *ELEMENT_GENERLIZED_SHELL and *ELEMENT_GENERALIZED_SOLID.
- The two keywords, *ELEMENT_INTERPOLATION_SHELL and *ELEMENT_INTERPOLATION_SOLID, are used to interpolate stresses and other solution variables from the generalized shell and solid element formulations for visualization. They are used together with the new keyword *CONSTRAINED_NODE_INTERPOLATION.
- The keyword, *ELEMENT_SHELL_NURBS_PATCH, is used to define 3D shell elements based on NURBS (Non-Uniform Ration B-Spline) basis functions. Currently four different element formulations, with and without rotational degrees of freedom are available.
- The keyword LOAD_SPCFORC is used to apply equivalent SPC loads, read in from the d3dump file during a full-deck restart, in place of the original constraints in order to facilitate the classical non-reflecting boundary on an outside surface.

Capabilities added in 2012 to create Version 971R6.1, of LS-DYNA:

- A new keyword *MAT_THERMAL_DISCRETE_BEAM defines thermal properties for ELFORM 6 beam elements.
- An option *CONTROL_THERMAL_SOLVER, invoked by TSF < 0, gives the thermal speedup factor via a curve. This feature is useful when artificially scaling velocity in metal forming.

- A nonlinear form of Darcy's law in *MAT_ADD_PORE_AIR allows curves to define the relationship between pore air flow velocity and pore air pressure gradient.
- An extension to the PART option in *SET_SEGMENT_GENERAL allows reference to a beam part. This allows for creation of 2D segments for traction application.
- Options "SET_SHELL", "SET_SOLID", "SET_BEAM", "SET_TSHELL", "SET_SPRING" are added to *SET_NODE_GENERAL so users can define a node set using existing element sets.
- Options "SET_SHELL", "SET_SOLID", "SET_SLDIO", "SET_TSHELL", "SET_TSHIO" are added to *SET_SEGMENT_GENERAL so users can use existing element sets to define a segment set.
- *BOUNDARY_PRESCRIBED_MOTION_SET_BOX prescribes motion to nodes that fall inside a defined box.
- IPNINT > 1 in *CONTROL_OUTPUT causes d3hsp to list the IPNINT smallest element timesteps in ascending order.
- Section and material titles are echoed to d3hsp.
- A new parameter, MOARFL, in *DEFINE_CONNECTION_PROPERTIES permits reduction in modeled area due to shear.
- A new option HALF_SPACE in *FREQUENCY_DOMAIN_ACOUSTIC_BEM enables treatment of a half-space in boundary element method, frequency domain acoustic analysis.
- A shell script "kill_by_pid" is created during MPP startup. When executed, this script will run "kill -9" on every LS-DYNA process started as part of the MPP job. This is for use at the end of submission scripts, as a "fail safe" cleanup in case the job aborts.
- A new parameter IAVIS in *CONTROL_SPH selects the artificial viscosity formulation for the SPH particles. If set to 0, the Monaghan type artificial viscosity formulation is used. If set to 1, the standard artificial viscosity formulation for solid elements is used which may provide a better energy balance but is less stable in specific applications such as high velocity impact.
- Contact friction may be included in *CONTACT_2D_NODE_TO_SOLID for SPH.
- A new keyword *ALE_COUPLING_NODAL_CONSTRAINT provides a coupling mechanism between ALE solids and non-ALE nodes. The nodes can be from virtually any non-ALE element type including DISCRETE_SPHERE, EFG, and SPH, as well as the standard Lagrangian element types. In many cases, this coupling type may be a better alternative to *CONSTRAINED_LAGRANGE_IN_SOLID.
- The keyword *ALE_ESSENTIAL_BOUNDARY assigns essential boundary conditions to nodes of the ALE boundary surface. The command can be repeated multiple times and is recommended over use of EBC in *CONTROL_ALE.

- The keyword *DELETE_ALECPL in a small restart deck deletes coupling defined with *ALE_COUPLING_NODAL_CONSTRAINT. The command can also be used to reinstate the coupling in a later restart.
- *DEFINE_VECTOR_NODES defines a vector with two node points.
- *CONTACT_AUTOMATIC_SINGLE_SURFACE_TIED allows for the calculation of eigenvalues and eigenvectors for models that include *CONTACT_AUTOMATIC_SINGLE_SURFACE.
- A new parameter RBSMS in *CONTROL_RIGID affects rigid body treatment in Selective Mass Scaling (*CONTROL_TIMESTEP). When rigid bodies are in any manner connected to deformable elements, RBSMS = 0 (default) results in spurious inertia due to improper treatment of the nodes at the interface. RBSMS = 1 alleviates this effect but an additional cost is incurred.
- A new parameter T10JTOL in *CONTROL_SOLID sets a tolerance for issuing a warning when J_min/J_max goes below this tolerance value (i.e., quotient between minimum and maximum Jacobian value in the integration points) for tetrahedron type 16. This quotient serves as an indicator of poor tetrahedral element meshes in implicit that might cause convergence problems.
- A new option MISMATCH for *BOUNDARY_ACOUSTIC_COUPLING handles coupling of structural element faces and acoustic volume elements (ELFORMs 8 and 14) in the case where the coupling surfaces do not have coincident nodes.
- A porosity leakage formulation in *MAT_FABRIC (*MAT_034, FLC < 0) is now available for particle gas airbags (*AIRBAG_PARTICLE).
- *BOUNDARY_PRESCRIBED_ACCELEROMETER is disabled during dynamic relaxation.
- A new parameter CVRPER in *BOUNDARY_PAP defines porosity of a cover material encasing a solid part.
- A parameter TIEDID in *CONTACT_TIED_SURFACE_TO_SURFACE offers an optional incremental normal update in SMP to eliminate spurious contact forces that may appear in some applications.
- A new option SPOTSTP = 3 in *CONTROL_CONTACT retains spot welds even when the spot welds are not found by *CONTACT_SPOTWELD.
- The SMP consistency option (ncpu < 0) now pertains to the ORTHO_FRICTION contact option.
- Forces from *CONTACT_GUIDED_CABLE are now written to ncforc (both ASCII and binout).
- Discrete beam materials 70, 71, 74, 94, 121 calculate axial force based on change in length. Output the change in length instead of zero axial relative displacement to ASCII file disbout (*DATABASE_DISBOUT).
- *DATABASE_RCFORC_MOMENT is now supported in implicit.

- After the first implicit step, the output of projected cpu and wall clock times is written and the termination time is echoed.
- *DATABASE_MASSOUT is upgraded to include a summary table and to optionally add mass for nodes belonging to rigid bodies.
- Generate and store resultant forces for the LaGrange Multiplier joint formulation so as to give correct output to jntforc (*DATABASE_JNTFORC).
- Control the number of messages for deleted and failed elements using parameter MSGMAX in *CONTROL_OUTPUT.
- Nodal and resultant force output is written to nodfor for nodes defined in *DATABASE_NODAL_FORCE_GROUP in *FREQUENCY_DOMAIN_SSD analysis (SMP only).
- Ncforc data is now written for guided cables (*CONTACT_GUIDED_CABLE) in MPP.
- Jobid handling is improved in l2a utility so that binout files from multiple jobs, with or without a jobid-prefix, can be converted with the single command "l2a -j *binout*". The output contains the correct prefix according to the jobid.
- ALE_MULTI-MATERIAL_GROUP (AMMG) info is written to matsum (both ASCII and binout).
- Shell formulation 14 is switched to 15 (*SECTION_SHELL) in models that include axisymmetric SPH.
- *ELEMENT_BEAM_PULLEY is permitted with *MAT_CABLE_DISCRETE_BEAM.
- A warning during initialization is written if a user creates DKT triangles, either by ELFORM = 17 on *SECTION_SHELL or ESORT = 2 on *CONTROL_SHELL, that are thicker than the maximum edge length.
- Account is taken of degenerate acoustic elements with ELFORM 8. Tria and quad faces at acoustic-structure boundary are handled appropriately according to shape.
- The compression elimination option for 2D seatbelts, CSE = 2 in *MAT_SEATBELT is improved.
- Detailed material failure (*MAT_ADD_EROSION) messages in messag and d3hsp are suppressed when number of messages > MSGMAX (*CONTROL_OUTPUT).
- Implement SMP consistency (ncpu < 0) in *MAT_COHESIVE_GENERAL (*MAT_186) solids and shells.
- Viscoelastic model in *MAT_077_O now allows up to twelve terms in Prony series instead of standard six.
- Large curve IDs for friction table (*CONTACT_… with FS = 2) are enabled.
- Efficiency of GISSMO damage in *MAT_ADD_EROSION is improved.

- *MAT_ADD_PERMEABILITY_ORTHOTROPIC is now available for pore pressure analysis (*…_PORE_FLUID).
- For *MAT_224 solids and shells, material damage serves as the failure variable in *CONSTRAINED_TIED_NODES_FAILURE.
- The behavior of *MAT_ACOUSTIC is modified when used in combination with dynamic relaxation (DR). Acoustic domain now remains unperturbed in the DR phase but hydrostatic pressure from the acoustic domain is applied to the structure during DR.
- Option for 3D to 2D mapping is added in *INITIAL_ALE_MAPPING.
- *CONTACT_ERODING_NODES_TO_SURFACE contact may be used with SPH particles.
- Total Lagrangian SPH formulation 7 (*CONTROL_SPH) is now available in MPP.
- The output formats for linear equation solver statistics now accommodate very large numbers as seen in large models.
- *CONTROL_OUTPUT keyword parameter NPOPT is now applicable to thermal data. If NPOPT = 1, then printing of the following input data to d3hsp is suppressed:
    - *INITIAL_TEMPERATURE
    - *BOUNDARY_TEMPERATURE
    - *BOUNDARY_FLUX
    - *BOUNDARY_CONVECTION
    - *BOUNDARY_RADIATION
    - *BOUNDARY_ENCLOSURE_RADIATION
- Beam energy balance information is written to TPRINT file.
- MPP performance for LS-DYNA/MADYMO coupling is improved.
- Shell adaptivity (*CONTROL_ADAPTIVE) is improved to reduce the number of elements along curved surfaces in forming simulations.
- One-step unfolding (*CONTROL_FORMING_ONESTEP) is improved to accommodate blanks with small initial holes.
- Efficiency of FORM 3 isogeometric shells is improved.
- The processing of *SET_xxx_GENERAL is faster.
- *KEYWORD_JOBID now works even when using the *CASE command.
- Parts may be repositioned in a small restart by including *DEFINE_TRANSFORMATION and *NODE_TRANSFORM in the small restart deck to move nodes of a specified node set prior to continuing the simulation.

Capabilities added during 2012/2013 to create LS-DYNA R7.0:

- Three solvers, EM, CESE, and ICFD, and a volume mesher to support the latter two solvers, are new in Version 7. Brief descriptions of those solvers are given below. Keyword commands for the new solvers are in Volume III of the LS-DYNA Keyword User's Manual. These new solvers are only included in double precision executables.
- Keyword family: *EM_, the keywords starting with *EM refer to and control the Electromagnetic solver problem set up:
    - EM Solver Characteristics:
        - Implicit
        - Double precision
        - Dynamic memory handling
        - SMP and MPP
        - 2D axisymmetric solver / 3D solver
        - Automatic coupling with structural and thermal LS-DYNA solvers
        - FEM for conducting pieces only, no air mesh needed (FEM-BEM system)
        - Solid elements for conductors, shells can be insulators
    - EM Solver Main Features:
        - Eddy Current (a.k.a Induction-Diffusion) solver
        - Induced heating solver
        - Resistive heating solver
        - Imposed tension or current circuits
        - Exterior field
        - Magnetic materials (beta version)
        - Electromagnetic contact
        - EM Equation of states (Conductivity as a function of temperature)
    - EM Solver Applications (Non-exhaustive) :
        - Electromagnetic forming
        - Electromagnetic welding
        - Electromagnetic bending
        - Inductive heating
        - Resistive heating
        - Rail-gun
        - Ring expansions
- Keyword family: *CESE_, the keywords starting with *CESE refer to and control the Compressible CFD solver problem set up:

- CESE Solver Characteristics:
  - Explicit
  - Double precision
  - Dynamic memory handling
  - SMP and MPP
  - 3D solver / special case 2D solver and 2D axisymmetric solver
  - Automatic coupling with structural and thermal LS-DYNA solvers
  - Eulerian fixed mesh or moving mesh (Either type input with *ELEMENT_SOLID cards or using *MESH cards)
- CESE Solver Main Features:
  - The CESE (Conservation Element / Solution Element) method enforces conservation in space-time
  - Highly accurate shock wave capturing
  - Cavitation model
  - Embedded (immersed) boundary approach or moving (fitting) approach for FSI problems
  - Coupled stochastic fuel spray solver (See *STOCHASTIC keywords)
  - Coupling with chemistry (See *CHEMISTRY keywords) solver
- CESE Solver Applications (Non-exhaustive):
  - Shock wave capturing
  - Shock/acoustic wave interaction
  - Cavitating flows
  - Conjugate heat transfer problems
  - Many different kinds of stochastic particle flows, e.g, dust, water, fuel.
  - Chemically reacting flows, e.g, detonating flow, supersonic combustion.

- Keyword family: *ICFD_, the keywords starting with *ICFD refer to and control the incompressible CFD solver problem set up:
  - ICFD Solver Characteristics:
    - Implicit
    - Double precision
    - Dynamic memory handling
    - SMP and MPP
    - 2D solver / 3D solver
    - Makes use of an automatic volume mesh generator for fluid domain (See *MESH keywords)
    - Coupling with structural and thermal LS-DYNA solvers

- ICFD Solver Main Features:
  - Incompressible fluid solver
  - Thermal solver for fluids
  - Free Surface flows
  - Two-phase flows
  - Turbulence models
  - Transient or steady-state problems
  - Non-Newtonian fluids
  - Boussinesq model for convection
  - Loose or strong coupling for FSI (Fluid-structure interaction)
  - Exact boundary condition imposition for FSI problems
- ICFD Solver Applications (Non-exhaustive) :
  - External aerodynamics for incompressible flows
  - Internal aerodynamics for incompressible flows
  - Sloshing, Slamming and Wave impacts
  - FSI problems
  - Conjugate heat transfer problems

- Keyword family: *MESH_, the keywords starting with *MESH refer to and control the tools for the automatic volume mesh generator for the CESE and ICFD solvers.
  - Mesh Generator Characteristics:
    - Automatic
    - Robust
    - Generic
    - Tetrahedral elements for 3D, Triangles in 2D
    - Closed body fitted mesh (surface mesh) needs to be provided for volume generation
  - Mesh Generator Main Features:
    - Automatic remeshing to keep acceptable mesh quality for FSI problems (ICFD only)
    - Adaptive meshing tools (ICFD only)
    - Anisotropic boundary layer mesh
    - Mesh element size control tools
    - Remeshing tools for surface meshes to ensure mesh quality
  - Mesh Generator Applications :
    - Used by the Incompressible CFD solver (ICFD).
    - Used by the Compressible CFD solver (CESE).

Other additions to Version 7 include:

- Add new parameter VNTOPT to *AIRBAG_HYBRID, that allows user more control on bag venting area calculation.
- Allow heat convection between environment and CPM bag (*AIRBAG_PARTICLE) bag. Apply proper probability density function to part's temperature created by the particle impact.
- *AIRBAG_PARTICLE and *SENSOR_SWITCH_SHELL_TO_VENT allows user to input load curve to control the venting using choking flow equation to get proper probability function for vents. Therefore, this vent will have the same vent rate as real vent hole.
- Add new option NP2P in *CONTROL_CPM to control the repartition frequency of CPM particles among processors (MPP only).
- Enhance *AIRBAG_PARTICLE to support a negative friction factor (FRIC or PFRIC) in particle to fabric contact. Particles are thus able to rebound at a trajectory closer to the fabric surface after contact.
- Use heat convection coefficient HCONV and fabric thermal conductivity KP to get correct effective heat transfer coefficient for heat loss calculation in *AIRBAG_PARTICLE. If KP is not given, H will be used as effective heat transfer coefficient.
- Extend CPM inflator orifice limit from 100 to unlimited (*AIRBAG_PARTICLE).
- Support dm_in_dt and dm_out_dt output to CPM chamber database (*DATABASE_ABSTAT) to allow user to study mass flow rate between multiple chambers.
- Previously, the number of ships (rigid bodies) in *BOUNDARY_MCOL, as specified by NMCOL, was limited to 2. Apparently, this was because the code had not been validated for more than 2 rigid bodies, but it is believed that it should not be a problem to remove this restriction. Consequently, this limit has been raised to 10, with the caveat that the user should verify the results for NMCOL > 2.
- Implemented a structural-acoustic mapping scheme (*BOUNDARY_ACOUSTIC_MAPPING), for mapping transient structural nodal velocity to acoustic volume surface nodes. This is useful if the structure finite element mesh and the acoustic boundary/finite element mesh are mismatched.
- *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURACE_ORTHO_FRICTION can now be defined by part set IDs when supplemented by *DEFINE_FRICTION_ORIENTATION. Segment sets with orientation per *DEFINE_FRICTION_ORIENTATION are generated automatically.
- Contact force of *CONTACT_ENTITY is now available in intfor (*DATABASE_BINARY_INTFOR).
- *CONTACT_FORCE_TRANSDUCER_PENALTY will now accept node sets for both the slave and master sides, which should allow them to work correctly for eroding materials. BOTH sides should use node sets, or neither.

- Added option to create a backup penalty-based contact for a tied constraint-based contact in the input (IPBACK on Card E of *CONTACT).
- New option for *CONTACT_ENTITY. If variable SO is set to 2, then a constraint-like option is used to compute the forces in the normal direction. Friction is treated in the usual way.
- *CONTACT_ENTITY: allow friction coefficient to be given by a "coefficient vs time" load curve (input < 0 -> absolute value is the load curve ID). Also, if the friction coefficient bigger or equal 1.0, the node sticks with no sliding at all.
- Minor tweak to the way both MPP and SMP handle nodes sliding off the ends of beams in *CONTACT_GUIDED_CABLE.
- Frictional energy output in sleout (*DATABASE_SLEOUT) supported for *CONTACT_…_MORTAR.
- Tiebreak damage parameter output as "contact gap" in intfor file for *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_MORTAR, OPTION = 9.
- Added MPP support for *CONTACT_2D_AUTOMATIC_SINGLE_SURFACE and *CONTACT_2D_AUTOMATIC_SURFACE_TO_SURFACE.
- Added keyword *CONSTRAINED_MULTIPLE_GLOBAL for defining multi-node constraints for imposing periodic boundary conditions.
- Enhancement for *CONSTRAINED_INTERPOLATION_SPOTWELD (SPR3): calculation of bending moment is more accurate now.
- If *CONSTRAINED_NODAL_RIGID_BODY nodes are shared by several processors with mass scaling on, the added mass is not summed up across processors. This results in an instability of the NRB. (MPP only)
- *ALE_REFINE has been replaced and expanded upon by the *CONTROL_REFINE family of commands. These commands invoke local mesh refinement of shells, solids, and ALE elements based on various criteria.
- Shells or solids in a region selected for refinement (parent element) are replaced by 4 shells or 8 solids, respectively. *CONTROL_REFINE_SHELL applies to shells, *CONTROL_REFINE_SOLID applies to solids and *CONTROL_REFINE_ALE and *CONTROL_REFINE_ALE2D applies to ALE elements. Each keyword has up to 3 lines of input. If only the 1st card is defined, the refinement occurs during the initialization. The 2nd card defines a criterion CRITRF to automatically refine the elements during the run. If the 3rd card is defined, the refinement can be reversed based on a criterion CRITM. All commands are implemented for SMP and MPP, but SMP is not parallelized.
- *CONTROL_REFINE_MPP_DISTRIBUTION distributes the elements required by the refinement across the MPP processes.

- Eliminate automatic writing of a d3plot plot state after each 3D tetrahedral remeshing operation (*CONTROL_REMESHING) to reduce volume of output.
- Generate disbout output (*DATABASE_DISBOUT) for MPP and SMP binout files.
- Extend *DATABASE_MASSOUT to include option to output mass information on rigid body nodes.
- Added new keyword *CHANGE_OUTPUT for full deck restart to override default behavior of overwriting existing ASCII files. For small restart, this option has no effect since all ASCII output is appended to the result of previous run already.
- Added new option (NEWLENGD) to 2nd field of 3rd card of *CONTROL_OUTPUT to write more detailed legend in ASCII output files. At present, only rcforc and jntforc are implemented.
- Increased default binary file size scale factor (x=) from 7 to 1024. That means the default binary file size will be 1 Gb for single version and 2 Gb for double version.
- Add echo of new "max frequency of element failure summaries" flag (FRFREQ in *CONTROL_OUTPUT) to d3hsp file.
- Support LSDA/binout output for new pllyout file (*DATABASE_PLLYOUT, *ELEMENT_BEAM_PULLEY) in both SMP and MPP.
- Allow degenerated hexahedrons (pentas) for cohesive solid elements (ELFORM = 19, 20) that evolve from an extrusion of triangular shells. The input of nodes on the element cards for such a pentahedron is given by: N1, N2, N3, N3, N4, N5, N6, N6.
- Add new option to activate drilling constraint force for shells in explicit calculations. This can be defined by parameters DRCPSID (part set) and DRCPRM (scaling factor) on *CONTROL_SHELL.
- Add SMP ASCII database "pllyout" (*DATABASE_PLLYOUT) for *ELEMENT_BEAM_PULLEY.
- *FREQUENCY_DOMAIN_ACOUSTIC_BEM:
    - Added an option to output real part of acoustic pressure in time domain.
    - Enabled BEM acoustic computation following implicit transient analysis.
    - Implemented coupling between steady state dynamics and collocation acoustic BEM.
    - Implemented Acoustic Transfer Vector (ATV) to variational indirect BEM acoustics.
    - Enabled boundary acoustic mapping in BEM acoustics.
- *FREQUENCY_DOMAIN_ACOUSTIC_FEM:
    - Added boundary nodal velocity to binary plot file d3acs.
    - Implemented pentahedron elements in FEM acoustics.
    - Enabled using boundary acoustic mapping in FEM acoustics.

- *FREQUENCY_DOMAIN_FRF:
    - Updated FRF to include output in all directions (VAD2 = 4).
    - Added treatment for FRF with base acceleration (node id can be 0).
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION:
    - Updated calculation of PSD and RMS von Mises stress in random vibration environment, based on Sandia National Laboratories report, 1998.
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION_FATIGUE:
    - Implemented an option to incorporate initial damage ratio in random vibration fatigue.
- *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM:
    - Implemented double sum methods (based on Gupta-Cordero coefficient, modified Gupta-Cordero coefficient, and Rosenblueth-Elorduy coefficient).
    - Updated calculating von Mises stress in response spectrum analysis.
    - Implemented treatment for multi simultaneous input spectra.
    - Improved double sum methods by reducing number of loops.
- *FREQUENCY_DOMAIN_SSD:
    - Added the option to output real and imaginary parts of frequency response to d3ssd.
    - Added the option to output relative displacement, velocity and acceleration in SSD computation in the case of base acceleration. Previously only absolute values were provided.
- Implemented keyword *FREQUENCY_DOMAIN_MODE_{OPTION} so that user can select the vibration modes to be used for frequency response analysis.
- Implemented keyword *SET_MODE_{OPTION} so that user can define a set of vibration modes, to be used for frequency response analysis.
- Implemented keyword *FREQUENCY_DOMAIN_PATH to define the path of binary databases containing mode information, used in restarting frequency domain analysis, e.g. frf, ssd, random vibration.
- Compute normal component of impulse for oblique plates in *INITIAL_MINE_IMPULSE. The feature is no longer limited to horizontal plates.
- Disable license security for *INITIAL_IMPULSE_MINE. The feature is no longer restricted.

- Enabled hourglass type 7 to work well with *INITIAL_FOAM_REFERENCE_GEOMETRY so that initial hourglass energy is properly calculated and foam will spring back to the initial geometry.
- Accommodate erosion of thin shells in *LOAD_BLAST_ENHANCHED.
- *LOAD_VOLUME_LOSS has been changed such that after the analysis time exceeds the last point on the curve of volume change fraction versus time, the volume change is no longer enforced.
- *LOAD_BODY_POROUS new option AOPT added to assign porosity values in material coordinate system.
- Added *LOAD_SEGMENT_FILE.
- Add new sensor definition, *SENSOR_DEFINE_ANGLE. This card traces the angle formed between two lines.
- *SENSOR_DEFINE_NODE can be used to trace the magnitude of nodal values (coordinate, velocity or accleration) when VID is “0” or undefined.
- Add two new parameters to *SENSOR_DEFINE_ELEMENT, scale factor and power, so that user can adjust the element-based sensor values (strain, stress, force, …).
- Change history variables 10-12 in *MAT_054/*MAT_ENHANCED_COMPOSITE_DAMAGE (thin shells only) to represent strains in material coordinate system rather than in local element coordinate system. This is a lot more helpful for post-processing issues. This change should not lead to different results other than due to different round-off errors.
- New features and enhancements to *MAT_244/*_MAT_UHS_STEEL:
    - Added implicit support for MAT_244.
    - Changed the influence of the austenite grain size in Mat244 according to Li et al.
    - Changed the start temperatures to fully follow WATT et al and Li et al.
    - Hardness calculation is now improved when noncontinuous cooling is applied i.e., tempering.
    - Added temperature dependent Poisson ratio and advanced reaction kinetics.
    - Added new advanced option to describe the thermal expansion coefficients for each phase.
    - Added option to use Curve ID or a Table ID for describing the latent heat generation during phase transormations.
    - Added support for table definition for Youngs modulus. Now you can have one temperature dependent curve for each of the 5 phases
- Added support for implicit to *MAT_188.

- Added material model *MAT_273/*MAT_CDPM/*MAT_CONCRETE_DAMAGE_PLASTIC_MODEL. This model is aimed at simulations where failure of concrete structures subjected to dynamic loadings is sought. The model is based on effective stress plasticity and has a damage model based on both plastic and elastic strain measures. Implemented for solids only but both for explicit and implicit simulations. Using an implicit solution when damage is activated may trigger a slow convergense. IMFLAG = 4 or 5 can be useful.
- Added an option in *MAT_266 (*MAT_TISSUE_DISPERSED) so that the user can tailor the active contribution with a time dependent load curve instead of using the internal hardcoded option. See ACT10 in the User's Manual.
- *MAT_173/*MAT_MOHR_COULOMB is available in 2D.
- Enable *MAT_103 and *MAT_104 to discretize the material load curves according to the number of points specified by LCINT in *CONTROL_SOLUTION.
- Implement Prony series up to 18 terms for shells using *MAT_076/*MAT_GENERAL_VISCOELASTIC.
- Added *DEFINE_STOCHASTIC_VARIATION and the STOCHASTIC option for *MATs 10, 15, 24, 81, 98 for shells, solids, and type 13 tets. This feature defines a stochastic variation in the yield stress and damage/failure of the aforementioned material models.
- Add Moodification for *DEFINE_CONNECTION_PROPERTIES, PROPRUL = 2: thinner weld partner is first partner, PROPRUL = 3: bottom (nodes 1-2-3-4) weld partner is first partner.
- Add spotweld area to debug output of *DEFINE_CONNECTION_PROPERTIES which is activated by *CONTROL_DEBUG.
- Add support of *MAT_ADD_EROSION option NUMFIP < 0 for standard (non-GISSMO) failure criteria. Only for shells.
- Improve implicit convergence of *MAT_ADD_EROSION damage model GISSMO by adding damage scaling (1-D) to the tangent stiffness matrix.
- Provide plastic strain rates (tension/compression, shear, biaxial) as history variables no. 16, 17, and 18 for *MAT_187.
- Add new variables to user failure routine matusr_24 (activated by FAIL < 0 on *MAT_024 and other materials): integration point numbers and element id.
- Add new energy based, nonlocal failure criterion for *MAT_ADD_EROSION, parameters ENGCRT (critical energy) and RADCRT (critical radius) after EPSTHIN. Total internal energy of elements within a radius RADCRT must exceed ENGCRT for erosion to occur. Intended for windshield impact.
- Add new option to *MAT_054 for thin shells: Load curves for rate dependent strengths and a rate averaging flag can be defined on new optional card 9.

- Add new option for *MAT_MUSCLE: Input parameter SSP < 0 can now refer to a load curve (stress vs. stretch ratio) or a table (stress vs. stretch ratio vs. normalized strain rate).
- Expand list of variables for *MAT_USER_DEFINED_MATERIAL_MODELS by characteristic element size and element ID.
- Enable *MAT_USER_DEFINED_MATERIAL_MODELS to be used with tetrahedron element type 13. New sample routines “umat41_t13” and “umat41v_t13” show corresponding pressure calculation in the elastic case.
- Add a new feature to *MAT_125 allowing C1 and C2 to be used in calculation of back stress. When plastic strain < 0.5%, C1 is used, otherwise C2 is used as described in Yoshida's paper.
- Extend non-linear strain path (_NLP_FAILURE) in *MAT_037 to implicit.
- *MAT_173/*MAT_MOHR_COULOMB now works in ALE. A new option has been added to suppress the tensile limit on hydrostatic stress recommended for ALE multi-material use.
- Upgraded *MAT_172/*MAT_CONCRETE_EC2.
    - Corrections to DEGRAD option.
    - Concrete and reinforcement types 7 and 8 have been added to reflect changes to Eurocode 2.
    - Extra history variables for reinforcement stress and strain are now output as zero for zero-fraction reinforcement directions.
- Added RCDC model for solid *MAT_082.
- Added Feng's failure model to solid *MAT_021.
- Added *MAT_027 for beams.
- Added *DEFINE_HAZ_PROPERTIES and *DEFINE_HAZ_TAILOR_WELDED_BLANK for modifying material behavior near a spot weld.
- Added fourth rate form to viscoplastic Johnson-Cook model (*MAT_015).
- Added option to *MAT_224 to not delete the element if NUMINT = -200.
- New damage initiation option 3 in multi fold damage criteria in *MAT_ADD_EROSION. Very similar to option 2 but insensitive to pressure.
- Added rotational resistance in *MAT_034/*MAT_FABRIC. Optionally the user may specify the stiffness, yield and thickness of and elastic-perfectly-plastic coated layer of a fabric that results in a rotational resistance during the simulation.
- FLDNIPF < 0 in *MAT_190/*MAT_FLD_3-PARAMETER_BARLAT for shell elements means that failure occurs when all integration points within a relative distance of -FLDNIPF from the mid surface has reached the fld criterion.

- A computational welding mechanics *MAT_270/*MAT_CWM material is available that allows for element birth based on a birth temperature as well as annealing based on an annealing temperature. The material is in addition a thermo-elasto-plastic material with kinematic hardening and temperature dependent properties.
- Added *MAT_271/*MAT_POWDER, a material for manufacturing (i.e., compaction and sintering) of cemented carbides. It is divided into an elastic-plastic compaction model that is supposed to be run in a first phase, and a viscoelastic sintering model that should be run in a second phase. This model is for solid elements.
- For IHYPER = 3 on a *MAT_USER_DEFINED_… shell material, the deformation gradient is calculated from the geometry instead of incremented by the velocity gradient. The deformation gradient is also passed to the user defined subroutines in the global system together with a transformation matrix between the global and material frames. This allows for freedom in how to deal with the deformation gradient and its transformations in orthotropic (layered) materials.
- The Bergstrom-Boyce viscoelastic rubber model is now available in explicit and implicit analysis as *MAT_269/*MAT_BERGSTROM_BOYCE_RUBBER. The Arruda-Boyce elastic stress is augmented with a Bergstrom-Boyce viscoelastic stress corresponding to the response of a single entangled chain in a polymer gel matrix.
- Added a new parameter IEVTS to *MAT_USER_DEFINED_MATERIAL_MODELS (*MAT_041-050). IEVTS is optional and is used only by thick shell formulation 5. It points to the position of E(a) in the material constants array. Following E(a), the next 5 material constants must be E(b), E(c), v(ba), v(ca), and v(cb). This data enables thick shell formulation 5 to calculate an accurate thickness strain, otherwise the thickness strain will be based on the elastic constants pointed to by IBULK and IG.
- Implemented enhancements to fabric material (*MAT_034), FORM = 14. Stress-strain curves may include a portion for fibers in compression. When unload/reload curves with negative curve ID are input (curve stretch options), the code that finds the intersection point now extrapolates the curves at their end rather than simply printing an error message if an intersection point cannot be found before the last point in either curve.
- Map 1D to 3D by beam-volume averaging the 1D data over the 3D elements (*INITIAL_ALE_MAPPING).
- In a 3D to 3D mapping (*INITIAL_ALE_MAPPING), map the relative displacements for the penalty coupling in *CONSTRAINED_LAGRANGE_IN_SOLID.
- The [name].xy files associated with *DATABASE_ALE_MAT are now created when sense switches sw1, sw2, quit, or stop are issued.
- *ALE_ESSENTIAL_BOUNDARY is available in 2D.
- *DATABASE_FSI is available for 2D (MPP).

- *ALE_ESSENTIAL_BOUNDARY implemented to apply slip-only velocity BC along ALE mesh surface.
- *CONTROL_ALE flag INIJWL = 2 option added to balance initial pressure state between ALE Soil and HE.
- Include SPH element (*ELEMENT_SPH) in time step report.
- Time step and internal energy of 2D axisymmetric SPH elements are calculated in a new way more consistent with the viscosity force calculation.
- Only apply viscosity force to x and y components of 2D axisymmetric SPH element, not on hoop component.
- MAXV in *CONTROL_SPH can be defined as a negative number to turn off velocity checking.
- Improve calculation of 2D axisymmetric SPH contact force in *DEFINE_SPH_TO_SPH_COUPLING.
- Added the following material models for SPH particles: *MAT_004/*MAT_ELASTIC_PLASTIC_THERMAL (3D only) and *MAT_106/*MAT_ELASTIC_VISCOPLASTIC_THERMAL
- Added a new parameter DFACT for *DEFINE_SPH_TO_SPH_COUPLING. DFACT invokes a viscous term to damp the coupling between two SPH parts and thereby reduce the relative velocity between the parts.
- Added BOUNDARY_CONVECTION and BOUNDARY_RADIATION for explicit SPH thermal solver.
- *CONTROL_REMESHING_EFG:
    - Add eroding failed surface elements and reconstructing surface in EFG adaptivity.
    - Add a control parameter for monotonic mesh resizing in EFG adaptivity.
    - Add searching and correcting self-penetration for adaptive parts in 3D tetrahedron remeshing.
- Enhance 3D axisymmetric remeshing with 6-node/8-node elements
- *CONTROL_REMESHING:
    - Use RMIN/RMAX along with SEGANG to determine element size.
    - Remove the restriction that the reference point of computational model must be at original point $(0, 0, 0)$.
    - Rewrite the searching algorithm for identifying the feature lines of cross-sections in order to provide more stable remeshing results.
- Improve rigid body motion in EFG shell type 41.

- Support EFG pressure smoothing in EFG solid type 42 for *MAT_ELASTIC_VISCOPLASTIC_THERMAL.
- Add visco effect for implicit EFG solid type 42.
- Add new EFG solid type 43 (called Meshfree-Enriched FEM, MEFEM) for both implicit and explicit. This element formulation is able to relieve the volumetric locking for nearly incompressible material (eg. rubber) and performs strain smoothing across elements with common faces.
- EFG shell adaptivity no longer requires a special license.
- Application of EFG in an implicit analysis no longer requires a special license.
- Add *SENSOR_CONTROL for prescribed motion constraints in implicit.
- Update *INTERFACE_LINKING_NODE in implicit to catch up with explicit, including adding scaling factors.
- Add support for *DATABASE_RCFORC_MOMENT for implicit.
- Enhance Iterative solvers for Implicit Mechanics.
- Add, after the first implicit time step, the output of projected cpu and wall clock times. This was already in place for explicit. Also echo the termination time.
- Add variable MXDMP in *CONTROL_THERMAL_SOLVER to write thermal conductance matrix and right-hand side every MXDMP time steps.
- Add keyword *CONTROL_THERMAL_EIGENVALUE to calculate eigenvalue(s) of each thermal conductance matrix.
- Added thermal material model *MAT_THERMAL_ORTHOTROPIC_TD_LC. This is an orthotropic material with temperature dependent properties defined by load curves.
- Changed structured file format for control card 27 (first thermal control card). Several input variables used i5 format limiting their value to 99,999. A recent large model exceeded this limit. The format was changed to i10. This change is not backward compatible. Old structured input files will no longer run unless control card 27 is changed to the new i10 format. This change does not affect the KEYWORD file.
- Add thermal material *MAT_T07/*MAT_THERMAL_CWM for welding simulations, to be used in conjunction with mechanical counterpart *MAT_270/*MAT_CWM.
- Modify decomposition costs of *MAT_181 and *MAT_183.
- Introduce new timing routines and summary at termination.
- Echo “MPP contact is groupable” flag to d3hsp
- Bodies using *MAT_RIGID_DISCRETE were never expected to share nodes with non-rigid bodies, but this now works in MPP.

- There is no longer any built-in limitation on the number of processors that may be used in MPP.
- Echo contents of the MPP pfile (including keyword additions) to the d3hsp and mes0000 files.
- Add new keyword *CONTROL_MPP_PFILE, which allows for insertion of text following this command to be inserted into the MPP pfile (p = pfile).
- Change in MPP treatment of *CONSTRAINED_TIEBREAK. They now share a single MPI communicator, and a single round of communication. This should improve performance for problems with large numbers of these, without affecting the results.
- Added two input variables for *CONTROL_FORMING_ONESTEP simulation, TSCLMIN is a scale factor limiting the thickness reduction and EPSMAX defines the maximum plastic strain allowed.
- Added output of strain and stress tensors for onestep solver *CONTROL_FORMING_ONESTEP, to allow better evaluation of formability.
- Improved *CONTACT_AUTO_MOVE: before changes the termination time, and it causes problems when several tools need to be moved. Now *CONTACT_AUTO_MOVE does not change the termination time, but changes the current time. In this way, several tools can be moved without the need to worry about the other tool's move. This is especially useful in multi-flanging and hemming simulations.
- Made improvements to previously undocumented keyword *INTERFACE_BLANKSIZE, including adding the options INITIAL_TRIM, and INITIAL_ADAPTIVE. This keyword was developed for blank size development in sheet metal forming. Generally, for a single forming process, only the option DEVELOPMENT is needed and inputs are an initial estimated blank shape, a formed blank shape, and a target blank shape in either mesh or boundary coordinates. Output will be the calculated/corrected initial blank shape. Initial blank mesh and formed blank mesh can be different (e.g. adaptive). For a multi-stamping process involving draw, trimming and flanging, all three options are needed. Related commands for blank size estimation are *CONTROL_FORMING_ONESTEP, and for trim line development, *CONTROL_FORMING_UNFLANGING.
- Made improvements and added features to previously undocumented keyword *CONTROL_FORMING_UNFLANGING, this keyword unfolds flanges of a deformable blank, e.g., flanged or hemmed portions of a sheet metal part, onto a rigid tooling mesh using the implicit static solver. It is typically used in trim line mapping during a draw die development process. The roots of the flanges or hemmed edges are automatically processed based on a user input of a distance tolerance between the flanges/hemmed edges and rigid tool. It includes the ability to handle a vertical flange wall. Other keywords related to blank size development are, *CONTROL_FORMING_ONESTEP, and *INTERFACE_BLANKSIZE_DEVELOPMENT.

- Added keyword *CONTROL_FORMING_OUTPUT which allows control of d3plot output by specifying distances to tooling home. It works with automatic position of stamping tools using *CONTROL_FORMING_AUTOPOSITION_PARAMETER.
- Added the LOCAL_SMOOTH option to *INTERFACE_COMPENSATION_NEW which features smoothing of a tool's local area mesh, which could otherwise become distorted due to, e.g., bad/coarse mesh of the original tool surface, tooling pairs (for example, flanging post and flanging steel) do not maintain a constant gap and several compensation iterations. This new option also allows for multiple regions to be smoothed. Local areas are defined by *SET_LIST_NODE_SMOOTH.
- Added output to rcforc for *DEFINE_DE_TO_SURFACE_COUPLING.
- Implement traction surface for *DEFINE_DE_TO_SURFACE_COUPLING.
- Add keyword *DATABASE_BINARY_DEMFOR with command line option dem = dem_int_force. This will turn on the DEM interface force file for DEM coupling option. The output frequency is controlled by the new keyword.
- Add new feature *DEFINE_DE_INJECTION to allow DEM particle dropping from user defined plane.
- Add new option_VOLUME to *ELEMENT_DISCRETE_SPHERE. This will allow DEM input based on per unit density and use *MAT card to get consistent material properties.
- Added FORM = -4 for *ELEMENT_SHELL_NURBS_PATCH. Rotational dofs are automatically set at control points at the patch boundaries, whereas in the interior of the patch only translational dofs are present. This helps for joining multiple nurbs patches at their C0-boundaries.
- Disabled FORM = 2 and 3 for *ELEMENT_SHELL_NURBS_PATCH. These formulations are experimental and not fully validated yet.
- Added energy computation for isogeometric shells (*ELEMENT_SHELL_NURBS_PATCH) to matsum.
- Allow isogeometric shells (*ELEMENT_SHELL_NURBS_PATCH) to behave as rigid body (*MAT_RIGID).
- Added “g” as abbreviation for gigawords in specification of memory on execution line, e.g, memory = 16g is 16 billion words.
- Suppress non-printing characters in *COMMENT output.
- Add command line option “pgpkey” to output the current public PGP key used by LS-DYNA. The output goes to the screen as well as a file named “lstc_pgpkey.asc” suitable for directly importing into GPG.
- When reading the NAMES file, allow a “+” anywhere on a line to indicate there will be a following line, not just at the end. This was never intended, but worked before r73972 and some customers use it that way.

- Check for integer overflow when processing command line arguments and the memory value on the *KEYWORD card.
- Added new capability for *INTERFACE_LINKING_NODE to scale the displacements of the moving interface.
- Support for *KEYWORD_JOBID with internal *CASE driver.
- *DAMPING_FREQUENCY_RANGE now works for implicit dynamic solutions. An error check has been added to ensure that the timestep is small enough for the damping card to work correctly.
- Added new option *DAMPING_FREQUENCY_RANGE_DEFORM to damp only the deformation instead of the global motion.
- Added *DEFINE_VECTOR_NODES. A vector is defined using two node IDs.
- Add sense switch "prof" to output current timing profile to message (SMP) file or mes#### (MPP) files. Also, for MPP only, collect timing information from processor and output to prof.out when sense switch "prof" is detected.

Capabilities added during 2013/2014 to create LS-DYNA R7.1:

- Add MUTABLE option for *PARAMETER so that parameter values can be redefined later in the input deck.
- Change MPP treatment of two-sided *CONTACT_FORCE_TRANSDUCER so that proper mass and moment values can be output to the rcforc file.
- MPP support for non-zero birthtime for *CONTACT_SINGLE_EDGE.
- Add new command line option "ldir=" for setting a *local* working directory. In MPP, this has the same effect as setting the "directory { local }" pfile option (and it overrides that option). For SMP, it indicates a directory where local, working files should be placed.
- Add support for SMOOTH option in MPP groupable contact.
- Add new keyword card *CONTROL_REQUIRE_REVISION to prevent the model from being run in old versions of LS-DYNA.
- Add part set specification for dynamic relaxation with implicit using *CONTROL_DYNAMIC_RELAXATION. This is a new feature specified with idrflg = 6 on *CONTROL_DYNAMIC_RELAXATION. This allows implicit to be used for the dynamic relaxation phase for models involving parts being modeled with SPH and/or ALE while excluding those parts from the dynamic relaxation phase.
- Add new feature for implicit automatic time step control to cooperate with thermal time step control. On *CONTROL_IMPLICIT_AUTO, IAUTO = 2 is the same as IAUTO = 1 with the extension that the implicit mechanical time step is limited by the active thermal time step.

- On *CONTROL_IMPLICIT_SOLUTION, add negative value of MAXREF for implicit mechanics. Nonlinear iteration will terminate after |MAXREF| iterations. With MAXREF < 0 convergence is declared with a warning. Simulation will continue. Positive values of MAXREF still cause failure of convergence to be declared leading to either a time step reduction or an error termination.
- Add *CONTROL_IMPLICIT_MODAL_DYNAMIC keywords and features. This elevates the modal dynamic features of IMASS = 2 on *CONTROL_IMPLICIT_-DYNAMICS. It also adds additional features of damping and mode selection and stress computations.
- New material model *MAT_DRY_FABRIC / MAT_214, which can be used in modeling high strength woven fabrics with transverse orthotropic behavior.
- Add *ALE_COUPLING_NODAL_PENALTY, penalty-based nodal coupling with ALE.
- Add type 8 *ELEMENT_SEATBELT_PRETENSIONER which takes energy-time curve, instead of pull-in or force curve.
- Add type 9 *ELEMENT_SEATBELT_PRETENSIONER for energy-based buckle / anchor pretensioner.
- Add *DATABASE_BINARY_FSILNK. This feature stores coupling pressure from *CONSTRAINED_LAGRANGE_IN_SOLID in a binary time history file for use in a separate model that does not include ALE.
- Add *LOAD_SEGMENT_FSILNK. Use pressure loads stored in aforementioned binary time history file to load model that does not have ALE elements.
- Add new keyword *DEFINE_SPH_DE_COUPLING to allow SPH particles to contact discrete element spheres (DES).
- Add MOISTURE option to *MAT_076 solids. Allows moisture content to be input as a function of time. Material parameters are then scaled according to the moisture and a moisture strain is also introduced.
- Add *RIGIDWALL_FORCE_TRANSDUCER to output forces from rigidwalls acting on node sets.
- Add LOG_INTERPOLATION option to *MAT_024. This offers an alternate means of invoking logarithmic interpolation for strain rate effects. The other way is to input the natural log of strain rate in the table LCSS.
- Add capability in *MAT_ADD_EROSION (NUMFIP < -100) to set stress to zero in each shell integration point as it reaches the failure criterion. When |NUMFIP|-100 integration points have failed, the shell is eroded. In contrast, when NUMFIP > 0, failed integration points continue to carry full load as though they were unfailed until element erosion occurs.

- Add new keyword, *PARAMETER_TYPE, for use by LS-PrePost when combining keyword input files. The appropriate offset is applied to each ID value defined using *PARAMETER_TYPE, according to how that ID is used.
- Allow use of load curve to specify damping as a function of time in *DAMPING_-RELATIVE.
- Add a segment based (SOFT = 2) contact option to include the overlap area in the contact stiffness calculation. This is good for improving the friction calculation and possibly for implicit convergence. The option is turned on by setting FNLSCL > 0 and DNLSCL = 0. As DNLSCL = 0, the contact stiffness is not non-linear. This new option is also useful when used with another improvement that was made to the FS = 2 friction coefficient by table lookup option in segment based contact. When the above mentioned FNLSCL > 0, option is used, the FS = 2 option is now very accurate.
- Add a new RCDC damage option, *MAT_PLASTICITY_WITH_DAMAGE_OR-THO_RCDC1980 which is consistent with the WILKINS paper. It uses the principal values of stress deviators and a different expression for the A_d term.
- Add a TIETYP option to *CONTACT_2D_AUTOMATIC. By default the tied contact automatically uses constraint equations when possible for 2D tied contact. If a conflict is detected with other constraints, or to avoid 2-way constraints, penalty type ties are used when constraints are not possible. The TIETYP option, when set to 1, causes all ties to use the penalty method. This is useful if in spite of the code's best efforts to avoid problems, there is still a conflict in the model.
- Add a scale factor for scaling the frictional stiffness for contact. The parameter is FRICSF on optional card E and it's only supported for segment based (SOFT = 2) contact. This was motivated by a rubber vs. road skidding problem where the friction coefficent had static, dynamic and decay parameters defined. The growth of the frictional force was too slow so the static coulomb value could not be achieved. By scaling the frictional stiffness higher, the coulomb value could approach the static value.
- Add keyword *CONTACT_2D_AUTOMATIC_FORCE_TRANSDUCER. Like the 3D force transducers, it does no contact calculation but only measures the contact forces from other contact definitions. When only a slave side is defined, the contact force on those segments is measured. Currently, two surface force transducers are not available.
- Add options to *MAT_058:
    - Load curves for rate dependent strain values (E11C, E11T, …) can be defined on new optional card 9.
    - Load curves for rate dependent strengths (XC, XT, …) and a rate averaging flag can be defined on new optional card 8.
    - Abscissa values in above curves are taken to be natural log of strain rate when the first value is negative.

- Add optional transverse shear damage to *MAT_058.

- Add MAT_261 and MAT_262 for general use. *MAT_261 is *MAT_LAMINATED_FRACTURE_DAIMLER_PINHO. *MAT_262 is *MAT_LAMINATED_FRACTURE_DAIMLER_CAMANHO.
- Add pentahedra cohesive solid element types (TYPE = 21 & 22). Type = 21 is the pentahedra version of Type = 19 and Type = 22 is the pentahedra version of Type = 20. Using ESORT > 0 in *CONTROL_SOLID will automatically sort out the pentahedra elements (19 to 21 and 20 to 22).
- Add *DEFINE_DE_BY_PART to define control parameters for DES by part ID, including damping coefficient, friction coefficient, spring constant, etc. If defined, it will overwrite the parameters in *CONTROL_DISCRETE_ELEMENT.
- Add new feature for *MAT_030 (*MAT_SHAPE_MEMORY) as optional 3rd card. Curves or tables (strain rate dependency) can be defined to describe plastic loading and unloading behavior.
- New feature for *ELEMENT_BEAM_PULLEY. Beam elements BID1 and BID2 can now both be defined as "0" (zero). In that case, adjacent beam elements are automatically detected. Therefore, the first two beam elements with nodal distance < 10^{-6} to the pulley node (PNID) will be chosen.
- Add new feature to *MAT_ADD_EROSION's damage model GISSMO. By default, damage is driven by equivalent plastic strain. Now, users can optionally define another history variable as driving quantity by setting DMGTYP.
- Add volumetric plastic strain to *MAT_187 as history variable 6.
- Add internal energy calculation for *ELEMENT_BEAM_PULLEY.
- Add viscoplastic option to *MAT_157: new parameter VP on Card 5, Column 6.
- Add new keyword *MAT_ADD_COHESIVE which is intended to make 3D material models available for cohesive elements.
- Add new parameters to *MAT_CABLE_DISCRETE / *MAT_071. MXEPS (Card 2, Column 4) is equal the maximum strain at failure and MXFRC (Card 2, Column 5) is equal to the maximum force at failure
- Add *MAT_124 as potential weld partner material for PROPRUL = 2/3 of *DEFINE_CONNECTION_PROPERTIES.
- Add new material *MAT_TOUGHENED_ADHESIVE_POLYMER (TAPO) or *MAT_252 for epoxy-based, toughened, ductile adhesives.
- Add new option to *MAT_002_ANIS: parameter IHIS on Card 4, Column 8. IHIS = 0: terms C11, C12, … from Cards 1, 2, and 3 are used. IHIS = 1: terms C11, C12, … initialized by *INITIAL_STRESS_SOLID's extra history variables.
- Add new option to *MAT_102. Instead of constant activation energy Q, one can define a load curve LCQ on Card 2, Column 7:

- LCQ.GT.0: Q as function of plastic strain
- LCQ.LT.0: Q as function of temperature

- Add new option to *MAT_071 (MAT_CABLE_DISCRETE_BEAM). New parameter FRACL0 (Card 2, Column 3) is fraction of initial length that should be reached over time period of TRAMP. That means the cable element length gets modified from L0 to FRACL0 × L0 between t = 0 and t = TRAMP.
- Add internal energy calculation for SPR models *CONSTRAINED_INTERPOLATION_SPOTWELD (SPR3) and *CONSTRAINED_SPR2. Their contribution was missing in energy reports like glstat.
- Add new failure model OPT = 11 to *MAT_SPOTWELD/*MAT_100 for beam elements.
- Add three new failure criteria for shell elements to *MAT_ADD_EROSION on optional card 4, columns 6-8:

  - LCEPS12: load curve in-plane shear strain limit vs. element size.
  - LCEPS13: load curve cross-thickness shear strain limit vs. element size.
  - LCEPSMX: load curve in-plane major strain limit vs. element size.

- Add new capability to *MAT_ADD_EROSION damage model GISSMO. Strain rate scaling curve LCSRS can now contain natural logarithm values of strain rates as abscissa values. This is automatically assumed when the first value is negative.
- Add new parameter NHMOD to *MAT_266. The constitutive model for the isotropic part can now be chosen:

  - NHMOD = 0: original implementation (modified Neo-Hooke)
  - NHMOD = 1: standard Neo-Hookeon (as in umat45)

- New keyword *DEFINE_TABLE_MATRIX is an alternative way of defining a table and the curves that the table references from a single unformatted text file, e.g., as saved from an Excel spreadsheet.
- Change long format so that all data fields are 20 columns and each line of input can hold up to 200 columns. In this way, the number of input lines is the same for long format as for standard format.

  - 8 variables per line in long format = 160 columns
  - 10 variables per line in long format = 200 columns

- Add a new option (SOFT = 6) in *CONTACT_FORMING_NODES_TO_SURFACE for blank edge and guide pin contact.
- Add user-defined criteria for mesh refinement (or coarsening) in *CONTROL_REFINE_….

- Add new contact option that currently only works for MPP SINGLE_SURFACE contact with SOFT = 0 or 1. If SRNDE (field 4 of optional card E) is a 1, then free edges of the contact definition will be rounded WITHOUT extending the segments. Rather than having cylindrical caps on the ends of the segments, the "corners" of the squared off thickness are rounded over.
- Add geometric contact entity type -3 "finite cylinder".
- Add irate = 2 to *CONTROL_IMPLICIT_DYNAMICS to turn off rate effects for both implicit and explicit.
- Add quadratic 8-node and 6-node shells (shell formulations 23 and 24).
- Add LOG_LOG_INTERPOLATION option for table defining strain rate effects in *MAT_083, *MAT_181, and *MAT_183.
- Add automatic generation of null shells for quadratic shell contact (*PART_DUPLICATE_NULL_OVERLAY).
- Add beam contact forces to rcforc output (*DATABASE_RCFORC).
- Add SHL4_TO_SHL8 option to *ELEMENT_SHELL to automatically convert 4-node shells to 8-node quadratic shells.
- Add 3-node beam element with quadratic interpolation that is tailored for the piping industry. It includes 12 degrees of freedom, including 6 ovalization degrees of freedom, per node for a total of 36 DOF. An internal pressure can be given that can stiffen and elongate the pipe.
    - ELFORM = 14 in *SECTION_BEAM.
    - *ELEMENT_BEAM_ELBOW.
    - NEIPB in *DATABASE_EXTENT_BINARY to direct output of elbow loop-stresses to d3plot. Otherwise, output goes to ASCII file elbwls.k.
    - Supported by a subset of material models including mats 3, 4, 6, 153, 195.
- Add discrete element option DE to *DATABASE_TRACER.
    - Includes variable RADIUS. average result of all
        - RADIUS > 0: Reports the average result of all DE particles in a spherical volume having radius equal to RADIUS and centered at the tracer.
        - RADIUS < 0: Reports result of the closest particle to the tracer.
    - If a tracer node NID is given, then the tracer moves with this node. The node must belong to a DES.
- Add new options *PART_COMPOSITE_LONG and *ELEMENT_SHELL_COMPOSITE_LONG. In contrast to "COMPOSITE", one integration point is defined per card. This is done to allow for more information, such as "ply id".

- Add support of *MAT_ADD_EROSION option NUMFIP < 0 for standard (non-GISSMO) failure criteria. Only for shells.
- Add viscoplastic behavior to *MAT_157, i.e., parameter LCSS can now refer to a table with strain rate dependent yield curves.
- Add singular finite element with midside nodes for 2D plane strain fracture analysis (ELFORM = 55 in *SECTION_SHELL). This is an 8-noded element and can induce a singular displacement field by moving mid-side nodes to quarter locations.
- If HCONV < 0 in *AIRBAG_PARTICLE, |HCONV| is a curve of heat convection coefficient vs. time.
- Add new option DECOMPOSITION for *AIRBAG_PARTICLE -- MPP only.This will automatically invoke the recommended decomposition commands, *CONTROL_MPP_DECOMPOSITION_BAGREF (if applicable) and *CONTROL_-MPP_DECOMPOSITION_ARRANGE_PARTS, for the bag.
- Add new blockage option for vents in *AIRBAG_PARTICLE:
    - blockage considered
        - .eq.0: no
        - .eq.1: yes
        - .eq.2: yes, exclude external vents
        - .eq.3: yes, exclude internal vents
        - .eq.4: yes, exclude all vents
- Add option in *CONTROL_CPM to consider CPM in the time step size calculation.
- When using *AIRBAG_PARTICLE with IAIR = 2, user should keep mole / particle similar between inflator gas and initial air particles to ensure the correct elastic collision. If different by more than 10%, code will issue warning message and provide the suggested initial air particle number.
- Enable *DEFINE_CURVE_FUNCTION for *SECTION_POINT_SOURCE_MIXTURE and *SECTION_POINT_SOURCE.
- Make *BOUNDARY_PRESCRIBED_MOTION_SET compatible with *CONTROL_REFINE
- Change *BOUNDARY_ACOUSTIC_COUPLING_MISMATCH to rank order opposing acoustic faces and structural segments by proximity, thereby accelerating the preprocessing stage, enhancing reliability and allowing some liberalization of the search parameters.
- Implement hemispherical geometry for particle blast (*DEFINE_PBLAST_GEOMETRY).
- Add explosive type for *PARTICLE_BLAST.

- For particle-based blast *PARTICLE_BLAST:
    - Include random distribution of initial air molecules
    - Modify algorithm to account for the non-thermally-equilibrated state of high velocity gas.
- Improve particle contact method for particle-based blast loading *PARTICLE_-BLAST.
- *CONTACT now works for parts refined using *CONTROL_REFINE_SOLID or *CONTROL_REFINE_SHELL.
- Improve calculation of shell element contact segment thicknesses, particularly at material boundaries.
- MPP: Add output to rcforc file for *CONTACT_AUTOMATIC_TIEBREAK to record the # of nodes tied, and the total tied area.
- MPP: Add calculation of "contact gap" for master side of FORMING contact.
- MPP: Add support for table-based friction (FS = 2.0) to groupable contact.
- Implement splitting-pinball contact, Belytschko & Yeh (1992, 1993). This new contact option is invoked by setting SOFT = 2, SBOPT = 3 and DEPTH = 45. A penetration check method based on LS-PrePost version 4.0 is implemented for the new bilinear-patch-based contact, SOFT = 2, DEPTH = 45 & Q2TRI = 0. The new method provides more accurate intersection information when Q2TRI = 0.
- Add support for birth time for *CONTACT_2D_AUTOMATIC_TIED.
- Improve the segment based single surface contact search for thick segment pairs that are too close together. The code was not working well with triangluar segments. This change affects models with shell segments that have thickness greater than about 2/3 of the segment length.
- Enable segment based quad splitting options to work when shell sets or segment sets are used to define the surface that will be split. This is really a bug fix because there was no check to prevent this and the result was writing past the allocated memory for segment connectivites.
- Allow *CONSTRAINED_INTERPOLATION to use node set to define the independent nodes.
- Add a length unit to the tolerance used for the checking of noncoincident nodes in *CONSTRAINED_JOINTs excluding spherical joints. The old tolerance was 1.e-3. The new tolerance is 1.e-4 times the distance between nodes 1 and 3. The error messages were changed to warnings since this change might otherwise cause existing models to stop running.
- Add d3hsp output for *CONSTRAINED_INTERPOLATION_SPOTWELD (SPR3) and *CONSTRAINED_SPR2. Can be deactivated by setting NPOPT = 1 on *CONTROL_OUTPUT.

- Support NFAIL1 and NFAIL4 of *CONTROL_SHELL in coupled thermal-mechanical analysis, i.e. erode distorted elements instead of error termination.
- PTSCL on *CONTROL_CONTACT can be used to scale contact force exerted on shell formulations 25, 26, 27 as well as shell formulations 2, 16 (IDOF = 3).
- Use SEGANG in *CONTROL_REMESHING to define positive critical angle (unit is radian) to preserve feature lines in 3D tetrahedral remeshing (ADPOPT = 2 in *PART).
- For 3D solid adaptive remeshing including ADPOPT = 2 and ADPOPT = 3 (*PART), the old mesh will be used automatically if the remesher fails generating a new mesh.
- Add option INTPERR on *CONTROL_SHELL (Optional Card 3, Column 8). By default, warning messages INI+143/144/145 are written in case of non-matching number of integration points between *INITIAL_STRESS_SHELL and *SECTION_SHELL. Now with INTPERR = 1, LS-DYNA can terminate with an error.
- Add variable D3TRACE on *CONTROL_ADAPTIVE: The user can now force a plot state to d3plot just before and just after an adaptive step. This option is necessary for tracing particles across adaptive steps using LS-PrePost.
- By putting MINFO = 1 on *CONTROL_OUTPUT, penetration info is written to message files for mortar contact., see also *CONTACT_…. Good for debugging implicit models, not available for explicit.
- Change the default scale factor for binary file sizes back to 70. This value can be changed using "x=" on the execution line. In version R7.0, the default value of x is 4096, and that sometimes leads to difficulty in postprocessing owing to the large size of the d3plot file(s).
- Enable *CONTROL_OUTPUT flag, EOCS, which wasn't having any effect on the shells output to elout file.
- *DATABASE_FSI_SENSOR: Create sensors at solid faces in 3D and at shell sides in 2D.
- *DATABASE_PROFILE: Implement the option DIR = 4 to plot data with curvilinear distributions and the flag UPDLOC to update the profile positions.
- In *CONTROL_SHELL, add options for deletion of shells based on:
    - diagonal stretch ratio (STRETCH)
    - w-mode amplitude in degrees (W-MODE)
- New element formulation ELFORM = 45 in *SECTION_SOLID: Tied Meshfree-enriched FEM (MEFEM). This element is based on the 4-noded MEFEM element (ELFORM = 43, *SECTION_SOLID). Combined with *CONSTRAINED_TIED_NODES_FAILURE, *SET_NODE_LIST and cohesive model, this element can be used to model dynamic multiple-crack propagation along the element boundaries.

- New high order tetrahedron CPE3D10 based on Cosserat Point theory can be invoked by specifying element formulation ELFORM = 16 and combining this with hourglass formulation IHQ = 10. See *SECTION_SOLID and *HOURGLASS.
- Add database D3ACS for collocation acoustic BEM (*FREQUENCY_DOMAIN_ACOUSTIC_BEM) to show the surface pressure and normal velocities.
- Implement biased spacing for output frequencies for random vibration (*FREQUENCY_DOMAIN_RANDOM_VIBRATION).
- Add frequency domain nodal or element velocity output for acoustic BEM (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- Implement boundary acoustic mapping to acoustic BEM in MPP (*BOUNDARY_ACOUSTIC_MAPPING). This is enabled only for segment sets at present.
- Implement panel contribution analysis capability to Rayleigh method (*FREQUENCY_DOMAIN_ACOUSTIC_BEM_PANEL_CONTRIBUTION).
- Implement a scheme to map velocity boundary condition from dense BEM mesh to coarse mesh to speed up the computation (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- Add user node ID for acoustic field points in D3ATV (*FREQUENCY_DOMAIN_ACOUSTIC_BEM). Now D3ATV is given for multiple field points, and multiple frequencies.
- Add database D3ATV for acoustic transfer vector binary plot (*FREQUENCY_DOMAIN_ACOUSTIC_BEM_ATV, *DATABASE_FREQUENCY_BINARY_D3ATV).
- Implement acoustic panel contribution analysis to collocation BEM and dual collocation BEM (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- Enable *FREQUENCY_DOMAIN_MODE in response spectrum analysis (*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM).
- Implement an option to read in user-specified nodal velocity history data for running BEM acoustics (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- Extend Kirchhoff acoustic method to MPP (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- Extend response spectrum analysis to multiple load spectra cases (*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM).
- Add BAGVENTPOP for *SENSOR_CONTROL. This allows user more flexibilty controlling the pop-up of the venting hole of *AIRBAG_HYBRID and *AIRBAG_WANG_NEFSKE
- Add command *SENSOR_DEFINE_FUNCTION. Up to 15 *DEFINE_SENSORs can be referenced in defining a mathematical operation.

- LAYER of *SENSOR_DEFINE_ELEMENT can now be an integer “I” representing the Ith integration point at which the stress/strain of the shell or tshell element will be monitored.
- Add control of *LOAD_MOVING_PRESSURE by using *SENSOR_CONTROL.
- Add thick shells to the ETYPE option list of *SENSOR_DEFINE_ELEMENT.
- Add *CONTROL_MPP_MATERIAL_MODEL_DRIVER in order to enable the Material Model Driver for MPP (1 core).
- Add table input of thermal expansion coefficient for *MAT_270. Supports temperature-dependent curves arranged according to maximum temperature.
- Add table input of heat capacity for *MAT_T07. Supports temperature dependent curves arranged according to maximum temperature.
- Add two more kinematic hardening terms for *MAT_DAMAGE_3/MAT_153, c2 & gamma2.
- Add materials *MAT_CONCRETE_DAMAGE_REL3/*MAT_072R3 and *MAT_-CSCM_CONCRETE/*MAT_159 to Interactive Material Model Driver.
- Enable *MAT_JOHNSON_COOK/*MAT_015 for shell elements to work with coupled structural / thermal analysis.
- Allow *MAT_SOIL_AND_FOAM/*MAT_005 to use positive or negative abscissa values forload curve input of volumetric strains.
- Add *MAT_ACOUSTIC elform = 8 support for pyramid element case using 5-pt integration.
- Add support to *MAT_219 (*MAT_CODAM2) for negative AOPT values which point to coordinate system ID's.
- Modify *MAT_224 so it uses the temperatures from the thermal solution for a coupled thermal-mechanical problem.
- Add alternative solution method (Brent) for *MAT_015 and *MAT_157 in case standard iteration fails to converge.
- Add shell element IDs as additional output to messag file for *MAT_036's warning “plasticity algorithm did not converge”.
- For *MAT_USER_DEFINED_MATERIAL_MODELS, the subroutines crvval and tabval can be called with negative curve / table id which will extract values from the user input version of the curve or table instead of the internally converted “100-point” curve / table.
- In the damage initiation and evolution criteria of *MAT_ADD_EROSION (invoked by IDAM < 0), add the option Q1 < 0 for DETYP = 0. Here, |Q1| is the table ID defining the ufp (plastic displacement at failure) as a function of triaxiality and damage value, i.e., ufp = ufp(eta, D), as opposed to being constant which is the default.

- In *MAT_RHT, ONEMPA = -6 generates parameters in g, cm, and μS and ONEMPA = -7 generates parameters in g, mm, and mS
- In *MAT_SIMPLIFIED_RUBBER/FOAM, STOL > 0 invokes a stability analysis and warning messages are issued if an unstable stretch point is found within a logarithmic strain level of 100%.
- Implement *DATABASE_ALE to write time history data (volume fractions, stresses, …) for a set of ALE elements. Not to be confused with *DATABASE_ALE_MAT.
- Implement *DELETE_PART in small restarts for ALE2D parts.
- Add conversion of frictional contact energy into heat when doing a coupled thermal-mechanical problem for SPH (variable FRCENG in *CONTROL_CONTACT). This option applys to all 3D contact types supported by SPH particles.
- For keyword *DEFINE_ADAPTIVE_SOLID_TO_SPH, add support of explicit SPH thermal solver for the newly generated SPH particles which were converted from solid elements. The temperatures of those newly generated SPH particles are mapped from corresponding solid elements.
- Implement DE to surface tied contact *DEFINE_DE_TO_SURFACE_TIED. The implementation includes bending and torsion.
- Implement keyword *DEFINE_DE_HBOND to define heterogeneous bond for discrete element spheres (DES). DES (*ELEMENT_DISCRETE_SPHERE) with different material models can be bonded.
- Implement keyword *INTERFACE_DE_BOND to define multiple failure models for various bonds within one part or between different parts through the keyword *DEFINE_DE_HBOND.
- Implement *DEFINE_DE_TO_BEAM_COUPLING for coupling of discrete element spheres to beam elements.
- Add variable MAXGAP in *DEFINE_DE_BOND to give user control of distance used in judging whether to bond two DES together or not, based on their initial separation.
- Add IAT = -3 in *CONTROL_REMESHING_EFG, which uses FEM remapping scheme in EFG adaptivity. Compared to IAT = -2, -1, 1, 2, IAT = -3 is faster and more robust but less accurate.
- Add control flag MM in *CONTROL_REMESHING_EFG to turn on/off monotonic mesh resizing for EFG 3D general remeshing (ADPOPT = 2 in *PART).
- *CONTROL_IMPLICIT_BUCKLING - Extend Implicit Buckling Feature to allow for Implicit problems using Inertia Relief. This involves adding the Power Method as a solution technology for buckling eigenvalue problems. Using the power method as an option for buckling problems that are not using inertia relief has been added as well.

- Extend Implicit Buckling to allow for Intermittent extraction by using negative values of NMODE on *CONTROL_IMPLICIT_BUCKLING similar to using negative values of NEIG on *CONTROL_IMPLICIT_EIGENVALUE.
- Extend implicit-explicit switching specified on *CONTROL_DYNAMIC_RELAXATION to allow explicit simulation for the dynamic relaxation phase and implicit for the transient phase.
- New implementation for extracting resultant forces due to joints for implicit mechanics.
- New implementation of extracting resultant forces due to prescribed motion for implicit mechanics.
- Add support for IGAP > 2 in implicit, segment based (SOFT = 2) contact.
- Add constraint-based, thermal nodal coupling for *CONSTRAINED_LAGRANGE_IN_SOLID. HMIN < 0 turns it on.
- Add FRCENG = 2 on CONTROL_CONTACT keyword.
    - if FRCENG = 1, convert contact frictional energy to heat.
    - if FRCENG = 2, do not convert contact frictional energy to heat.
- Add effect of thermal time scaling (TSF in *CONTROL_THERMAL_SOLVER) to 2D contact.
- Add new pfile decomposition region option: partsets. Takes a list of part sets (*SET_PART) from the keyword input and uses them to define a region, e.g., region { partsets 102 215 sy 1000 } This example would take partsets, scale y by 1000, and decompose them and distribute them to all processors.
- Reduce MPP memory usage on clusters.
- Add MPP support for *ELEMENT_SOURCE_SINK.
- Add new pfile options:
    - decomp { d2r_as_rigid }
    - decomp { d2ra_as_rigid }

  which cause materials appearing in "*DEFORMABLE_TO_RIGID" and "*DEFORMABLE_TO_RIGID_AUTOMATIC" to have their computational costs set as if they were rigid materials during the decomposition.
- Add option ISRCOUT to *INCLUDE_STAMPED_PART to dump out the transformed source/stamp mesh.
- *CONTROL_FORMING_OUTPUT: Allow NTIMES to be zero; support birth and death time; support scale factor in curve definition.
- Add a new option (INTFOR) to *CONTROL_FORMING_OUTPUT to control the output frequency of the INTFOR database.

- Add new features (instant and progressive lancing) in *ELEMENT_LANCING for sheet metal lancing simulation.
- Add a new keyword: *CONTROL_FORMING_INITIAL_THICKNESS.
- Add a new option for springback compensation: *INCLUDE_COMPENSATION_ORIGINAL_TOOLS.
- Add a new keyword: *INTERFACE_COMPENSATION_NEW_PART_CHANGE.
- Add a new keyword (*DEFINE_CURVE_BOX_ADAPTIVITY) to provide better control of mesh refinement along two sides of the curve.
- Isogeometric analysis: contact is available in MPP.
- Normalize tangent vectors for local coordinate system for the rotation free isogeometric shells.
- Add support for dumping shell internal energy density for isogeometric shells (*ELEMENT_SHELL_NURBS_PATCH) via interpolation shells.
- Add support for dumping of strain tensor (STRFLG.EQ.1) for isogeometric shells (*ELEMENT_SHELL_NURBS_PATCH) via interpolation shells.
- Add H-field, magnetization and relative permeability to d3plot output.
- *ICFD_INITIAL: Add a reference pressure (pressurization pressure) for when no pressure is imposed on the boundaries.
- Add the initialization of all nodes at once by setting PID = 0.
- Add the non-inertial reference frame implementation defined by the keyword *ICFD_DEFINE_NONINERTIAL.
- Add several new state variables to LSO. Please refer to the LSO manual to see how to print out the list of supported variables.
- Add support for FSI with thick shells.
- 2D shells are now supported for FSI in MPP. In the past only beams could be used in MPP and beams and shells could be used in SMP.
- The keyword ICFD_CONTROL_FSI has a new field to control the sensitivity of the algorithm to find the solid boundaries used in FSI calculations.
- The 2D mesh now generates semi-structured meshes near the boundaries.
- Add heat flux boundary condition using ICFD_BOUNDARY_FLUX_TEMP.
- Add divergence-free and Space Correlated Synthetic Turbulence Inlet Boundary Condition for LES (Smirnov et al.) using *ICFD_BOUNDARY_PRESCRIBED_VEL.
- *ICFD_BOUNDARY_PRESCRIBED_VEL: Add inflow velocities using the wall normal and a velocity magnitude using the 3rd field VAD.
- Add the activation of synthetic turbulence using the 3rd field VAD.

- Add the option to control the re-meshing frequency in both keywords: see *ICFD_-CONTROL_ADAPT_SIZE and *ICFD_CONTROL_ADAPT.
- *ICFD_CONTROL_TURB_SYNTHESIS: control parameters for the synthetic turbulence inflow.
- *ICFD_BOUNDARY_PRESCRIBED_MOVEMESH: Allows the mesh to slide on the boundaries following the cartesian axis.
- Add a PART_SET option for *CESE_BOUNDARY_…_PART cards.
- Bring in more 2D mesh support, both from the PFEM mesher and a user input 2D mesh (via *ELEMENT_SOLID with 0 for the last 4 of 8 nodes).
- Enable the 2D ball-vertex mesh motion solver for the 2D CESE solver.
- Add new input cards:
    - *CESE_BOUNDARY_CYCLIC_SET
    - *CESE_BOUNDARY_CYCLIC_PART
- Add code for 2D CESE sliding boundary conditions.
- Add support in CESE FSI for 2D shells in MPP.
- Add support for CESE FSI with thick shells.
- Add 2D & 2D-axisymmetric cases in the CESE-FSI solver (including both immersed boundary method & moving mesh method) .
- Add the CSP reduced chemistry model with 0D, 2D, and 3D combustion. The 2D and 3D combustion cases couple with the CESE compressible flow solver.
- Add the G-scheme reduced chemistry model only for 0D combustion.
- Add two different reduced chemistry models.
    - The Computational Singular Perturbation (CSP) reduced model is implemented with existing compressible CESE solver. The CSP is now working on 0-dimensional onstant volume and pressure combustion, 2-D, and 3-D combustion problems.
    - The new reduced chemistry model, G-scheme, is implemented, but currently works only 0-dimensional problems such as constant combustors.
- Jobid can now be changed in a restart by including “jobid=“ on the restart execution line. Previously, the jobid stored in d3dump could not be overwritten.
- Part labels (PID) can be up to 8 characters in standard format; 20 characters in long format.
- Labels for sections (SID), materials (MID), equations of state (EOSID), hourglass IDs (HGID), and thermal materials (TMID) can be up to 10 characters in standard format; 20 characters in long format.

- Create bg_switch and kill_by_pid for SMP. Both files will be removed at the termination of the run.
- Increase the overall length of command line to 1000 characters and length of each command line option to 50 characters.
- Increase MPP search distance for tied contacts to include slave and master thicknesses.
- For *CONTACT_AUTOMATIC_…_MORTAR, the mortar contact now supports contact with the lateral surface of beam elements.
- On *CONTACT_..._MORTAR, IGAP.GT.1 stiffens the mortar contact for large penetrations. The mortar contact has a maximum penetration depth DMAX that depends on geometry and input parameters; if penetration is larger than this value the contact is released. To prevent this release, which is unwanted, the user may put IGAP.GT.1 which stiffens the behavior for penetrations larger than 0.5*DMAX without changing the behavior for small penetrations. This should hopefully not be as detrimental to convergence as increasing the overall contact stiffness.
- For initialization by prescribed geometry in dynamic relaxation (IDRFLG = 2, *CONTROL_DYNAMIC_RELAXATION), add an option where displacements are not imposed linearly but rather according to a polar coordinate system. This option was added to accommodate large rotations.
- The flag RBSMS on *CONTROL_RIGID is now active for regular and selective mass scaling to consistently treat interfaces between rigid and deformable bodies
- Remove static linking for l2a as many systems do not have the required static libraries.
- Add IELOUT in *CONTROL_REFINE to handle how child element data is handled in elout (*DATABASE_HISTORY_SOLID and *DATABASE_HISTORY_-SHELL). Child element data are stored if IELOUT = 1 or if refinement is set to occur only during initialization.
- Include eroded hourglass energy in hourglass energy in glstat file to be consistent with KE & IE calculations so that the total energy = kinetic energy + internal energy + hourglass energy + rigidwall energy.
- Remove *DATABASE_BINARY_XTFILE since it is obsolete.
- When using *PART_AVERAGED for truss elements (beam formulation 3), calculate the time step based on the total length of the combined macro-element instead of the individual lengths of each element.
- Enable writing of midside nodes to d3plot or 6- and 8-node quadratic shell elements.
- Write complete history variables to dynain file for 2D solids using *MAT_NULL and equation-of-state.

- Shell formulations 25, 26, and 27 are now fully supported in writing to dynain file (*INTERFACE_SPRINGBACK_LSDYNA).
- Shell formulations 23 (quad) and 24 (triangle) can now be mixed in a single part. When ESORT = 1 in *CONTROL_SHELL, triangular shells assigned by *SECTION_SHELL to be type 23 will automatically be changed to type 24.
- Enable hyperelastic materials (those that use Green's strain) to be used with thick shell form 5. Previously, use of these materials (2, 7, 21, 23, 27, 30, 31, 38, 40, 112, 128, 168, and 189) with thick shell 5 has been an input error.
- Update acoustic BEM to allow using *DEFINE_CURVE to define the output frequencies (*FREQUENCY_DOMAIN_ACOUSTIC_BEM).
- When using *CONTROL_SPOTWELD_BEAM, convert *DATABASE_HISTORY_BEAM to *DATABASE_CROSS_SECTION and *INITIAL_AXIAL_FORCE_BEAM to *INITIAL_STRESS_CROSS_SECTION for the spot weld beams that are converted to hex spot welds.
- Improve output of *INITIAL_STRESS_BEAM data to dynain via *INTERFACE_SPRINGBACK_LSDYNA. Now, large format can be chosen, history variables are written, and local axes vectors are included.
- Update *MAT_214 (*MAT_DRY_FABRIC) to allow fibers to rotate independently.
- Enable regularization curve LCREGD of *MAT_ADD_EROSION to be used with FLD criterion, i.e. load curve LCFLD. Ordinate values (major strain) will be scaled with the regularization factor.
- Modify *MAT_ADD_EROSION parameter EPSTHIN:
    - EPSTHIN > 0: individual thinning for each IP from *z*-strain (as before).
    - EPSTHIN < 0: averaged thinning strain from element thickness (new).
- Enable regularization curve LCREGD of *MAT_ADD_EROSION to be used with standard (non-GISSMO) failure criteria. Users can now define a failure criterion plus IDAM = 0 plus LCREGD = scaling factor vs. element size to get a regularized failure criterion.
- *MAT_ADD_EROSION: equivalent von Mises stress SIGVM can now be a function of strain rate by specifying a negative load curve ID.
- *SECTION_ALE1D and *SECTION_ALE2D now work on multiple processors (SMP and MPP).
- *CONSTRAINED_LAGRANGE_IN_SOLD ctype 4/5 now converts friction energy to heat. Note it only works for ALE elform 12.

Capabilities added September 2013 – January 2015 to create LS-DYNA R8.0:

See release notes (published separately) for further details.

- Add RDT option for *AIRBAG_SHELL_REFERENCE_GEOMETRY.
- LCIDM and LCIDT of *AIRBAG_HYDRID can now be defined through *DEFINE_CURVE_FUNCTION.
- New variable RGBRTH in *MAT_FABRIC to input part-dependent activation time for airbag reference geometry.
- Negative PID of *AIRBAG_INTERACTION considers the blockage of partition area due to contact.
- Enhancements to *AIRBAG_PARTICLE:
    - New blockage (IBLOCK) option for vents.
    - External work done by inflator gas to the structure is reported to glstat.
    - Enhance segment orientation checking of CPM bag and chambers.
    - Allow user to excluded some parts surface for initial air particles.
    - Support compressing seal vent which acts like flap vent.
    - Support Anagonye and Wang porosity equation through *MAT_FABRIC.
    - Add keyword option _MOLEFRACTION.
- Add ID keyword option to *AIRBAG_REFERENCE_GEOMETRY and *AIRBAG_SHELL_REFERENCE which includes optional input of variables for scaling the reference geometry.
- Enable *DEFINE_CURVE_FUNCTION for *AIRBAG_SIMPLE_AIRBAG_MODEL.
- Calculate heat convection (HCONV) between environment and airbag in consistent fashion when TSW is used to switch from a particle airbag to a control volume.
- For *AIRBAG_PARTICLE, add ENH_V = 2 option for vent hole such that two-way flow can occur, i.e., flow with or against the pressure gradient.
- *BOUNDARY_ALE_MAPPING: add the following mappings: 1D to 2D, 2D to 2D, 3D to 3D.
- *SET_POROUS_ALE: new keyword to define the properties of an ALE porous media by an element set. The porous forces are computed by *LOAD_BODY_POROUS.
- *ALE_FSI_SWITCH_MMG: applies also now to 2D.
- *ALE_SWITCH_MMG: new keyword to switch multi-material groups based on criteria defined by the user with *DEFINE_FUNCTION.
- *CONTROL_ALE: Allow PREF (reference pressure) to be defined by materials.
- Implement *ALE_COUPLING_NODAL_DRAG to model the drag force coupling between discrete element spheres or SPH particles and ALE fluids.

- Implement *ALE_COUPLING_RIGID_BODY as an efficient alternative for constraint type coupling between ALE fluids and a Lagrangian rigid body.
- Error terminate if *BOUNDARY_SPC_NODE_BIRTH_DEATH is applied to a node that belongs to a rigid body.
- Modify *BOUNDARY_PRESCRIBED_ORIENTATION_VECTOR to accommodate bodies which undergo no changes in orientation.
- Add a new keyword *BOUNDARY_SPC_SYMMETRY_PLANE.
- Solid part or solid part set is now allowed for *PARTICLE_BLAST.
- Add ambient pressure boundary condition flag BC_P for *PARTICLE_BLAST.
- New command *DEFINE_PBLAST_GEOMETRY allows the high explosive domain for*PARTICLE_BLAST to be defined by various geometric shapes.
- Allow multiple *PARTICLE_BLAST definitions.
- Add *DATABASE_PBSTAT to output particle blast statistics.
- Output the initial volume and initial mass of HE particles and air particles for *PARTICLE_BLAST to d3hsp.
- Add the command *CESE_BOUNDARY_BLAST_LOAD to allow a blast described by the *LOAD_BLAST_ENHANCED command to be used as a boundary condition in CESE.
- Modify the FSI interface reflective boundary condition pressure treatment in some calculations for the moving mesh and immersed boundary solvers.
- Change the CESE derivatives calculation method to use the current values of flow variables.
- Add two new MAT commands for CESE solver, *CESE_MAT_000 and *CESE_-MAT_002.
- Add a non-inertial reference frame solver for fluid and FSI problems using the moving-mesh method.
- For the moving mesh CESE solver, replace the all-to-all communication for conjugate heat and FSI quantities with a sparse communication mechanism.
- Add structural element erosion capability to the immersed boundary method CESE FSI solver (serial capability only).
- Add 2D cyclic boundary conditions capability.
- Add a NaN detection capability for the CESE solver.
- Switch all CESE boundary conditions that use a mesh surface part to define the boundary to use the character string "MSURF" instead of "PART" in the option portion of the keyword name.

- Add missing temperature interpolation in time for imposing solid temperatures as a boundary condition in the CESE solver.
- Optimize the IDW-based mesh motion for the CESE moving mesh solver.
- Treat the input mesh as 3D by default for the CESE solver.
- All of the chemistry features mentioned below are coupled only to the CESE compressible flow solver when 2D or 3D calculations are involved.
- Chemical source Jacobians have been added.
- Introduce *CHEMISTRY_CONTROL_PYROTECHNIC and *CHEMISTRY_PROPELLANT_PROPERTIES for airbag applications. In conjunction with these commands, basic airbag inflator models are implemented.
- The pyrotechnic inflator model using NaN3/Fe2O3 propellant is newly implemented. To connect with the existing ALE airbag solver, two load curves, mass flow rate and temperature, are saved in "inflator_outfile" as a function of time. This model computes three sub-regions: combustion chamber, gas plenum, and discharge tank. Each region can be initialized with different *CHEMISTRY_COMPOSITION models, which means that user can compute Propellant+Gas hybrid mode.
- The following 0-dimensional combustion problems have been improved: constant volume, constant pressure, and CSP.
- For iso-combustion. temperature and species mass fractions as a function of time are displayed on screen and saved in "isocom.csv" to plot with LS-PrePost.
- Another chemical ODE integration method has been implemented.
- The output file of the pyrotechnic inflator is updated so that this file can be read from ALE solver for an airbag simulation.
- 2-D and 3-D TNT gaseous blast explosives, categorized as TBX (thermobaric explosives), are implemented for the Euler equation systems (CESE-only). Also, 3-D TNT blast + aluminum combustion for serial problems is now implemented.
- Implement a mix modeling method for use with CESE solvers.
- Modify *CHEMISTRY-related keyword commands to allow multiple chemistry models in the same problem.
- Add command *CHEMISTRY_MODEL which identifies the files that define a Chemkin chemistry model.
- Modify the following commands such that the files related to the chemistry model have been removed. These commands are only used to select the type of chemistry solver:
    - *CHEMISTRY_CONTROL_CSP
    - *CHEMISTRY_CONTROL_FULL
    - *CHEMISTRY_CONTROL_1D

- Modify *CHEMISTRY_DET_INITIATION where the files related to the chemistry model have been removed, and the Model ID used is inferred through a reference to a chemistry composition ID.
- Modify *CHEMISTRY_COMPOSITION and *CESE_CHEMISTRY_D3PLOT to add model ID.
- Add *CONTACT_TIED_SHELL_EDGE_TO_SOLID for transferring moments from shells into solids.
- Add frictional energy calculation for beams in *CONTACT_AUTOMATIC_GENERAL.
- Enhance ERODING contacts for MPP. The new algorithm uses a completely different approach to determining the contact surface. The old algorithm started from scratch when identifying the exterior of the parts in contact. The new algorithm is smarter about knowing what has been exposed based on what is eroded, and is faster.
- Force EROSOP = 1 for all ERODING type contacts, with a warning to the user if they had input it as 0.
- Add error check in case of a contact definition with an empty node set being given for the slave side.
- Modify output of ncforc (*DATABASE_NCFORC) in order to support output in a local coordinate system.
- For ERODING contacts, reduce memory allocated for segments so each interior segment is only allocated once.
- Add keyword *DEFINE_CONTACT_EXCLUSION (MPP only) to allow for nodes tied in some contacts to be ignored in certain other contacts.
- Rewrite meshing of *CONTACT_ENTITY to use dynamic memory, which removes the previous limit of 100 meshed contact entities. There is now no limit.
- Remove undocumented release condition for MPP's *CONTACT_AUTOMATIC_TIEBREAK, options 5 and greater.
- Add new experimental "square edge" option to select SOFT = 0,1 contacts. This new option applies only to AUTOMATIC_SINGLE_SURFACE and the segment-to-segment treatment of AUTOMATIC_GENERAL, and is invoked by setting SRNDE = 2 on *CONTACT's Optional Card E. This new option does not apply to SOFT = 2; SOFT = 2 square edge option is set using SHLEDG in *CONTROL_CONTACT.
- BT and DT in *CONTACT can be set to define more than one pair of birthtime/death-time for the contact by pointing to a curve or table. These pairs can be unique for the dynamic relaxation phase and the normal phase of the simulation.

- Add EDGEONLY option to *CONTACT_AUTOMATIC_GENERAL to exclude node-to-segment contact and consider only edge-to-edge and beam-to-beam contact.
- VDC defines the coefficient of restituion when variable CORTYP is defined. Available for *CONTACT_AUTOMATIC_NODES_TO_SURFACE, *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE, and *CONTACT_AUTOMATIC_SINGLE_SURFACE; SOFT = 0 or 1 only.
- Enhancements for *CONTACT_AUTOMATIC_GENERAL:
  - Add beam to beam contact option CPARM8 in *PART_CONTACT (MPP only).
  - Add option whereby beam generated on exterior shell edge will be shifted into the shell by half the shell thickness. In this way, the shell-edge-to-shell-edge contact starts right at the shell edge and not at an extension of the shell edge (see OPT2 = 10, CPARM8, MPP Card 1).
- Implement *CONTROL_CONTACT PENOPT = 3 option to *CONTACT_AUTOMATIC_NODES_TO_SURFACE and *CONTACT_ERODING_NODES_TO_SURFACE for SMP.
- Update segment based (SOFT = 2) contact to improve accuracy at points away from the origin. The final calculations are now done with nodal and segment locations that have been shifted towards the origin so that coordinate values are small.
- Enable user defined friction (*USER_INTERFACE_FRICTION; subroutine usrfrc) for MPP contact SOFT = 4.
- Unify automatic tiebreak messages for damage start and final failure. SMP and MPP should now give the same output to d3hsp and messag. This affects *CONTACT_AUTOMATIC_...TIEBREAK, OPTIONs 6, 7, 8, 9, 10, and 11.
- *CONTACT_ADD_WEAR: Associates wear calculations to a forming contact interface whose quantities can be posted in the intfor database file. Adaptivity is supported.
- *CONTACT_..._MORTAR:
  - Detailed warning outputs activated for mortar contact, also clarifies echoed data in d3hsp.
  - Contact thickness made consistent with other contacts in terms of priority between ISTUPD on CONTROL_SHELL, SST on CONTACT and OPTT on PART_CONTACT.
  - Efficency improvement of bucket sort in mortar contact allowing for significant speedup in large scale contact simulations.
- *CONTACT_..._MORTAR, *DEFINE_FRICTION, *PART_CONTACT:

    - Mortar contact supports FS = -1.0, meaning that frictional coefficients are taken from *PART_CONTACT parameters.
    - Mortar contact supports FS.EQ.-2 meaning that friction is taken from *DEFINE_FRICTION.

- *CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR: Using IGNORE < 0 for single surface mortar contact will ignore penetrations of segments that belong to the same part.
- Friction factors are now a function of temperature for *CONTACT_..._THERMAL_FRICTION.
- *SET_POROUS_LAGRANGIAN: new keyword to define the porosity of Lagrangian elements in an element set. The porous forces are computed by *CONSTRAINED_LAGRANGE_IN_SOLID CTYPE = 11 or 12.
- *CONSTRAINED_LAGRANGE_IN_SOLID: CTYPE = 12 is now also available in 2D.
- Add helix angle option for *CONSTRAINED_JOINT_GEARS.
- Change keyword from *CONSTRAINED_BEARING to *ELEMENT_BEARING.
- Enhance explicit to use the implicit inertia relief constraints. This allows implicit-explicit switching for such problems.
- Add new input options to *CONTROL_IMPLICIT_INERTIA_RELIEF:

    - user specified number of nodes
    - user specified list of modes to constrain out.

- Implement *CONSTRAINED_BEAM_IN_SOLID. This feature is basically an overhauled constraint couping between beams and Lagrangian solids that includes features that make it more attractive in some cases than *CONSTRAINED_LAGRANGE_IN_SOLID, for example, in modeling coupling of rebar in concrete.
- Allow *CONSTRAINED_INTERPOLATION to use node set to define the independent nodes.
- Add new feature MODEL.GE.10 to *CONSTRAINED_INTERPOLATION_SPOTWELD (SPR3). This allows parameters STIFF, ALPHA1, RN, RS, and BETA to be defined as *DEFINE_FUNCTIONs of thicknesses and maximum engineering yield stresses of connected sheets.
- Add failure reports for *CONSTRAINED_SPR2.
- Add more d3hsp output for *CONSTRAINED_INTERPOLATION_SPOTWELD and *CONSTRAINED_SPR2. Can be deactivated by setting NPOPT = 1 on *CONTROL_OUTPUT.

- Add option to *CONSTRAINED_JOINT: Relative penalty stiffness can now be defined as function of time when RPS < 0 refers to a load curve. Works for SPHERICAL, REVOLUTE, CYLINDRICAL in explixit analyses.
- Variable MODEL invokes new SPR4 option in *CONSTRAINED_INTERPOLATION_SPOTWELD.
- *CONSTRAINED_JOINT_GEARS: Gear joint now supports bevel gears and similar types, i.e., the contact point does not necessarily have to be on the axis between the gear centers.
- *CONSTRAINED_MULTIPLE_GLOBAL: Support multiple constraints defined on the extra DOFs of user-defined elements.
- Make the *CONTROL_SHELL PSNFAIL option work with the W-MODE deletion criterion for shells.
- New subcycling scheme activated for *CONTROL_SUBCYCLE and *CONTROL_SUBCYCLE_MASS_SCALED_PART. By default the ratio between the largest and smallest time step is now 16 and the external forces are evaluated every time step. The old scheme had a hard wired ratio of 8. The ratios can be optionally changed by *CONTROL_SUBCYCLE_K_L where K is the maximum ratio between time steps for internal forces and L is likewise the ratio for external forces.
- *DATABASE_PROFILE:
    - output kinetic and internal energy profiles,
    - output volume fraction profiles,
    - add a parameter MMG to specify the ALE group for which element data can be output.
- *DATABASE_ALE_MAT: can now use *DEFINE_BOX to compute the material energies, volumes and masses for elements inside boxes (instead of the whole mesh).
- *DATABASE_TRACER_GENERATE: new keyword to create ALE tracer particles along iso-surfaces.
- *DATABASE_FSI: add option to output moments created by FSI forces about each node in a node set. These moments about nodes are reported in dbfsi.
- Add *DATABASE_BEARING to write brngout data pertaining to *ELEMENT_BEARINGs.
- Include eroded hourglass energy in hourglass energy in glstat file to be consistent with KE & IE calculations so that the total energy = kinetic energy + internal energy + hourglass energy + rigidwall energy.
- Add support for new database pbstat (*DATABASE_PBSTAT) for *PARTICLE_BLAST.
    - internal energy and translational energy of air and detonation products

    - force/pressure of air and detonation products for each part

- *DATABASE_EXTENT_INTFOR: New parameter NWEAR on optional card governs the output of wear depth to the intfor database.
- Using CMPFLG = -1 in *DATABASE_EXTENT_BINARY will work just as CMPFLG = 1, except that for *MAT_FABRIC (form 14 and form -14) and *MAT_FABRIC_MAP the local strains and stresses will be engineering quantities instead of Green-Lagrange strain and 2nd Piola-Kirchhoff stress.
- For some materials and elements, thermal and plastic strain tensors can be output to d3plot database, see STRFLG in *DATABASE_EXTENT_BINARY.
- Add option for output of detailed (or long) warning/error messages to d3msg. See MSGFLG in *CONTROL_OUTPUT. Only a few "long" versions of warnings/errors at this time but that list is expected to grow.
- Add two new options for rigid body data compression in d3plot; see DCOMP in *DATABASE_EXTENT_BINARY.
- Add option to write revised legend to jntforc, secforc, rcforc, deforc and nodout files via input flag NEWLEG in *CONTROL_OUTPUT. This helps to avoid confusion over unassigned IDs and duplicated IDs.
- If any input data is encrypted and dynain is requested, the code issues an error message and stops the job.
- Solid part or solid part set is now allowed for *DEFINE_DE_TO_SURFACE_COUPLING.
- Implement *DELETE_PART for Discrete Element Sphere.
- The unit of contact angle changed from radian to degree for *CONTROL_DISCRETE_ELEMENT.
- Implement Archard's wear law to *DEFINE_DE_TO_SURFACE_COUPLING for discrete element spheres. Wear factor is output to DEM binout database.
- Add damping energy and frictional energy of discrete elements to "damping energy" and "sliding interface energy" terms in glstat.
- Introduce a small perturbation to the initial position of newly generated discrete elements for *DEFINE_DE_INJECTION. This allows a more random spatial distribution of the generated particles.
- *INTERFACE_DE_HBOND replaces *INTERFACE_DE_BOND. Used to define the failure models for bonds linking various discrete element (DE) parts within one heterogeneous bond definition (*DEFINE_DE_HBOND).
- *DEFINE_ADAPTIVE_SOLID_TO_DES: Embed and/or transform failed solid elements to DES (*ELEMENT_DISCRETE_SPHERE) particles. The DES particles inherit the material properties of the solid elements. All DES-based features are available through this transformation, including the bond models and contact

algorithms. This command is essentially to DES what *DEFINE_ADAPTIVE_-SOLID_TO_SPH is to SPH particles.

- Add EM orthotropic materials where the electric conductivity is a 3x3 tensor, see new card, *EM_MAT_003.
- Add new keyword family, *EM_DATABASE_... which triggers the output of EM quantities and variables. All EM related ASCII outputs now start with em_***. Keywords are:
    - EM_DATABASE_CIRCUIT
    - EM_DATABASE_CIRCUIT0D
    - EM_DATABASE_ELOUT
    - EM_DATABASE_GLOBALENERGY
    - EM_DATABASE_NODOUT
    - EM_DATABASE_PARTDATA
    - EM_DATABASE_POINTOUT
    - EM_DATABASE_ROGO
    - EM_DATABASE_TIMESTEP
- Add capability to plot magnetic field lines in and around the conductors at given times, see *EM_DATABASE_FIELDLINE. ASCII output files are generated (lspp_fieldLine_xx) and are readable by LSPP in order to plot the field lines. In the future, LSPP will be capable of directly generating the field lines.
- Add EM quantities in *DEFINE_CURVE_FUNCTION:
    - EM_ELHIST for element history (at element center).
    - EM_NDHIST for node history.
    - EM_PAHIST for part history (integrated over the part).
- Add *EM_EOS_TABULATED2 where a load curve defines the electrical conductivity vs time.
- Introduce capability to use the EM solver on (thin) shells: An underlying solid mesh (hexes and prisms) is built where the EM is solved and the EM fields are then collapsed onto the corresponding shell. The EM mat for shells is defined in *EM_MAT_004. This works for EM solvers 1, 2 and 3 and the EM contact is available for shells.
- Add different contact options in the *EM_CONTACT card.
- Add new methods to calculate electric contact resistance between two conductors for Resistive Spot Welding applications (RSW). See *EM_CONTACT_RESISTANCE.
- Add Joule Heating in the contact resistance (*EM_CONTACT_RESISTANCE). The Joule heating is evenly spread between the elements adjacent to the faces in contact.

- Add new circuit types 21 and 22 (see *EM_CIRCUIT) allowing users to put in their own periodic curve shape when using the inductive heating solver. This is useful in cases where the current is not a perfect sinusoidal.
- Provide default values for NCYCLEBEM and NCYCLEFEM (=5000) and set default value of NUMLS to 100 in *EM_CIRCUIT.
- Add two additional formulations, FORM = 3 and 4, to *PART_MODES.
- Add 20-node solid element, ELFORM = 23 in *SECTION_SOLID.
- Add H8TOH20 option to *ELEMENT_SOLID to convert 8-node to 20-node solids.
- Add option SOLSIG to *CONTROL_OUTPUT which will permit stresses and other history variables for multi-integration point solids to be extrapolated to nodes. These extrapolated nodal values replace the integration point values normally stored in d3plot. NINTSLD must be set to 8 in *DATABASE_EXTENT_BINARY when a nonzero SOLSIG is specified. Supported solid formulations are solid elements are: -1, -2, 2, 3, 4, 18, 16, 17, 23.
- Activate contact thickness input from *PART_CONTACT for solids.
- Made many enhancements for *PART_MODES for robustness and MPP implementation.
- Add new cohesive shell element (elform = 29) for edge-to-edge connectivity between shells. This element type takes bending into account and supports MPP and implicit solvers.
- Error terminate with message, STR+1296, if same node is defined multiple times in *ELEMENT_MASS_MATRIX.
- Add support for negative MAXINT option in *DATABASE_EXTENT_BINARY for thick shell elements.
- *ELEMENT_TSHELL: Add "BETA" as option for *ELEMENT_TSHELL to provide an orthotropic material angle for the element.
- Add Rayleigh damping (*DAMPING_PART_STIFFNESS) for triangular shell element types 3 and 17.
- Add new keyword *ELEMENT_BEAM_SOURCE. Purpose: Define a nodal source for beam elements. This feature is implemented for truss beam elements (ELFORM = 3) with material *MAT_001 and for discrete beam elements (ELFORM = 6) with material *MAT_071.
- Add new option to *DEFINE_ELEMENT_DEATH. New variable IDGRP defines a group id for simultaneous deletion of elements.
- Convert cohesive solid type 20 and 22 to incremental formulation to properly handle large rotations. Also use consistent mass.
- Add Smoothed Particle Galerkin (SPG) method for solid analysis (ELFORM = 47) and corresponding keyword option *SECTION_SOLID_SPG. SPG is a true

particle method in Galerkin formulation that is suitable for severe deformation problems and damage analysis.

- Enhance *ELEMENT_LANCING by supporting *PARAMETER, *PARAMETER_-EXPRESSION.
- Add a new feature, *CONTROL_FORMING_TRIMMING, for 2D and 3D trimming of a 3-layer, sandwich laminate blank via *DEFINE_CURVE_TRIM.
- Add 3D normal trimming of solid elements via *DEFINE_CURVE_TRIM_3D.
- Add new features for solid elements 2D trimming *DEFINE_CURVE_TRIM_-NEW:
    - Allow support of arbitrary trimming vector (previously only global z direction was allowed).
    - Improve trimming algorithm for speed up.
    - Allow trimming curves to project to either the top or bottom surface.
- Add a new AUTO_CONSTRAINT option to *CONTROL_FORMING_ONESTEP which is convenient for blank nesting.
- Add new features to *CONTROL_FORMING_SCRAP_FALL. Previously the user was required to define the trimmed blank properly. Now the blank is trimmed by the cutting edge of the trim steel, which is defined by a node set and a moving vector.
- Enhance *CONTROL_FORMING_SCRAP_FALL: Allow the node set (NDSET) on the trim steel edge to be defined in any order.
- Improve *CONTROL_FORMING_ONESTEP:
    - Reposition the initial part before unfolding, using the center element normal.
    - Add a message showing that the initial unfolding is in process.
- Add 2D trimming for solid elements *DEFINE_CURVE_TRIM_NEW, support *DEFINE_TRIM_SEED_POINT_COORDINATES.
- Add *CONTROL_FORMING_AUTOCHECK to detect and fix flaws in the mesh for the rigid body that models the tooling.
- Add new features to *CONTROL_FORMING_UNFLANGING:
    - The incoming flange mesh will be automatically checked for mesh quality and bad elements fixed.
    - Allow thickness offset of deformable flange to use the blank thickness from user's input.
    - Allow definition any node ID in the outer boundary of the flange, to speed up the search when holes are present in the part.
    - Add a new parameter CHARLEN to limit the search region.

- Allow holes to exist in the flange regions.
- Output a suggested flange part after unflanging simulation, with the failed elements deleted from the unflanged part.
- Automatically define a node set and constraints for the flange boundary nodes through the user definition of three nodes.
- Add output of forming thickness, effective strain and trim curves after unflanging simulation.

- Add a new keyword *CONTROL_FORMING_TRIM to replace *ELEMENT_TRIM.
- Add a new keyword: *CONTROL_FORMING_UNFLANGING_OUTPUT: Failed elements are removed to come up with the trim curves.
- Add new features to *INTERFACE_BLANKSIZE_DEVELOPMENT including allowing for trimming between initial and final blank.
- Enhance *CONTROL_FORMING_OUTPUT for controlling the number of states.
- Add *CONTROL_FORMING_TRIM_MERGE to close a user specified (gap) value in the trim curves, so each trim curve will form a closed loop, which is required for a successful trimming.
- Add *CONTROL_FORMING_MAXID to set a maximum node ID and element ID for the incoming dynain file (typically the blank) in the current simulation.
- Enhance *FREQUENCY_DOMAIN_ACOUSTIC_BEM:

  - Update the boundary condition definition for BEM acoustics so that impedance and other user defined boundary conditions can be combined with time domain velocity boundary condition.
  - Implement Burton-Miller BEM to MPP.
  - Implement impedance boundary condition to Burton-Miller BEM.
  - Implement half space option (*FREQUENCY_DOMAIN_ACOUSTIC_BEM_HALF_SPACE) to variational indirect BEM.
  - Implement half space option to acoustic scattering problems.
  - Extend acoustic ATV computation to elements, in addition to nodes.
  - Support element based ATV output in d3atv.
  - Add an option (_MATV) to run modal acoustic transfer vector. Implement MATV to MPP.
  - Implement running BEM Acoustics based on modal ATV (SSD excitation only).

- *FREQUENCY_DOMAIN_ACOUSTIC_FEM: Enable running FEM acoustics based on restarting SSD (*FREQUENCY_DOMAIN_SSD).
- Add *FREQUENCY_DOMAIN_ACOUSTIC_INCIDENT_WAVE to define the incident waves for acoustic scattering problems. To be used with *FREQUENCY_DOMAIN_ACOUSTIC_BEM.

- Add *FREQUENCY_DOMAIN_ACOUSTIC_SOUND_SPEED to define frequency dependent complex sound speed, which can be used in BEM acoustics. By using complex sound speed, the damping in the acoustic system can be considered. To be used with *FREQUENCY_DOMAIN_ACOUSTIC_BEM.
- *FREQUENCY_DOMAIN_FRF: Add mode dependent rayleigh damping to frf and ssd (DMPMAS and DMPSTF).
- *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM:
  - Add output of nodout_spcm and elout_spcm, to get nodal results and element results at user specified nodes and elements.
  - Add von Mises stress computation.
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION: Add semi-log, and linear-linear interpolation on PSD curves (parameter LDFLAG).
- *FREQUENCY_DOMAIN_SSD:
  - Add strain computation.
  - Add parameter LC3 to define the duration of excitation for each frequency.
  - Implement fatigue analysis option (_FATIGUE) based on ssd (sine sweep).
  - Add option to use *DAMPING_PART_MASS and *DAMPING_PART_-STIFFNESS in SSD (DMPFLG = 1).
- Add *MAT_ADD_FATIGUE to define material's SN fatigue curve for application in vibration fatigue and SSD fatigue analysis.
- Add *FREQUENCY_DOMAIN_ACCELERATION_UNIT to facilitate the acceleration unit conversion.
- The icfd_mstats.dat file now outputs the ten worst quality element locations (ICFD solver).
- Add option in *ICFD_CONTROL_OUTPUT allowing terminal output to be written to messag file.
- Add keyword *ICFD_CONTROL_OUTPUT_SUBDOM to output only part of the domain. Available for vtk, dx and gmv formats.
- Add new keyword family, *ICFD_DATABASE_... which triggers the output of ICFD variables. All ICFD related output files now start with icfd_***.
- Add new keyword family *ICFD_SOLVER_TOL_... which allows the user to control tolerances and iteration number for the fractional step solve, the mesh movement solve, and the heat equation solve.
- Curves in *ICFD_BOUNDARY_PRESCRIBED_VEL each provide a scaling factor vs. x, y, or z coordinate, respectively. These scaling factors are applied to the velocity boundary condition.
- Enable free-slip condition for FSI walls (ICFD solver).

- Add new variable IDC to *ICFD_CONTROL_FSI that allows the modification of the scaling parameter that multiplies the mesh size to detect contact.
- Add automatic squeezing to the ICFD elements of the boundary layer when there are two very close surfaces with poor (coarse) mesh resolution.
- Add the initialization for all nodes using *ICFD_INITIAL with PID = 0.
- Add a curve (LCIDSF in *ICFD_CONTROL_TIME) that scales the CFL number as a function of time.
- Add a Heaviside function that allows the solution of simple multiphase problems (ICFD).
- Add the computation of the heat convection coefficient (ICFD).
- Add MPP support for y+ and shear for output (ICFD).
- Add uniformity index (ICFD).
- Add *ICFD_CONTROL_TAVERAGE to control the restarting time for computing the time average values.
- Implement the XMl format for vtk. See *ICFD_CONTROL_OUTPUT.
- Improve temperature stabilization for thermal problems (ICFD).
- Add the Generalized Flow Through Porous Media model monolithically coupled to the incompressible Navier-Stokes model. See keyword *ICFD_MAT for the new options.
- Add the Anisotropic version of the Generalized Flow in Porous Media. See *ICFD_MAT for details.
- Add the capability to define the porous properties using the Pressure-Velocity (P-V) experimental curves. See *ICFD_MAT.
- Compute drag forces around anisotropic/isotropic porous domains (ICFD).
- Extend implicit debug checking when LPRINT = 3 on *CONTROL_IMPLICIT_-SOLVER.
- Add option for implicit dynamic relaxation so that only a subset of parts is active during the dynamic relaxation phase.
- Extend implicit time step control via IAUTO < 0 in *CONTROL_IMPLICIT_AU-TO to linear analysis.
- Add self-piercing rivet capability to implicit (*CONSTRAINED_SPR2, *CON-STRAINED_INTERPOLATION_SPOTWELD).
- Add MTXDMP in *CONTROL_IMPLICIT_SOLVER to dump the damping matrix from implicit mechanics.
- Improve stress and strain computation induced by mode shapes. See MSTRES in *CONTROL_IMPLICIT_EIGENVALUE.

- Add variable MSTRSCL to *CONTROL_IMPLICIT_EIGENVALUE for user control of geometry scaling for the stress computation.
- Make SMP and MPP treatment of autospc constraints consistent. See AUTOSPC on *CONTROL_IMPLICIT_SOLVER.
- Enhance output for *ELEMENT_DIRECT_MATRIX_INPUT (superelements) to describe how they are attached to the LS-DYNA model.
- Enhance superelement computation (*CONTROL_IMPLICIT_MODES or *CONTROL_IMPLICIT_STATIC_CONDENSATION):
    - The computation of the inertia matrix in the presense of rigid bodies is correct.
    - Adjust superelement computation to accept initial velocities.
    - Add null beams for the visualization of superelements.
- Enhance implicit to allow the use of *CONSTRAINED_RIVET in conjunction with axisymmetric shell element problems.
- Add output of performance statistics for the MPP implicit eigensolver to mes0000.
- Add stress computation to modal dynamics (*CONTROL_IMPLICIT_MODAL_DYNAMIC).
- Allow nonsymmetric terms to the assembled stiffness matrix from some implicit features.
- Enhance implicit-explicit switching (IMFLAG < 0 in *CONTROL_IMPLICIT_GENERAL) so that curve |IMFLAG| can be defined using *DEFINE_CURVE_FUNCTION.
- Upgrade the implicit implementation of rack and pinion and screw joints so the joint is driven by relative motion of the assembly instead of absolute motion.
- Add *CONTACT_1D to implicit mechanics.
- *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS is added to study Rotordynamics using the implicit time integrator.
- *MAT_SEATBELT is supported for implicit by introducing bending stiffness.
- *INITIAL_LAG_MAPPING added to initialize a 3D Lagrangian mesh from the last cycle of a 2D Lagrangian simulation.
- *ELEMENT_SHELL_NURBS_PATCH:
    - Add support for dumping of strain tensor and shell internal energy density for isogeometric shells via interpolation shells.
    - Add conventional mass-scaling for isogeometric shells.
- *LOAD_BODY_POROUS: applies also now to 1D and 2D problems.

- Add *LOAD_SEGMENT_CONTACT_MASK, which currently works in MPP only. This feature masks the pressure from a *LOAD_SEGMENT_SET when the pressure segments are in contact with another material.
- Curve LCID of *LOAD_NODE can be defined by *DEFINE_CURVE_FUNCTION.
- *USER_LOADING: pass more data to user-defined loading subroutine loadud including nodal moment, nodal rotational displacement and velocity, and nodal translational mass and rotational inertia.
- Add load curves for dynamic relaxation for *LOAD_THERMAL_VARIABLE.
- *LOAD_SEGMENT_NONUNIFORM, *LOAD_SEGMENT_SET_NONUNIFORM: By specifying a negative load curve ID the applied load becomes a follower force, i.e., the direction of the load is constant with respect to a local coordinate system that rotates with the segment.
- Make several enhancements to *MAT_172.
- *MAT_HYPERELASTIC_RUBBER (*MAT_077_H) has new thermal option for material properties.
- Add *MAT_ORTHOTROPIC_PHASE_CHANGE, *MAT_ELASTIC_PHASE_CHANGE, and *MAT_MOONEY-RIVLIN_PHASE_CHANGE whereby elements change phase as they cross a plane in space.
- Add P1DOFF to 2D seatbelt material, *MAT_SEATBELT_2D, to specify a part ID offset for the internally created 1D seatbelt elements.
- All load curves for *MAT_067 can be defined via *DEFINE_FUNCTION.
- Enhance *MAT_CWM:
    - Add support for shell elements.
    - Add support for hardening curves. Yield stress can be supplied as table depending on plastic strain and temperature.
- Check diagonal elements of C-matrix of *MAT_002/MAT_*OPTION*-TROPIC_ELASTIC and error terminate with message, STR+1306, if any are negative.
- Add a keyword option called MIDFAIL for *MAT_024, (MAT_PIECEWISE_LINEAR_PLASTICITY). When MIDFAIL appears in the keyword, failure by plastic strain will only be checked at the mid-plane. If the mid-plane fails, then the element fails. If there are an even number of integration points through the thickness, then the two points closest to the middle will check for failure and the element fails when both layers fail.
- Enable solid and solid assembly spot welds (*MAT_SPOTWELD) to use the NF parameter for force filtering.
- Add the shear angle in degrees as the first history variable for shell material *MAT_214 (DRY_FABRIC).

- Expand from 2 to 5 the number of additional cards that can be used for the user defined weld failure, OPT = 12 or OPT = 22 on *MAT_SPOTWELD. Now a total of 46 user variables are possible.
- Add a solid spot weld material option in *MAT_SPOTWELD to treat the stress state as uniaxial. This option is available for solid assemblies also.
- Add *MAT_FABRIC form 24 which is a modified version of form 14. The main improvement is that the Poisson's effects work correctly with the nonlinear curves for fiber stress. Also, the output of stress and strain to d3plot are engineering stress and strain instead of 2nd PK stress and Green's strain. Added an option to input curves in engineering stress and strain rather than 2nd PK stress vs. Green's strain. To use this, set DATYP = -2 on *DEFINE_CURVE.
- Increase maximum number of plies from 8 to 24 in a sublaminate with *MAT_CODAM2.
- Add *MAT_THERMAL_CHEMICAL_REACTION to model a material undergoing a chemical reaction such as an epoxy used in manufacturing composite materials.
- *MAT_058:
    - Add option to use nonlinear (elastic) stress-strain curves instead of constant stiffnesses (EA, EB, GAB).
    - Add option to use strain-rate dependent nonlinear (elastic) stress-strain curves instead of constant stiffnesses (EA, EB, GAB).
    - Add option to define proper poisson ratios PRCA and PRCB (also added in *MAT_158).
- Add option to use yield curve or table in *MAT_100 (*MAT_SPOTWELD) for solid elements.
- Add *MAT_157 for solid elements. This includes an optional variable IHIS that invokes *INITIAL_STRESS_SOLID to initialize material properties on an element-by-element basis. This was developed to allow a user to map/initialize anisotropic material properties from an injection molding simulation.
- *MAT_157 (shells):
    - Add anisotropic scale factor for plastic strain rate (VP = 1 only).
    - Improve local stress projection for VP = 1.
    - Add optional variable IHIS, similar to that described for solids above.
- Add strain rate dependence to *MAT_103 for solids via a table (isotropic hardening only).
- *MAT_136 (*MAT_CORUS_VEGTER): Implemented an alternative, implicit plasticity algorithm (define N.lt.0) for enhanced stability.
- *MAT_244 (*MAT_UHS_STEEL):

- In plasticity with non-linear hardening, temperature effects and strain rate effects are now dealt with the same way they are implemented in *MAT_106. In particular, strain rate now refers to the plastic strain rate.
- Allow for the definition of start temperatures for each phase change, for cooling and heating.
- Account for elastic transformation strains, given as a curve wrt temperature.
- Add feature to *MAT_244 for welding simulations. Similar to *MAT_270, material can be initialized in a quiet (ghost) state and activated at a birth temperature.

- Furthermore, annealing is accounted for.
- Modify formula for Pearlite phase kinetics based on Kirkaldy and Venugoplan (1983).
- *MAT_249 (*MAT_REINFORCED_THERMOPLASTIC): Implement new material formulation for shells, which is based on additive split of stress tensor.

  - For the thermoplastic matrix, a thermo-elasto-plastic material is implemented,where the temperature dependence is defined by load curves/tables in the input file.
  - Includes hyperelastic fiber contribution.
  - For any integration point, up to three different fiber directions can be defined. Their (non-linear) response to elongation and shear deformations can also be defined with load curves.
  - Includes input parameters for anisotropic transverse shear stiffness.

- *MAT_T07 (*MAT_THERMAL_CWM): Add HBIRTH and TBIRTH which are specific heat and thermal conductivity, resp., used for time t < TISTART.
- One additional parameter (exponent GAMMA) for B-K law of *MAT_138.
- MAT_187: Speed-up of load curve lookup for curves with many points.
- Add new option "MAGNESIUM" to *MAT_233. Differences between tension and compression are included.
- Add enhanced damage model with crack closure effects to *MAT_104.
- Some improvements for *MAT_075 (BILKHU/DUBOIS_FOAM): Volumetric strain rate can now be averaged over NCYCLE cycles, original input curve LCRATE is instead of a rediscretized curve, and averaged strain rate is stored as history variable #3.
- Add new history variables to *MAT_123: A mixed failure indicator as history variable #10 and triaxiality as #11.
- Decrease memory requirements for *MAT_ADD_EROSION by 50%.
- Add *MAT_098 for tetrahedral solid type 13.

- Add new history variable #8 to *MAT_157 for shell elements: "Anisotropic equivalent plastic strain".
- Add tangent stiffness to *MAT_224 for implicit analyses with solid and shell elements.
- Put internal enery on "plastic strain" location for *MAT_027 solids.
- Add new option *MAT_224: BETA < 0: strain rate dependent amount given by load curve ID = -BETA
- Add new flag to switch off all MAT_ADD_EROSION definitions globally.
- This will be the 1st parameter "MAEOFF" on new keyword *CONTROL_MAT.
- Add option to define a load curve for isotropic hardening in *MAT_135.
- *MAT_CDPM is reimplemented by its original developers (Peter Grassl and Dimitros Xenos at University of Glasgow) for enhanced robustness. A new parameter EFC is introduced governing damage in compression and the bilinear law is exchanged for an exponential one.
- *MAT_3-PARAMETER_BARLAT: HR = 7 is complemented with biaxial/shear hardening curves.
- *MAT_FABRIC_MAP:
    - A stress map material for detailed stress response in fabrics, stress can be prescribed through tables PXX and PYY corresponding to functions of biaxial strain states.
    - A compaction effect due to packing of yarns in compression is obtained by specifying BULKC (bulk modulus) and JACC (critical jacobian for the onset of compaction effect). This results ib increasing pressure that resists membrane elements from collapsing and/or inverting.
    - Strain rate effects can be obtained by specifying FXX and FYY which in effect scales the stress based on engineering strain rate. A smoothing effect is applied by using a time window DT.
    - A hysteresis option TH is implemented for stability, given in fraction dissipated energy during a cycle. Can also depend on the strain state through a table.
- *MAT_GENERAL_HYPERELASTIC_RUBBER, *MAT_OGDEN_RUBBER: By specifying TBHYS.LT.0 a more intuitive interpolation of the damage vs. deviatoric strain energy is obtained. It requires however that the damage and strain energy axes are swapped.
- *MAT_SIMPLIFIED_RUBBER: For AVGOPT.LT.0 the absolute value represents a time window over which the strain rates are averaged. This is for suppressing extensive noise used for evaluating stress from tables.

- *MAT_FABRIC: The bending stiffness contribution in material 34, ECOAT/SCOAT/TCOAT, is now supported in implicit calculations.
- Add *MAT_122_3D which is an extension of *MAT_122 to solid elements. This material model combines orthotropic elastic behavior with Hill's 1948 anisotropic plasticity theory and its applicability is primarily to composite materials.
- MPP groupable tied contact: Output messages about initial node movement due to projection like non-groupable routines do.
- MPP tied contact initialization:
    - Change a tolerance in groupable tied contact bucketsort to match the non-groupable code and fix the slave node thickness used for beam nodes during initial search in non-groupable contact to match groupable contact.
    - Update the slave node from beam thickness calculation for type 9,11, and 12 beams.
- For MPP, set a "last known location" flag to give some indication of where the processors were if an error termination happens. Each writes a message to their own message file. Look for a line that says "When error termination was triggered, this processor was".
- MPP BEAMS_TO_SURFACE contact: Remove "beam" node mass from the penalty stiffness calculation when soft = 1 is used, which matches SMP behavior.
- Make sure the pfile.log file gets created in case of termination due to *CONTROL_-STRUCTURED_TERM.
- Add two new decomposition region-related pfile options "nproc" and "%proc" so that any given decomposition region can be assigned to some subset of all the processors. nproc takes a single argument, which is a specific number of processors. %proc takes a single argument, which is a percentage of processors to use. The old options "lump" and "distribute" are still available and are mapped to the new options thusly:
    - lump => "nproc 1"
    - distribute => "%proc 100.0"
- Tweak MPP beam-to-beam contact routine for better handling of parallel beams.
- MPP: Add support for new solid and shell cost routines, invoked with the pfile option "decomp { newcost }". Will be expanded to include beams, thick shells, etc. in the future.
- MPP contact: add support for IGAP > 2 added to the SINGLE_SURFACE, AUTOMATIC_GENERAL, and *_TO_SURFACE contacts.
- Improve the way MPP computes slave node areas for AUTOMATIC_TIEBREAK contacts (and other that use areas). This should result in less mesh dependency in the failure condition of AUTOMATIC_TIEBREAK contacts.

- MPP: synchronize rigid body flags for shared nodes during rigid-to-deformable switching so that these nodes are handled consistently across processors.
- Add new pfile decomposition region option “partsets”. Takes a list of part sets (SET_PART) from the keyword input and uses them to define a region.
- Apply decomposition transformation (if defined) to:
    - *CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE
    - *CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE
    - *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS.
- Honor TIEDPRJ flag on *CONTROL_CONTACT for MPP groupable tied interfaces.
- Increase initial search distance in MPP tied contact to include slave and master thicknesses.
- Tweak MPP_INTERFERENCE contact to better handle deep initial penetrations.
- MPP: Reorganize how *RIGIDWALL_PLANAR_FORCES is handled, which greatly improves scaling.
- Add new MPP pfile option: directory { local_dirs { path1 path2 path3 } which will assign different local working directories to different processors, to balance the I/O load.
- Miscellaneous MPP enhancements:
    - Restructure and reduce memory usage of 3D ALE searching of neighboring algorithm. Now, the code can handle hundreds of millions ALE elements during decomposition.
    - Support *PARTICLE_BLAST.
    - Support SPH 2D contact.
    - Greatly speed up reconstruction of eroding contact surface, (soft = 0,1) when using large number of cores.
- Add the following options for small restarts:
    - *CHANGE_VELOCITY_GENERATION,
    - *CHANGE_RIGIDWALL_option,
    - PSNFAIL option to *CONTROL_SHELL
- MPP full deck restart: Restore behavior consistent with SMP which is that only the nodes of materials being initialized (not all nodes) are initialized from d3full.
- MPP: add full deck restart support for AUTOMATIC_TIEBREAK contact types.
- Implement *DELETE_PART for seatbelt parts. The associated slipring, retractors and pretensioners will be deactivated as well.
- Add support for MPP restarts with USA coupling.

- Add NREP option to *SENSOR_CONTROL to repeat NREP cycles of switches given on Card 2.
- Implement *SENSOR_CONTROL TYPEs BELTPRET, BELTRETRA and BELTSLIP control the pretensioners, retractors and sliprings of a 2D seatbelt.
- Add function SENSORD to *DEFINE_CURVE_FUNCTION to return the value of a sensor.
- Replace *SENSOR_DEFINE_ANGLE with more general *SENSOR_DEFINE_MISC. MTYPEs include ANGLE, RETRACTOR, RIGIDBODY, and TIME.
- Add rcforc output for *CONTACT_2D_NODE_TO_SOLID (supported for ASCII output only; not binout).
- Add temperature output (when applicable) to sphout file (*DATABASE_SPHOUT).
- Add support of *MAT_ALE_VISCOUS for SPH particles. This allows modeling of non-viscous fluids with constant or variable viscosity, i.e, non-newtonian type fluid using SPH.
- Add support of *EOS for *MAT_272 with SPH particles.
- Add support of *MAT_255, *MAT_126, and *MAT_26 (with AOPT = 2 only) for SPH particles.
- Add new keyword command *SECTION_SPH_INTERACTION: Combined with CONT = 1 in *CONTROL_SPH card, this keyword is used to define the partial interaction between SPH parts through normal interpolation method and partial interaction through the contact option. All the SPH parts defined through this keyword will interact with each other through normal interpolation method automatically.
- Add support for *DATABASE_TRACER for axisymmetric SPH (IDIM = -2 in *CONTROL_SPH).
- ICONT in *CONTROL_SPH now affects *DEFINE_SPH_TO_SPH_COUPLING in the sense of enabling or disabling the coupling for deactivated particles.
- The commands *STOCHASTIC_TBX_PARTICLES and *CHEMISTRY_CONTROL_TBX are now available for use (along with the CESE solver) in TBX-based explosives simulations.
- Multi-nozzle injection mode is implemented for spray injection.
- Add logic to skip thermal computations during dynamic relaxation for a coupled thermal-structual problem (i.e. when SOLN = 2 on the *CONTROL_SOLUTION keyword). This does not affect the use of *LOAD_THERMAL keywords during dynamic relaxation.
- Implement *DEFINE_CURVE_FUNCTION for convection, flux, radiation boundary

- conditions in thermal-only analyses, both 2D and 3D.
- *BOUNDARY_CONVECTION, *BOUNDARY_FLUX, and thermal dynamics are implemented for 20 node brick element.
- Include the reading of thermal data to *INCLUDE_BINARY.
- Allow *DEFINE_FUNCTION_TABULATED to be used in any place that requires a function of 1 variable. Specifically, as a displacement scale factor with *INTERFACE_LINKING_NODE.
- Add new MUTABLE option for *PARAMETER and *PARAMETER_EXPRESSION to indicate that it is OK to redefine a specific parameter even if *PARAMETER_DUPLICATION says redefinition is not allowed. Also, only honor the first *PARAMETER_DUPLICATION card.
- Add functions DELAY and PIDCTL to *DEFINE_CURVE_FUNCTION for simulating PID (proportional-integral-derivative) controllers.
- *DEFINE_TABLE: Add check of table's curves for mismatching origin or end points.
- Update ANSYS library to version 16.0.
- Enhance report of "Elapsed time" in d3hsp.
- Add keyword *INCLUDE_UNITCELL to create a keyword file containing user-defined unit cell information with periodic boundary conditions.
- Add *INCLUDE_AUTO_OFFSET: the node and element IDs of the include file will be checked against IDs of the previously read data to see if there is any duplication. If duplicates are found, they will be replaced with another unique ID.

Capabilities added to create LS-DYNA R9.0:

See release notes (published separately) for further details.

- ***AIRBAG**
  - Disable CPM airbag feature during DR and reactivate in the transient phase.
  - *AIRBAG_WANG_NEFSKE_POP_ID pop venting based on RBIDP is now supported correctly (MPP only).
  - *AIRBAG_INTERACTION:
    - Fixed MPP airbag data sync error to allow final pressure among interacted airbags to reach equilibrium.
  - *AIRBAG_PARTICLE:
    - When IAR = -1 and Pbag or Pchamber is lower than Patm, ambient air will inflate the bag through external vents and also fabric porosity.

- Treat heat convection when chamber is defined.
- Output pres+ and pres- to CPM interface forces file for internal parts.
- Allow IAIR = 4 to gradually switch to IAIR = 2 to avoid instability.
- Allow using shell to define inflator orifice. The shell center and normal will be used as orifice node and flow vector direction.
- Bug fix for porous leakage for internal fabric parts using CPM.
- New feature to collect all ring vents into a single vent in order to correctly treat enhanced venting option. All the vent data will only be output to the first part defined in the part set.
- Evaluate airbag volume based on relative position to avoid truncation. The bag volume becomes independent of coordinate transformation.
- Support explicit/implicit switch and dynamic relaxation for *AIRBAG_PARTICLE.
- Support vent/fabric blockage for CPM and ALE coupled analysis.
- New option in *CONTROL_CPM to allow user defined smoothing of impact forces.
- Fixed bug affecting *AIRBAG_PARTICLE_ID with PGP encryption.

- ***ALE**
  - *ALE_REFERENCE_SYSTEM_GROUP: For prtype = 4, allow the ALE mesh to follow the center of mass of a set of nodes.
  - *CONTROL_ALE:
    - Add a variable DTMUFAC to control the time step related to the viscosity from
    - *MAT_NULL (if zero, the viscosity does not control the time step).
    - Implement a 2D version of BFAC and CFAC smoothing algorithm.
  - *ALE_SMOOTHING: Automatically generate the list of 3 nodes for the smoothing constraints and implement for MPP.
  - *SECTION_ALE2D, *SECTION_SOLID_ALE: Allow a local smoothing controlled by AFAC,...,DFAC.
  - *ALE_SWITCH_MMG: Allow the variables to be modified at the time of the switch.
  - *CONTROL_REFINE_ALE: Add a variable to delay the refinement after removal (DELAYRGN), one to delay the removal after the refinement (DELAYRMV), and one to prevent any removal in a certain radius around latest refinements (RADIUSRMV).
  - *ALE_STRUCTURED_MESH: Implemented structured ALE mesh solver to facilitate rectilinear mesh generation and to run faster.

- ***BOUNDARY**
  - *BOUNDARY_AMBIENT_EOS: Implement *DEFINE_CURVE_FUNCTION for the internal energy and relative volume curves.

- *BOUNDARY_AMBIENT: Apply ambient conditions to element sets.
- Fix for adaptivity dropping SPCs in some cases (MPP only).
- Added conflict error checking between rigid body rotational constraints and joints between rigid bodies (*CONSTRAINED_JOINT) with *BOUNDARY_PRESCRIBED_ORIENTATION.
- The first rigid body of the prescribed orientation cannot have any rotational constraints. Only spherical joints or translational motors can be used between the two rigid bodies of the prescribed orientation. For now explicit will be allowed to continue with these as warnings.Implicit will terminate at end of input checking.
- Instead of error terminating with warning message, STR+1371, when *BOUNDARY_PRESCRIBED_MOTION and *BOUNDARY_SPC are applied to same node and dof, issue warning message, KEY+1106, and release the conflicting SPC.
- Fix erroneous results if *SET_BOX option is used for *BOUNDARY_PRESCRIBED_MOTION.
- Fix *BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID for MPP. It may error terminate or give wrong results if more than one of this keyword are used.
- Fix segmentation fault when using *BOUNDARY_PRESCRIBED_ORIENTATION with vad = 2, i.e. cubic spline interpolation.
- Fix incorrect behavior if multiple *BOUNDARY_SPC_SYMMETRY_PLANE, i.e. > 1, are used.
- Fix incorrect motion if *BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL is on a rigid part which is merged with a deformable part that has been switched to rigid using *DEFORMABLE_TO_RIGID.
- Fix incorrect external work when using *BOUNDARY_PRESCRIBED_MOTION with or without_RIGID option. The dof specified in *BPM was not considered when computing the external work. Also, when multiple *BPM applied to the same node/rigid body with different dof may also cause incorrect computation of external work.
- Fix incorrect velocities when using *BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL and *INITIAL_VELOCITY_RIGID_BODY for rigid bodies.
- Implement check for cases where *MAT_ACOUSTIC nodes are merged with structural nodes on both sides of a plate element and direct the user to the proper approach to this situation - *BOUNDARY_ACOUSTIC_COUPLING.
- *BOUNDARY_ACOUSTIC_COUPLING with unmerged, coincident node coupling now implemented in MPP.
- MPP logic corrected so *MAT_ACOUSTIC and *BOUNDARY_ACOUSTIC_COUPLING features may be used with 1 MPP processor.
- Fixed bug for *BOUNDARY_PRESCRIBED_MOTION if part label option is used.

- **BLAST**
    - Improve *LOAD_BLAST_ENHANCED used with ALEPID option in *LOAD_BLAST_SEGMENT:
    - Rearrange the ambient element type 5 and its adjacent element into same processor to avoid communications.
    - Eliminate several n-by-n searches for segment set and ambient type 5 with its neighboring elements to speed up the initialization.
    - Change the name of keyword *DEFINE_PBLAST_GEOMETRY to *DEFINE_PBLAST_HEGEO. Both names will be recognized.

- ***CESE (Compressible Flow Solver)**
    - Modified the CESE moving mesh CHT interface condition calculation to deal with some occasional MPP failures that could occur with mesh corner elements.
    - Improved the CESE spatial derivatives approximation in order to bring better stability to the CESE solvers.
    - The 3D SMP and MPP CESE immersed boundary solvers now work with structural element erosion.
    - A new energy conservative conjugate heat transfer method has been added to the following 2D and 3D CESE Navier-Stokes equation solvers:
        - Fixed mesh (requires use of *CESE_BOUNDARY_CONJ_HEAT input cards)
        - Moving mesh FSI
        - Immersed boundary FSI
    - Prevent the fluid thermal calculation from using too short a distance between the fluid and structure points in the new IBM CHT solvers.
    - In the under resolved situation, prevent the CHT interface temperature from dipping below the local structural node temperature.
    - Add detection of blast wave arrival at CESE boundary condition face first sensing the leading edge of the pulse (used with *LOAD_BLAST_ENHANCED).
    - Set CESE state variable derivatives to more stable values for the blast wave boundary condition.
    - Corrected time step handling for the CESE Eulerian conjugate-heat transfer solver. This affected only the reported output time.
    - Added CESE cyclic BC capability to the moving mesh CESE solver.
    - Fixed some issues with 2D CESE solvers where the mesh is created via *MESH cards.
    - For the CESE solver coupled with the structural solver (FSI), corrected the time step handling.

- For the CESE mesh motion solvers, and the ICFD implicit ball-vertex mesh motion solver, added a mechanism to check if all of the imposed boundary displacements are so small that it is not necessary to actually invoke the mesh motion solver. This is determined by comparing the magnitude of the imposed displacement at a node with the minimum distance to a virtual ball vertex (that would appear in the ball-vertex method). The relative scale for this check can be input by the user via field 4 of the *CESE_CONTROL_-MESH_MOV card.
- Changed the NaN check capability for the CESE solvers to be activated only upon user request. This is input via a non-zero entry in field 7 of the *CESE_-CONTROL_SOLVER card.
- Much like the ICFD solver, added a mechanism to adjust the distance used by the contact detection algorithm for the *CESE_BOUNDARY_FSI cards, as well as the new moving mesh conjugate heat transfer solvers. This is available through field 6 of the *CESE_CONTROL_SOLVER card.
- Added a correction to the moving mesh CESE solver geometry calculation.
- Corrected the initial time step calculation for both the 2D and 3D moving mesh CESE solvers.
- For the moving mesh CESE solver, replaced the all-to-all communication for fsi quantities with a sparse communication mechanism.

- ***CHEMISTRY**

  - The immersed boundary FSI method coupled with the chemistry solver is released.

    - Only Euler solvers, both in 2D and 3D, are completed with full chemistry.
    - Using this technique, CESE FSI Immerged Boundary Method coupled to the chemistry solver can be applied to high speed combustion problems such as explosion, detonation shock interacting with structures, and so on.
    - Some examples are available on our ftp site.

- ***CONTACT**

  - Fix MPP groupable contact problem that could in some cases have oriented the contact surfaces inconsistently.
  - Fix bug in *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIED_-WELD.
  - Fix seg fault when using *CONTACT_AUTOMATIC_SINGLE_SURFACE_-TIED with consistency mode, .i.e. ncpu < 0, for SMP.
  - Fix false warnings, SOL+1253, for untied nodes using *CONTACT_AUTO-MATIC_SURFACE_TO_SURFACE_TIEBREAK and *CONTACT_AUTO-MATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK.

- Fix *CONTACT_TIED_SHELL_EDGE_TO_SURFACE when rigid nodes are not tied even when ipback = 1. This applies to SMP only.
- Issue warning if SOFT = 4 is used with an unsupported contact type, and reset it to 1.
- Change "Interface Pressure" report in intfor file from abs(force/area) to -force/area, which gives the proper sign in case of a tied interface in tension.
- Increase MPP contact release condition for shell nodes that contact solid elements in SINGLE_SURFACE contact.
- Fix for MPP IPBACK option for creating a backup penalty-based tied contact.
- Fix for MPP orthotropic friction in contact.
- Fix for MPP *CONTACT_SLIDING_ONLY that was falsely detecting contact in some cases.
- Skip constraint based contacts when computing the stable contact time step size.
- Add error trap if node set is input for slave side of single surface contact.
- MPP: some fixes for constrained tied contact when used with adaptivity. The behavior of the slave nodes in adaptive constraints was not correct if they were also master nodes of a tied interface. This has been fixed, and support for the rotations required for CONTACT_SPOTWELD have also been added.
- MPP: update to AUTOMATIC_TIEBREAK option 5 to release the slave nodes (and report them as having failed) when the damage curve reaches 0.
- Fix made to routine that determines the contact interface segments, which was not handling pentahedral thick shell elements correctly.
- MPP: fix for strange deadlock that could happen if a user defines a *CONTACT_FORCE_TRANSDUCER that has no elements in it and so gets deleted.
- MPP contact: add support for *DEFINE_REGION to define an active contact region. Contact occurring outside this region is ignored. This is only for MPP contact types:

  - AUTOMATIC_SINGLE_SURFACE
  - AUTOMATIC_NODES_TO_SURFACE
  - AUTOMATIC_SURFACE_TO_SURFACE
  - AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

- MPP fix for table based friction in non-groupable contact.
- MPP: add frictional work calculation for beams in *CONTACT_AUTOMATIC_GENERAL.
- Added new option "FTORQ" for contact. Currently implemented only for beams in *CONTACT_AUTOMATIC_GENERAL in MPP. Apply torque to the nodes to compensate for the torque introduced by friction. Issue error message when users try to use SOFT = 2/DEPTH = 45 contact for solid elements.

- R-adaptivity, ADPTYP = 7 in *CONTROL_ADAPTIVE, is now available for SMP version of *CONTACT_SURFACE_TO_SURFACE, _NODES_TO_SURFACE, _AUTOMATIC_SURFACE_TO_SURFACE, and _AUTOMATIC_NODES_TO_SURFACE (SOFT = 0 or 1 only).
- The options AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE has been added to model composite processing. The same option may be used to model certain types of lubrication, and AUTOMATIC_SURFACE_TO_SURFACE_LUBRICATION may be used instead of the COMPOSITE option for clarity. (The two keyword commands are equivalent.)
- Added AUTOMATIC_SURFACE_TO_SURFACE_TIED_WELD to model the simulation of welding. As regions of the surfaces are heated to the welding temperature and come into contact, the nodes are tied.
- Added *CONTACT_TIED_SHELL_EDGE_TO_SOLID. This contact transmits the shell moments into the solid elements by using forces unlike the SHELL_EDGE_TO_SURFACE contact with solid elements. This capability is easier for users than *CONSTRAINED_SHELL_TO_SOLID. The input is identical to *CONTACT_TIED_SHELL_EDGE_TO_SURFACE (except for the keyword).
- Fix incorrect motion of displayed rigidwall between 0.0 < time < birth_time when birth time > 0.0 for *RIGIDWALL_GEOMETRIC_FLAT_MOTION_DISPLAY. The analysis was still correct. Only the displayed motion of the rigidwall is incorrect.
- Fix corrupted intfor when using parts/part sets in *CONTACT_AUTOMATIC_.... This affects SMP only.
- Fix incorrect stonewall energy when using *RIGIDWALL_PLANAR_ORTHO.
- Fix unconstrained nodes when using *CONTACT_TIED_SURFACE_TO_SURFACE_CONSTRAINED_OFFSET resulting in warning message, SOL+540. This affects SMP only.
- Fix spurious repositioning of nodes when using *CONTACT_SURFACE_TO_SURFACE for SMP.
- Enable MAXPAR from optional card A to be used in *CONTACT_TIED_SURFACE_TO_SURFACE. It was originally hard-coded to 1.07.
- The shells used for visualisation of *RIGIDWALL_PLANAR_MOVING_DISPLAY and *RIGIDWALL_PLANAR_MOVING_DISPLAY in d3plot were not moving with the rigidwall. This is now fixed.
- Fix incorrect frictional forces if ORTHO_FRICTION is used in *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE.
- Fix seg fault when using *CONTACT_ENTITY and output to intfor file with MPP, i.e. s = intfor in command line.
- Fix ineffective birth time for *CONTACT_TIED_NODES_TO_SURFACE.
- Fix untied contacts when using *CONTACT_TIED... with *MAT_ANISOTROPIC_ELASTIC_PLASTIC/*MAT_157.
- Fix MPP hang up when using *CONTACT_ENTITY.

- Allow *CONTACT_AUTOMATIC_GENERAL to use MAXPAR from contact optional card A instead of using the hard coded value of 1.02. This will better detect end to end contact of beams. This applies to SMP only.
- Fix *CONTACT_TIED_SHELL_EDGE_TO_SURFACE for SMP which ignores MAXPAR in contact optional card A.
- Fix seg fault when using *CONTACT_GUIDED_CABLE.
- Fix segmentation fault when using *CONTACT_AUTOMATIC_SINGLE_-SURFACE_TIED in consistency mode, i.e. ncpu < 0 in command line, for SMP.
- Fix incorrect contacts when using *CONTACT_AUTOMATIC_GENERAL_-INTERIOR for beams with large differences in thickness and when the thinner beams are closer to each other than to the thicker beams. Affects SMP only.
- Fixed force transducers with MPP segment-based contact when segments are involved with multiple 2 surface force transducers. The symptom was that some forces were missed for contact between segments on different partitions.
- Fixed an MPP problem in segment-based contact that cased a divide by zero during the bucket sort. During an iteration of the bucket sort, all active segments were somehow in one plane which was far from the origin such that a dimension rounded to zero. The fix for this should affect only this rare case and have no effect on most models.
- Fixed thermal MPP segment-based contact. The message passing of thermal energy due to friction was being skipped unless peak force data was written to the intfor file.
- Fixed MPP segment based implicit contact. A flaw in data handling caused possible memory errors during a line search.
- Fixed implicit dynamic friction for segment-based contact. For sliding friction, the implicit stiffness was reduced to an infinitesimal value. Also, the viscous damping coefficient is now supported for implicit dynamic solutions.
- Fixed segment-based contact when the data has all deformable parts that are switched to rigid at the start of the calculation and then switched back to deformable prior to contact occurring. A flaw was causing contact to be too soft. This is now corrected.
- Fixed a flaw in segment-based contact with DEPTH = 25 that could allow penetration to occur.
- Improved edge-to-edge contact checking (DEPTH = 5, 25, 35) and the sliding option (SBOPT = 4, 5) in areas where bricks have eroded when using segment based eroding contact.
- Improved the initial penetration check (IGNORE = 2 on *CONTROL_CONTACT) of segment-based contact to eliminate false positives for shell segments. Previously, the search was done using mid-plane nodes and the gap or penetration adjusted to account for segment thicknesses after. The new way projects the nodes to the surface first and uses the projected surface to

measure penetration. For brick segments with zero thickness there should be no difference. For shell segments, the improved accuracy will me more noticeable for thicker segments.

- Improved segment-based contact when SHAREC = 1 to run faster when there are rigid bodies in the contact interface.
- Fixed a possible problem during initialization of segment-based contact. Options that use neighbor segment data such as the sliding option and edge-to-edge checking could access bad data if the same nodes were part of both the slave and master surfaces. This would not be a normal occurrence but could happen.
- Updated segment-based contact to improve accuracy at points away from the origin. The final calculations are now done with nodal and segment locations that have been shifted towards the origin so that coordinate values are small.
- The reporting of initial penetrations and periodic intersection reports by segment-based contact was corrected for MPP solutions which were reporting incorrect element numbers.
- Fixed memory errors in 2D automatic contact initialization when friction is used.
- Fixed 2D force transducers in the MPP version which could fail to report master surface forces. Also fixed 2 surface 2D force transducers when the smp parallel consistency option is active.
- Fixed *CONTACT_2D_AUTOMAITC_SINGLE_SURFACE and SURFACE_TO_SURFACE which could exhibit unpredictable behavior such as a force spike or penetration.
- Fixed a serious MPP error in the sliding option of *CONTACT_2D_AUTOMATIC that could lead to error termination.
- Fixed a problem with birth time for *CONTACT_2D_AUTOMATIC_TIED when used with sensor switching. Also, fixed a problem in the contact energy calculation that could lead to abnormal terminations. Finally, I made the process of searching for nodes to tie more robust as some problem was found with nodes being missed.
- Fixed a 2D automatic contact bug that occurred if a segment had zero length. An infinite thickness value was calculated by A/L causing the bucket sort to fail.
- Added support for *CONTACT_ADD_WEAR for smp and mpp segment based (SOFT = 2) contact. This option enables wear and sliding distance to be measured and output to the intfor file.
- Added support to segment-based contact for the SRNDE parameter on optional card E of *CONTACT.
- Added support to segment based eroding contact for SBOXID and MBOXID on card 1 of *CONTACT.
- Added support for *ELEMENT_SOURCE_SINK used with segment-based contact. With this update, inactive elements are no longer checked for contact.

- Added a segment-based contact option to allow the PSTIFF option on *CONTROL_CONTACT to be specified for individual contact definitions. The new parameter is PSTIFF on *CONTACT on optional card F, field 1. Prior to this change, setting PSTIFF on *CONTROL_CONTACT set all contact to use the alternate penalty stiffness method. With this update, PSTIFF on *CONTROL_CONTACT now sets a default value, and PSTIFF on card F can be used to override the default value for an individual contact interface.
- Added support for REGION option on optional card E of *CONTACT when using segment based, SOFT = 2 contact. This works for all supported keywords, SMP and MPP.
- Added master side output in the MPP version for 2-surface force transducers when used with segment based (soft = 2) contact.
- Added contact friction energy to the sleout database file for
  - 2D_AUTOMATIC_SURFACE_TO_SURFACE and
  - 2D_AUTOMATIC_SINGLE_SURFACE contact.
- Enabled segment-based contact (SMP and MPP) to work with type 24 (27-node) solid elements.
- Enabled the ICOR parameter on *CONTACT, optional card E to be used with segment based (SOFT = 2) contact.
- Fixed output to d3hsp for *CONTACT_DRAWBEAD using negative curve ID for LCIDRF
- Add slave node thickness and master segment thickness as input arguments to the *USER_INTERFACE_FRICTION subroutine usrfrc (SMP).
- Forming mortar contact can now run with deformable solid tools and honors ADPENE to account for curvatures and penetrations in adaptive step. This applies to h- as well as r-adaptivity.
- Single surface and surface-to-surface mortar contact accounts for rotational degrees of freedom when contact with beam elements. This allows for beams to "roll" on surfaces and prevents spurious friction energy to be generated when in contact with rotating parts.
- Maximum allowable penetration in forming and automatic mortar contacts is hereforth .5*(tslav+tmast)*factor where tmast = thickness of slave segment and tmast = thickness of master segment. The factor is hardwired to 0.95, but is subject to change. Prior to this it was .5*tslav, which seems inadequate (too small) in coping with initial penetrations in automotive applications using standard modeling approaches.
- Up to now, mortar contact has only acted between flat surfaces, now account is taken for sharp edges in solid elements (the angle must initially be larger than 60 degrees), may have to increase the corresponding stiffness in the future.
- When solid elements are involved in mortar contact the default stiffness is increased by a factor of 10. This is based on feedback from customers indicating that the contact behavior in those cases has in general been too soft.

This may change the convergence characteristics in implicit but the results should be an improvement from earlier versions.

- The OPTT parameter on *PART_CONTACT for the contact thickness of beams is now supported in mortar contact.
- *CONTACT_ADD_WEAR: A wear law, Archard's or a user defined, can be associated with a contact interface to assess wear in contact. By specifying WTYPE < 0 a user defined wear subroutine must be written to customize the wear law. For the Archard's wear law, parameters can depend on contact pressure, relative sliding velocity and temperature. Contacts supported are *CONTACT_FORMING_SURFACE_TO_SURFACE, *CONTACT_-FORMING_ONE_WAY_TO_SURFACE and *CONTACT_AUTOMATIC_-SURFACE_TO_SURFACE. To output wear data set NWEAR = 1 or NWEAR = 2 on *DATABASE_EXTENT_INTFOR. If NWEAR is set to 2 then the sliding distance is output to the intfor file, in addition to the wear depth. Otherwise only wear depth is output. Also, the parameter NWUSR specifies the number of user wear history variables to be output in case a user defined wear routine is used. By specifying CID (contact interface id) to a negative number, the wear depth will couple to the contact in the simulation in the sense that the penetration is reduced with wear. The effect is that contact pressure will be redistributed accordingly but is only valid for relatively small wear depths. A formulation for larger wear depths lie in the future which will require modification of the actual geometry.
- Fixed bug affecting *CONTACT_RIGID_NODE_SURFACE (broken at rev. 86847). The bug was in reading *NODE_RIGID_SURFACE.
- A bug fix in *CONTACT_DRAWBEAD_INITILIZE. - The bug was caused by a sudden increase in effective strain after the element passed the drawbead. When the increase in strain is too big, the search algorithm was not working reasonably in the material routine. At the drawbead intersection point, an element could be initialized twice by two bead curves, and cause abnormal thickness distribution.
- Fix a bug in *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_SMOOTH which removes the limitation that the contact must be defined by segment set.
- SMOOTH option does not apply to FORMING_SURFACE_TO_SURFACE contact. When the SMOOTH option is used, we now write a warning message and disregard the SMOOTH option.

- ***CONSTRAINED**

  - *CONSTRAINED_LAGRANGE_IN_SOLID: Implement *CONSTRAINED_-LAGRANGE_IN_SOLID_EDGE in 2D.
  - Fixed bug in *DAMPING_RELATIVE. If the rigid part PIDRB is the slave part in *CONSTRAINED_RIGID_BODIES, the damping card did not work correctly. There is a work-around for previous LS-DYNA versions: set

PIDRB to the master part in *CONSTRAIEND_RIGID_BODIES, not the slave part.

- *CONSTRAINED_RIGID_BODY_INSERT: This keyword is for modeling die inserts. One rigid body, called slave rigid body, is constrained to move with another rigid body, called the master rigid body, in all directions except for one.
- A variety of enhancements for *CONSTRAINED_INTERPOLATION.
  - Enhanced the error message when nodes involved in the constraint have been deleted.
  - Removed printing of 0 node ID in MPP.
  - Added a warning if there are too many (now set at 1000) nonzeroes in a constraint row for *CONSTRAINED_INTERPOLATION or *CONSTRAINED_LINEAR to protect implicit's constraint processing. These constraints will be processed differently in future releases. We modified the constraint processing software to robustly handle constraint rows with thousands of nonzero entries. We added error checking for co-linear independent nodes as these constraints allow singularities in the model.
- Improved implicit's treatment of the constraints for *CONSTRAINED_BEAM_IN_SOLID.
- Added error checking on the values of the gear ratios in *CONSTRAINED_JOINTS.
- *CONSTRAINED_BEAM_IN_SOLID:
  - Thick shell elements supported.
  - Wedge elements supported.
  - Debonding law by user-defined subroutine (set variable AXFOR > 1000).
  - Debonding law by *DEFINE_FUNCTION (set variable AXFOR < 0).
- Error terminate with message, SOL+700, if CIDA and CIDB is not defined for *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED.
- Fix incorrect constraints on rotary dof for adaptivity.
- Fix incorrect motion if NRBF = 2 in *DEFORMABLE_TO_RIGID_AUTOMATIC and if any of the *CONSTRAINED_NODAL_RIGID_BODY nodes belongs to a solid element.
- Fix input error when using large load curve ID for FMPH, FMT, FMPS in card 3 of *CONSTRAINED_JOINT_STIFFNESS with GENERALIZED or TRANSLATIONAL options.
- Fix seg fault if using tables for FMPH of *CONSTRAINED_JOINT_STIFFNESS and if the angle of rotation is less than the abscissa of the table or load curves.

- Fixed a problem with *CONSTRAINED_BEAM_IN_SOLID when used in a model that also uses segment based eroding contact in the MPP version. This combination now works.
- Improved the precision of spot weld constraints (*CONSTRAINED_SPOTWELD) to prevent possible divide by zeroes when the inertia tensor is inverted. This affects the single precision version only.
- Fix for damage function in *CONSTRAINED_INTERPOLATION_SPOTWELD, MODEL = 2.
- Add some user-friendly output (rigid body ID) to d3hsp for *CONSTRAINED_NODAL_RIGID_BODY_INERTIA.
- Add new option to *CONSTRAINED_SPR2 to connect up to 6 shell element parts (metal sheets) with only one rivet location node. This is invoked by defining extra part IDs for such a multi-sheet connection.
- Add more flexibility to *CONSTRAINED_SPR2: Load curve function exponent values originally hardwired as "8" can now be defined with new input parameters EXPN and EXPT.
- Fixed bug wherein the joint ID in *CONSTRAINED_JOINT_COOR was read incorrectly.
- Fixed duplicate ID for *CONSTRAINED_SPOTWELD, ..._NODE_SET, _POINTS and _SPR2.
- Fix keyword reader for SPR4 option in *CONSTRAINED_INTERPOLATION_SPOTWELD, where BETA2 was replaced by BETA3.
- Significantly reduce the memory demand in the initialization stage of *CONSTRAINED_MULTIPLE_GLOBAL for implicit analysis.
- The unit cell mesh and constraint generated by *INCLUDE_UNITCELL now supports job ID.

- ***CONTROL**

  - Terminate and print error KEY+1117 for cases that use *INCLUDE_TRANSFORM in 3d r-adaptvity. More work is needed to make this combination work.
  - Changed SOL+41 message ("reached minimum step") from an error to a warning and terminate normally. This message is triggered when the DTMIN criterion set in *CONTROL_TERMINATION is reached.
  - Fixed bug in which h-adaptivity missed some ADPFREQ-based adaptations when IREFLG < 0 (*CONTROL_ADAPTIVE).
  - Fixed bug: MS1ST in *CONTROL_TIMESTEP causes non-physical large mass and inertia on Nodal Rigid Bodies if Dynamic Relaxation is active. The error occurs at the start of the transient solution. The mass can become very large, so the model may appear to be over-restrained.
  - Add new input check for curves. After rediscretizing curves, check to see how well the original values can be reproduced. If the match is poor, write out See variable CDETOL in *CONTROL_OUTPUT.

- Added the ability to specify unique values LCINT for each curve, which override the value set in *CONTROL_SOLUTION. Note: the largest value of LCINT that appears will be used when allocating memory for each load curve, so a single large value can cause significant increases in the memory required for solution.
- The DELFR flag in *CONTROL_SHELL has new options for controlling the deletion of shell elements. This feature is aimed at eliminating single, detached elements and/or elements hanging on by one shared node.
- Fix spurious deletion of elements when using TSMIN.ne.0.0 in *CONTROL_TERMINATION, ERODE = 1 in *CONTROL_TIMESTEP and initialized implicitly in dynamic relaxation.
- Fix spurious error, STR+755, if using *DAMPING_FREQUENCY_RANGE with *CONTROL_ADAPTIVE.
- Add new feature to *CONTROL_SOLUTION, LCACC, to truncate load curve to 6 significant figures for single precision & 13 significant figures for double precision. The truncation is done after applying the offset and scale factors.
- Fix "*** termination due to mass increase ***' error when using mass scaling with *ELEMENT_MASS_PART.
- Fix input error 'node set for nodal rigid body # not found' when using *PART_INERTIA with *CONTROL_SUBCYCLE.
- Fixed the negative DT2MS option on *CONTROL_TIMESTEP for thick shell types 5, 6, and 7.
- Fixed bug in *CONTROL_CHECK_SHELL if PSID.lt.0 (part set ID) is used
- Add new option NORBIC to *CONTROL_RIGID to bypass the check of rigid body inertia tensors being too small.
- Add new option ICRQ to *CONTROL_SHELL for continuous treatment of thickness and plastic strain across element edges for shell element types 2, 4, and 16 with max. 9 integration points through the thickness.
- Add new option ICOHED to *CONTROL_SOLID. If this value is set to 1, solid cohesive elements (ELFORM 19-22) will be eroded when neighboring (nodewise connected) shell or solid elements fail.
- Beam release conditions are now properly supported in selective mass scaling, see IMSCL on *CONTROL_TIMESTEP.
- Modified MSGMAX in *CONTROL_OUPUT: MSGMAX is the maximum number of each error/warning message

  - > 0 number of message to screen output, all messages written to messag/d3hsp
  - < 0 number of messages to screen output and messag/d3hsp
  - = 0 the default is 50

- Fix bugs in 3D solid adaptivity (*CONTROL_ADAPTIVE, ADPTYP = 7) so that the solid adaptivity will still work when there are any of the following in the model:

- thick shells (*SECTION_TSHELL),
- massless nodes,
- *LOAD_SEGMENT_*{OPTION}*.

◦ Added PARA = 2 to *CONTROL_PARALLEL which activates consistent force assembly in parallel for SMP. An efficient parallel algorithm is implemented for better performance when the consistency flag is turned on. It shows better scaling with more cpus. This option is overridden by parameter "para=" on the execution line.

- **DEM (Discrete Element Method)**

◦ Added output of following DES history variables to d3plot:

- nodal stress and force
- pressure
- density
- force chain
- damage calculation when *DEFINE_DE_BOND is defined

◦ Added output of following DES history variables to demtrh (*DATABASE_-TRACER_DE):

- coordination number
- porosity and void ratio
- stress
- pressure
- density

◦ Output ASCII format for demrcf if BINARY.eq.3.
◦ Implement gauss distribution of DE sphere radius for *DEFINE_DE_INJECTION. The mean radius is 0.5 × (rmin + rmax) and standard deviation is 0.5 × (rmax - rmin).
◦ For DE sphere, implement the stress calculation for REV (Representative Elementary Volume) using *DATABASE_TRACER_DE and specific RADIUS.
◦ Add *BOUNDARY_DE_NON_REFLECTING for defining non-reflecting boundary conditions for DE spheres.
◦ For *CONTROL_DISCRETE_ELEMENT, add the option to create the liquid bridge if the initial distance between two DE spheres is smaller than predefined gap.
◦ Added *DATABASE_DEMASSFLOW, see *DEFINE_DE_MASSFLOW_-PLANE, for measuring the mass flow of DE spheres through a surface. The surface is defined by part or part set. Output file is 'demflow'.
◦ Add *DEFINE_DE_INJECTION_ELLIPSE, to define a circular or elliptical injection plane.

- Add *DEFINE_PBLAST_AIRGEO for *PARTICLE_BLAST which defines initial geometry for air particles.
- Add DEM stress calculation when coupling with segment (*DEFINE_DE_TO_SURFACE_COUPLING).
- Fix error in demtrh file output (Windows platform only).

- **EFG (Element Free Galerkin)**

  - Fix bug for ELFORM = 41 implicit when there are 6-noded/4-noded elements.

- ***ELEMENT**

  - Fix a 2d seatbelt bug triggered by having both 1d and 2d seatbelts, and a 1d pretensioner of type 2, 3 or 9.
  - Fix MPP bug initializing multiscale spot weld in the unexpected case where the spot weld beam is merged with the shells rather than tied via contact.
  - Fix bug for *INCLUDE_UNITCELL.
  - *CONTROL_REFINE_...: Implement the parent-children transition in *CONTACT_2D_SINGLE_SURFACE when a shell refinement occurs.
  - Fix error traps for *ELEMENT_SEATBELT_..., for example, error termination due to convergence failure in retractors. These error traps worked but could lead to a less graceful termination than other LS-DYNA error traps.
  - Correct calculation of wrap angle in seatbelt retractor.
  - Add MPP support for *ELEMENT_LANCING.
  - *ELEMENT_SEATBELT:

    - Fix an MPP belt bug that can happen when buckle pretensioner is modeled as a type-9 pretensioner.
    - 2D belt and 1D belt now can share the same *MAT_SEATBELT.
    - The section force for 2d belt is recoded to provide more robust and accurate results.
    - The loading curve LLCID of *MAT_SEATBELT can be a table defining strain-rate dependent stiffness curve.
    - IGRAV of *ELEMENT_SEATBELT_ACCELEROMETER can be a curve defining gravitation flag as a function of time.

  - Add *NODE_THICKNESS to override shell nodal thickness otherwise determined via *SECTION_SHELL, *PART_COMPOSITE, or *ELEMENT_SHELL_THICKNESS.
  - Fix input error when using *DEFINE_ELEMENT_DEATH with BOXID > 0 for MPP.
  - Implement subcycling for thick shells.
  - Fix ineffective *DEFINE_HAZ_PROPERTIES when solid spotwelds and hex spotweld assemblies are both present.

- Fix incorrect beta written out for *ELEMENT_SHELL_BETA in dynain file when *PART_COMPOSITE keyword is present in the original input.
- Fix NaN output to elout_det and spurious element deletion if NOD-OUT = STRAIN or STRAIN_GL or ALL or ALL_GL.
- Fix incorrect reading of TIME in card 3 of *ELEMENT_SEATBELT_SENSOR SBSTYP = 3 when long = s in command line.
- *PART_COMPOSITE: Increased the explicit solution time step for thin shell composite elements. The existing method was overly conservative. The new method is based on average layer stiffness and density.
- In conjunction with the above change in composite time step calculation, increase nodal inertia in the rare cases of *PART_COMPOSITE in which the bending stability is not satisfied by the membrane stability criterion. The inertia is only increased in the cases where it is necessary; for most models this change has no effect, but this can occur in the case of sandwich sections with stiffer skins around a less stiff core.
- Corrected rotational inertia of thin shells when layers have mixed density and the outer layers are more dense than inner layers. The fix will mostly affect elements that are very thick relative to edge length.
- Fixed default hourglass control when the *HOURGLASS control card is used but no HG type is specified. We were setting to type 1 instead of 2. Also, fixed the default HG types to match the User's Manual for implicit and explicit.
- Fixed the part mass that was reported to d3hsp when *ELEMENT_SHELL_-SOURCE_SINK is used. The inactive elements were being included causing too high mass.
- Prevent inactive shell elements (from *ELEMENT_SHELL_SOURCE_SINK) from controlling the solution time step.
- Fixed the reported strain tensor in elements created by *ELEMENT_-SHELL_SOURCE_SINK when strain output is requested. The history was being retained from the previous elements with the same ID.
- Fixed torsion in linear beam form 13. A failure to add the torsional moment at node 2 caused an inability to reach equilibrium in the torsional mode.
- Fixed solid element 4 so that rigid body translation will not cause strain and stress due to round-off error.
- Mixed parallel consistency when used with solid element type 20. A buffer was not being allocated leading to a memory error.
- Changed the MPP behavior of discrete beams (ELFORM = 6) when attached to elements that fail. They were behaving like null beams, in the sense that it was possible for beam nodes to become dead due to attached elements failing, and discrete beams would be no longer visualized even if the beams themselves had not failed. With this change, the MPP discrete beams now behave like other beams in that the beams have to fail before they are re-moved. MPP and SMP behavior is now consistent.
- Improved the precision of the type 2 Belytschko Schwer resultant beam to prevent energy growth in single precision.

- Fixed the NLOC option on *SECTION_SHELL for the BCIZ triangle elements (ELFORM = 3) and the DKT triangle elements (ELFORM = 17). The offset was scaled by the solution time step so typically the offset was much smaller than expected.
- Fixed elout stress output for shell element forms 23 and 24. The in-plane averaging was incorrect causing wrong output.
- Changed *ELEMENT_TSHELL so that both the COMPOSITE and BETA options can be read at the same time. Prior to the fix, only the first one would be read.
- Fixed all thick shells to work with anisotropic thermal strains which can be defined by *MAT_ADD_THERMAL. Also, this now works by layer for layered composites.
- Fixed implicit solutions with thick shells with *MAT_057 when there are also solid elements in the model that use *MAT_057. Thick shells support only the incremental update of the F tensor but a flag was set incorrectly in the material model.
- Fixed *MAT_219 when used with thick shell types 3, 5, and 7. A failure to initialize terms for the time step caused a possible wrong time step.
- Fixed orthotropic user defined materials when used with thick shell elements. The storing of the transformation matrix was in the wrong location leading to wrong stress and strain.
- For thick shell composites that use element forms 5 and 7, the user can now use laminated shell theory along with the TSHEAR = 1 on *SECTION_TSHELL to get a constant shear stress through the thickness with a composite.
- Fixed the initialization of *MAT_CODAM2/*MAT_219 when used with thick shell forms 3, 5, or 7. The 3D thick shell routine uses only 2 terms for the transformation and therefore needs unique initialization of the transformation data.
- Fixed thick shell types 3 and 5 when used in implicit solutions with *MATs 2, 21, 261, and 263. The material constitutive matrix for *MATs 2 and 21 was not rotated correctly causing wrong element stiffness. The constitutive matrix for *MATs 261 and 263 was not orthotropic. Also, for *MAT_021, type 5 thick shell needed some material terms defined to correct the assumed strain.
- Fixed thick shell forms 3 and 5 when used in implicit solutions with non-isotropic materials. The stiffness matrix was wrong due to incorrect transformations.
- Also, fixed the implicit stiffness of thin and thick shells when used with laminated shell theory by assumed strain (LAMSHT = 3,4,5 on *CONTROL_ACCURACY). Elements were either failing to converge or converging more slowly due to the failure to adjust the stiffness matrix to be consistent with the assumed strain.
- Added support for *ELEMENT_SHELL_SOURCE_SINK to form 2 elements with BWC = 1 on *CONTROL_SHELL.

- Fixed the s-axis and t-axis orientation of beam spot welds in the MPP version when those beam weld elements are defined with a 3rd node. The 3rd node was being discarded prior to initializing the beam orientation so the s and t-axes were being randomly assigned as if the 3rd node had not been assigned. The effect on solutions is likely fairly minimal since beam material is isotropic and failure typically is too but may not be.
- Added Rayleigh damping (*DAMPING_PART_STIFFNESS) for thick shell formulations 1, 2, and 6. Previously, it was available for only the thick shells that call 3D stress updates, (forms 3 and 5), but now it is available for all thick shell formulations.
- Added new SCOOR options for discrete beam section 6 (*SECTION_-BEAM). A flaw was found in how the discrete beam accounts for rigid body rotation when SCOOR = -3, -2, +2, and +3. A correction for this is made and introduced as new options, SCOOR = -13, -12, +12, and +13. A decision was made to leave the existing options SCOOR = -2, +2, -3 and +3 unchanged so that legacy data could run without changes.
- Enabled the ELFORM 18 linear DKT shell element to work with *PART_-COMPOSITE and with an arbitrary number of through thickness integration points. It was limited to a single material and 10 Gauss points.
- Added the possibility to write *ELEMENT_SOLID_ORTHO into dynain file if requested. To activate this, add OPTCARD to *INTERFACE_SPRING-BACK and set SLDO = 1.
- Refine characteristic length calculation for 27-node solid (ELFORM 24). This change may increase the time step substantially for badly distorted elements.
- Implement selective reduced integration for 27-node solid (ELFORM 24).
- Allow part sets to be used in *DEFORMABLE_TO_RIGID_AUTOMATIC. Either PID is defined negative or "PSET" is set in column 3 (D2R) or 2 (R2D).
- Add new option STRESS = 2 to *INCLUDE_STAMPED_PART: no stresses and no history variables are mapped with that setting.
- New keyword *PART_STACKED_ELEMENTS provides a method to define and to discretize layered shell-like structures by an arbitrary sequence of shell and/or solid elements over the thickness.
- The geometric stiffness matrix for the Belytschko beam element type 2 has been extended to include nonsymmetric terms arising from nonzero moments. Provides "almost" quadratic convergence, still some terms missing to be added in the future. Also support a strongly objective version activated by IACC on *CONTROL_ACCURACY.
- The geometric stiffness for the Hughes-Liu element type 1 is fixed.
- Fix parsing error in *SECTION_BEAM_AISC.

- **EM (Electromagnetic Solver)**

  - Add the new EM 2d axi solver in SMP and MPP for EM solver 1 (eddy current). It is coupled with the mechanics and thermal solvers.

- The new EM 2d can be used with RLC circuits on helix/spiral geometries using *EM_CIRCUIT_CONNECT.
- Add EM contact into new EM 2d axi, in SMP and MPP.
- Add *EM_BOUNDARY support in new EM 2d axi solver.
- Introduce scalar potential in new EM 2d axi. The 2d axi can also be coupled with imposed voltage.
- Add new keyword *EM_CIRCUIT_CONNECT to impose linear constraints between circuits with imposed currents in 3d solvers. This allows for example to impose that the current in circuit 1 is equal to the current in circuit 2 even if the 2 corresponding parts are not physically connected.
- Add *EM_VOLTAGE_DROP keyword to define a voltage drop between 2 segment sets. This voltage drop constraint is coupled to the contact constraint so that the contact (voltage drop = 0) has priority over the *EM_VOLTAGE_DROP constraint.
- Add *EM_CONTROL_SWITCH_CONTACT keyword to turn the EM contact detection on and off.
- NCYCLBEM/NCYCLFEM in *EM_SOLVER_... can now be different than 1 when EM_CONTACT detected.
- Add RLC circuit for type 3 solver (resistive heating).
- Add computation of mutuals/inductances in 2d axi for output to em_circuit.dat
- Add criteria on autotimestep calculation when RLC circuit used to take into account RLC period.
- Fix keyword counter in d3hsp.
- Better and clearer output to terminal screen.
- Support jobid for EM ascii file outputs.

- **Forming**

  - Improvements to trimming:

    - *DEFINE_CURVE_TRIM_NEW: if trim seed node is not defined, we will search a seed node based on nodes from the sheet blank and the inside/outside flag definition for the trimming curves.
    - Map strain tensors to triangular elements after trimming.

  - Add a new function to the trim of solid elements in normal (3-D) trimming case, related to *DEFINE_CURVE_TRIM_3D. If the trimming curve is close to the bottom side, set TDIR = -1. If the trimming curve is close to the upper side, set TDIR = 1.
  - Add to *ELEMENT_LANCING. Allow parametric expression for variables END and NTIMES.
  - A bug fix for *CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET: Fix distance calculation error when the target mesh is too coarse.
  - Improvements to springback compensations:

- Output the new trimming curve with *DEFINE_CURVE_TRIM_3D (previously *DEFINE_CURVE_TRIM), so that it can be easily converted to IGES curve by LS-PrePost. or used in another trimming calculation.
- Output each curve to IGES format in the following name format: newcurve_scp001.igs, newcurve_scp002.igs, newcurve_scp003.igs, etc.
- Output change in file "geocur.trm". This update will allow change from *DEFINE_CURVE_TRIM(_3D, _NEW), whatever is used for input.

- Add a new keyword: *DEFINE_FORMING_CONTACT to facilitate the forming contact definitions.
- Add a new keyword *DEFINE_FORMING_CLAMP, to facilitate clamping simulation.
- A new feature in mesh fusion, which allows a moving box to control the fusion, only if the center of the elements is inside the box can the elements be coarsened. Can be used in conjunction with *DEFINE_BOX_ADAPTIVE.
- Add a new feature to *DEFINE_BOX_ADAPTIVE: Moving box in adaptivity, useful in roller hemming and incremental forming.
- In mesh coarsening, if the node is defined in a node set, the connected elements will be kept from being coarsened. Previously, only *SET_NODE_LIST was supported. Now option *SET_NODE_GENERAL is allowed.
- Add a new function: mesh refinement for sandwich part. The top and bottom layers are shell elements and the middle layer is solid elements. Set IFSAND to 1 in *CONTROL_ADAPTIVE.

  - Applies to both 8-noded and 6-noded solid elements.
  - Map stress and history variables to the new elements.

- New features related to blank size development *INTERFACE_BLANKSIZE_DEVELOPMENT:

  - Add *INTERFACE_BLANKSIZE_SYMMETRIC_PLANE to define symmetric plane in blank size development
  - Add *INTERFACE_BLANKSIZE_SCALE_FACTOR. For each trimming, different scale factors can be used to compensate the blanksize. This is especially useful when the inner holes are small. Includes an option of offset the target curve which is useful if multiple target curves (e.g., holes) and formed curves are far from each other.
  - Allow target curve to be outside of the surface of the blank.
  - Add sorting to the mesh so the initial mesh and the formed mesh do not need to have the same sequence for the nodes.
  - Add a new variable ORIENT, set to "1" to activate the new algorithm to potentially reduce the number of iterations with the use of *INTERFACE_BLANKSIZE_SCALE_FACTOR (scale = 0.75 to 0.9).

- Fix smooth problem along calculated outer boundary.
- Automatically determine the curve running directions (IOPTION = 2 and -2 now both give the same results).
- Accept parameteric expression.

- A bug fix for springback compensation: *INCLUDE_COMPENSATION_-SYMMETRIC_LINES Fix reading problem of free format in the original coding.
- Add a new keyword *CONTROL_FORMING_BESTFIT. Purpose: This keyword rigidly moves two parts so that they maximally coincide. This feature can be used in sheet metal forming to translate and rotate a spring back part (source) to a scanned part (target) to assess spring back prediction accuracy. This keyword applies to shell elements only.
- Improvements to *CONTROL_FORMING_AUTOCHECK:

  - When IOFFSET = 1, rigid body thickness is automatically offset, based on the MST value defined in *CONTACT_FORMING_ONE_WAY_-SURFACE.
  - Add new variable IOUTPUT that when set to 1 will output the offset rigid tool mesh, and the new output tool file is: rigid_offset.inc. After output the simulation stops. See R9.0 Manual for further details.
  - When both normal check and offset are used, small radius might cause problem for offsetting. The new modification will check the normal again after offsetting the tool
  - When outputting the rigid body mesh, output the bead nodes also.
  - Changes to *CONTROL_FORMING_AUTOCHECK when used together with SMOOTH option: check and fix rigid body bad elements before converting the master part ID to segment set id to be used by SMOOTH option.
  - Set IOUTPUT.eq. 3 to output rigid body mesh before and after offset.
  - Fix problems offseting a small radius to a even smaller radius.
  - Remove T-intersection.

- For *CONTROL_IMPLICIT_FORMING, fix output messages in d3hsp that incorrectly identified steps as implicit dynamic when they were actually implicit static.
- Improve *CONTROL_FORMING_UNFLANGING:

  - Automatically calculate CHARLEN, so user does not need to input it anymore.
  - Allow nonsmooth flange edge.
  - Instead of using preset value of 0.4 (which works fine for thin sheet metal), blank thickness is now used to offset the slave node (flanges) from the rigid body (die).

- ***FREQUENCY_DOMAIN**
  - *FREQUENCY_DOMAIN_RANDOM_VIBRATION: Fixed a bug in dumping d3psd binary database, when both stress and strain are included.
  - *FREQUENCY_DOMAIN_SSD_ERP: Implemented the Equivalent adiated Power (ERP) computation to MPP.
  - *FREQUENCY_DOMAIN_ACOUSTIC_BEM:
    - Enabled running dual collocation BEM based on Burton-Miller formulation (METHOD = 4) with vibration boundary conditions provided by Steady State Dynamic analysis (*FREQUENCY_DOMAIN_SSD).
    - Added exponential window function for FFT (FFTWIN = 5).
    - Implemented a new forward and backward mixed radix FFT.
    - Implemented acoustic computation restart from frequency domain boundary conditions, in addition to time domain boundary conditions (RESTRT = 1).
    - Enabled out-of-core velocity data storage, to solve large scale problems.
    - Implemented option HALF_SPACE to Rayleigh method (METHOD = 0) to consider acoustic wave reflection.
    - Added velocity interpolation to take care of mismatching between acoustic mesh and structural mesh (*BOUNDARY_ACOUSTIC_MAPPING), for the case that the boundary conditions are provided by Steady State Dynamic analysis.
    - Added weighted SPL output to acoustic computation (DBA = 1,2,3,4).
    - Implemented radiated sound power, and radiation efficiency computation to collocation BEMs (METHOD = 3,4). Added new ASCII xyplot databases Press_Power and Press_radef to save the sound power and radiation efficiency results.
    - Enabled using both impedance and vibration (velocity) boundary conditions in acoustic simulation.
  - *FREQUENCY_DOMAIN_ACOUSTIC_FEM:
    - Added weighted SPL output to FEM acoustics (DBA = 1,2,3,4).
    - Implemented option EIGENVALUE to perform acoustic eigenvalue analysis; added ASCII database eigout_ac to save acoustic eigenvalue results; added binary plot database d3eigv_ac to save acoustic eigenvectors.
    - Enabled consideration of nodal constraints in acoustic eigenvalue analysis.
    - Enabled FEM acoustic analysis with frequency dependent complex sound speed.
    - Implemented pressure and impedance boundary conditions.

- *FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT:
  - Added this keyword to 1) generate acoustic field points as a sphere or plate mesh (options SPHERE and PLATE), or 2) define acoustic field points mesh based on existing structure components (options PART, PART_SET and NODE_SET) so that user can get fringe plot of acoustic pressure and SPL. The results are saved in binary plot database d3acs (activated by keyword *DATABASE_FREQUENCY_BINARY_-D3ACS).
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION:
  - Changed displacement rms output in d3rms to be the displacement itself, without adding the original nodal coordinates.
  - Implemented von mises stress PSD computation in beam elements.
  - Implemented fatigue analysis with beam elements.
  - Added strain output to binary plot databases d3psd and d3rms, and binout database elout_psd.
  - Added initial damage ratio from multiple loading cases (INFTG > 1).
- *FREQUENCY_DOMAIN_SSD:
  - Implemented option ERP to compute Equivalent Radiated Power. It is a fast and simplified way to characterize acoustic behavior of vibrating structures. The results are saved in binary plot database d3erp (activated by keyword *DATABASE_FREQUENCY_BINARY_D3ERP), and ASCII xyplot files ERP_abs and ERP_dB.
  - Implemented fatigue analysis based on maximum principal stress and maximum shear stress.

- **ICFD (Incompressible Flow Solver)**
  - *ICFD_BOUNDARY_FSWAVE: Added a boundary condition for wave generation of 1st order stokes waves with free surfaces.
  - *ICFD_DATABASE_DRAG_VOL: For computing pressure forces on volumes ID (useful for forces in porous domains), output in icfdragivol.dat and icfdragivol.#VID.dat.
  - *ICFD_CONTROL_DEM_COUPLING: Coupling the ICFD solver with DEM particles is now possible.
  - *ICFD_CONTROL_MONOLITHIC: Added a monolithic solver (=1) which can be selected instead of the traditional fractional step solver (=0).
  - *ICFD_CONTROL_POROUS: This keyword allows the user to choose between the Anisotropic Generalized Navier-Stokes model (=0) or the Anisotropic Darcy-Forchheimer model (=1) (for Low Reynolds number flows). The Monolithic solver is used by default for those creeping flows.
  - *ICFD_CONTROL_TURBULENCE:

- Modified existing standard k-epsilon.
- Added Realizable k-epsilon turbulence model.
- Added Standard 98 and 06 Wilcox and Menter SST 03 turbulence models.
- Added Several laws of the wall.
- Added Rugosity law when RANS turbulence model selected.

◦ *ICFD_MODEL_POROUS:

- Added Porous model 5 for anistropic materials defined by P-V experimental curves.
- Added porous model 6 for moving domain capabilities for Porous Media volumes using load curves for permeabilities directions.
- Added porous model 7 for moving domain capabilities for Porous Media volumes using ICFD_DEFINE_POINT for permeabilities directions.
- Added porous model 9 for a new Anisotropic Porous Media flow model (PM model ID = 9): It uses a variable permeability tensor field which is the result of solid dynamic problems. The model reads the solid mesh and the field state and maps elemental permeability tensor and solid displacements to the fluid mesh.

◦ *ICFD_MODEL_NONNEWT:

- Added a few models for non newtonian materials and temperature dependant viscosity :
  - model 1 : power law non newtonian (now also temperature dependant)
  - model 2 : carreau fluid
  - model 3 : cross fluid
  - model 4 : herschel-bulkley
  - model 5 : cross fluid II
  - model 6 : temperature dependant visc (sutherland)
  - model 7 : temperature dependant visc (power law)
  - model 8 : load curve dependant visc, model 8 is especially interesting since a DEFINE_FUNCTION can be used (for solidification applications).

◦ *ICFD_SOLVER_TOL_MONOLITHIC: Used to define atol, rtol, dtol and maxits linear solver convergence controls of the monolithic NS time integration

◦ *MESH_BL: Added support for boundary layer mesh creation by specifying the thickness, number of layers, first node near the surface and the strategy to use to divide and separate the elements inside the BL adding.

- *ICFD_BOUNDARY_PRESCRIBED_VEL: Added the support of DEFINE_-FUNCTION making the second line of the keyword obsolete.
- *ICFD_CONTROL_TIME: Min and Max timestep values can be set.
- *ICFD_DATABASE_DRAG:
  - Added frequency output.
  - Added option to output drag repartition percentage in the d3plots as a surface variable.
- *ICFD_CONTROL_IMPOSED_MOVE: This keyword now uses *ICFD_-PART and *ICFD_PART_VOL instead of *MESH_VOL for ID. It is now possible to impose a rotation on a part using Euler angles.
- *ICFD_CONTROL_OUTPUT:
  - Field 4 now to output mesh in LSPP format and in format to be run by the icfd solver (icfd_fluidmesh.key and icfd_mesh.key)
  - icfd_mesh.key now divides the mesh in ten parts, from best quality element decile to worst.
  - A new mesh is now output at every remeshing.
  - Added support for parallel I/O for Paraview using the PVTU format.
- *ICFD_DEFINE_POINT: Points can now be made to rotate or translate.
- *ICFD_MAT:
  - Nonnewtonian models and Porous media models are now selected in the third line by using the new ICFD_MODEL keyword family.
  - HC and TC can now be made temperature dependent.
- *ICFD_CONTROL_DIVCLEAN: Added option 2 to use a potential flow solver to initialize the Navier Stokes solver.
- *ICFD_CONTROL_FSI: Field 5 provides a relaxation that starts after the birthtime.
- *ICFD_CONTROL_MESH: Field 3 added a new strategy to interpolate a mesh size during the node insertion. In some cases it speeds up the meshing process and produces less elements. Field 4 changes the meshing strategy in 2d.
- *ICFD_CONTROL_SURFMESH: Added support for dynamic re-meshing/adaptation of surface meshing.
- *ICFD_BOUNDARY_PRESCRIBED_VEL:VAD = 3 now works with DOF = 4.
- SF can be lower than 0.
- PID can be over 9999 in *ICFD_DATABASE_FLUX.
- Fixed d3hsp keyword counter.
- Clarified terminal output.
- Y+ and Shear now always output on walls rather than when a turbulence model was selected.

- Added coordinate of distorted element before remeshing occurs. Output on terminal and messag file
- Fixed bug in conjugate heat transfer cases. When an autotimestep was selected in *ICFD_CONTROL_TIME, it would always only take the thermal timestep.
- An estimation of the CFL number is now output in the d3plot files. This is not the value used for the autotimestep calculation.
- Turbulence intensity is now output in the d3plots.
- Jobid now supported for ICFD ASCII File outputs.
- Fixed communication of turbulent constants in MPP.
- Fixed the Near Velocity field output.
- Increasing the limit of number of parts for the model.
- Temperature added as a surface variable in output.
- Fixed non-linear conjugate heat solver.

- **Implicit**

  - Fixed Implicit for the case of Multi-step Linear (*CONTROL_IMPLICIT_GENERAL with NSOLVR = 1) with Intermittent Eigenvalue Computation (*CONTROL_IMPLICIT_EIGENVALUE with NEIG < 0).
  - Recent fix for resultant forces for Multi-step Linear cause segmentation fault when Intermittent Eigenvalue Computation was also active.
  - Fix possible issue related to constrained contacts in MPP implicit not initializing properly.
  - Fixed label at beginning of implicit step to be correct for the case of controlling implicit dynamics via a load curve (*CONTROL_IMPLICIT_DYNAMICS).
  - Corrected the computation of modal stresses with local coordinate terms and for some shell elements (see MSTRES on *CONTROL_IMPLICIT_EIGENVALUE).
  - Corrected *CONTROL_IMPLICIT_INERTIA_RELIEF logic in MPP. In some cases the rigid body modes were lost.
  - Enhanced implicit's treatment of failing spotwelds (*CONSTRAINED_SPOTWELD).
  - Added additional error checking of input data for *CONTROL_IMPLICIT_MODAL_DYNAMICS_DAMPING.
  - Per user request we added the coupling of prescribed motion constraints for Modal Dynamics by using constraint modes. See *CONTROL_IMPLICIT_MODAL_DYNAMIC.
  - Added reuse of the matrix reordering for MPP implicit execution. This will reduce the symbolic processing time which is noticeable when using large numbers of MPP processes. Also added prediction of non-tied contact connections for standard contact and mortar contact. This allows reuse of the ordering when contact interfaces are changing very slightly but can increase the cost of the numerical factorization. Useful only for MPP using large

numbers of processes for large finite element models. This reuse checking happens automatically for MPP and is not required for SMP.

- Apply improvements to Metis memory requirements used in Implicit MPP.
- Enhanced Metis ordering software (ORDER = 2, the default, on *CONTROL_IMPLICIT_SOLVER).
- Added new keyword *CONTROL_IMPLICIT_ORDERING to control of features of the ordering methods for the linear algebra solver in MPP Implicit. Only should be used by expert users.
- The following 4 enhancements are applicable when IMFLAG > 1 on *CONTROL_IMPLICIT_GENERAL.
  - Implicit was modified to reset the time step used in contact when switching from implicit to explicit.
  - Adjusted implicit mechanical time step for the case of switching from explicit to implicit so as not to go past the end time.
  - Explicit with intermittent eigenvalue analysis was getting incorrect results after the eigenvalue analysis because an incorrect time step was used for the implicit computations. For this scenario implicit now uses the explicit time step.
  - The implicit time step is now reset for the dump file in addition to explicit's time.
- Implicit's treatment of prescribed motion constraints defined by a box had to be enhanced to properly handle potential switching to explicit.
- The following 6 enhancements are for matrix dumping (MTXDMP > 0 on *CONTROL_IMPLICIT_SOLVER) or for frequency response (*FREQUENCY_DOMAIN) computations.
- Corrected the collection of *DAMPING_PART_STIFFNESS terms for elements like triangles and 5, 6, and 7 node solid elements.
- Corrected Implicit's access of *DAMPING_PART_STIFFNES parameter when triangle and tet sorting is activated.
- Fixed Implicit's collecting of damping terms for beams that have reference nodes.
- There is an internal switch that turns off damping for beams if the run is implicit static. This switch needed to be turned off for explicit with intermittent eigenvalue analysis.
- Fixed collecting of stiffness damping terms for implicit. Corrected the loading of mass damping terms when collecting damping terms for post processing.
- Extend matrix dumping to include dumping the solution vector in addition to the matrix and right-hand-side.
- Adjusted Implicit's handling of sw1. and sw3. sense switches to properly handle dumping. If sw1. sense switch is issued when not at equilibrium, then reset time and geometry to that at the end of last implicit time step. If

sw3. sense switch is issued, then wait until equilibrium is reached before dumping and continuing.

- Enable the use of intermittent eigenvalue computation for models using inertia relief and/or rotational dynamics. See NEIG < 0 on *CONTROL_IMPLICIT_EIGENVALUE and *CONTROL_INERTIA_RELIEF and *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS. Due to round-off, an implicit intermittent eigenvalue computation was occasionally skipped. A fudge factor of 1/1000 of the implicit time step was added to compensate for round-off error in the summation of the implicit time. See NEIG < 0 on *CONTROL_IMPLICIT_EIGENVALUE.
- Added support for *CONSTRAINED_LINEAR for 2D implicit problems. It was already supported for standard 3D problems.
- Added warning for implicit when the product of ILIMIT and MAXREF (two parameters on *CONTROL_IMPLICIT_SOLUTION) is too small. For the special case when the user changes the default of ILIMIT to 1 to choose Full Newton and does not change MAXREF then MAXREF is reset to 165 and a warning is generated. Reinstate the option of MAXREF < 0.
- Fixed the display of superelements in LS-PrePost. Enhanced reading of Nastran dmig files to allow for LS-DYNA-like comment lines starting with '$'. Fixed a problem with implicit initialization in MPP with 2 or more superelements. See *ELEMENT_DIRECT_MATRIX_INPUT.
- Turned off annoying warning messages associated with zero contact elemental stiffness matrices coming from mortar contact. See *CONTACT_..._MORTAR
- Fixed construction of d3mode file in MPP. Involves proper computation of the reduced stiffness matrix. See *CONTROL_IMPLICIT_MODES
- Fixed up *PART_MODES to correctly handle constraint modes.
    - removed rigid body modes
    - correct construction of reduced stiffness matrix
- Enhanced the error handling for input for *PART_MODES.
- Modified open statements for binary files used by implicit to allow for use of *CASE.
- Removed internal use files such as spooles.res when not required for debugging.
- Fixed implicit static condensation and implicit mode computation to properly deal with the *CASE environment. See *CONTROL_IMPLICIT_STATIC_CONDENSATION and *CONTROL_IMPLICIT_MODES. Sort node/dof sets for implicit_mode to get correct results. Properly handle cases with only solid elements.
- Add implicit implementation of the new "last location" feature for MPP error tracking.
- Fixed problem with implicit processing of rigid body data with deformable to rigid switching (*DEFORMABLE_TO_RIGID).

- Extended Implicit model debugging for LPRINT = 3 (*CONTROL_IMPLICIT_SOLVER) to isogeometric and other large elemental stiff matrices.
- Added beam rotary mass scaling to the modal effective mass computation. Enhanced implicit computation of modal effective mass that is output to file eigout with *CONTROL_IMPLICIT_EIGENVALUE. We had to account for boundary SPC constraints as well as beam reference nodes to get the accumulated percentage to add up to 100%.
- Fixed a problem reporting redundant constraints for MPP Implicit.
- Enhanced *CONTACT_AUTO_MOVE for implicit.
- Fixed Implicit handling of *CONSTRAINED_TIE-BREAK in MPP.
- Added support for implicit dynamics to *MAT_157 and *MAT_120.
- Skip frequency damping during implicit static dynamic relaxation.
- Added feature to simulate brake squeal. Transient and mode analysis can be combined to do the brake squeal study by intermittent eigenvalue analysis. Besides *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS, *CONTROL_IMPLICIT_SOLVER should also be used, setting LCPACK = 3 to enable nonsymmetric stiffness matrix. In the nonsymmetric stiffness matrix analysis such as brake squeal analysis, the damping ratio, defined as -2.0*RE(eigenvalue)/ABS(IMG(eigenvalue)), can be output to the eigout file and plotted in LS-PrePost. A negative damping ratio indicates an unstable mode.
- Add a warning message if the defined rotational speed is not the same as NOMEG in *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS.
- *CONTROL_IMPLICIT: Fixed a bug to initialize velocity correctly when using a displacement file in dynamic relaxation for implicit MPP.
- Nonlinear implicit solver 12 is made default implicit solver, which is aimed for enhanced robustness in particular relation to BFGS and line search.
- Parameter IACC available on *CONTROL_ACCURACY to invoke enhanced accuracy in selected elements, materials and tied contacts. Included is strong objectivity in the most common elements, strong objecitity and physical respons in most commont tied contacts and full iteration plasticity in *MATs 24 and 123. For more detailed information refer to the manual.
- Bathe composite time integration scheme implemented for increased stability and conservation of energy/momentum, see *CONTROL_IMPLICIT_DYNAMICS. Time integration parameter ALPHA on CONTROL_IMPLICIT_DYNAMICS is used for activation.
- For NLNORM.LT.0 all scalar products in implicit are with respect to all degrees of freedom, sum of translational and rotational (similar to NLNORM.EQ.4), just that the rotational dofs are scaled using ABS(NLNORM) as a characteristic length to appropriately deal with consistency of units.
- The message 'convergence prevented due to unfulfilled bc...' has annoyed users. Here this is loosened up a little and also accompanied with a check that the bc that prevents convergence is actually nonzero. Earlier this prevention has activated even for SPCs modelled as prescribed zero motion, which does not make sense.

- Implicit now writes out the last converged state to the d3plot database on error termination if not already written.
- Fixed bug for *CONTROL_IMPLICIT_MODAL_DYNAMIC if jobid is used.

- ***INITIAL**

  - Fix incorrect NPLANE and NTHICK for *INITIAL_STRESS_SHELL when output to dynain file for shell type 9.
  - Fix *INITIAL_STRAIN_SHELL output to dynain for shell types 12 to 15 in 2D analysis.
  - Write out strain at only 1 intg point if INTSTRN = 0 in *INTERFACE_SPRINGBACK_LSDYNA and all strains at all 4 intg points if INTSTRN = 1 and nip = 4 in *SECTION_SHELL.
  - *INITIAL_EOS_ALE: Allow initialization of internal energy density, relative volume, or pressure in ALE elements by part, part set, or element set.
  - *INITIAL_VOLUME_FRACTION_GEOMETRY: Add option (FAMMG < 0) to form pairs of groups in *SET_MULTI-MATERIAL_GROUP_LIST to replace the first group of the pair by the second one.
  - *INITIAL_STRESS_DEPTH can now work with parts that have an Equation of State (EOS types 1, 4, 6 only). Note however that *INITIAL_STRESS_DEPTH does not work with ALE.
  - Fix several instances of overwriting the initial velocities of any interface nodes read in from a linking file (SMP only).
  - *INITIAL_VOLUME_FRACTION_GEOMETRY: Add local coordinate system option for box.
  - The initial strain and energy is calculated for *INITIAL_FOAM_REFERENCE_GEOMETRY.
  - Add the option of defining the direction cosine using two nodes for *INITIAL_VELOCITY_GENERATION.
  - Fix incorrect transformation of *DEFINE_BOX which results in incorrect initial velocities if the box is used in *INITIAL_VELOCITY.
  - Fix incorrect initial velocity when using *INITIAL_VELOCITY with NX = -999.
  - Fix seg fault when using *INITIAL_INTERNAL_DOF_SOLID_TYPE4 in dynain file.
  - Do not transform the translational velocities in *INITIAL_VELOCITY or *INITIAL_VELOCITY_GENERATION if the local coordinate system ICID is defined.
  - Fix uninitialized velocities when using *INITIAL_VELOCITY_GENERATION with STYP = 2, i.e. part id, for *ELEMENT_SHELL_COMPOSITE/*ELEMENT_TSHELL_COMPOSITE.
  - Fix incorrect initialization of velocities if using *INITIAL_VELOCITY_GENERATION with STYP = 1, i.e. part set for shells with formulation 23 & 24.
  - Fix incorrect initial velocity and also mass output to d3hsp for shell types 23 & 24.

- Fix incorrect initial velocities when using *INITIAL_VELOCITY_GENERATION with irigid = 1 and *PART_INERTIA with xc = yc = zc = 0 and nodeid > 0 with *DEFINE_TRANSFORMATION.
- Fix incorrect stress initialization of *MAT_057/MAT_LOW_DENSITY_FOAM using dynain file with *INITIAL_STRESS_SOLID when NHISV is equal to the number of history variables for this mat 57.
- Fix seg fault when reading dynain.bin
- Fixed stress initialization (*INITIAL_STRESS_SECTION) for type 13 tetrahedral elements. The pressure smoothing was causing incorrect pressure values in the elements adjacent to the prescribed elements.
- Assign initial velocities (*INITIAL_VELOCITY) to beam nodes that are generated when release conditions are defined (RT1, RT2, RR1, RR2 on *ELEMENT_BEAM.)
- Added an option to retain bending stiffness in spot weld beams that have prescribed axial force. To use is, set KBEND = 1 on *INITIAL_AXIAL_FORCE_BEAM.
- Fix for *INITIAL_STRESS_BEAM when used with spotweld beam type 9. It was possible that error/warning message INI+140 popped up even if number of integration points matched exactly.
- Fix for the combination of type 13 tet elements and *INITIAL_STRESS_SOLID. The necessary nodal values for averaging (element volume, Jacobian) were not correctly initialized. Now the initial volume (IVEFLG) is used to compute the correct initial nodal volume.

- **Isogeometric Elements**

  - Enable spc boundary condition to be applied to extra nodes of nurbs shell, see *CONSTRAINED_NODES_TO_NURBS_SHELL
  - Fix a bug for isogeometric element contact, IGACTC = 1, that happens when more than one NURBS patches are used to model a part so that a interpolated elements have nodes belonging to different NURB patches.
  - *ELEMENT_SOLID_NURBS_PATCH:

    - Enable isogeometric analysis for solid elements, it is now able to do explicit and implicit analysis, such as contact and eigenvalue analysis, etc.
    - Add mode stress analysis for isogeometric solid and shell elements so that the isogeometric element is also able to do frequency domain analysis.

  - Add reduced, patch-wise integration rule for C1-continuous quadratic NURBS. This can be used by setting INT = 2 in *ELEMENT_SHELL_NURBS_PATCH.

- Add trimmed NURBS capability. Define NL trimming loops to specify a trimmed NURBS patch. Use *DEFINE_CURVE (DATTYP = 6) to specify define trimming edges in the parametric space.
- Fix bug in added mass report for *ELEMENT_SHELL_NURBS_PATCH in MPP.

- ***LOAD**

  - *LOAD_GRAVITY_PART and staged construction (*DEFINE_STAGED_CONSTRUCTION_PART) were ignoring non-structural mass MAREA (shells) and NSM (beams). Now fixed.
  - Fix for *INTERFACE_LINKING in MPP when used with adaptivity.
  - Updates for *INTERFACE_LINKING so that it can be used with adaptivity, provided the linked parts are adapting.
  - Fix for *INTERFACE_LINKING when used with LSDA based files generated by older versions of the code.
  - *DEFINE_CURVE_FUNCTION:

    - Functions "DELAY", PIDCTL" and "IF" of are revised.
    - Add sampling rate and saturation limit to PIDCTL of *DEFINE_CURVE_FUNCTION.
    - "DELAY" of *DEFINE_CURVE_FUNCTION can delay the value of a time-dependent curve by "-TDLY" time steps when TDLY < 0.

  - Add edge loading option to *LOAD_SEGMENT_SET_NONUNIFORM.
  - Fix insufficient memory error,SOL+659, when using *LOAD_ERODING_PART_SET with mpp.
  - Fix incorrect loading when using *LOAD_ERODING_PART_SET with BOXID defined.
  - Fix incorrect pressure applied if the directional cosines, V1/V2/V3, for *LOAD_SEGMENT_SET_NONUNIFORM do not correspond to a unit vector.
  - Add *DEFINE_FUNCTION capability to *LOAD_SEGMENT_SET for 2D analysis.
  - Fix incorrect behavior when using arrival time, AT, or box, BOXID, in *LOAD_ERODING_PART_SET.
  - Fix error when runing analysis with *LOAD_THERMAL_CONSTANT_ELEMENT_(OPTION) in MPP with ncpu > 1.
  - Fixed *LOAD_STEADY_STATE_ROLLING when used with shell form 2 when used with Belytschko-Wong-Chang warping stiffness (BWC = 1 *CONTROL_SHELL).
  - Add "TIMESTEP" as a code defined value available for *DEFINE_FUNCTION and *DEFINE_CURVE_FUNCTION. It holds the current simulation timestep.
  - Fixed issues involving *LOAD_THERMAL_D3PLOT.

- Allow extraction of node numbers in loadsetud for all values of LTYPE in *USER_LOADING_SET. Comments included appropriately in the code. Argument list of loadsetud is changed accordingly.
- Implemented SPF simulation (*LOAD_SUPERPLASTIC_FORMING) for 2d problems.
- Added effective stress as target variable for SPF simulation.
- Added box option for SPF simulation to limit target search regions.

- ***MAT**

  - Fix output to d3hsp for *MAT_HYPERELASTIC_RUBBER. Broken in r93028.
  - Error terminate with message, KEY+1115, if_STOCHASTIC option is invoked for *MATs 10,15,24,81,98, 123 but no *DEFINE_STOCHASTIC_VARIATION or *DEFINE_HAZ_PROPERTIES keyword is present in the input file.
  - Fix spurious error termination when using *DEFINE_HAZ_PROPERTIES with adaptivity.
  - Fixed *MATs 161 and 162 when run with MPP. The array that is used to share delamination data across processors had errors.
  - *MAT_261/*MAT_262: Fixed problem using *DAMPING_PART_STIFFNESS together with RYLEN = 2 in *CONTROL_ENERGY.
  - Added safety check for martensite phase kinetics in *MAT_244.
  - Fix for combination of *MAT_024_STOCHASTIC and shell elementstype 13, 14, and 15 (with 3d stress state).
  - Fix bug in *MATs 21 and 23 when used with *MAT_ADD_THERMAL_EXPANSION.
  - *MAT_ALE_VISCOUS: Implement a user defined routine in dyn21.F to compute the dynamic viscosity.
  - Add histlist.txt to usermat package. This file lists the history variables by material.
  - Bug in *MAT_089 fixed: The load curve LCSS specifies the relationship between "maximum equivalent strain" and the von Mises stress. The "maximum equivalent strain" includes both elastic and plastic components. The material model was not calculating this variable as intended, so was not following LCSS accurately. The error was likely to be more noticeable when elastic strains are a significant proportion of the total strain e.g. for small strains or low initial Youngs modulus.
  - Fixed bug affecting *MAT_119: unpredictable unloading behaviour in local T-direction if there are curves only for the T-direction and not for the S-direction.
  - Fixed bug in *MAT_172: Occured when ELFORM = 1 (Hughes-Liu shell formulation) was combined with Invariant Numbering (INN > 0 on *CONTROL_ACCURACY). In this case, the strain-softening in tension did not work: after cracking, the tensile strength remained constant.

- New option for *MAT_079: Load curve LCD defining hysteresis damping versus maximum strain to date. This overrides the default Masing behaviour.
- *MAT_172:
  - Added error termination if user inputs an illegal value for TYPEC. Previously, this condition could lead to abnormal terminations that were difficult to diagnose.
  - Fixed bug affecting ELFORM = 16 shells made of *MAT_172 – spurious strains could develop transverse to the crack opening direction.
- Fixed bug in *MAT_ARUP_ADHESIVE (*MAT_169). The displacement to failure in tension was not as implied by the inputs TENMAX and GCTEN. For typical structural adhesives with elastic stiffness of the order of 1000-10000 MPa, the error was very small. The error became large for lower stiffness materials.
- *MAT_SPR_JLR:
  - Modify output variables from *MAT_SPR_JLR (see Manual).
  - Fix bug that caused spurious results or unexpected element deletion if TELAS = 1.
- Fixed bug in *MAT_174 - the code could crash when input parameters EUR = 0 and FRACR = 0..
- Fix MPP problem when writing out aea_crack file for *MAT_WINFRITH.
- Include *MAT_196 as one that triggers spot weld thinning.
- *MAT_ADD_FATIGUE: Implemented multi slope SN curves to be used in random vibration fatigue (*FREQUENCY_DOMAIN_RANDOM_VIBRATION_FATIGUE) and SSD fatigue (FREQUENCY_DOMAIN_SSD_FATIGUE).
- Guard against possible numerical round off that in some cases might result in unexpected airflow in *MAT_ADD_PORE_AIR.
- Added new material *MAT_115_O/*MAT_UNIFIED_CREEP_ORTHO.
- *MAT_274: Added support for 2D-solids. New flag (parameter 8 on card 2) is used to switch normal with in-plane axis.
- *MAT_255: Fixed bug in plasticity algorithm and changed from total strain rate to plastic strain rate for stability. Added VP option (parameter 5 on card 2) for backwards compatibility: VP = 0 invokes total strain rate used as before.
- Added new cohesive material *MAT_279/*MAT_COHESIVE_PAPER to be used in conjunction with *MAT_274/*MAT_PAPER.
- User materials: Added support for EOS with user materials for tshell formulations 3 and 5.
- Fixed bug in dyna.str when using EOS together with shells and orthotropic materials.

- *MAT_122: A new version of *MAT_HILL_3R_3D is available. It supports temperature dependent curves for the Young's/shear moduli, Possion ratios, and Hill's anisotropy parameters. It also supports 2D-tables of yield curves for different temperatures. Implicit dynamics is supported. The old version is run if parameter 5 on card 3 is set to 1.0.
- Added the phase change option to *MAT_216, *MAT_217, *MAT_218 to allow material properties to change as a function of location. This capability is designed to model materials that change their properties due to material processing that is otherwise not modeled. For example, increasing the mass and thickness due to the deposition of material by spraying. It is not used for modeling phase changes caused by pressure, thermal loading, or other mechanical processes modeled within LS-DYNA.
- Fix internal energy computation of *MAT_ELASTIC_VISCOPLASTIC_-THERMAL/MAT_106.
- Fix incorrect results or seg fault for *MAT_FU_CHANG_FOAM/MAT_083 if KCON > 0.0 and TBID.ne.0.
- If SIGY = 0 and S = 0 in *MAT_DAMAGE_2/MAT_105, set S = EPS1/200, where EPS1 is the first point of yield stress input or the first ordinate point of the LCSS curve.
- Set xt = 1.0E+16 as default if user inputs 0.0 for *MAT_ENHANCED_COMPOSITE_DAMAGE/MAT_054. Otherwise, random failure of elements may occur. Implemented for thick shells and solids.
- Allow *MAT_ENHANCED_COMPOSITE_DAMAGE/MAT_054 failure mechanism to work together with *MAT_ADD_EROSION for shells.
- Fix incorrect erosion behavior if *MAT_ADD_EROSION is used with failure criteria defined for *MAT_123/MAT_MODIFIED_PIECEWISE_LINEAR_PLASTICITY.
- Fix non-failure of triangular elements type 4 using *MAT_ADD_EROSION with NUMFIP = -100.
- Implement scaling of failure strain for *MAT_MODIFIED_PIECEWISE_-LINEAR_PLASTICITY_STOCHASTIC/MAT_123_STOCHASTIC for shells.
- Fix incorrect behavior for *MAT_LINEAR_ELASTIC_DISCRETE_-BEAM/MAT_066 when using damping with implicit(statics) to explicit switching.
- Fix error due to convergence when using *MAT_CONCRETE_-EC2/MAT_172 in implicit and when FRACRX = 1.0 or FRACRY = 1.0
- Fix incorrect fitting results for *MAT_OGDEN_RUBBER/MAT_077_O if the number of data points specified in LCID is > 100.
- Fix incorrect fitting results for *MAT_MOONEY-RIVLIN-RUBBER/MAT_027 if the number of data points specified in LCID is > 100.
- Fix incorrect forces/moments when preloads are used for *MAT_-067/NONLINEAR_ELASTIC_DISCRETE_BEAM and the strains changes sign.

- Implement *MAT_188/MAT_THERMO_ELASTO_VISCO-PLASTIC_CREEP for 2D implicit analysis.
- Support implicit for *MAT_121/MAT_GENERAL_NONLIN-EAR_1DOF_DISCRETE_BEAM.
- Fix seg fault when using *DEFINE_HAZ_TAILOR_WELDED_BLANK with *DEFINE_HAZ_PROPERTIES.
- Fix ineffective *MAT_ADD_EROSION if the MID is defined using a alpha-numeric label.
- Fix seg fault when using *MAT_PIECEWISE_LINEAR_PLASTIC_THER-MAL/MAT_255 for solids.
- Zero the pressure for *MAT_JOHNSON_HOLMQUIST_JH1/MAT_241 af-ter it completely fractures, i.e. D>=1.0, under tensile load.
- Fix incorrect element failure when using EPSTHIN and VP = 0 for *MAT_-123/MODIFIED_PIECEWISE_LINEAR_PLASTICITY.
- Fix error termination when using adaptive remeshing for 2D analysis with *MAT_015/JOHNSON_COOK and NIP = 4 in *SECTION_SHELL and ELFORM = 15.
- Fix erosion due to damage, max shear & critical temperature in elastic state for *MAT_MODIFIED_JOHNSON_COOK/MAT_107 for solids.
- Check diagonal elements of C-matrix of *MAT_002/MAT_*OPTION* TROP-IC_ELASTIC and error terminate with message, STR+1306, if any of them are negative.
- Fix plastic strain tensor update for *MAT_082/*MAT_PLASTICITY_-WITH_DAMAGE.
- Fix error when using *MAT_144/MAT_PITZER_CRUSHABLE_FOAM with solid tetahedron type 10.
- Fix out-of-range forces after dynamic relaxation when using VP = 1 for *MAT_PIECEWISE_LINEAR_PLASTICITY and non-zero strain rate parameters, C & P, and the part goes into plastic deformation during dynamic relaxation.
- Fixed unit transformation for GAMAB1 and GAMAB2 on *MAT_DRY_-FABRIC. We were incorrectly transforming them as stress.
- Fixed implicit solutions with shell elements that use *MAT_040 and laminated shell theory.
- Fixed the stress calculation in the thermal version of *MAT_077.
- Corrected the AOPT = 0 option of ortho/anisotropic materials when use with skewed solid elements. Previously, the material direction was initialized to be equivalent to the local coordinate system direction. This is not consistent with the manual for skewed elements which states that the material a-axis is in the 1-2 directions for AOPT = 0. This is now fixed and the manual is correct.
- Fixed the AOPT = 0 option of ortho/anisotropic materials for tetrahedral element forms 10, 13, and 44.
- Fixed *MAT_082 for solid elements. An error in the history data was causing possible energy growth or loss of partially damaged elements.

- Modified *MAT_FABRIC/*MAT_034 FORM = 24 so that Poisson's effects occur in tension only.
- Modified *MAT_221/*MAT_ORTHOTROPIC_SIMPLIFIED_DAMAGE to correct the damage behavior. Prior to this fix, damage was applied to new increments of stress, but not the stress history, so material softening was not possible.
- Fixed *MAT_106 when used with curves to define the Young's modulus and Poisson's ratio and when used with thick shell form 5 or 6. The assumed strain field was unreasonable which caused implicit convergence to fail.
- Added 2 new erosion criteria for *MAT_221/*MAT_ORTHOTROPC_SIMPLIFIED_DAMAGE. The new options are NERODE = 10: a or b directions failure (tensile or compressive) plus out of plane failure bc or ca. NERODE = 11: a or b directions failure (tensile only) plus out of plane failure bc or ca.
- Added a new option for shell *MAT_022/*MAT_COMPOSITE_DAMAGE. When ATRACK = 1, the material directions will follow not only element rotation, but also deformation. This option is useful for modeling layered composites, that have material a-directions that vary by layer, by allowing each layer to rotate independently of the others. Within each layer, the b-direction is always orthogonal to the a-direction.
- Fixed the TRUE_T option on *MAT_100 and *MAT_100_DA. If the weld connects shells with different thickness and therefore different bending stiffness, the scheme used by TRUE_T to reduce the calculated moment could behave somewhat unpredictably. With the fix, TRUE_T behaves much better, both for single brick welds and brick assemblies.
- Added a warning message and automatically switch DMGOPT > 0 to DMGOPT = 0 on *MAT_FABRIC when RS < EFAIL or RS = EFAIL. This prevents a problem where weld assemblies did not fail at all when RS = 0.
- *MATs 9, 10, 11, 15, 88, and 224 are now available for thick shells, however only *MATs 15, 88, and 224 are available for the 2D tshell forms 1,2, and 6.
- Added thick shell support for the STOCHASTIC option of *MATs 10, 15, 24, 81, and 98.
- Added support for *MAT_096 for several solid element types including ELFORMs 3, 4, 15, 18, and 23.
- Added a MIDFAIL keyword option for *MAT_024, (MAT_PIECEWISE_LINEAR_PLASTICITY). With this option, element failure does not occur until the failure strain is reached in the mid plane layer. If an even number of layers is used, then the failure occurs when the 2 closest points reach the failure strain.
- Enabled *MATs 26 and 126 (HONEYCOMB) to be used with thick shell forms 3, 5, and 7. These was initialized incorrectly causing a zero stress.
- Enabled *MAD_ADD_EROSION to be used with beams that have user defined integration. Memory allocation was fixed to prevent memory errors.
- Enabled OPT = -1 on *MAT_SPOTWELD for solid elements.

- Enabled thick shells to use *MATs 103 and 104 in an implicit solution. These materials were lacking some data initialization so they would not converge.
- Enabled solid elements with user-defined orthotropic materials to work with the INTOUT and NODOUT options on *DATABASE_EXTENT_BINARY. The transformation matrix was stored in the wrong place causing strain and stress transformations to fail.
- Enabled *MAT_017 to run with thick shell forms 3 and 5. Neither element was initialized correctly to run materials with equations of state.
- Add degradation factors and strain rate dependent strength possibility for *MAT_054/*MAT_ENHANCED_COMPOSITE_DAMAGE solids.
- Fixed bug in *MAT_058/*MAT_LAMINATED_COMPOSITE_FABRIC when used with strain-rate dependent tables for stiffnesses EA, EB and GAB and LAMSHT = 3.
- Add strain rate dependency of ERODS in *MAT_058.
- Add possibility to use *DEFINE_FUNCTION for *MAT_SPOTWELD_DAMAGE_FAILURE (*MAT_100), OPT = -1/0. If FVAL = FunctionID, then a *DEFINE_FUNCTION expression is used to determine the weld failure criterion using the following arguments: func (N_rr, N_rs, N_rt, M_rr, M_ss, M_tt).
- Store tangential and normal separation (delta_II & delta_I) as history variables 1&2 of *MAT_138/*MAT_COHESIVE_MIXED_MODE.
- Add second normalized traction-separation load curve (TSLC2) for Mode II in *MAT_186/*MAT_COHESIVE_GENERAL.
- Fixed bug in using *MAT_157/*MAT_ANISOTROPIC_ELASTIC_PLASTIC with IHIS.gt.0 for shells. Thickness strain update d3 was not correct and plasticity algorithm failed due to typo.
- Fixed bug in *MAT_157 for solids: This affected the correct stress transformation for post-processing using CMPFLG = 1 in *DATABASE_EXTENT_BINARY.
- Fixed bug in *MAT_225 (*MAT_VISCOPLASTIC_MIXED_HARDENING) when using Table-Definition together with kinematic hardening.
- Add load curves for rate dependent strengths (XC, XT, YC, YT, SC) in *MAT_261/*MAT_LAMINATED_FRACTURE_DAIMLER_PINHO (shells only).
- Add table definition for LCSS for rate dependency in *MAT_261 (shells only).
- Add load curves for rate dependent strengths (XC, XCO, XT, XTO, YC, YT, SC) in *MAT_262/*MAT_LAMINATED_FRACTURE_DAIMLER_CAMANHO (shells only).
- Fixed bug when using *MAT_261 or *MAT_262 solids (ELFORM = 2).
- Add load curves for SIGY and ETAN for rate dependency of *MAT_262 (shells only)
- *MAT_021_OPTION

- Fixed a bug for defining different orientation angles through the thickness of TSHELL elements (formulations 2 and 3)
- Added new option CURING:
- Two additional cards are read to define parameters for curing kinetics. Formulation is based on Kamal's model and considers one ODE for the state of cure.
- State of cure does not affect the mechanical parameters of the material.
- CTE's for othotropic thermal expansione can be defined in a table with respect to state of cure and temperature.
- An orthotropic chemical shrinkage is accounted for.

◦ *MAT_REINFORCED_THERMOPLASTICS_OPTION (*MAT_249_OPTION):

- Fiber shear locking can be defined wrt to the fiber angle or shear angle.
- Output of fiber angle to history variables.
- Simplified input: Instead of always reading 8 lines, now the user only has to specify data for NFIB fibers.
- Added fiber elongation to history variables in *MAT_249 for posprocessing.
- New Option UDFIBER (based on a user defined material by BMW):
  - Transversely isotropic hyperelastic formulation for each fiber family (see Bonet&Burton,1998).
  - Anisotropic bending behavior based on modified transverse shear stiffnesses.
  - Best suited for dry NCF's.

◦ *MAT_GENERALIZED_PHASE_CHANGE (*MAT_254):

- New material that is a generalized version of *MAT_244 with application to a wider range of metals.
- Up to 24 different phases can be included.
- Between each of the phases, the phase transformation can be defined based on a list of generic transformation laws. For heating JMAK and Oddy are implemented. For cooling Koistinen-Marburger, JMAK and Kirkaldy can be chosen.
- Constant parameters for the transformations are given as 2d tables, parameters depending on temperature (rate) or phase concentration employ 3d tables.
- Plasticity model (temperature and strain rate dependent) similar to MAT_244.
- Transformation induced strains.
- TRIP algorithm included.
- Temperature dependent mixture rules.

- Parameter 'dTmax' that defines the maximum temperature increment within a cycle. If the temperature difference at a certain integration point is too high, local subcycling is performed.
- Implemented for explicit/implicit analysis and for 2d/3d solid elements.

- *MAT_ADHESIVE_CURING_VISCOELASTIC (*MAT_277):

  - New material implementation including a temperature dependent curing process of epoxy resin based on the Kamal-Sourour-model.
  - Material formulation is based on *MAT_GENERAL_VISCOELASTIC.
  - Viscoelastic properties defined by the Prony series, coefficients as functions of state of cure.
  - Chemical and thermal shrinkage considered (differential or secant formulations).
  - Available for shell and solid elements.
  - Can be used in combination with *MAT_ADD_COHESIVE.
  - Implemented for explicit and implicit analysis.
  - An incremental and a total stress calculation procedure available.

- Enable *MAT_ADD_EROSION to be safely used with material models that have more than 69 history variables, for now the new limit is 119.
- Use correct element ID for output of failed solid elements when GISSMO (*MAT_ADD_EROSION) is used with *CONTROL_DEBUG.
- Improve performance of GISSMO (*MAT_ADD_EROSION with IDAM = 1), especially when used with *MAT_024, no other failure criteria, shell elements, and DMGEXP = 1 or 2. Allows speed-up of 10 to 20 percent.
- Add new keyword *MAT_ADD_GENERALIZED_DAMAGE. It provides a very flexible approach to add non-isotropic (tensorial) damage to standard materials in a modular fashion. Solely works with shell elements at the moment.
- Correct the computation of effective strain for options ERODS < 0 in *MAT_058 (*MAT_LAMINATED_COMPOSITE_FABRIC) and EFS < 0 in *MAT_261 and *MAT_262 (*MAT_LAMINATED_FRACTURE_DAIMLER_...). The shear strain term was twice the size as it should have been.
- Adjust stiffness for time step calculation in *MAT_076 and subsequent models (*MAT_176, *MAT_276, ...) to prevent rarely observed instabilities.
- Add output of original and fitted curves to messag and separate file (curveplot_<MID>) for *MAT_103.
- In *MAT_104 (*MAT_DAMAGE_1), stress-strain curve LCSS can now be used directly with all FLAG options (-1,0,1,10,11), no fitting.
- Correct strain calculation for anisotropic damage in *MAT_104 (*MAT_DAMAGE_1) with FLAG = -1.
- Initialize stress triaxiality of *MAT_107 (*MAT_MODIFIED_JOHNSON_COOK) to zero instead of 1/3.

- Avoid negative damage in *MAT_107 (*MAT_MODIFIED_JOHNSON_-COOK) with FLAG2 = 0 for solid elements.
- Rectify the characteristic element length in *MAT_138 (*MAT_COHESIVE_-MIXED_MODE) for solids type 21 and 22 (cohesive pentas) and shell type 29 (cohesive shell) for "curve" options T < 0 and S < 0.
- Correct/improve material tangent for *MAT_181 with PR > 0 (foam option).
- Add possibility to define logarithmically defined strain rate table LCID-T in material *MAT_187 (*MAT_SAMP-1).
- Fix missing offset when using *DEFINE_TRANSFORMATION with load curve LCID-P in *MAT_187 (*MAT_SAMP-1).
- Add reasonable limit for biaxial strength in *MAT_187 with RBCFAC > 0.5 to avoid concave yield surface.
- Improve performance of *MAT_187 to reach speed-up of 10 to 40 percent, depending on which options are used.
- Add new option for *MAT_224 (*MAT_TABULATED_JOHNSON_COOK). With BETA < 0 not only a load curve but now also a table can be referred to. The table contains strain rate dependent curves, each for a different temperature.
- Fix for implicit version of *MAT_224 (*MAT_TABULATED_JOHNSON_-COOK). Computations with shell elements should converge faster now.
- *MAT_224 (*MAT_TABULATED_JOHNSON_COOK) can now be used in implicit even with temperature dependent Young's modulus (parameter E < 0).
- Always store the Lode parameter as history variable #10 in *MAT_224 (*MAT_TABULATED_JOHNSON_COOK), not just for LCF being a table.
- Variable LCI of *MAT_224 / *MAT_224_GYS can now refer to a *DEFINE_-TABLE_3D. That means the plastic failure strain can now be a function of Lode parameter (TABLE_3D), triaxiality (TABLE), and element size (CURVE).
- For thick shells type 1 and 2, the element size in *MAT_224 is now correct.
- Add new option for definition of parameters FG1 and FG2 in *MAT_240 (*MAT_COHESIVE_MIXED_MODE_ELASTOPLASTIC_RATE).
- Add new option to *MAT_240: new load curves LCGIC and LCGIIC define fracture energies GIC and GIIC as functions of cohesive element thickness. GIC_0, GIC_INF, GIIC_0, and GIIC_INF are ignored in that case.
- Add new feature to *MAT_248 (*MAT_PHS_BMW). Estimated Hocket-Sherby parameters are written to history variables based on input functions and phase fractions.
- Add new option ISLC = 2 to *MAT_248 (*MAT_PHS_BMW) which allows to define load curves (cooling rate dependent values) for QR2, QR3, QR4, and all parameters on Cards 10 and 11.
- Add new option LCSS to *MAT_252 (*MAT_TOUGHENED_ADHESIVE_-POLYMER): A load curve, table or 3d table can now be used to define rate and temperature dependent stress-strain behavior (yield curve).

- Fix for *MAT_255, evaluation of 2d tables LCIDC and LCDIT. Negative temperatures were interpreted as logarithmic rates.
- Add new material model *MAT_280 (*MAT_GLASS) for shell elements. It is a smeared fixed crack model with a selection of different brittle, stress-state dependent failure criteria and crack closure effects.
- *DEFINE_FABRIC_ASSEMBLIES: Assemblies of *MAT_FABRIC part sets can be specified to properly treat bending of t-intersecting fabrics that are stitched or sewn together. See ECOAT, TCOAT and SCOAT on *MAT_FABRIC_... Bending can only occur within an assembly, aka a part set.
- *MAT_USER_DEFINED_MATERIAL_MODELS: In user defined material models, a logical parameter 'reject' can be set to .true. to indicate to the implicit solver that equilibrium iterations should be aborted. The criterion is the choise of the implementor, but it could be if plastic strain increases by more than say 5% in one step or damage increases too much, whatever that might render an inaccurate prediction and bad results. Setting this parameter for explicit won't do anything.
- IHYPER = 3 for user shell materials now supports thickness train update, see *MAT_USER_DEFINED_MATERIAL_MODELS.
- *MAT_SIMPLIFIED_RUBBER/FOAM: AVGOPT < 0 is now supported for the FOAM option, which activates a time averaged strain rate scheme to avoid noisy response.
- MAT_181 is now supported for 2D implicit simulations.
- *MAT_ADD_EROSION:

  - A number of extensions and improvements to the DIEM damage model were made, IDAM < 0.
  - General efficiency, it was slow, now it's GOT to be faster.
  - NCS can be used as a plastic strain increment to only evaluate criteria in quantifications of plastic strain.
  - NUMFIP < 0 is employing the GISSMO approach, number of layers for erosion.
  - A new ductile damage criterion based on principal stress added (DMITYP = 4).
  - MSFLD and FLD can be evaluated in mid or outer layers to separate membrane and bending instability (P2).
  - MSFLD and FLD can use an incremental or direct update of instability parameter (P3).
  - Output of integration point failure information made optional (Q2).
  - Specifying DCTYP = -1 on the damage evolution card will not couple damage to stress but the damage variable is only calculated and stored.

- *MAT_SMOOTH_VISCOELASTIC_VISCOPLASTIC, *MAT_275: An elastic-plastic model with smooth transition between elastic and plastic mode is available. It incorporates viscoelasticity and viscoplasticity and is based on hyper-elastoplasticity so it is valid for arbitrarily large deformations and

rotations. A sophisticated parameter estimation is required to match test data, it is available for implicit and explicit analyisis but perhaps mostly suited for implicit.

- *MAT_FABRIC_MAP: Stress map material 34 is equipped with bending properties identical to that of the form 14 and form -14 version of the fabric. Coating properties are set in terms of stiffness, thickness and yield. The material is supported in implicit, including optional accounting for the non-symmetric tangent. Should be used with bending stiffness on, and convergence is improved dramatically if geometric stiffness is turned on.
- *MAT_084 with predefined units (CONM < 0) is now transformed correctly with INCLUDE_TRANSFORM.
- If LCIDTE = 0 in *MAT_121, then LS-DYNA was crashing on some platforms, including Windows. This is fixed.
- Fix initialization issues so that PML models can be run with *CASE commands.
- *MAT_027 is revised to avoid accuracy issues for single precision executables.
- The nearly imcompressible condition is enhanced for *MAT_027 shell elements.
- Add a new material model as a option for *MAT_165. *MAT_PLASTIC_NONLINEAR_KINEMATIC_B is a mixed hardening material model, and can be used for fatigue analysis.
- Output local z-stress in *MAT_037, when *LOAD_STRESS_SURFACE is used. This was previously calculated and saved as another history variable.
- Add a new material model *MAT_260 (2 forms).
  - Uses non-associated flow rule and Hill's yield surface; including strain rate effect and temperate effect. MIT failure criteria is also implemented.
  - Implemented for solids and shells.
  - Strain rate sensitivity for solids.
  - Option to directly input the Pij and Gij values.
  - Separate the material model *MAT_260 into *MAT_260A and *MAT_260B:
    - MAT260A=*MAT_STOUGHTON_NON_ASSOCIATED_FLOW
    - MAT260B=*MAT_MOHR_NON_ASSOCIATED_FLOW
  - Incorporates FLD into the fracture strain, so as to consider the mesh size effect.
  - Calculates the characteristic length of the element for *MAT_260B, so that an size-dependent failure criterial can be used.
  - When failure happens for half of the integration points through the thickness, the element is deleted.
- Add Formablitiy Index to *MAT_036, *MAT_037, *MAT_226.

- Add new history variables for Formability Index, affecting *MAT_036, *MAT_037, *MAT_125, *MAT_226. Those new history variables are FI, beta, effective strain. These comes after the 4 regular history variables.
- *MAT_036, *MAT_125: New option_NLP is added to evaluate formability under non-linear strain paths. User inputs a forming limit diagram (FLD), and Formablitiy Index (F.I.) will be automatically converted to effective stain vs. beta based space.

- **MPP**

  - Fix problem of MPP pre-decomposition that can occur if the local directory specified in the pfile has very different lengths in the initial run vs the actual run The difference resulted in a line count difference in the size of the structured files created, throwing off the reading of the file in the actual run.
  - Straighten out some silist/sidist issues in MPP decomp:

    - silist and sidist outside of a "region" in the pfile are no longer supported, and an error message is issued which suggests the use of "region { silist" instead.
    - They have been undocumented for several years (since "region" was introduced), and had other issues.

  - Fix the keywords, CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE and CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE, which were not treating each contact interface individually (as the manual states), but collectively.
  - Fix for MPP decomp of part sets.
  - Fixed *CONTROL_MPP_PFILE (when used inside an include file) so that it honors ID offsets from *INCLUDE_TRANSFORM for parts, part sets, and contact ids referenced in "decomp { region {" specifications. Furthermore, such a region can contain a "local" designation, in which case the decomposition of that region will be done in the coordinate system local to the include file, not the global system. For example:
    *CONTROL_MPP_PFILE
    decomp { region {partset 12 local c2r 30 0 -30 0 1 0 1 0 0}}
    would apply the c2r transformation in the coordinate system of the include file, which wasn't previously possible. The local option can be useful even if there are no such transformations, as the "cubes" the decomposition uses will be oriented in the coordinate system of the include file, not the global system. Furthermore, the following decomposition related keywords now have a_LOCAL option, which has the same effect:

    - *CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE_LOCAL

        - *CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE_LOCAL
        - *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS_LOCAL
        - *CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE_LOCAL
          (**Note:** LOCAL option is only valid when *INCLUDE_TRANSFORM with *DEFINE_TRANSFORMATION is used)

    - Revert revision 86884, which was:

        - "MPP: change to the decomposition behavior of *CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE
          *CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE
          *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS
          in the case where a decomposition transformation is also used. Previously, any such regions were distributed without the transformation being applied. This has been fixed so that any given transformation applies to these regions also. So now the transformations will NOT apply to these keywords. Really, the "region" syntax should be used together with *CONTROL_MPP_PFILE as it is more specific.

    - Modify behavior of DECOMPOSITION_AUTOMATIC so that if the initial velocity used is subject to *INCLUDE_TRANSFORM, the transformed velocities are used.
    - Fix MPP decomposition issue with "decomp { automatic }" which was not honored when in the pfile.
    - Save hex weld creation orientation to the pre-decomposition file so that the subsequent run generates the welds in the same way.
    - Fix for MPP not handling element deletion properly in some cases at decomposition boundaries.
    - Add new pfile option "contact { keep_acnodes }" which does NOT exclude slave nodes of adaptive constraints from contact, which is the default behavior. (MPP only.)
    - MPP Performance-Related Improvements:

        - Allow user input of *LOAD_SEGMENT_FILE through familied files.
        - Bug fix for *LOAD_SEGMENT_FILE to get correct time history data for pressure interpolation.
        - Output two csv files for user to check MPP performance:
            - load_profile.csv: general load balance
            - cont_profile.csv: contact load balance
        - Allow user to control decomp/distribution of multiple airbags using *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS
        - memory2 = option on *KEYWORD line

        - Disable unreferenced curves after decomposition using *CONTROL_MPP_DECOMPOSITION_DISABLE_UNREF_CURVES. This applies to the curves used in the following options to speed up the execution several times.
            - *BOUNDARY_PRESCRIBED_MOTION_NODE
            - *LOAD_NODE
            - *LOAD_SHELL_ELEMENT
            - *LOAD_THERMAL_VARIABLE_NODE

    - Bug fix for *CONTROL_MPP_DECOMPOSITION_SHOW with *AIRBAG_PARTICLE.
    - Fix cpu dependent results when using function RCFORC() in *DEFINE_CURVE_FUNCTION. This affects MPP only.
    - Fix hang up when using *DEFINE_CURVE_FUNCTION with element function BEAM(id,jflag,comp,rm) and running MPP with np > 1.
    - *CONTROL_MPP_DECOMPOSITION: The cpu cost for solid elements -1 and -2 are accounted for in the mpp domain decomposition.
    - Fix bug in *CONTROL_MPP_IO (Windows platform only) related to insufficient administrative privileges for writing tmp file on root drive.
    - Revise l2a utility on Windows platform to create identical node output format as Linux.

- **Output**

    - Fix for MPP external work when bndout is output and there are *BOUNDARY_PRESCRIBED_MOTION_RIGID commands in the input.
    - Fixed the output of forces and associated energy due to *LOAD_RIGID_BODY for both explicit and implicit (*DATABASE_BNDOUT).
    - Fixed stress and strain output of thick shells when the composite material flag is set on *DATABASE_EXTENT_BINARY. The transformation was backwards.
    - If the size of a single plot state was larger than the d3plot size defined by x=<factor> on the execution line, the d3plot database may not be readable by LS-PrePost. This issue is now fixed.
    - *DATABASE_PROFILE: Output data profiles for beams (TYPE = 5) and add density as DATA = 20.
    - New option HYDRO = 4 on *DATABASE_EXTENT_BINARY. Outputs 7 additional variables: the same 5 as HYDRO = 2 plus volumetric strain (defined as Relative Volume - 1.0) and hourglass energy per unit initial volume.
    - Fix for binout output of swforc file which can get the data vs. ids out of sync when some solid spotwelds fail.
    - Fix for d3plot output of very large data sets in single precision.
    - Fix for output of bndout data for joints in MPP, which was writing out incorrect data in some cases.

- Added new option *INTERFACE_SPRINGBACK_EXCLUDE to exclude selected portions from the generated dynain file.
- Add a new option to *INTERFACE_COMPONENT_FILE to output only 3 degrees of freedom to the file, even if the current model has 6.
- Minor change to how pressure is computed for triangles in the INTFOR output.
- Fix MPP output issue with intfor file.
- Fixes for writing and reading of dynain data in LSDA format.
- Corrected the summation of rigid body moments for output to bndout for some special cases in MPP.
- Corrected the output to d3iter when 10 node tets are present (D3ITCTL on *CONTROL_IMPLICIT_SOLUTION).
- Enhanced implicit collection of moments for the rcforc file.
- For implicit, convert spc constraint resultant forces to local coordinate system for output. Also corrected Implicit's gathering of resultant forces due to certain SPC constraints.
- Fixed the gathering of resultant forces in implicit for prescribed motion on nodes of a constrained rigid body for output to bndout.
- Added output of modal dynamics modal variables to a new file moddynout. Output is controlled by *CONTROL_IMPLICIT_MODAL_DYNAMICS.
- Corrected the output of resultant forces for Implicit Linear analysis. Corrected the output of resultant forces for MPP executions. These enhancements affect a number of ASCII files including bndout.
- The following 4 enhancements are to the eigensolvers, including that used for *CONTROL_IMPLICIT_EIGENVALUE.

    - Standardized and enhance the warning/error messages for Implicit eigensolution for the case where zero eigenmodes are computed and returned in eigout and d3eigv.
    - Added nonsymmetric terms to the stiffness matrix for the implicit rotational dynamics eigenanalysis. This allows brake squeal analysis with the contact nonsymmetric terms from mortar contact now included in the analysis.
    - Updated implicit eigensolution for problems with nonsymmetric stiffness matrices. Fixed Rotational Dynamics eigensolution to work correctly when first order matrix (W) is null. (See *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS).
    - Added the eigensolution for problems with stiffness (symmetric or nonsymmetric), mass, and damping.

- Improve Implicit's treatment of constrained joints to account for rounding errors. Applicable to *CONSTRAINED_JOINT with *CONTROL_IMPLICIT_GENERAL.
- For implicit springback, zero out the forces being reported to rcforc for those contact interfaces disabled at the time of springback. Also enhance the

removal of contact interfaces for springback computations. For *INTERFACE_SPRINGBACK.
- *DATABASE_RECOVER_NODE is available to recover nodal stress.
- Fix a bug for detailed stress output, eloutdet, for SOLID type 18.
- Support new format of interface force files for ALE, DEM, and CPM. LS-PrePost can display the correct label for each output component.
- Added *DATABASE_NCFORC_FILTER option to allow the NCFORC data to be filtered using either single pass or double pass Butterworth filtering to smooth the output. Added the same filtering capability to *DATABASE_BINARY_D3PLOT. This capability is specified on the additional card for the D3PLOT option and does not require "_FILTER" in the keyword input.
- Fix incorrect mass properties for solids in SSSTAT file when using *DATABASE_SSSTAT_MASS_PROPERTIES.
- Fix seg fault during writing of dynain file if INSTRN = 1 in *INTERFACE_SPRINGBACK and STRFLG ≠ 0 in *DATABASE_EXTENT_BINARY and the *DATABASE_EXTENT_BINARY comes after *INTERFACE_SPRINGBACK. Also output warning message, KEY+1104.
- Fix zero strain values output to curvout for *DEFINE_CURVE_FUNCTION using function, ELHIST, for solid elements.
- Fix missing parts in d3part when MSSCL = 1 or 2 in *DATABASE_EXTENT_BINARY.
- Fix incorrect damping energy computation for glstat.
- Fix incorrect part mass in d3plot for shells, beams & thick shells.
- Fix incorrect curvout values when using BEAM(id,jflag,comp,rm) for *DEFINE_CURVE_FUNCTION and if the beam formulation is type 3, i.e. truss.
- Fix incorrect output to curvout file if using ELHIST in *DEFINE_CURVE_FUNCTION for shells.
- Output stresses for all 4 intg points to eloutdet for cohesive element types 19 & 20.
- Fix incorrect rotational displacement to nodout when REF = 2 in *DATABASE_HISTORY_NODE_LOCAL. Affects MPP only.
- Fix incorrect strains output to elout for shell type 5 and when NIP > 1.
- Fix incorrect acceleration output to nodout file when IACCOP = 1 in *CONTROL_OUTPUT and IGRAV = 1 in *ELEMENT_SEATBELT_ACCELEROMETER.
- Fix corrupted d3plot when RESPLT = 1 in *DATABASE_EXTENT_BINARY and idrflg.ge.5 in *CONTROL_DYNAMIC_RELAXATION.
- Fix missing element connectivities in nastin file when using *INTERFACE_SPRINGBACK_NASTRAN_NOTHICKNESS.
- Fix seg fault when using *DATABASE_BINARY_D3PART with *CONTACT_TIED_SHELL_EDGE_TO_SURFACE. This affects SMP only.
- Fix incorrect output to bndout when using multiple *LOAD_NODE_POINT for the same node and running MPP with ncpu > 1.
- Fix incorrect dyna.inc file when using *MAT_FU_CHANG_FOAM/MAT_83, *DEFINE_COORDINATE_NODES, and *CON-

STRAINED_JOINT_STIFFNESS_GENERALIZED with *INCLUDE_-TRANSFORM.

- Fix IEVERP in *DATABASE_EXTENT_D3PART which was not honored in writing out d3part files.
- Fix incorrect stresses written out to dynain for thick shells with formulations 1,2 and 4.
- Fix incorrect output to disbout data for discrete beams.
- Fix incorrect output to binary format of disbout. Affects SMP only.
- Fix error when writing initial stresses for thick shells to dynain. Affects MPP only.
- Fix thick shells strain output to dynain.
- Fix incorrect writing of material data to dyna.str for *MAT_SEATBELT when using long = s.
- Fix coordinate/disp output to d3plot of *CONSTRAINED_NODAL_-RIGID_BODY's pnode.
- Fixed the initial d3plot state in SMP runs when tied contact is used with theCNTCO parameter on *CONTROL_SHELL. The geometry was wrong in that state.
- Add cross section forces output (*DATABASE_SECFORC) for cohesive elements ELFORM type 19, 20, 21, and 22.
- Slight increase of precision for values in nodout file.
- Add new option FSPLIT to *INTERFACE_SPRINGBACK_LSDYNA to split the dynain file into two files (geometry and initial values).
- *DEFINE_MATERIAL_HISTORIES: New keyword for organizing material history outputs, currently only for solids, shells and beams and the d3plot output but to be extended to tshells and ascii/binout. The purpose is to customize the history variables that otherwise are output via NEIPS/NEIPH/NEIPB on *DATABASE_EXTENT_BINARY, to avoid variable conflict and large d3plots and thus facilitate post-processing of these variables. Currently available in small scale but to be continuously extended.
- Fixed bug affecting IBINARY = 1 (32 bit ieee format) in *DATABASE_FORMAT. This option was not working.
- Fixed incorrect printout of node ID for *ELEMENT_INERTIA.
- Increased the header length to 80 for the following files in binout: matsum, nodout, spcforc, ncforc
- Fixed bug in which d3msg was not written for SMP.
- The d3plot output for rigid surface contact was incorrect for MPP.
- Fixed bugs when using curve LCDT to control d3plot output.
- Fixed abnormal increase in d3plot size caused by outputting velocity and acceleration when data compression is on.
- Added new variable GEOM in *CONTROL_OUTPUT for choosing geometry or displacement in d3plot, d3part, and d3drlf.

- Added command line option "msg=" to output warning/error descriptions. See MSGFLG in *CONTROL_OUTPUT for alternate method of requesting such output. Accepted values for "msg=" are message# or all.

  - message#, e.g., KEY+101 or 10101. This option will print the error/warning message to the screen.
  - all. this option will print all error/warning messages to d3msg file.

- Fixed bug for *DATABASE_BINARY_D3PROP file if adaptivity used. The error caused blank d3prop output.
- *DATABASE_HISTORY_SHELL_SET combined with *CONTROL_ADAPTIVITY caused error 20211. The error involves the BOX option being used for shell history output.
- Added *INTEGRATION... data to d3prop.

- **Restarts**

  - Fix bug when deleted uniform pressure (UP) airbag during simple restart.
  - Fix for index error that could cause problems for accelerometers during full deck restart in MPP.
  - Fix for MPP output of LSDA interface linking file when restarting from a dump file.
  - Fix incorrect strains in d3plot after restart when STRLG > 1.
  - Fix incorrect velocity initialization for SMP full deck restart when using
  - *INITIAL_VELOCITY_GENERATION and *INITIAL_VELOCITY_GENERATION_START_TIME.
  - Fix incorrect behavior of *CONTACT_ENTITY in full deck restart.
  - Fix incorrect full deck restart analysis if initial run was implicit and the full deck restart run is explicit.
  - Fix ineffective boundary condition for *MAT_RIGID when using *CHANGE_RIGID_BODY_CONSTRAINT with *RIGID_DEFORMABLE_R2D for small deck restart.
  - Fix initialization of velocities of *MAT_RIGID_DISCRETE nodes after restart using *CHANGE_VELOCITY_GENERATION.
  - Fix internal energy oscillation after full deck restart when using *CONTACT_TIED_SURFACE_TO_SURFACE_OFFSET with TIEDID = 1 in optional card D. This affects SMP only.
  - Corrected bug affecting full restart that included any change to node/element IDs. This bug has existed since version R6.
  - Fixed bug affecting d3plot times following fulldeck restart with curve in SMP.
  - Fixed bug in simple restart: *INTERFACE_COMPONENT_FILE forgets the filename and writes to infmak instead.

- ***SENSOR**
  - Enable full restart for *SENSOR.
  - Add optional filter ID to SENSORD of *DEFINE_CURVE_FUNCTION.
  - Enable LOCAL option of *CONSTRAINED_JOINT to be used with *SENSOR_DEFINE_FORCE.
  - Fix a MPP bug that happens when *SENSOR_DEFINE_NODE has a defined N2.
  - *SENSOR_CONTROL:
    - Fix a bug for TYPE = JOINTSTIF
    - Fix an MPP bug for TYPE = PRESC-MOT when the node subject to prescribed motion is part of a rigid body
    - Add TYPE = BELTSLIP to control the lockup of *ELEMENT_SEATBELT_SLIPRING.
    - Add TYPE = DISC-ELES to delete a set of discrete elements.
  - Add FTYPE = CONTACT2D to *SENSOR_DEFINE_FORCE to track the force from *CONTACT_2D.
  - Add the variable SETOPT for *SENSOR_DEFINE_NODE_SET and *SENSOR_DEFINE_ELEMENT_SET to sense and process data from a node set or element set, resp., resulting in a single reported value.
  - *SENSOR can be used to control *CONTACT_GUIDED_CABLE.
  - Fix a bug related to *SENSOR_DEFINE_FUNCTION triggered by more than 10 sensor definitions.
- **SPG (Smooth Particle Galerkin)**
  - *SECTION_SOLID_SPG (KERNEL = 1): The dilation parameters (DX, DY, DZ) of SPG Eulerian kernel are automatically adjusted according to the local material deformation to prevent tensile instability.
- **SPH (Smooth Particle Hydrodynamics)**
  - Retain user IDs of SPH particles in order to ensure consistent results when changing the order of include files.
  - Add feature to inject SPH particles, *DEFINE_SPH_INJECTION.
  - Added support of various material models for 2D and 3D SPH particles:
    - *MAT_098 (*MAT_SIMPLIFIED_JOHNSON_COOK)
    - *MAT_181 (*MAT_SIMPLIFIED_RUBBER)
    - *MAT_275 (*MAT_SMOOTH_VISCOELASTIC_VISCOPLASTIC)
  - Added support of *DEFINE_ADAPTIVE_SOLID_TO_SPH for 2D shell elements and 2D axisymmetric shell elements.

- When using *DEFINE_ADAPTIVE_SOLID_TO_SPH, eliminated duplicate kinetic energy calculation for SPH hybrid elements (both SPH particles and solid elements contributed kinetic energy into global kinetic energy).
- Added support of second order stress update (OSU = 1 in *CONTROL_ACCURACY keyword) for 2D and 3D SPH particles. This is necessary for simulation of spinning parts.
- Added ISYMP option in *CONTROL_SPH to define as a percentage of original SPH particles the amount of memory allocated for generation of SPH ghost nodes used in *BOUNDARY_SPH_SYMMETRY_PLANE.
- Fixed unsupported part and part set option in *BOUNDARY_SPH_FLOW.
- Fixed unsupported ICONT option from *CONTROL_SPH when combined with *BOUNDARY_SPH_FLOW.
- *DEFINE_SPH_TO_SPH_COUPLING: Output contact forces between two SPH parts (x, y, z and resultant forces) into sphout. The forces can be plotted by LS-PrePost.
- *CONTACT_2D_NODE_TO_SOLID: Added bucket sort searching algorithm to speed up the process of finding contact pairs between SPH particles and solid segments.

- **Thermal**

  - Corrected a long standing bug in MPP thermal associated with spotwelds (*CONSTRAINED_SPOTWELD) using thermal linear solver option 11 or greater. The spotweld loads were not being loaded correctly due to an indexing issue in MPP.
  - Fix for thermal with *CASE.
  - Fix MPP support for thermal friction in SOFT = 4 contact.
  - Fixed bug where thermal solver gives a non-zero residual even though no loads are present.
  - Added SOLVER = 17 (GMRES solver) to *CONTROL_THERMAL_SOLVER for the conjugate heat transfer problem. The GMRES solver has been developed as an alternative to the direct solvers in cases where the structural thermal problem is coupled with the fluid thermal problem in a monolithic approach using the ICFD solver. A significant savings of calculation time can be observed when the problem reaches 1M elements. This solver is implemented for both SMP and MPP.
  - *CONTACT_(option)_THERMAL (3D contact only): Add variable FTOSLV to specify fraction of frictional energy applied to slave surface. It follows that 1.-FTOSLV is applied to master surface. Default is 0.5 which gives a 50% - 50% split between the slave and master surfaces which was hardwired in prior releases.
  - First release of AUTOMATIC_SURFACE_TO_SURFACE_TIED_WELD_THERMAL. This will only work when used with BOUNDARY_THERMAL_WELD. This combination of keywords will activate a condition where sliding contact will become tied contact on cooldown when the temperature

of the segments in contact go above an input specified temperature limit during welding.

- *LOAD_THERMAL_D3PLOT: The d3plot data base was changed such that the 1st family member contains control words, geometry, and other control entities. Time state data begins in the 2nd family member. This change allows the new d3plot data structure to be read in by LS-DYNA when using the *LOAD_THERMAL_D3PLOT keyword. This change is not backward compatible. The old d3plot data structure will no longer be read correctly by LS-DYNA.
- Synchronize data in TPRINT for SMP and MPP:
  - Fixed output to tprint/binout for thermal contact.
  - Fixed part IDs for part energies.
  - Fixed format of TPRINT file generated by l2a.
- Fixed handling of start time defined with *CONTROL_START for thermal solver.
- Change the maximum number of *LOAD_HEAT_CONTROLLER definitions from 10 to 20.
- Added a third parameter to the TIED_WELD contact option. The parameter specifies heat transfer coefficient h_contweld for the welded contact. Before welding, the parameter from the standard card of the thermal contact is used.
- Parameter FRCENG supported for mortar contact to yield heat in coupled thermomechanical problems.

- **XFEM (eXtended Finite Element Method)**
  - Added ductile failure to XFEM using critical effective plastic strain as failure criterion.

- **Miscellaneous**
  - Support *SET_NODE_GENERAL PART with SPH and DES.
  - *DEFINE_POROUS_...: Compute the coefficients A and B with a user defined routine in dyn21.F.
  - Fixed bugs in Staged Construction (*DEFINE_STAGED_CONSTRUCTION_PART):
    - Staged construction not working on SMP parallel. Symptoms could include wrong elements being deleted.
    - Staged construction with beam elements of ELFORM = 2: when these beams are dormant, they could still control the time step.
    - Staged construction with *PART_COMPOSITE. The bug occurred when different material types were used for different layers within the same part, and that part becomes active during the analysis. The

symptom of the bug was that stresses and/or history variables were not set to zero when the part becomes active.

- Bugs fixed in *DAMPING_FREQUENCY_RANGE_DEFORM:
  - Incorrect results when large rigid body rotations occur.
  - If RYLEN on *CONTROL_ENERGY = 2, the energy associated with this damping should be included in the Internal Energy for the relevant part(s). This energy was being calculated only if there was also *DAMPING_PART_STIFFNESS in the model. Now fixed - the damping energy will be included in the internal energy whenever RYLEN = 2.
- Fixed NID option of *DEFINE_COORDINATE_VECTOR (bug occurred in MPP only).
- Fix lsda open mode to require only minimal permissions to avoid unnecessary errors, for example if using an interface linking file that is read only.
- Fix for DPART processing (*SET_..._GENERAL) for solid and thick shell elements.
- Fix for JOBID > 63 characters.
- Fix input processing problem (hang) that could happen in some unusual cases if encrypted *INCLUDE files are used.
- Fix interaction of *CASE with jobid = on command line, so the jobid on the command line is combined with the generated case ids instead of being ignored.
- *INCLUDE_NASTRAN:
  - Integration defaults to Lobatto for Nastran translator.
  - The default number of integration points is set to 5 for Nastran translator.
- Issue error message and terminate when illegal *DEFINE_TRANSFORMATION is specified.
- Add OPTION = POS6N to *DEFINE_TRANSFORMATION to define transformation with 3 reference nodes and 3 target nodes.
- Add OPTION = MIRROR to *DEFINE_TRANSFORMATION.
- Fix a bug that could occur when adapted elements are defined in a file included by *INCLUDE_TRANSFORM.
- Fix a bug that could occur when *BOUNDARY_SPC_SYMMETRIC_PLANE is used together with *INCLUDE_TRANSFORM.
- Fix a bug that occurs when *DEFINE_BOX is included by *INCLUDE_TRANSFORM.
- Make *SET_NODE_COLLECT work together with *NODE_SET_MERGE.
- Fix incorrect shell set generated when using *SET_SHELL_GENERAL with OPTION = PART.

- Add error trap for *SET_PART_LIST_GENERATE_COLLECT to catch missing part IDs.
- Fixed bug in *INCLUDE_TRANSFORM for adaptive case if JOBID is used.
- Fixed bug in memory allocation for *DEFINE_CURVE if total number of points in curve is more than 100.
- Fixed bug with *INCLUDE_TRANSFORM and *CONTROL_ADAPTIVITY due to an *INCLUDE inside *INCLUDE_TRANSFORM file. Added new files: adapt.inc# for *INCLUDE_TRANSFORM file. The *NODE, *ELEMENT_SHELL and *ELEMENT_SOLID are removed from include file.
- Fixed bug for DPART option in *SET_SEGMENT_GENERAL. DPART option was treated as PART option before.
- Fixed failure of *PARAMETER definition in long format.
- Fixed error in reading solid id for *SET_SOLID_GENERAL.
- Ignore any nonexistant part set IDs in *SET_PART_ADD.
- Fix bug in which sense switches sw2 and sw4 don't work when the output interval for glstat is small.
- Fixed bug if *DEFINE_CURVE is used to define adaptivity level.
- Three new keywords are implemented in support of user defined subroutines: *MODULE_PATH[_RELATIVE], MODULE_LOAD, MODULE_USE.

    - The MODULE feature allows users to compile user subroutines into dynamic libraries without linking to the LS-DYNA main executable.
    - The dynamic libraries are independent from the main executable and do not need to be recompiled or linked if the main executable is updated.
    - This feature loads multiple dynamic libraries on demand as specified in the keywords.
    - Without the MODULE feature, only one version of each umat (such as umat41) can be implemented. With the MODULE feature, most umat subroutines can be have multiple versions in multiple dynamic libraries, and used simultaneously.
    - The MODULE feature supports all user subroutines.
    - The LS-DYNA main executable may also run without any dynamic libraries if no user subroutines are required.

Capabilities added to create LS-DYNA R10:

See release notes (published separately) for further details.

- ***AIRBAG**

    - Enhance the robustness of *AIRBAG_INTERACTION to help avoid instability in MPP when the interaction involves more than two bags.
    - *AIRBAG_PARTICLE:

- Adjust dm_out calculation of vent hole to avoid truncation error.
- Fix bug in chamber output when there are multiple airbags and multiple chambers not in sequential order.
- Bug fix for closed volume of airbag/chamber with intersecting tubes.
- Add new feature to allow user to define local coordinates of jetting of particles through internal vents.
- Support *SENSOR_CONTROL for CPM airbag.
- CPM is not supported for dynamic relaxation. Disable CPM airbag feature during DR and reactivate airbag following DR.
- Allow solid parts in definition of internal part set. The solid volume will be excluded from the airbag volume.
- Allow additional internal part set for shells. The shell part should form a closed volume and its volume will be excluded from the airbag volume.

- ***ALE**
  - *LOAD_BLAST_SEGMENT: Automatically generate the ALE ambient elements attached to a segment or segment set.
  - *BOUNDARY_AMBIENT_EOS: implement *DEFINE_CURVE_FUNCTION for the internal energy and relative volume curves.
  - *CONTROL_ALE, *CONSTRAINED_LAGRANGE_IN_SOLID and *ALE_REFERENCE_SYSTEM: If NBKT < 0 in *CONTROL_ALE, call *DEFINE_CURVE to load a curve defining the number of cycles between bucket sorting in function of time. If NBKT > 0, the bucket sorting is activated if the mesh rotations and deformations are large.
  - *ALE_FSI_TO_LOAD_NODE: Implement a mapping of the FSI accelerations (penalty forces/masses) computed by *CONSTRAINED_LAGRANGE_IN_SOLID (ctype = 4) between different meshes.
  - DATABASE_FSI, *DATABASE_BINARY_FSIFOR and *DATABASE_BINARY_FSILNK: Add a parameter CID to output fsi forces in a local coordinate system.
  - Structured ALE (S-ALE) solver:
    - ALE models using rectilinear mesh can be directly converted to S-ALE models and run using S-ALE solver by assigning CPIDX = -1 in *ALE_STRUCTURED_MESH.
    - S-ALE progressive mesh generation via RATIO in *ALE_STRUCTURED_MESH_CONTROL_POINTS.
  - Recode ALE Donor Cell/Van Leer advection routines and restructure *CONSTRAINED_LAGRANGE_IN_SOLID communication algorithm. These give 30% improvement in run time.

- ***BOUNDARY**

- *BOUNDARY_PWP can now accept a *DEFINE_FUNCTION instead of a load curve. The input arguments are the same as for *LOAD_SEGMENT: (time, x, y, z, x0, y0, z0).
- Add option of "toffset" for *BOUNDARY_PRESCRIBED_ORIENTATION_-RIGID to offset the curves by the birth time.
- MPP now supports MCOL coupling, *BOUNDARY_MCOL.
- Fix bug of there being fully constrained motion of a rigid part when prescribing more than one translational dof with *BOUNDARY_PRESCRIBED_MOTION_RIGID while con2 = 7 in *MAT_RIGID, meaning all rotational dof are constrained.
- Instead of error terminating with warning message, STR+1371, when *BOUNDARY_PRESCRIBED_MOTION and *BOUNDARY_SPC is applied to same node and dof, issue warning message, KEY+1106, and release the conflicting SPC.
- Fix erroneous results if SET_BOX option is used for *BOUNDARY_PRESCRIBED_MOTION.
- Fix *BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID for MPP. It may error terminate or give wrong results if more than one of this keyword is used.
- Fix segmentation fault when using *BOUNDARY_PRESCRIBED_ORIENTATION with vad = 2, i.e. cubic spline interpolation.
- Added instruction *BOUNDARY_ACOUSTIC_IMPEDANCE for explicit calculations that applies an impedance boundary condition to the boundary of *MAT_ACOUSTIC element faces. This is a generalization of the non-reflecting boundary condition. Both *LOAD and *BOUNDARY_ACOUSTIC_IMPEDANCE may be used on the same faces, in which case the boundary acts like both and entrant and exit boundary.
- Fixed a problem with non-reflecting boundaries redefining the bulk modulus which caused contact to change behavior.
- Added support for acoustic materials with non-reflective boundaries.
- Fix the single precision version so that *INCLUDE_UNITCELL now has no problem to identify pairs of nodes in periodic boundaries.
- When using *INCLUDE_UNITCELL to generate Periodic Boundary Constraints (PBC) for an existing mesh, a new include file with PBCs is generated instead of changing the original mesh input file. For example, if users include a file named "mesh.k" through *INCLUDE_UNITCELL (INPT = 0), a new include file named "uc_mesh.k" is generated where all PBCs are defined automatically following the original model information in mesh.k.
- *INCLUDE_UNITCELL now supports long input format in defining the element IDs.
- Include SPC boundary conditions as part of H8TOH20 solid element conversion.
- Add a new option SET_LINE to *BOUNDARY_PRESCRIBED_MOTION: This option allows a node set to be generated including existing nodes and new nodes created from h-adaptive mesh refinement along the straight line

connecting two specified nodes to be included in prescribed boundary conditions.

- **BLAST**
  - *PARTICLE_BLAST and DES:
    - Consider eroding of shell and solid in particle_blast.
    - Support interface force file output for gas particle-structure coupling.
    - Bug fix for wet DES coupled with beam.
    - Support *SET_NODE_GENERAL PART with SPH or DES.
    - MPP now uses async communication for DES coupling to improve general performance.
    - Support for solid element when modeling irregular shaped charge with HECTYPE = 0/1 in *PARTICLE_BLAST.
    - Output adaptive generated DES and NODE to a keyword file.
  - Fix inadvertent detonation of HE part when there are more than one HE part and even though the HE part is not defined with *INITIAL_DETONATION.
  - Fixed explicit *BOUNDARY_USA_COUPLING to support *INITIAL_-STRESS and *INITIAL_STRAIN_ usage, typically from a dynain file.
  - Fixed explicit *BOUNDARY_USA_COUPLING to support *CONTROL_-DYNAMIC_RELAXATION IDRFLF = 5, so a static implicit calculation can be used to initialize/preload a model before conducting an explicit transient calculation. If inertia relief is used during the static phase, then it must be disabled with *CONTROL_IM-PLICIT_INERTIA_RELIEF for the explicit phase.
  - Support imperial unit system for *PARTICLE_BLAST. mass = lbf-s^2/in, length = inch, time = second, force = lbf, pressure = psi.
  - Add option to define detonation point using a node for *PARTICLE_BLAST.
  - Add interface force file output for *PARTICLE_BLAST with keyword *DA-TABASE_BINARY_PBMFOR and command line option "pbm=". This output of forces for gas-particle-structure coupling.
  - For *PARTICLE_BLAST, add built-in smoothing function for particle structure interaction.
  - For *PARTICLE_BLAST, when coupling with DEM, the DEM nodes that are inside HE domain are automatically deactivated.
  - Add support for solid elements when modeling irregular shaped charge with HECTYPE = 0/1 for *PARTICLE_BLAST. The original approach only supports shell elements and the initial coordinates of HE particle are at shell surface. The model had to relax several hundred time step to let particle fill in the interior space, which was not convenient. Using new approach, the initial positions of HE particles are randomly distributed inside the container by using solid element geometry. Both hex and Tet solids are supported.

- For particle blast method (PBM), consider reflecting plane as infinite.
- Change the name of keyword *DEFINE_PBLAST_GEOMETRY to *DEFINE_PBLAST_HEGEO.

- ***CESE (Compressible Fluid Solver)**

  - CESE time steps:

    - Modified the blast wave boundary condition treatment to make it more stable in blast wave calculations (with *LOAD_BLAST_ENHANCED).
    - The flow field calculation will be skipped if the structural time-step is much smaller than the fluid time step, until both time-steps reach the same order. This will save CPU time in some fluid/structure interaction (FSI) problem calculations.
    - In addition to depending upon the local CFL number, the fluid time step 'dt' calculation has been modified to also adjust dynamically to extreme flow conditions. This makes stiff flow problems more stable especially in 3D fluid problem calculations when the mesh quality is poor.

  - Moving mesh solvers:

    - Corrected several aspects of the implicit ball-vertex (BV) mesh motion solver for the following keywords:
      *ICFD_CONTROL_MESH_MOV
      *CESE_CONTROL_MESH_MOV.
    - The absolute tolerance argument is no longer used by the BV solver. As an example, the following is all that is needed for CESE moving mesh problems:
      *CESE_CONTROL_MESH_MOV
      $ ialg numiter reltol
      1 500 1.0e-4
    - Also corrected the CESE moving mesh solvers for a special case involving a wedge element. Also, fixed the d3plot output of wedge element connectivities for the CESE moving mesh solvers.

  - CESE d3plot output:

    - Added real 2D CESE output, and this is confirmed to work with LSP-P4.3 and later versions. This also works for d3plot output with the 2D CESE axisymmetric solver.
    - For all immersed-boundary CESE solvers, corrected the plotting of the Schlieren number and the chemical species mass fractions.
    - The following new CESE input cards are related to surface d3plot output:
      *CESE_SURFACE_MECHSSID_D3PLOT

*CESE_SURFACE_MECHVARS_D3PLOT
In conjunction with the above, new FSI and conjugate heat transfer output on solid (volume) mesh outside boundaries is now supported.

- CESE immersed-boundary method (IBM) FSI solvers:
  - *CESE_FSI_EXCLUDE is a new keyword for use with the CESE immersed boundary method FSI solvers. With it, unnecessary structural parts that are not actively participating in the FSI in the CESE IBM-FSI solver can now be excluded from the CESE FSI calculation. This is also supported for the case when some of the mechanics parts involve element erosion.
- CESE chemistry solvers:
  - In R10, we also updated several things in the FSI solver with chemistry called FSIC. In chemical reacting flow, a delta time between iterations is extremely important for code stabilization and thus, to get reasonable results. To this end, we optimized such an iterative delta time, which is based on the CFL number. This optimization is based on the gradient of the local pressure, which we think will dominate control of the CFL number.
  - Next, the total number of species are increased up to 60 species in chemical reacting flow, so that the reduced Ethylene (24~53 species) and Methane (20~60 species) combustion are possible with this version.
  - We will update more practical examples about FSIC problems including precise experimental validations.
  - Note that we can provide some related examples upon user request.
  - Other corrections of note include the following:
    Brought in enthalpy-related corrections to the CESE chemistry solvers.
    Fixed the conjugate heat transfer boundary condition for the 2D and 3D CESE fixed mesh chemistry solvers.
    Corrected the initialization of fluid pressure for CESE IBM chemistry solvers.
    Enabled output of the timing information for the CESE chemistry solvers.
    Added restart capability to the CESE chemistry solvers.

- ***CHEMISTRY**
  - New inflator models of Pyrotechnic and Hybrid type are updated. It is important to note that these are basically 0-dimensional models via the following two main keywords,
    *CHEMISTRY CONTROL_INFLATOR
    *CHEMISTRY_INFLATOR_PROPERTIES

- By using the *CHEMISTRY CONTROL_INFLATOR keyword, the user can select the type of the solver, output mode, running time, delta t, and time interval for output of time history data.
  For example, if we have a keyword set up as,
  *CHEMISTRY CONTROL_INFLATOR,
  $ isolver ioutput runtime delt p_time
  1 0 0.1 1.0e-6 5.0e-4

  with "isolver set to 1", the user can simulate a conventional Pyrotechnic inflator mode, while with "isolver" set to 2 or 3, Hybrid inflator simulation is possible.
- In addition, to continue an airbag simulation via an ALE or CPM method, the user can save the corresponding input data file by using "ioutput" option. For more details about airbag simulations using a saved data file, refer to the keyword manual.
- Also, note that the updated version has two options for the Hybrid models:

  - isolver = 2 => Hybrid model for the cold flow
  - isolver = 3 => Hybrid model for the heated flow.

- In the *CHEMISTRY_INFLATOR_PROPERTIES keyword, there are several cards to set up the required properties of an inflator model. The first two cards are for the propellant properties involved in inflator combustion.
  For example,
  $card1: propellants
  $ comp_id p_dia p_height p_mass p_tmass
  10 0.003 0.0013 2.0e-5 5.425e-3
  $card2: control parameters
  $ t_flame pindex A0 trise rconst
  2473. 0.4 4.45e-5 0.0 0.037
  In the first card, the user can specify the total amount of propellant particles and their shape.
  Using the second card, the user can also specify the thermodynamics of the propellant and its burning rate.
  To support the options in card2, especially the second option, pindex, and the third, A0, we provide a standalone program upon request for the propellant equilibrium simulation.
  The remaining cards are for the combustion chamber, gas chamber, and airbag, respectively.

- ***CONTACT**

  - *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED_-WELD for modeling welding has been added. Surfaces are tied based on

meeting temperature and proximity criteria. Non-MORTAR version of this contact was introduced at R9.0.1.

- Fix issue setting contact thickness for rigid shells in ERODING contact.
- Add MPP support for *CONTACT_AUTOMATIC_GENERAL with adaptivity.
- Change "Interface Pressure" report in intfor file from abs (force/area) to -force/area, which gives the proper sign in case of a tied interface in tension.
- Rework input processing so that more than one *CONTACT_INTERIOR may be used, and there can be multiple part sets in each one.
- Minor change to how pressure is computed for triangles in the intfor database.
- Fix 2 bugs for contact involving high order shell elements:
  - When high order shell elements are generated by SHL4_TO_SHL8.
  - When using a large part ID like 100000001.
- Implement a split-pinball based contact option for neighbor elements in segment-based contact. Invoke this option by setting |SFNBR|>=1000. The new algorithm is more compatible with DEPTH = 45 so that there is no longer a need to split quads.
- The effect of shell reference system offsets on contact surface location is now properly considered when running MPP. The shell offset may be specified using NLOC in *SECTION_SHELL or in *PART_COMPOSITE, or by using the OFFSET option of *ELEMENT_SHELL. This effect on contact is only considered when CNTCO is set to 1 or 2 in *CONTROL_SHELL.
- Fix bug of zero forces in rcforc at time = 0.0 for *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE after dynamic relaxation when consistency is on in SMP.
- Fix input error when using many *RIGIDWALL_GEOMETRIC_... with _DISPLAY option.
- Fix input error when *CONTACT_ENTITY is attached to a beam part, PID.
- Fix error termination due to negative volume, SOL+509, even when *CONTACT_ERODING... is set. This affects MPP only.
- Check whether a slave/master node belongs to a shell before updating the nodal thickness when ISTUPD > 0.0 in *CONTROL_SHELL and SST/MST.ne.0.0 and in SSFT/SMFT = 0.0 card 3 of *CONTACT_..... For SMP only.
- Fix penetrating nodes when using *CONTACT_ERODING_-NODES_TO_-SURFACE with SOFT = 1 in *MAT_142/*MAT_
- Fix seg fault when using *CONTACT_AUTOMATIC_-SINGLE_SURFACE_TIED with consistency mode, .i.e. ncpu < 0, for SMP.
- Fix corrupted intfor when using parts/part sets in *CONTACT_AUTOMATIC_....This affects SMP only.
- Implement incremental update of normal option, invoked by TIEDID = 1, for *CONTACT_TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET for SMP.

- Fix unconstrained nodes when using *CONTACT_TIED_-SURFACE_TO_-SURFACE_CONSTRAINED_OFFSET resulting in warning message, SOL+540. This affects SMP only.
- Fix spurious repositioning of nodes when using *CONTACT_-SURFACE_-TO_SURFACE for SMP.
- Added support to segment-based contact for the SRNDE parameter on optional card E. This option allows round edge extensions that do not extend beyond shell edges and also square edges. The latter overlaps with the SHLEDG parameter on card D.
- Fixed a potential memory error that could occur during segment-based contact input.
- Fixed an error that could cause an MPP job to hang in phase 3. The error could occur when SOFT = 2 contact is used with the periodic intersection check and process 0 does not participate in the contact.
- Modified SOFT = 2 contact friction when used with *PART_CONTACT to define friction coefficients, and the two parts in contact have different coefficient values. With this change, the mu values used for contact will be the average of the values that are calculated for each part. Prior to this change, mu was calculated for only the part that is judged to be the master. This change makes the behavior more predictable and also makes it behave like the other contacts with SOFT = 0 and SOFT = 1.
- Added a warning message (STR+1392) for when trying to use the ORTHO_-FRICTION contact option with SOFT = 2 contact, because that option is not available. The contact type is switched to SOFT = 1.
- Fixed serious error in MPP *CONTACT_2D_AUTOMATIC_-SURFACE_-TO_SURFACE when used with node sets to define the contact surfaces. The master side was likely to trigger a spurious error about missing nodes that terminated the job.
- Switched segment based (SOFT = 2) non-eroding contact to prevent it from adding any new segments when brick element faces are exposed when other elements are deleted. There were two problems. The first is that the interface force file could not support NFAIL = 1 on *DATA-BASE_EXTENT_-INTFOR because the intfor file does not expect new segments to replace the old, so it just undeletes the old segments instead of adding the new. The second problem is that when non-eroding contact is used, we only have enough memory in fixed length arrays for the segments that exist at t = 0. When segments are deleted, I was using the space that they vacated to create new segments, but it was very likely that some segments could not be created when the number of open spaces was less than the number of new segments that are needed. In this case, some segments would not be created and there would be surfaces that could be penetrated with no resistance. This behavior is impossible to predict, so it seems better to prevent any new segments from being created unless eroding contact is used.
- Fixed rcforc output for MPP 2D automatic contact. The forces across processors were missed.

- Fixed a segment-based contact error in checking airbag segments. This affects only airbags that are defined by control volumes, that is defined by *AIRBAG. The symptom was a segmentation fault.
- Fixed SMP eroding segment based (SOFT = 2) contact which was not activating the negative volume checking of brick elements. The MPP contact and the other SMP contacts were doing this but not SMP SOFT = 2.
- Fixed support for CNTCO on *CONTROL_SHELL by segment based (SOFT = 2) contact. It was adjusting the contact surface only half of what it should have done.
- Fixed eroding segment-based contact when used with the CNTCO > 0 on *CONTROL_CONTACT. A segmentation fault was occurring.
- Modified MPP segment based (soft = 2) contact to use R8 buffers to pass nodal coordinates. This should reduce MPP scatter when decomposition changes.
- Added support for using a box to limit the contact segments to those initially in the box when using eroding segment-based contact. The box option has not been available for any eroding contact up until now. (SOFT = 2 and SBOXID, MBOXID on *CONTACT_ERODING_...).
- Fixed force transducers with MPP segment-based contact when segments are involved with multiple, 2-surface force transducers. The symptom was that some forces were missed for contact between segments on different partitions.
- Added support for *ELEMENT_SOURCE_SINK used with segment-based contact. With this update, inactive elements are no longer checked for contact.
- Fixed an MPP problem in segment-based contact that cased a divide by zero during the bucket sort. During an iteration of the bucket sort, all active segments were somehow in one plane which was far from the origin such that a dimension rounded to zero. The fix for this should affect only this rare case and have no effect on most models.
- Modified segment based (SOFT = 2) contact to make SMP and hybrid faster, particularly for larger numbers of processors.
- Fixed thermal MPP segment-based contact. The message passing of thermal energy due to friction was being skipped unless peak force data was written to the intfor file.
- Fixed likely memory errors in MPP problems with 2D automatic contact when friction is used.
- Support the VC parameter (coefficient for viscous friction) in the case of segment-based contact, which has previously been unsupported. This option will work best with FNLSCL > 0, DNLSCL = 0 on optional card D. The card D option causes the contact force to be proportional to the overlap area which causes even pressure distribution.
- Enabled segment-based contact (SMP and MPP) to work with type 24 (27-node) brick elements.

- Fixed MPP segment-based contact for implicit solutions. During a line search, some data was not restored correctly when the solver goes back to the last converged state. This caused possible memory errors.
- Fixed friction for MPP segment-based contact in the implicit solver. The sliding velocity was calculated incorrectly using the explicit time step rather than the implicit step.
- Fixed a bug in MPP *CONTACT_2D_AUTOMATIC..., where a flaw in code used during MPP initialization could cause segments to fail to detect penetration.
- Fixed the thick beam checking of *CONTACT_2D_AUTO-MATIC_SINGLE_SURFACE in the MPP version. There was a memory error that could occur if thick beams were in the model.
- New values for user friction element history (*USER_INTERFACE_FRICTION): material directions, relative velocity components and yield stress.
- Add new user-defined interface for tiebreak contact invoked by *CONTACT_-AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK_USER.
- MORTAR CONTACT
  - PENMAX and SLDTHK has taken over the meanings of SST and TKSLS in R9 and earlier, although in a different way. Now PENMAX corresponds to the maximum penetration depth for solid elements (if nonzero, otherwise it is a characteristic length). SLDTHK is used to offset the contact surface from the physical surface of the solid element, instead of playing with SST and TKSLS, which was rather awkward. This update also saves the pain of having to treat shells and solids in separate interfaces if these features are wanted. This changes the behavior in some inputs that did have SST turned on for solids, but a necessary measure to make the contact decent for future versions.
  - The characteristic length for solid elements has been revised to not result in too small sizes that would lead to high contact stiffnesses and less margin for maximum penetration.
  - SFS on CONTACT_..._MORTAR can be input as negative, then contact pressure is the -SFS load curve value vs penetration.
  - Smooth roundoffs of sharp edges in MORTAR contact has been extended to high order segments, meaning that edge contact is valid even in this case.
  - The MORTAR contact now honors the NLOC parameter for shells, see *SECTION_SHELL, adjusting the contact geometry accordingly. Note that CNTCO on *CONTROL_SHELL applies as if always active, meaning that if NLOC is on, then CNTCO will also be "on" for MORTAR contacts.
  - Output of contact gaps to the intfor file is now supported for MORTAR contact, see *DATABASE_EXTENT_INTFOR.

- Transducer contacts, *CONTACT_..._FORCE_TRANSDUCER, are supported for MORTAR contact in SMP and MPP. A disclaimer is that the slave and master sets in the transducer have to be defined through parts or part sets. Warnings are issued if this is violated.
- Option 2 is now supported for tiebreak MORTAR contact, *CONTACT_..._MORTAR_TIEBREAK, but only for small sliding. Options 4 and 7 are supported in the MORTAR tiebreak contact for any type of sliding.
- For explicit analysis, the bucket sort frequency for MORTAR contact is 100, but can be changed through parameter BSORT on the CONTACT_..._MORTAR card or NSBCS on CONTROL_CONTACT. Note that the MPP bucket sort parameter does not apply. This assumes to improve the efficiency of MORTAR explicit contact significantly compared to R9 and earlier versions.
- Dynamic friction is supported in MORTAR contact for explicit and implicit dynamic analysis. See FD and DC on *CONTACT_... card.
- Wear calculations are supported for the MORTAR contact. See CONTACT_ADD_WEAR.
- Triangular shell form 24 is supported with MORTAR forming contact and accounts for high order shape functions.
- Automatic MORTAR contact now supports contact with end faces of beam elements and not just the lateral surfaces.
- Mortar contact is available in 2D plane strain and axisymmetric simulations, but only for SMP implicit. See CONTACT_2D_...MORTAR.

◦ Wear computed from *CONTACT_ADD_WEAR can optionally be output to dynain on optional card of *INTERFACE_SPRINGBACK_LSDYNA. This will generate *INITIAL_CONTACT_WEAR cards for subsequent wear simulations, and LS-DYNA will apply this wear and modify geometry accordingly. Restrictions as described in the manual apply.

◦ Improve SOFT = 6 under *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE to allow users to define part ID and a node set is automatically generated.

- ***CONSTRAINED**

◦ Add frictional energy calculation for constraint-based rigid walls.

◦ *CONSTRAINED_BEAM_IN_SOLID:

- Works with r-adaptivity now.
- Can now constrain beams in tshells as well as solids.

◦ Fix a bug for *CONSTRAINED_LOCAL that might mistakenly constrain z-translation when RC = 0.

- The following options do not support MEMORY = auto properly. The MEMORY = auto option will be turned off in this section and report an error if additional memory allocation is needed.
  *CONSTRAINED_LINEAR_OPTION
  *CONSTRAINED_MULTIPLE_GLOBAL
- Switched translational joints with stiffness to use double precision storage for the displacement value so that the calculated forces are more accurate. This prevents round-off error that can become significant.
- Fixed *CONSTRAINED TIED_NODES_FAILURE when used with MPP single surface segment based contact. Non-physical contact between segments that share tied constraints was being penalized leading to failure of the constraints.
- The SPR models (*CONSTRAINED_SPR2, *CONSTRAINED_INTER-POLATION_SPOTWELD) now support the SPOTDEL option of *CONTROL_-CONTACT. That means if shell elements involved in the SPR domain fail, the SPR gets deactivated.

- ***CONTROL**

  - Fix possible error termination with single precision MPP when PSFAIL.ne.0 in *CONTROL_SOLID and using solid formulation 10/13/44.
  - Fix spurious deletion of elements when using TSMIN.ne.0.0 in *CONTROL_TERMINATION, erode = 1 in *CONTROL_TIMESTEP and initialized implicitly in dynamic relaxation.
  - Added keyword *CONTROL_ACOUSTIC to calculate the nodal motions of *MAT_ACOUSTIC nodes for use in d3plot and time history files. Without this option the *MAT_ACOUSTIC mesh propagates pressure but does not deform because it uses a linear Eulerian solution method. The structural response is unaffected by this calculation; it is only for visualization and will roughly double the time spent computing acoustic element response.
  - When IACC = 1 on *CONTROL_ACCURACY and for shell type 16/-16 in nonlinear implicit, shell thickness change due to membrane strain when ISTUPD > 0 in *CONTROL_SHELL is now included in the solution process and will render continuity in forces between implicit time steps. The output contact forces will reflect the equilibrated state rather than the state prior or after the thickness update.
  - Fix bug when RBSMS in *CONTROL_RIGID, affecting mass scaled solutions, is used in conjunction with *ELEMENT_INERTIA and/or *PART_INERTIA, specifically with choices of IFLAG on *CONSTRAINED_RIGID_-BODIES and *CONSTRAINED_EXTRA_-NODES.
  - Tshells added to the subcycling scheme (*CONTROL_SUBCYCLE).
  - Tshells and spotweld beams are supported in selective mass scaling. See IMSCL in *CONTROL_TIMESTEP.
  - Add a new keyword: *CONTROL_FORMING_SHELL_TO_TSHELL to convert shell elements to tshell elements.

- If a parent node has SPCs, the same SPC constraints will be applied to the corresponding tshell nodes.
- If adaptivity is invoked, *BOUNDARY_SPC_SET is automatically updated to include newly generated nodes.
- Allows the normal of the segment set to be changed.
- Can offset the generated tshells from the mid-surface of the parent shells.
- Automatically generate segment sets for the top and bottom surfaces, which can be used for contact.

- **DISCRETE ELEMENT METHOD**

  - Implement generalized law of erosion for *DEFINE_DE_TO_SURFACE_COUPLING based on the following article in the journal "Wear": Magnee, A., Generalized law of erosion: application to various alloys and intermetallics, Wear, Vol. 181, 500, 1995.
  - Modify tangential force calculation to get better rigid body rotation behavior for *DEFINE_DE_BOND
  - Support restart feature for DEM interface force file and DATABASE output.
  - Instead of using bulk modulus, use mass and time step to estimate contact stiffness for SPH-DEM coupling. This should be better if DEM material is quite different from SPH material.
  - Fix *DEFINE_DE_MASSFLOW_PLANE bug if DE injection is defined.
  - Add CID_RCF to *DEFINE_DE_TO_SURFACE_COUPLING for force output in local coordinates to 'demrcf' file.
  - Update the *DEFINE_DE_BY_PART card so that it matches the capabilities of the *CONTROL_DISCRETE_ELEMENT card.
  - Add penalty stiffness scale factor, thickness scale factor, birth time and death time to *DEFINE_DE_TO_SURFACE_COUPLING.
  - Add dynamic coefficient of friction to *CONTROL_DISCRETE_ELEMENT.
  - Implement Finnie's wear law and user defined wear model to *DEFINE_DE_TO_SURFACE_COUPLING.
  - Implement user-defined curve for DEM frictional coefficient as function of time.
  - Implement user-defined curve for contact force calculation for *CONTROL_DISCRETE_ELEMENT.
  - Fix inconsistent results between *DEFINE_DE_BY_PART and *CONTROL_DISCRETE_ELEMENT.

- ***ELEMENT**

  - Fixed bug affecting output from beam elements ELFORM = 2 when certain uncommon inputs are present. Forces and moments in the output files could be wrongly rotated about the beam axis. This affected the output files only, not the solution inside LS-DYNA. The error could occur under two

circumstances: (a) if IST on *SECTION_BEAM is non-zero, the output forces and moments are supposed to be rotated into the beam's principal axis system, but this rotation could be applied to the wrong beam elements; and (b) when no ELFORM = 2 elements have IST, but the model also contains beams with ELFORM = 6 and RRCON = 1 on the SECTION_BEAM card, some of the ELFORM = 2 elements can have their output forces and moments rotated by 1 radian.

- Fix a bug affecting 2d seatbelt with time-dependent slipring friction.
- Fix erroneous 1d seatbelt slipring message.
- Fix seatbelt consistency issue in SMP (ncpu < 0).
- Add error message when 2d seatbelt part doesn't have shell formulation of 5 and *MAT_SEATBELT.
- Fix a bug for 2d seatbelt that could occur when a model has both 1d and 2d belts, and a 1d pretensioner of type 2, 3 or 9.
- Fix an MPP seatbelt bug that could occur when using a type 9 pretensioner.
- Allows shell formulation 9 to be used for 2d seatbelt. It was reset to formulation 5 by LS-DYNA, no matter what formulation was input. Now, only formulation 5 and 9 are accepted as input. Other formulations will incur error message.
- MPP now supports *ELEMENT_MASS_MATRIX_NODE_(SET).
- Added cohesive shell formulation -29. This formulation uses a cohesive midlayer where local direction q1 coincides with the average of the surrounding shell normals. This formulation is better suited for simulating normal shear.
- Cohesive shell formulation +/-29: Fixed absence of part mass in d3hsp.
- Make *TERMINATION_DELETED_SOLIDS work with hex spot weld failures.
- Fix incorrect load curve used if large value is used for FC < 0 and/or FCS < 0 in *ELEMENT_SEATBELT_SLIPRING.
- Fix incorrect velocity on accelerometer if

  - velocity is prescribed on the rigid body that the accelerometer is attached to, and
  - INTOPT = 1 in *ELEMENT_SEATBELT_ACCELEROMETER, and
  - *INITIAL_VELOCITY_GENERATION_START_TIME is used.

- Fix incorrect discrete spring behavior when used with adaptivity.
- Fix input error when using *DEFINE_ELEMENT_DEATH with BOXID > 0 for MPP.
- Modify tolerances on error messages SOL+865 and SOL+866 to prevent unnecessary error terminations when translational or rotational mass of a discrete beam was close to zero.
- Made the solid element negative volume warning SOL+630 for penta formulatgion 15 consistent with the volume calculation in the element. With this change, elements are deleted rather than the job terminating with error SOL+509.

- Fixed the default hourglass control for shell form 16. It was defaulting to type 5 hourglass control rather than 8.
- Fixed default hourglass control when the *HOURGLASS control card is used but no HG type is specified. We were setting to type 1 instead of 2. Also, fixed the default HG types to match the user's manual for implicit and explicit.
- Fixed the fully integrated membrane element (shell ELFORM = 9) when used with NFAIL4 = 1 on *CONTROL_SHELL and there are triangular elements in the mesh. Triangular elements were being deleted by the distorted element check.
- Fixed a divide by zero error that occurred with *SECTION_BEAM, ELFORM = 6, SCOOR = 12 or -12, and node 3 was omitted on *ELEMENT_BEAM, and nodes 1 and 2 are along the global y-direction or z-direction.
- Fixed laminated shell theory for type 6 and 7 shell elements when made active by LAMSHT = 3 or 5 on *CONTROL_SHELL.
- Added an int.pt. variable for *PART_COMPOSITE_LONG and *PART_COMPOSITE_TSHELL_LONG called SHRFAC which is a scale factor for the out-of-plane shear stress that allows the user to choose the stress distribution through thickness. This was motivated by test data that shows that for large differences is layer shear stiffness, the parabolic assumption is poor.
- Fixed implicit hourglass stiffness in viscoelastic materials when used with tshell forms 5 or 6. The stiffness was much too small.
- Modified tshell type 5 to use the tangent stiffness for calculating the Poisson's affects and hourglass control for *MAT_024. This makes the behavior softer during buckling which is much more realistic.
- Fixed a significant bug in segment based contact when SHLEDG = 1 and SBOPT = 3 or 5 and DEPTH < 45, and shell segments in contact have different thicknesses. A penetration check was using incorrect thicknesses causing contact to be detected too late, particularly for edge to surface contact.
- Improved the time step calculation for triangular tshell elements. The time step was too conservative for elements with significant thickness. This fix does not affect tshell type 7.
- Fixed all tshells to work with anisotropic thermal strains which can be defined by *MAT_ADD_THERMAL. Also, this now works by layer for layered composites.
- Enabled tshell form 5 to recalculate shear stiffness scale factors when plasticity material models 3, 18, 24, 123, or 165 are included in a composite section. Prior to this change the scale factors were based on elastic properties so after yielding, the stress distribution was not what was expected. This new capability supports the constant stress option, the parabolic option, and the SHRFAC option on *PART_COMPOSITE_TSHELL_LONG.
- Improved tshell 5 when used with mixed materials in the layers. A failure to use the correct Poisson's ratio was causing a less accurate stress tensor.

- Modified the time step calculation for tshell forms 3 and 5. A dependence on volumetric strain rate was removed in order to prevent oscillations in the time step which caused stability problems, particularly for tshell 5.
- Fixed tshell constant shear stress option (TSHEAR = 1 on *SECTION_TSELL or *PART_COMPOSITE). It was producing a not very constant stress distribution.
- Fixed stress and strain output of tshells when the composite material flag CMPFLG is set on *DATABASE_EXTENT_BINARY. The transformation was backwards.
- Fixed mass of parts reported to d3hsp when *ELEMENT_SHELL_SOURCE_SINK is used. The mass of inactive elements was being included.
- Enabled *MAT_026 and *MAT_126 (HONEYCOMB) to be used with tshell forms 3, 5, and 7. It was initialized incorrectly causing a zero stress.
- Added a missing internal energy calculation for tshell form 6.
- Enabled tshell forms 1, 2, and 6 to work with material types 54, 55, and 56.
- Modified the z-strain distribution in tshell forms 5 and 6 when used in composites with mixed materials that are isotropic. The existing assumed strain scheme was doing a poor job of creating a constant z-stress through the thickness.
- Increased the explicit solution time step for thin shell composite elements. The existing method calculated a sound speed using the stiffness from the stiffest layer and dividing it by the average density of all layers. This could be overly conservative for composites with soft layers of low density. The new method uses the average stiffness divided by average density. This is still conservative, but less so.
- Corrected rotational inertia of thin shells when layers have mixed density and the outer layers are denser than inner layers. The fix will mostly affect elements that are very thick relative to edge length.
- Added support for *ELEMENT_SHELL_SOURCE_SINK to type 2 shells with BWC = 1 on *CONTROL_SHELL.
- Prevent inactive shell elements (from *ELEMENT_SHELL_SOURCE_SINK) from controlling the solution time step.
- Fixed *LOAD_STEADY_STATE_ROLLING when used with shell form 2 and Belytschko- Wong-Chang warping stiffness (BWC = 1 *CONTROL_SHELL). The load was not being applied.
- Improved the brick element volume calculation that is used by the option erode elements (ERODE = 1 on *CONTROL_TIMESTEP or PSFAIL.ne.0 on *CONTROL_SOLID). It was not consistent with the element calculation which caused an error termination.
- Fixed all tshell forms to work with anisotropic thermal strains which can be defined by *MAT_ADD_THERMAL. Also, this now works by layer for layered composites.
- Reworked shell output so that we can correctly output stress in triangular shells when triangle sorting is active, that is when ESORT = 1 or 2 on *CONTROL_SHELL.

- *ELEMENT_T/SHELL_COMPOSITE(_LONG) and *PART_COMPOSITE_-T/SHELL_(LONG): Permit the definition of zero thickness layers in the stacking sequence. This allows the number of integration points to remain constant even as the number of physical plies varies and eases post-processing since a particular integration point corresponds to a physical ply. Such a capability is important when plies are not continuous across a composite structure.
  To represent a missing ply, set THK to 0.0 for the corresponding integration point and additionally, either set MID = -1 or set PLYID to any nonzero value. Obviously, the PLYID option applies only to the keywords containing LONG.
- Implemented sum factorization for 27-node quadratic solid that may increase speed by a factor of 2 or 3.
- Support second order solid elements (formulations 23,24,25,26) for *SET_-NODE_GENERAL.
- Invoke consistent mass matrix of 27-node hex element for implicit dynamics and eigenvalues.
- Reorder node numbering when assembling global stiffness matrix for 27-node hex. This fixes a bug in which it was reported than the implicit 27-node element didn't work
- Automatically transfer nodal boundary conditions for newly generated nodes if H8TOH27 option is used in *ELEMENT_SOLID.
- Modify initialization of material directions for solid elements. If there are only zeros for all the 6 values in *INITIAL_STRESS_SOLID, then the values from the other input (e.g. *ELEMENT_SOLID_ORTHO) are kept.
- Enable *PART_STACKED_ELEMENTS to pile up shell element layers. Before, it was necessary that solid element layers were placed between shell element layers. Now, shell element layers can follow each other directly. Contact definitions have to be done separately.
- Allow *PART_STACKED_ELEMENTS to be used in adaptive refinement simulations.
- Add alternative mass calculation for critical time step estimate of cohesive elements. This hopefully resolves rarely occurring instability issues. Option ICOH on *CONTROL_SOLID is used for that.
- Correct the strain calculation for tet formulation 13. This did not affect the stress response, only output of strains. Nodal averaging was not accounted for.
- User defined elements (ELFORM = 101 to 105 on *SECTION_-SHELL/SOLID) can now be used together with *MAT_ADD_EROSION.
- Add option to define a pull-out force vs. time curve in *ELEMENT_BEAM_-SOURCE by defining a negative variable FPULL. |FPULL| can refer to *DEFINE_CURVE or *DEFINE_CURVE_FUNCTION.
- Solid tet form 13 supported for all materials in implicit, including a presumable consistency improvement for the future.

- The Hughes-Liu beam is supported in *INTEGRATION_BEAM such each integration point may refer to a different part ID and thus have a different coef. Of thermal expansion. See *MAT_ADD_THERMAL_EXPANSION.
- Shell types 2 and 16 that combines thermal expansion and thick thermal shells, see *MAT_ADD_THERMAL_EXPANSION and TSHELL on *CONTROL_SHELL, now correctly treat temperature gradient through the thickness to create bending moments. All shell types are to be supported in due time.
- *SECTION_BEAM_AISC now provides predefined length conversion factors for specific unit systems.
- 3D tet r-adaptivity now supports *DEFINE_BOX_ADAPTIVE.
  - For every adaptive part, users can define multiple boxes where different BRMIN & BRMAX (corresponding to RMIN & RMAX in *CONTROL_REMESHING) can be specified for 3D tet remesher to adjust the mesh size.
  - Current implementation does not support LOCAL option.
- Fix bug in 3D adaptivity so that users can now have both non-adaptive tshell parts and 3D adaptive parts in one analysis.
- Fix the bug in 3D adaptivity so that users can now have both dummy nodes and 3D adaptive parts in one analysis.

- ***EM (Electromagnetic Solver)**
  - Randles Circuits for Battery Modeling
    - A Randles circuit is an equivalent electrical circuit that consists of an active electrolyte resistance r0 in series with the parallel combination of the capacitance c10 and an impedance r10. The idea of the distributed Randles model is to use a certain number of Randles circuits between corresponding nodes on the two current collectors of a battery unit cell. These Randles circuits model the electrochemistry that happens in the electrodes and separator between the current collectors. The EM solver can then solve for the EM fields in the current collectors, and the connections between them.
    - Added analysis of distributed Randles circuits to MPP.
    - Added d3plot output for distributed Randles circuits:
      D3PL_RAND_r0_EM,
      D3PL_RAND_r10_EM,
      D3PL_RAND_c10_EM,
      D3PL_RAND_soc_EM,
      D3PL_RAND_i_EM,
      D3PL_RAND_u_EM,
      D3PL_RAND_v_EM,

D3PL_RAND_vc_EM,
D3PL_RAND_temperature_EM,
D3PL_RAND_P_JHR_EM,
D3PL_RAND_P_dudt_EM,
D3PL_RAND_i_vector_EM

This output can be visualized in LS-PrePost versions 4.3 and 4.5 on the separator part of the battery cell using Post/FriComp/Extend/EM node.

- Added tshells for EM analysis for use in battery modeling.
- Added new capability for modeling Randles short, based on *DEFINE_FUNCTION so that the user has a lot of freedom to define where and when the short happens as well as the short resistance.
- Added a new capability for battery exothermal reactions also based on *DEFINE_FUNCTION. The new keyword *RANDLE_EXOTHERMAL_REACTION makes it possible to complement the heating of a short circuit created by a short by exothermal reactions if, for example, the temperature becomes higher than a threshold value.

- **FORMING ANALYSIS**

  - Extend *INCLUDE_AUTO_OFFSET to solid and beam elements (draw beads).
  - Add a new keyword for springback compensation: *INTER-FACE_COMPENSATION_NEW_REFINE_RIGID to refine and break rigid tool mesh along the user supplied trim curves so compensated tool mesh follows exactly the blank mesh (file "disp.tmp"). This needs to be done only once in the beginning of the springback compensation (ITER0).
  - *CONTROL_FORMING_ONESTEP:

    - Change the default element formulation option for onestep method to QUAD2.
    - Add a new option QUAD to allow quadrilateral elements to be considered.
    - Limit the maximum thickening by using a new variable TSCLMAX for the sheet blank.
    - Set the value of OPTION to a negative value to output the file 'onestepresult' in large format (E20.0).
    - Calculate and add the damage factor and output to the 6th history variable in the output file "onestepresult". Add the variable for a curve ID to define the fracture strain vs. triaxility. Add another variable DMGEXP (damage parameter), as used in GISSMO model.
    - Keep the original coordinates for the onestep output "onestepresults".

  - Add a new option VECTOR to *CONTROL_FORMING_BESTFIT to output deviation vector (in the format of: NODEID, xdelta, ydelta, zdelta) for each

node to its closest target element. The deviation vectors are output under the keyword *NODE_TO_TARGET_VECTOR.

- *CONTROL_FORMING_OUTPUT:
  - Output will skip any negative abscissa (Y1) value.
  - When CIDT < 0, the positive value defines the time dependent load curve.
- Add a warning in springback compensation *INTERFACE_SPRING-BACK_-COMPENSATION to identify which input file (typically the blank with adaptive mesh not output directly by LS-DYNA) has the wrong adaptive constraints.
- *INTERFACE_COMPENSATION_3D: turn off the output of nikin file.
- *ELEMENT_LANCING:
  - Allow some unused lancing curves to be included in the input.
  - When the gap between the two ends of a lancing curve is not zero, but small enough, then this curve is automatically closed.
  - Allow several parts to be cut during lancing; the parts can be grouped in *SET_PART_LIST, and defined using a negative value IDPT.
  - Specify the distance to bottom dead center as AT and ENDT when the new variable CIVD is defined.
  - Set IREFINE = 1 (default) in lancing, to refine blank mesh automatically along the lancing curves.
  - Re-set the adaptive level to be 1 to prevent those elements along the lancing route to be further refined.
  - When IREFINE = 1, elements along the lancing curve will be refined to make sure that no adapted nodes exist in the neighborhood. This helps get improved lancing boundary.
  - Change of tolerance for lancing to merge the small elements into bigger ones.
- Add a new keyword to perform trimming after lancing (shell elements only): *DEFINE_LANCE_SEED_POINT_COORDINATES. Maximum of two seed nodes can be defined.
- Extend *CONTROL_FORMING_TOLERANC to *MAT_036, *MAT_037, *MAT_125, and *MAT_226. When beta is less than -0.5, there is no necking and no calculation of FI. When beta is greater than 1.0, beta = 1.0/beta. This keyword adds a smoothing method to calculate the strain ratios for a better formability index.
- Sandwiched parts (*CONTROL_ADAPTIVE, *DEFINE_CURVE_TRIM):
  - Disable *CONTROL_ADAPTIVE_CURVEs for sandwich parts, since refinement along the curve is automatically done during trimming.

        - Refine the elements along the trimming curve to make sure no slave nodes are be cut by trimming curves.
        - Allow mesh adaptivity.
        - Allow multi-layers of solids.
        - Add a check to the variable IFSAND in *CONTROL_ADAPTIVE for sandwich part to be refined to exclude solid elements.

    - Solid element trimming (*DEFINE_CURVE_TRIM):

        - Refine those elements along the trimming curve.
        - Improve solid trimmig to allow the trimming of one panel into two panels with two seed nodes.

    - Add a new keyword *CONTROL_FORMING_REMOVE_ADAPTIVE_-CONSTRAINTS to remove adaptive constraints on a formed, adapted blank, and replaced them with triangular elements.
    - *DEFINE_CURVE_TRIM_NEW: Allow trimming of tshells.
    - Add a new keyword: *INTERFACE_WELDLINE_DEVELOPMENT to obtain initial weld line from the final part and the final weld line position.

        - When Ioption = -1, convert weld line from its initial position to the final part.
        - Output the element nodes that intersect the weld line in the final part, and the output file is: affectednd_f.ibo
        - Output the element nodes that intersect the weld line in the initial part, and the output file is: affectednd_i.ibo

    - Add a new variable DT0 to *CONTROL_IMPLICIT_FORMING so there is no need to use *CONTROL_IMPLICIT_GENERAL to specify DT0.
    - *INTERFACE_BLANKSIZE:

        - Add a new feature DEVELOPMENT option. When ORIENT = 2, then a reference mesh file for the formed part should be included. The calculated and compensated boundary will be based on the reference mesh.
        - Add a new option SCALE_FACTOR that allows the target curve to be moved. This is useful when multiple target curves (e.g. holes) and formed curves are far away from each other.

- ***FREQUENCY_DOMAIN**

    - Added new keyword *CONTROL_FREQUENCY_DOMAIN to define global control parameters for frequency domain analysis. Currently two parameters are defined:

- REFGEO: flag for reference geometry in acoustic eigenvalue analysis (either the original geometry at t = 0, or the deformed geometry at the end of transient analysis).
- MPN: large mass added per node, to be used in large mass method for enforced motion.

◦ *FREQUENCY_DOMAIN_ACOUSTIC_BEM:

- Enabled using incident wave (*FREQUENCY_DOMAIN_ACOUSTIC_INCIDENT_WAVE) in Rayleigh method (METHOD = 0).
- Enabled acoustic pressure fringe plot (*FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT) in Rayleigh method (METHOD = 0).
- Fixed bug in running acoustic analysis with multiple boundary conditions in MPP.
- Fixed running MATV (Modal Acoustic Transfer Vector) approach in MPP (*FREQUENCY_DOMAIN_ACOUSTIC_BEM_MATV).
- Added treatment for triangular elements used in Rayleigh method (METHOD = 0).
- Added output of acoustic intensity to binary database D3ACS (defined by *DATABASE_FREQUENCY_BINARY_D3ACS).
- Fixed bug in acoustic pressure fringe plot for collocation BEM (METHOD = 3) and dual BEM based on Burton-Miller formulation (METHOD = 4).

◦ *FREQUENCY_DOMAIN_ACOUSTIC_FEM:

- Fixed bug in acoustic analysis by FEM, when dimensions of mass and k (stiffness) matrices are mismatched.

◦ *FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT:

- Implemented acoustic fringe plot for MPP for the options PART, PART_SET, and NODE_SET.

◦ *FREQUENCY_DOMAIN_FRF:

- Added new loading types:
  VAD1 = 5: enforced velocity by large mass method
  = 6: enforced acceleration by large mass method
  = 7: enforced displacement by large mass method
  = 8: torque
  = 9: base angular velocity
  = 10: base angular acceleration
  = 11: base angular displacement
- Added rotational dof output for FRF.

- *FREQUENCY_DOMAIN_MODE:
  - Added option _EXCLUDE to exclude some eigenmodes in modal superposition in frequency domain analysis.
- *FREQUENCY_DOMAIN_RANDOM_VIBRATION:
  - Fixed bug in running random vibration with random pressure wave load (VAFLAG = 2) in MPP.
  - Improved random vibration analysis by allowing using complex variable cross PSD functions. Previously cross PSD was defined as real variables thus the phase difference was ignored.
  - Added PSD and RMS computation for Von Mises stress in beam elements.
- *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM:
  - Added Von Mises stress output for beam elements in database D3SPCM.
  - Corrected computation of response spectrum at an intermediate damping value by interpolating spectra at two adjacent damping values. Now the algorithm is based on ASCE 4-98 standard.
- *FREQUENCY_DOMAIN_SSD:
  - Added new loading types:
    VAD = 5: enforced velocity by large mass method
    = 6: enforced acceleration by large mass method
    = 7: enforced displacement by large mass method
    = 8: torque
  - Fix for running SSD fatigue in MPP (affected keyword: *FREQUENCY_DOMAIN_SSD_FATIGUE).
  - Updated ssd computation with local damping, and enabled the restart feature by reading damping matrix.
  - Implemented ERP (Equivalent Radiated Power, keyword *FREQUENCY_DOMAIN_SSD_ERP) for MPP.
- *DATABASE_FREQUENCY_ASCII:
  - Added keyword *DATABASE_FREQUENCY_ASCII_{OPTION} to define the frequency range for writing frequency domain ASCII databases NODOUT_SSD, ELOUT_SSD, NODOUT_PSD and ELOUT_PSD.

- ***ICFD (Incompressible Fluid Solver)**
  - New ICFD features and major modifications

- Simple restart is now supported for ICFD.
- Added wave damping capabilities. See *ICFD_DEFINE_WAVE_-DAMPING.
- Added steady state solver. See *ICFD_CONTROL_GENERAL and *ICFD_CONTROL_STEADY.
- Added steady state potential flow solver. See *ICFD_CONTROL_-GENERAL.
- Weak thermal coupling for conjugate heat transfer is now possible in addition to the classic monolithic approach. See *ICFD_CONTROL_-CONJ.
- Windkessel boundary conditions are now available for blood flow. See *ICFD_BOUNDARY_WINDKESSEL.
- It is now possible to output the heat transfer coefficient as a surface variable in LSPP or in ASCII format on segment sets for a subsequent solid-thermal only analysis. See *ICFD_DATABASE_HTC.
- Two way coupling is now possible with DEM particles. See *ICFD_-CONTROL_DEM_COUPLING.
- Modifications introduced in the SUPG stabilization term used in thermal and conjugate heat transfer problems for improved accuracy and speed.

◦ Additions and modifications to existing ICFD keywords

- *ICFD_BOUNDARY_FSWAVE:
  Added a boundary condition for wave generation of 2nd order stokes waves with free surfaces
- *ICFD_CONTROL_FSI:
  Added a flag which, when turned on will project the nodes of the CFD domain that are at the FSI interface onto the structural mesh. This is recommended for cases with rotation.
- *ICFD_CONTROL_MESH:
  Added a flag to allow the user control over whether there will be re-mesh or not. If there is no re-mesh then we can free space used to backup the mesh and lower memory consumption.
- *ICFD_CONTROL_MESH_MOVE:
  Added option to force the solver to turn off any mesh displacements. This can be useful in cases where the mesh is static to save a little bit of calculation time.
- *ICFD_CONTROL_OUTPUT:
  Added option to support output in Fieldview format, binary and ASCII.
  When output of the fluid volume mesh is requested, the mesh will be divided into ten distinct parts, grouping elements in ten deciles based on the mesh quality (Part 1 has the best quality elements, part 10 the worst).

- *ICFD_CONTROL_POROUS:
  Improvements for RTM problems.
- *ICFD_CONTROL_TIME:
  Added an option to define an initial timestep.
  Added an option to shut off the calculation of Navier Stokes after a certain time leaving only the heat equation. This can be useful to save calculation times in conjugate heat transfer cases where the fluid often reaches steady state before the thermal problem.
- *ICFD_DATABASE_DRAG:
  It is now possible to output the force on segment sets in a FSI run directly in LS-DYNA compatible format. This can be useful for a subsequent linear FSI analysis running only the solid mechanics part.
  Added flag to output drag as a surface variable in LSPP.
- *ICFD_DATABASE_FLUX:
  Added option to change output frequency
- *ICFD_DATABASE_NODOUT:
  The user node IDs are now required rather than the internal node IDs
- *ICFD_CONTROL_IMPOSED_MOVE:
  Added the option to choose between imposing the displacements or the velocity.
- *ICFD_CONTROL_TRANSIENT:
  Choose implicit time integration scheme for NS.
- *ICFD_CONTROL_DEM_COUPLING:
  Added a scale factor for the sphere radius in the computation of the DEM force.
- *ICFD_MODEL_POROUS:
  Added a scale factor option on the permeability for model 1 and 2. A *DEFINE_FUNCTION can also be used.
- *MESH_BL:
  Added option to generate boundary layer mesh using a growth factor.

- ICFD bug fixes and minor improvements

  - Fixed bug when multiple *DEFINE_FUNCTIONs were used in an ICFD problem. Only the last one was taken into account.
  - LES turbulence model: fixed van Driest damping issue in the boundary layer. LES models can use wall functions.
  - RANS turbulence models: Standard k-epsilon, realizable k-epsilon, Wilcox k-omega uses HRN laws of the wall by default while SST and Spalart Allmaras use LRN. Improvements on the convergence of all those models.
  - The DEM particle volume is now taken into account in free surface problems.
  - Average shear is now output as a surface variable in the d3plots.

- *ICFD_CONTROL_MONOLITHIC is obsolete (replaced by *ICFD_-CONTROL_GENERAL).
- Added more output for the mesh generation indicating the stage of the meshing process and the amount of elements that are being generated as a multiple of 10000. Added progress % for the extrusion of the mesh during the BL mesh generation.
- Improvements on the element assemble speed in MPP.
- Fixed synchronization problem for the last timestep in an FSI problem.
- More options have been added to the timer output.
- Correction of the calculation of the flux in *ICFD_DATABASE_FLUX in free surface cases.
- Boundary layer mesh can go through free-surfaces or mesh size interfaces.
- The Center of Gravity of the fluid is output in the icfd_lsvol.dat ASCII file in free surface problems

- **Implicit (Mechanical) Solver**

  - Enhanced termination of MPP eigensolver when non eigenmodes are found.
  - Implicit was enforcing birth and death times on *BOUNDARY_SPC during dynamic relaxation contrary to the User's Manual. These times are now ignored by implicit during dynamic relaxation.
  - Corrected output of eigenvalues and frequencies to file eigout for the asymmetric eigenvalue problem.
  - Enhanced logic that determines when to write out the last state to d3plot for implicit.
  - Improved error message for reading d3eigv file for *PART_MODES for the case when the user inputs a d3eigv file from a different model than intended.
  - Corrected the reporting of kinetic and internal energy in file glstat for implicit.
  - Applied corrections to tied contact in implicit (MPP). This affects slave nodes coming from other processes.
  - Corrected output to file d3iter (implicit nonlinear search vectors) for restart.
  - Enhanced termination process when the implicit solver determined an early termination.
  - When implicit springback was following an explicit transient step, the implicit keywords with the _SPR were not properly handled. This is now corrected.
  - Added a warning about the combined use of *CONSTRAINED_RIGID_-BODY_STOPPERS and the Lagrange multiplier formulation for joints (*CONTROL_RIGID) for explicit. The warning recommends switching to the penalty joint formulation.
  - Applied numerous bug fixes to the implicit solver associated with *CONSTRAINED_INTERPOLATION where there are lots of independent degrees-of-freedom.

- Corrected initialization of MPP tied contact with implicit mechanics when the implicit phase follows explicit dynamic relaxation.
- Fixed an implicit problem where a linear implicit analysis follows inertia relief computation.
- Added gathering of damping terms from discrete elements from implicit especially for FRF computations and matrix dumping.
- Fixed Implicit for the case of multi-step Linear (NSOLVR = 1) with Intermittent Eigenvalue Computation.
- Corrected the output to d3iter when 10-noded tets are present.
- Keypoints specified in *CONTROL_IMPLICIT_AUTO are now enforced at the initial time step and on restart from explicit.
- Skip frequency damping during implicit static dynamic relaxation, i.e. IDRFLAG > 5.
- *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS:
  - The VID of the rotating axis can now be defined by both *DEFINE_VECTOR and *DEFINE_VECTOR_NODES. It enables the movement of the rotating axis. Previously, only *DEFINE_VECTOR could be used to define the VID.
  - The rotational dynamics now work in MPP.
- Shell forms 23 and 24 (high order shells), 1D seatbelts, Hughes-Liu and spotweld beams (types 1 and 9) are now supported with the implicit accuracy option (IACC = 1 in *CONTROL_ACCURACY) to render strong objectivity for large rigid body rotations. Also, shell type 16 is supported with implicit accuracy option, resulting in forms 16 and -16 giving the same solution.
- Translational and generalized stiffness joints are now strongly objective for implicit analysis. See CONSTRAINED_JOINT_STIFFNESS....
- In implicit it may happen that the initial loads are zero, for instance in forming problems. In addition, the goal is to move a tool in contact with a workpiece, and the way line search and convergence works, it is hard to get things going. We now attempt to handle this situation by automatically associating an augmented load to the prescribed motion simply to get off the ground.
- New tolerances on maximum norms are introduced for convergence in implicit: ratio of max displacement/energy/residual, and absolute values of nodal and rigid body translation/rotational residual can be specified. See DNORM.LT.0 on *CONTROL_IMPLICIT_SOLUTION for defining an additional card for these parameters DMTOL, EMTOL and RMTOL. Furthermore, maximum absolute tolerances on individual nodal or rigid body parameters can be set on NTTOL, NRTOL, RTTOL and RRTOL on the same card.
- If ALPHA < 0 on first *CONTROL_IMPLICIT_DYNAMICS card, the HHT implicit time integration scheme is activated.

- ***INITIAL**

- Fix *INITIAL_VELOCITY_GENERATION when used with *INCLUDE_-TRANSFORM, which was broken due to misplaced conditionals in r100504.
- Fix 3 bugs for *INITIAL_VELOCITY_GENERATION involving omega > 0 and icid > 0:

  - When nx = -999. Now the directional cosine defined by node NY to node NZ will be the final direction to rotate about. In other words, the direction from node NY to node NZ will not be projected along icid any more.
  - When nx != -999, (xc,yc,zc) should not be rotated along icid, since (xc,yc,zc) are global coordinates.
  - When *INITIAL_VELOCITY_GENERATION is included by *IN-CLUDE_TRANSFORM, (xc,yc,zc) is transformed.

- Add the option of ramping time steps, ndtrrg, for *INITIAL_FOAM_REF-ERENCE_GEOMETRY. The solid elements with reference geometry and ndtrrg > 0 will restore its reference geometry in ndtrrg time steps.
- Fix incorrect initial velocity when ICID.ne.0 in *INITIAL_VELOCITY_GEN-ERATION, and rotational velocity, omega, is not zero and *PART_INERTIA is also present.
- Add variable IZSHEAR in *INITIAL_STRESS_SECTION to initialize shear stress.
- Fix incorrect initial velocity for *INITIAL_VELOCITY if IRIGID = -2 and ICID > 0.
- Fix incorrect NPLANE and NTHICK for *INITIAL_STRESS_SHELL when writing dynain for shell type 9.
- Fix *INITIAL_STRAIN_SHELL output to dynain for shell types 12 to 15 in 2D analysis. Write out strain at only 1 intg point if INTSTRN = 0 in *INTER-FACE_SPRINGBACK_LSDYNA and all strains at all 4 intg points if INTSTRN = 1 and nip = 4 in *SECTION_SHELL.
- Skip transformation of the initial velocities if ICID > 0 and *INCLUDE_-TRANSFORM is used to transform the keyword input file with the *INI-TIAL_VELOCITY.... keyword. Also echo warning message, KEY+1109, that the transformation will be skipped since icid is specified.
- Fix incorrect transformation of *DEFINE_BOX which results in incorrect in-itial velocities if the box is used in *INITIAL_VELOCITY.
- Fixed *INITIAL_STRESS_DEPTH when used with 2D plane strain and ax-isymmetric elements. The prestress was being zeroed.
- Improved the precision of the initial deformation calculation for *INITIAL_-FOAM_REFERENCE_GEOMETRY in the single precision version.
- Fixed stress initialization (*INITIAL_STRESS_SECTION) for type 13 tet ele-ments. The pressure smoothing was causing incorrect pressure values in the elements adjacent to the prescribed elements.
- Add _SET option to *INITIAL_STRESS_SOLID for element sets.

- Fix bug in 3D adaptivity so that users can now define *INITIAL_TEMPERATURE for adaptive parts.

- **Isogeometric Elements**

  - The stability of the trimmed NURBS shell patches has been improved.
  - Add *LOAD_NURBS_SHELL to apply traction type loading directly on the surface of NURBS shell.
  - Users can use the PART option of *SET_SEGMENT_GENERAL to define segment set of a NURBS patch. The segment set will contain all segments of interpolated null shell elements.
  - *ELEMENT_SOLID_NURBS_PATCH:

    - Isogeometric solid analysis implemented for MPP.
    - Isogeometric solid analysis implemented for SMP with multiple CPUs, including consistency (ncpu < 0).
    - Activate user-defined materials for isogeometric solid.

  - *ELEMENT_SHELL_NURBS_PATCH:

    - Isogeometric shell analysis now implemented for SMP with multiple CPUs, including consistency (ncpu < 0).
    - Add a power iteration method to get the maximum eigen-frequency for each isogeometric element. This will be used to set a reasonable time step for trimmed elements.

  - *ELEMENT_SHELL_NURBS_PATCH:

    - Changed the way of projecting the results from isogeometric (NURBS) elements to the interpolation elements. Now a background mesh, spanned over the locations of the integration points of the isogeometric (NURBS) elements serves as basis to interpolate results from the integration points to the centroid of the interpolation elements. This change may lead to slightly different post-processing results in the interpolation elements.

  - Add support for trimmed NURBS to work in single precision. Anyway, it is still recommended to use double precision versions for trimmed NURBS patches.
  - Add post-processing of strains and thickness for interpolation shells.

- ***LOAD**

  - Fixed bugs affecting discrete beam elements (ELFORM = 6) when used with staged construction. Here, "dormant" refers to elements that have not yet

become active as defined on *DEFINE_STAGED_-CONSTRUCTION_-PART.

  - Dormant discrete beams could still control the timestep and attract mass-scaling, when they should not do so.
  - Dormant discrete beams reaching a failure criterion defined on the *MAT card were deleted, when they should not be.
  - The displacements output (see *DATABASE_DISBOUT) included displacements occurring while the elements were dormant. Now, the output displacements are reset to zero at the moment the element becomes active.

- Fixed bug in Staged Construction: if FACT on *CONTROL_-STAGED_-CONSTRUCTION had been left blank, and Dynamic Relaxation was active, an error termination occurred.
- Fixed bug: *LOAD_GRAVITY_PART (and also gravity loading applied by *DEFINE_STAGED_CONSTRUCTION_PART) was failing to account for non-structural mass when calculating gravity load: NSM on *SECTION_-BEAM and MAREA on *SECTION_SHELL.
- Fixed bug in *LOAD_VOLUME_LOSS: inconsistent results when run in SMP parallel.
- Fix bugs affecting *LOAD_SEGMENT_FILE:

  - Remove LOAD_SEGMENT_FILE file size limit (It used to be 200M).
  - Apply correct pressure on the shared boundary between processors.

- Fix GRAV = 1 in *PART which was not working correctly with *LOAD_-DENSITY_DEPTH. Make *LOAD_DENSITY_DEPTH work for Lagrangian 2D elements.
- Fix insufficient memory error, SOL+659, when using *LOAD_-ERODING_-PART_SET with MPP.
- Fix incorrect loading when using *LOAD_ERODING_PART_SET with BOXID defined.
- Added *LOAD_SUPERPLASTIC_FORMING for implicit analysis.
- *LOAD_SUPERPLASTIC_FORMING box option now works in MPP.

- ***MAT and *EOS**

  - *MAT_197 (*MAT_SEISMIC_ISOLATOR) could become unstable when the parameter DAMP was left at its default value. A workaround was to input DAMP as a small value such as 0.05. The timestep for *MAT_197 is now smaller than previously, irrespective of the DAMP setting, and the behavior is now stable even if DAMP is left at the default.
  - Fixed bug: Timestep calculation was wrong for *MAT_089 solid elements. Response could be unstable especially for higher values of Poisson's ratio, e.g. 0.4.

- Fixed bug: An error trap was wrongly preventing ELFORM = 15 for *MAT_-169 (*MAT_ARUP_ADHESIVE). Wedge elements with ELFORM = 15 are now permitted.
- *MAT_172 (*MAT_CONCRETE_EC2):
  Note that items (1) and (2) below can lead to different results compared to previous versions of LS-DYNA.

  - (1) The number of potential cracks in MAT_172 shell elements has been increased from 2 to 4. MAT_172 uses a fixed crack model: once the first crack forms, it remains at the same fixed angle relative to the element axes. Further cracks can then form only at pre-defined angles to the first crack. Previously, only one further crack could form, at 90 degrees to the first crack. Thus, if the loading direction subsequently changed so that the principal tension is at 45 degrees to the first crack, that stress could exceed the user-defined tensile strength by a considerable margin. Now, further cracks may form at 90, +45 and -45 degrees to the first crack. Although the maximum principal stress can still exceed the user-defined tensile strength, the "error" is much reduced. There is an option to revert to the 2-crack model as in R9 (to do this, add 100 to TYPEC).
  - (2) Add element erosion to MAT_172. This change may lead to different results compared to previous versions, because erosion strain limits are now added by default. Elements are now deleted when crack-opening strain becomes very large, or the material is crushed beyond the spalling limit. Plastic strain in the rebar is considered too. Previously, these elements that have passed the point of being able to generate any stress to resist further deformation would remain in the calculation, and sometimes showed very large non-physical deformations and could even cause error terminations. Such elements would now be deleted automatically. Default values are present for the erosion strains but these can be overridden in the input data, see new input fields ERODET, ERODEC, ERODER.
  - (3) New history variables 10,11,12 (maximum value so far of through-thickness shear stress). This is useful for checking results because MAT_172 cracks only in response to in-plane stress; before cracking occurs, the through-thickness shear capacity is unlimited. The data components are:
    Ex History Variable 10 - maximum out of 11 and 12
    Ex History Variable 11 - maximum absolute value of YZ shear stress
    Ex History Variable 12 - maximum absolute value of ZX shear stress
    These are in the element local axis system. Note that these variables are written only if TYPESC is zero or omitted. TYPESC is a pre-existing capability that requests a different type of shear check.

- (4) Fixed bug. Elastic stiffness for MAT_172 beams was not as described in the manual, and the axial response could sometimes become unstable. The bug did not affect shell elements, only beams.
- (5) *MAT_172 can now handle models with temperatures defined in Kelvin (necessary if the model also has heat transfer by radiation). *MAT_172 has thermally-sensitive material properties hard-wired to assume temperatures in Centigrade. A new input TMPOFF in *MAT_-172 offsets the model temperatures before calculating the material properties.
- (6) When the input parameter AGGSZ is defined, the maximum shear stress that can be transferred across closed cracks is calculated from a formula that has tensile strength and compressive stress as inputs. In MAT_172, the tensile strength of concrete is reduced when compressive damage has occurred (see description of UNLFAC). Up to now, compressive damage was therefore influencing the maximum shear across cracks. However, the Norwegian standard from which the shear formula is taken treats the tensile strength as a constant. Therefore, for the purpose of calculating the maximum shear stress across closed cracks only, the compressive damage effect is now ignored.
- (7) Added capability for water pressure in cracks, for offshore applications. The water pressure is calculated from the depth of the element below the water surface (calculated from the z-coordinate). The water pressure is applied as a compressive stress perpendicular to the plane of any crack in the element. See new input fields WRO_G and ZSURF.

◦ *MAT_119 (*MAT__GENERAL_NONLINEAR_6DOF_DISCRETE_BEAM): Fixed bug in UNLOAD option 2. The bug occurs if an unloading curve has been left zero (e.g. LCIDTUR) while the corresponding loading curve was non-zero (e.g. LCIDTR), and UNLOAD = 2. Depending on the computer system, the symptoms could be harmless or the code could crash. Now, if the unloading curve is left blank, it is assumed to be the same as the loading curve i.e. load and unload up and down the same curve. That behavior was already implemented for UNLOAD = 1.

◦ Added Equation Of State 19 (*EOS_MURNAGHAN). Used extensively for fluid modeling in SPH through Weakly-Compressible formulation, in conjunction with SPH formulations 15 (fluid form) and 16 (normalized fluid form).
◦ *MAT_ADD_FATIGUE: Added a new form of Basquin equation to define material's SN curve: LCID = -3: S = a*N^b, where a and b are material constants.
◦ Add the option of A0REF for *MAT_FABRIC. That allows the option of using reference geometry to calculate A0 for the purpose of porosity leakage calculation.

- Add optional parameter DVMIN for *MAT_ADD_PORE_AIR to define the min volume ratio change to trigger pore air flow analysis.
- *DEFINE_HAZ_PROPERTIES:
- Distance of shell from the weld center is treated consistently under MPP and the shell material's yield stress is scaled properly.
- *MAT_168 and *MAT_279: Fixed support for element erosion.
- *MAT_092: Improved of implicit convergence for shells.
- *MAT_224: Fixed bug where wrong shear modulus was used in EOS.
- *MAT_270: Increased stability for thickness strain iterations for shells.
- *MAT_240: Added support for cohesive shell formulation +/-29.
- Scale load curve, LCSRS, of *MAT_ADD_EROSION when used with *INCLUDE_TRANSFORM.
- Fix incorrect results when using *MAT_TABULATED_JOHN-SON_-COOK/*MAT_224 with table LCKT defined and the first abscissa value, temperature, is negative.
- Fix spurious element deletion when using table for LCF in *MAT_TABULATED_JOHNSON_COOK/*MAT_224 and *MAT_TABULA-TED_JOHNSON_COOK_GYS/*MAT_224GYS.
- Error terminate with message, KEY+1142, if *MAT_ADD_EROSION is applied to resultant materials 28,116,117,118,130,139,166,170 and 98(with 1 intg point).
- Increase robustness of *MAT_033/*MAT_BARLAT_ANISOTROPIC_-PLASTICITY for solids.
- Fix input error when using *MAT_ELASTIC_WITH_VISCOSITY_-CURVE/*MAT_060c when LCID = 0.
- Fix seg fault when using shell type 15, axisymmetric volume weighted, with *MAT_ADD_EROSION and also materials with equation-of-states.
- Store computed yield strength as history variable #6 for *MAT_255.
- Fix inconsistency for *MAT_MODIFIED_PIECEWISE_LINEAR_-PLASTICITY/*MAT_123 when ncpu < 0.
- Include original volume output to dynain file for 2D analysis when materials with an equation-of-state are used. This is needed to compute the deformation gradient when initializing a run using the dynain file.
- Fix improper stress initialization using *INITIAL_STRESS_SHELL via dynain for *MAT_018/*MAT_POWER_LAW_PLASTICITY with VP = 1.0.
- Make AOPT < 0 work for *MAT_170/*MAT_RESUL-TANT_ANISOTROPIC, i.e. with material coordinate system using *DEFINE_COORDINATE_(OPTION).
- Fix incorrect operation of TDEL for *MAT_MODIFIED_-PIECEWISE_LINEAR_PLASTICITY/*MAT_124 and *MAT_PLASTI-CITY_WITH_DAMAGE/*MAT_081/*MAT_082.
- Fix incorrect damping when using *DAMPING_PART_STIFFNESS for *MAT_16/*MAT_PSEUDO_TENSOR and *EOS_TABULATED_-COMPACTION.

- Fix incorrect computation of bulk modulus which caused complex sound speed error when using *EOS_TABULATED/EOS_09 with tabulated input.
- Fix moving part with *MAT_220 during dynamic relaxation when velocities are initialized.
- Fix convergence issue for *MAT_065/*MAT_MODIFIED_ZERILLI_-ARM-STRONG for shells when VP = 1.
- Error terminate with message, KEY+1115, if _STOCHASTIC option is invoked for materials 10,15,24,81,98, or 123 but no *DEFINE_STOCH-ASTIC_-VARIATION or *DEFINE_HAZ_PROPERTIES keyword is present in the input file.
- Fix spurious error termination when using *DEFINE_HAZ_PROPERTIES with adaptivity.
- Fix incorrect results or seg fault for *MAT_FU_-CHANG_FOAM/*MAT_-083 if KCON > 0.0 and TBID.ne.0.
- If SIGY = 0 and S = 0 in *MAT_DAMAGE_2/*MAT_105, set S = EPS1/200, where EPS1 is the first point of yield stress input or the first ordinate point of the LCSS curve.
- Allow *MAT_ENHANCED_COMPOSITE_DAMAGE/*MAT_054 failure mechanism to work together with *MAT_ADD_EROSION for shells.
- Fix incorrect erosion behavior if *MAT_ADD_EROSION is used with failure criteria defined for *MAT_123/*MAT_MODIFIED_PIECEWISE_-LIN-EAR_PLASTICITY.
- Implement *MAT_FHWA_SOIL/*MAT_147 for 2D analysis, shell types 13, 14 and 15.
- Implement scaling of failure strain for *MAT_MODIFIED_-PIECEWISE_-LINEAR_PLASTICITY_STOCHASTIC/*MAT_123_STOCHASTIC for shells.
- Fix incorrect behavior for *MAT_LINEAR_ELASTIC_DISCRETE_-BEAM/*MAT_066 when using damping with implicit (static) to explicit switching.
- Fixed *MAT_FABRIC/*MAT_034 with the negative unloading curve option. When searching for the intersection point of the load and unload curves, and extrapolation of one of the curves was needed to find the intersection point, the extrapolated stress was calculated incorrectly causing unpredictable behavior.
- Fixed fabric material forms 0 and 1 when used with a reference geometry. There were two problems, both occurring when there are mixed quad and triangular elements in the same block. A flaw in the strain calculation was leading to possible NaN forces in the elements. When a reference geometry was not used, the forces from triangular elements in mixed element blocks were 2 times too high.
- Added a new option for *MAT_SPOTWELD called FMODE. The FMODE option is available for DMGOPT = 10, 11, and 12. When the failure function is reached, and when FMODE > 0.0 and < 1.0, the value of FMODE will determine if a weld will fail immediately or will have damage initiated. The

failure function may include axial, shear, bending and torsion terms. If the sum of the squares of the shear and torsion terms divided by the sum of the square of all terms is greater than FMODE, then the weld will fail immediately. Otherwise, damage will be initiated.

- Enabled OPT = -1 on *MAT_SPOTWELD for brick elements which had not worked previously. Also, fixed TRUE_T when used with brick element forms 0, 1, and -1.
- Fixed spot welds with DMGOPT = 12 by removing warning STR+1327 which made it impossible to set a small value of RS without triggering this warning, or without setting EFAIL smaller. Setting EFAIL small however could lead to damage initiation by plastic strain when the user wanted only initiation by the failure function.
- If DMGOPT = 10, 11, or 12 and EFAIL = 0, on *MAT_SPOTWELD, damage will now initiate only by the failure function. If EFAIL > 0, then damage will initiate be either then failure function or when plastic strain exceeds EFAIL. Prior to this version, damage could initiate when plastic strain exceeds zero if the user set EFAIL = 0. This behavior is still true for DMGOPT = 0, 1, or 2, but no longer for DMGOPT = 10, 11, or 12.
- Allow solid spot welds and solid spot weld assemblies to have up to 300 points in the running average that is used to smooth the failure function. In other words, up to NF = 300 is possible.
- Fixed a problem with brick spot weld assemblies when OPT = 0 failure is used without defining any weld resultant values. Welds were being immediately deleted.
- Added new PID option for *DEFINE_SPOTWELD_FAILURE (applies to *MAT_SPOTWELD, OPT = 10). Changes the Card 3 input for static strength values to use part set ID's rather than material ID's.
- Modified shell *MAT_214/*MAT_DRY_FABRIC to calculate fiber strains based on the current distance between the points where the fibers intersect with the element edges. Previously, they were calculated from the rate-of-deformation, but this was not as accurate as the new total strain measure.
- Fixed unit scaling for GAMAB1 and GAMAB2 on *MAT_DRY_FABRIC. We were incorrectly transforming them as stress.
- Reworked the plastic stress update in *MAT_225/*MAT_VISCOPLASITC_-MIXED_HARDENING to prevent a divide by zero.
- Enabled *MAT_ADD_EROSION to be used with beams that have user defined integration. Memory allocation was fixed to prevent memory errors.
- Fixed *MAT_106 when used with tshell form 5 or 6. The elastic constants used in the assumed strain field were not reasonable.
- Fix issue that could have led to problems using *MAT_054 (or *MAT_058 or *MAT_158) in combination with TFAIL/TSIZE.gt.0.0 and damping.
- *MAT_054 - *MAT_ENHANCED_COMPOSITE_DAMAGE:

    - Add possibility to use failure criterion in *MAT_054 for solids in a transversal isotropic manner. It is assumed that the material 1-

direction is the main axis and that the behavior in the 2-3 plane is isotropic. This feature is invoked by setting TI = 1 in *MAT_054.

- *MAT_058 - *MAT_LAMINATED_COMPOSITE_FABRIC:
  - Bugfix for shear stiffness behavior in *MAT_058 when using a table definition for GAB and only providing stress-strain-curves for positive shear.
  - Bugfix for strain-rate dependent stiffness behavior in *MAT_058 when using a table definition for EA, EB or GAB under compressive loading.
  - Add default values for strengths (XT,XC,YT,YC,SC) 1.e+16 for *MAT_-058. If no values for the strengths were defined, unpredictable things could have happened.
- *MAT_138 - *MAT_COHESIVE_MIXED_MODE:
  - Store total mixed-mode and normal separation (delta_II & delta_I) on history variables 1&2 for *MAT_COHESIVE_MIXED_MODE (*MAT_-138). This is only for post-processing and should not lead to any changes in the results.
- *MAT_157 - *MAT_ANISOTROPIC_ELASTIC_PLASTIC:
  - Add Tsai-Hill failure criterion (EXTRA = 2).
  - Allow strain-rate dependent strength values (XT,XC,YT,YC,ZT,ZC,SXY,SYZ,SZX) using *DEFINE_CURVE. This is available for Tsai-Wu (EXTRA = 1) and Tsai-Hill.
  - Fixed bug in using *MAT_157 with IHIS.gt.0 for shells. Thickness strain update d3 was not correct and plasticity algorithm may have failed.
  - Add additional option to IHIS in *MAT_157 for SHELLs.
  - Now also the strength values (XT,XC,YT,YC,SXY) may be initialized via *INITIAL_STRESS_SHELL. See variable IHIS and remarks in the User's Manual for details of initializing various blocks of material parameters.
- *MAT_215 - *MAT_4A_MICROMEC:
  - Add new material *MAT_215 that is a micromechanical material model that distinguishes between a fiber/inclusion and a matrix material. The material is intended for anisotropic composite materials, especially for short (SFRT) and long fiber thermoplastics (LFRT). This model is available for shells, tshells and solids.
- *MAT_225 - *MAT_VISCOPLASTIC_MIXED_HARDENING:

- Fixed bug in *MAT_225 (*MAT_VISCOPLASTIC_MIXED_HARDENING) when using a table for LCSS together with kinematic hardening.

- *MAT_261 - *MAT_LAMINATED_FRACTURE_DAIMLER_PINHO:
  *MAT_262 - *MAT_LAMINATED_FRACTURE_DAIMLER_CAMANHO:

  - Allow table input for fracture toughness values for mats 261/262.Table represents fracture toughness vs. element length vs. strain rate (shells, tshells, solids)
  - Fixed bug in mats 261/262 when using *DAMPING_PART_STIFFNESS together with RYLEN = 2 in *CONTROL_ENERGY.
  - Correct shear failure behavior in *MAT_262. This will most probably have no effect to any real application, but could be seen in very special 1-element tests.

- Changed storage of history variables for *MAT_249 (*MAT_REINFORCED_THERMOPLASTIC). A new variable POSTV controls which variables are written and at what history variable location in d3plot.
- *MAT_254 (*MAT_GENERALIZED_PHASE_CHANGE) can now be used with shell elements and thermal thick shells.
- Added flag 'EZDEF' to *MAT_249_UDFIBER. In this case the last row of the deformation gradient is replaced by 0-0-1.
- Add opt. damage limitation curve/table LCDLIM for *MAT_ADD_GENERALIZED_DAMAGE.
- Add pre-defined damage tensors option PDDT to *MAT_ADD_GENERALIZED_DAMAGE.
- *MAT_ADD_GENERALIZED_DAMAGE now works for solid elements (only shells in R9).
- Add optional failure criterion FFCAP to *MAT_100 with OPT = -1 or 0.
- Enable *MAT_ADD_COHESIVE to be used in implicit analysis.
- Add alternative version of *MAT_280 invoked by new flag on 1st card. It is a physically based damage model with 4 new parameters.
- Enable *DEFINE_CONNECTION_PROPERTIES' option PROPRUL>=2 to be used with spotweld clusters, i.e. not only 1 hex element but several (via *DEFINE_HEX_SPOTWELD_ASSEMBLY or RPBHX > 1 on *CONTROL_SPOTWELD_BEAM).
- Enable *MAT_ADD_EROSION to be safely used with material models that have more than 119 history variables, for now the new limit is 169 (e.g. necessary for *MAT_157 with IHIS = 7).
- Add Tsai-Wu failure criterion to *MAT_157 for solid and shell elements invoked by EXTRA = 1 on card 6 and corresponding parameters on cards 8 and 9.
- Add viscoelastic option to *MAT_187 (SAMP-1). Rate dependent Young's modulus and associated settings can be defined on new optional card 5.

- Add new option IRNG for *DEFINE_STOCHASTIC_VARIATION to govern random number generation (deterministic or true random).
- Add option to define element size dependent parameters EN and SN for *MAT_120 and *MAT_120_JC by setting them to negative values (curves).
- Minor improvements for *MAT_252: Optional output of damage initiation information and more post-processing history variables.
- If the first abscissa value of *MAT_224's failure strain curve LCG is negative, it is assumed that all abscissa values are natural logarithms of a strain rate.
- Put *MAT_100_DA's "failure function" value to history variable 18.
- Add optional in-plane failure strain to *MAT_169 (ARUP_ADHESIVE): new input parameter FSIP.
- *MAT_USER_DEFINED_MATERIAL_MODELS now provides a few more variables for cohesive elements, i.e. additional arguments in subroutines umatXXc: temperature, element size, implicit rejection flag, integration point identifier, and total number of integration points.
- A modified version of the 3-parameter Barlat model (*MAT_036) is introduced as *MAT_EXTENDED_3-PARAMETER_BARLAT. In this model, hardening in 00, 45, 90, biaxial and shear can be specified as load curves. Furthermore, r-values in 00, 45, 90, biaxial and shear can be specified in terms of load curves vs plastic strain or constants. This is an extension of hardening law 7 of the original 3-parameter Barlat model.
- Improve implicit version of *MAT_098/*MAT_SIMPLIFIED_JOHNSON_-COOK.
- *MAT_181/*MAT_SIMPLIFIED_RUBBER/FOAM is now supported for 2D implicit simulations.
- Fixed issue in which *MAT_WINFRITH_CONCRETE wrote d3crack data too frequently.
- *EOS_JWL now has an AFTERBURN option. This adds afterburn energy to the EOS, where the energy can be added at a constant or linear rate, or can be added according to Miller's extension.
-
- *MAT_084 (*MAT_WINFRITH_CONCRETE) with predefined units (CONM < 0) is now transformed correctly with *INCLUDE_TRANSFORM.
- User-defined materials for Hughes-Liu beams can now be used with implicit analysis by defining the appropriate tangent modulus in the supplied routine urtanb.
- User-defined cohesive materials can now be used with implicit analysis by defining the appropriate tangent stiffness.
- *MODULE for user-defined materials and other user-defined capabilities:

    - A new command line option "module = filename" is added to load one module file without changing the input deck. It provides back compatibility to input deck without the MODULE keywords.
    - The system paths defined in LD_LIBRARY_PATH are also included for searching module files for those filenames start with "+".

- Add shell implementation to *MAT_277 (*MAT_ADHESIVE_CURING_-VISCOELASTIC).
- Add *MAT_278 for carbon fiber prepreg compression forming simulation. This material model is available for both solid and shell formulations.
- Add *MAT_293 non-orthogonal material model for carbon fiber prepreg forming simulation. This material model is only available for shell formulations.
- *MAT_260A:
  - Extend *MAT_260A to include solid elements.
  - Add a new option XUE for Xue's fracture criteria/theory for *MAT_-M260A (solid elements only).
- *MAT_260B:
  - Set default values for P's and G's in *MAT_260B.
  - Add a length scale to the fracture limit. The fracture limit strongly depends on the length scale in the measurement.
  - Add a new fracture criterion to *MAT_260B (Xue and Wierzbicki, Int. J. solids and Structures 46 (2009) 1423-1435). When the option XUE is activated, an additional card is needed, for example:

```
$     ef0     plim        q     gama        m
    0.70    925.7    0.970    0.296     2.04
```

- *MAT_037:
  - Improve *MAT_037 with negative R value in implicit calculation. The modification will allow the implicit method stress calculation to be more accurate.
  - Add a new option NLP2 to calculate formability index in *MAT_037. The previous method (option NLP_FAILURE) was based on the effective strain method, which assumes that necking happens at one instant. In fact, it might happen over a longer process. The new method calculates the damage accumulation.
- Add *MAT_165B (*MAT_PLASTIC_NONLINEAR_KINEMATIC_B) for shells and solids.

- **MPP**
  - Fix the report of decomp balance (shown as "Normalized element costs assigned during decomposition" in the d3hsp file), which was broken in r109760
  - MPP decomposition has not been properly balanced since r112652 due to a bug in that revision

- Fix MPP SYNC error due to inconsistent summation in *CONTACT_SLIDING_ONLY_PENALTY.
- Allow real values as the scale multipliers for "memory=" on the command line. For example, "memory = 2.5G memory2 = 1.1G" and the like.
- MPP: fix support for nlq setting in *CONTROL_SOLUTION which was not being honored on processors other than 0.
- Significant improvements in MPP groupable routines for FORMING contact.
- MPP: increase contact release distance for SINGLE_SURFACE contacts in the case of a node coming into contact with a solid element. The previous interpretation was releasing when the contact penetration was 0.5*solid thickness, but now when the node passes below the solid surface by 0.5*solid thickness (which is different by the half thickness of the slave material, in the case of a shell slave node).
- MPP: fix for viscous damping in automatic tiebreak contact.
- Implement new bucket sort based extent testing for MPP single surface contact.
- Added MPP support for *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_LUBRICATION.
- Fixed *CONTROL_MPP_PFILE so that it honors ID offsets from *INCLUDE_TRANSFORM for parts, part sets, and contact IDs referenced in "decomp { region {" specifications.
- Furthermore, such a region can contain a "local" designation, in which case the decomposition of that region will be done in the coordinate system local to the include file, not the global system. For example:
- *CONTROL_MPP_PFILE decomp { region { partset 12 local c2r 30 0 -30 0 1 0 1 0 0 } } would apply the c2r transformation in the coordinate system of the include file, which wasn't previously possible. The local option can be useful even if there are no such transformations, as the "cubes" that the decomposition uses will be oriented in the coordinate system of the include file, not the global system.
- Furthermore, the following decomposition related keywords now have a _LOCAL option, which has the same effect:

  *CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE_LOCAL
  *CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE_LOCAL
  *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS_LOCAL
  *CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE_LOCAL

- MPP job performance profiles are output to both .csv and .xy files.

- **OUTPUT**

- Fix for writing d3plot file when individual output states exceed 8GB in single precision
- Added new option *INTERFACE_SPRINGBACK_EXCLUDE to exclude selected portions from the generated dynain file.
- Add a new option to *INTERFACE_COMPONENT_FILE to output only 3 degrees of freedom to the file even if the current model has 6.
- Fix missing plastic strain tensors in d3plot when STRFLG in *DATA-BASE_-EXTENT_BINARY is set and INTSTRN = 1 in *INTERFACE_-SPRING-BACK.
- Fix no output to bndout when run with q = remap even though the keyword *DATABASE_BNDOUT was present in the remap run but was not present in the initial run.
- Fix d3plot output frequency which was different from the dt specified in *DATABASE_BINARY_D3PLOT when *CONTACT_AUTO_MOVE is used.
- Fix stress output to elout for solid elements which was in the global coordinates instead of local coordinates when CMPFLG = 1 in *DATABASE_EXTENT_BINARY and OPTION1 > 0 in *DATABASE_-ELOUT.
- Fix incorrect mass properties for solids in ssstat file when using *DATABASE_SSSTAT_MASS_PROPERTY.
- Fix seg fault during writing of dynain file if INSTRN = 1 in *INTERFACE_-SPRINGBACK and STRFLG.ne.0 in *DATABASE_-EXTENT_BINARY and the *DATABASE_EXTENT_BINARY comes after *INTERFACE_SPRING-BACK.
- If HYDRO is nonzero in *DATABASE_EXTENT_BINARY, LS-PrePost will now combine the solid and shell internal energy densities when fringing 'Internal Energy Density' in the Misc menu.
- By putting SIGFLG/EPSFLG = 3 in *DATABASE_EXTENT_BINARY, the stresses and plastic trains are excluded not only for shell elements but also for solids. This applies to d3plot and d3eigv.
- Added new file option *DATABASE_BINARY_INTFOR_FILE to define interface file name.
- Fixed legend in bndout in the case of multiple *BOUNDARY_PRESCRIBED_MOTION_SET_ID.
- Fix corrupt d3part data caused by DECOMP = 4 in *DATABASE_-EXTENT_BINARY.
- Fixed the legend of ssstat in binout.
- Added *DATABASE_EXTENT_SSSTAT_ID. The subsystem id will be included in the ASCII ssstat file.
- Fixed bug in stbout (seatbelt output) if NEWLEG = 0 in *CONTROL_OUTPUT.
- Fixed bug in which DECOMP = 2 corrupted d3part.
- Fixed d3plot bug if dynamic relaxation was activated in the input deck.
- Added another digit for coordinates in *NODE in dynain, e.g., what was written as 0.999266236E+00 is now written as 9.992662368E+00.

- Added *DATABASE_EXTENT_BINARY_COMP for alternative (simpler) control of output to d3plot and d3eigv.

  - Output control flags: 0-no 1-yes
  - IGLB : Global data
  - IXYZ : Current coordinate
  - IVEL : Velocity
  - IACC : Acceleration
  - ISTRS: 6 stress data + plastic strain
  - ISTRA: 6 strain data
  - ISED : Strain energy density

  This command can be used in combination with regular *DATABASE_EXTENT_BINARY but will disable most of the options in the latter, including output of extra history variables.
- Bugfix: *DATABASE_TRACER without the optional NID parameter was read incorrectly when used with *INCLUDE_TRANSFORM, but is now fixed
- Fixed incomplete output from Windows version of LS-DYNA. This affected demtrh (*DATABASE_TRACER_DE) and curvout (*DATABASE_-CURVOUT).

- **Restarts**

  - Enable definition of sensors in full restarts.
  - For a small restart in MPP, the value of "memory=" (M1) needed for each processor is stored in the dump files. This is the minimum requirement to read back the model info. If the value of "memory2=" (M2) is specified on the command line, the code will take the maximum of M1 and M2.
  - Fix input error during structured input when using *INITIAL_VELOCITY_-GENERATION and *CHANGE_VELOCITY_-GENERATION together in a full deck restart.
  - Fix incorrect full deck restart analysis if initial run was implicit and the full deck restart run is explicit. This affects MPP only.
  - Fix insufficient tying of nodes when doing full deck restart and the contact is newly added to the restart involving newly added parts. This applies to SMP contact only.
  - Fix incorrect velocity initialization for SMP full deck restart when using *INITIAL_VELOCITY_GENERATION and *INITIAL_VELOCITY_-GENERATION_START_TIME.
  - Fix incorrect initialization of velocities for SMP full deck restart when using *CHANGE_VELOCITY_OPTION & *INITIAL_VELOCITY_OPTION. Velocities of existing parts defined by *STRESS_INITIALIZATION should not be zeroed.
  - Fix *CHANGE_CURVE_DEFINITION for curve specifying d3plot output.

- Fixed bug in full deck restart if the new mesh has different part numbers.

- ***SENSOR**

  - Fix a bug regarding *SENSOR_JOINT_FORCE for *CONSTRAINED_-JOINT_STIFFNESS, that was triggered when the force refers to a local coordinate system.
  - Add the option of "ELESET" to *SENSOR_CONTROL to erode elements.
  - Add the option of NFAILE to *SENSOR_DEFINE_MISC to track number of eroded elements.
  - Fix a bug that was triggered when using a sensor to control spotwelds. The bug was triggered when the spotweld-connected nodal pairs happen to belong to more than 1 core (MPP only).
  - Add FAIL option to *SENSOR_DEFINE_ELEMENT to track the failure of element(s).
  - Fix a bug related to *SENSOR_DEFINE_FUNCTION when there are more than 10 sensor definitions.
  - Effect of TIMEOFF in *SENSOR_CONTROL is implemented for TYPE = PRESC-ORI.
  - *SENSOR_CONTROL can be used to control *BOUNDARY_PRESCRIBED_-ORIENTATION_RIGID.
  - Add optional filter id to SENSORD of *DEFINE_CURVE_FUNCTION.
  - Enable *CONSTRAINED_JOINT_..._LOCAL to be monitored by *SENSOR_-DEFINE_FORCE.
  - Allow moments in SPCFORC and BNDOUT to be tracked by *SENSOR_-DEFINE_FORCE.
  - Fix *SENSOR_CONTROL using TYPE="PRESC-MOT" which was not switching at all.

- **SPG (Smooth Particle Galerkin)**

  - MPP is ready in 3D SPG fluid particle stabilization (ITB = 1 & 2 in *SECTION_SOLID_SPG).
  - Added one SPG control parameter (itb = 2) for semi-brittle fracture analysis. In comparison to itb = 0 or itb = 1, itb = 2 is more efficient in modeling the fragmentation and debris in semi-brittle fracture analysis such as impact and penetration in concrete materials.
  - Fixed a bug related to E.O.S. in SPG.
  - Removed some temporary memory allocations to improve efficiency.
  - Changed the sequence of SPG initialization so that all state variables are properly initialized.
  - Subroutines were developed for SPG failure analysis with thermal effects. Both explicit and implicit (diagonal scaled conjugate gradient iterative only) SPG thermal solvers are available in SMP version only. However, thermal effect is applied only on material properties, which means thermal induced

deformation (i.e., thermal strain or thermal expansion) is not currently included.
- Modified *MAT_072R3 for SPG method in concrete applications.
- Fixed a bug for SPG method in using continuum damage mechanics. (IDAM = 0).
- Added the "fluid particle algorithm" (itb = 1) to SPG method. This algorithm is implemented in R10.0 as an alternative to the (itb = 0) option in previous version to enhance the numerical stability for SPG method. Users are recommended to use this new option for their ductile failure analysis.

- **SPH (Smooth Particle Hydrodynamics)**

  - Add ITHK flag in *CONTROL_SPH, card 3. If flag is set to 1, the volume of the SPH particles is used to estimate a node thickness to be employed by contacts.

    - Affects *AUTOMATIC_NODES_TO_SURFACE and *CONTACT_-2D_-NODE_TO_SOLID.
    - The thickness calculated by ITHK = 1 is used only if SST or OFFD are set to zero in the contact cards definitions.

  - Add SOFT = 1 option to *CONTACT_2D_NODE_TO_SOLID. This should help obtain reasonable contact forces in axisymmetric simulations. Default penalty PEN is 0.1 when SOFT = 1.
  - Implemented non-reflecting boundary conditions for SPH using a new keyword *BOUNDARY_SPH_NON_REFLECTING.
  - Bug fix for renormalized SPH formulations with symmetry planes. The renormalization was slightly incorrect in the vicinity of symmetry planes.
  - Density smoothing in SPH formulations 15 and 16 is now material sensitive. The smoothing only occurs over neighbors of the same material.
  - Resolved an MPP bug in SPH total Lagrangian formulations (FORM = 7/8) which was causing strain concentrations at the interfaces between CPU zones.
  - SPH total Lagrangian (FORM = 7/8) in SMP was pretty much serial, hence much slower than forms 0 or 1. SPH with FORM 7 and 8 now scales properly.
  - Added support for FORMs 0/1 in axisymmetric. Until now, renormalization was always active (equivalent to FORM = 1) which can be problematic for very large deformations or material fragmentation.
  - Improved tracer particles output for SPH: Use normalized kernel function for interpolation between particles.
  - Implemented enhanced fluid flow formulations (FORMs 15/16) with pressure smoothing.
  - Recode SPH neighborhood search algorithm to reduce the memory requirement and produce consistent results from MPP and HYBRID code.

- *DEFINE_ADAPTIVE_SOLID_TO_SPH now reports both active and inactive adaptive SPH particles in the fragment file sldsph_frag. This file gives a report of nodal mass, coordinates, and velocities.
- MPP now supports:
  - SPH type 3 inflow
  - Multiple *BOUNDARY_SPH_FLOW
  - Bulk viscosity option for SPH
- Sort SPH by part and then node ID to ensure consistent results while changing order of input files.
- *DEFINE_SPH_TO_SPH_COUPLING:
  - Corrected the SPH sphere radius (half of the particles distance) for node to node contact detecting algorithm.
  - Updated masses for SPH node to node coupling with damping contact force option.
  - Added a new option (Soft = 1) for SPH to SPH coupling: contact stiffness comes from particles masses and time step for softer contact.
- *DEFINE_ADAPTIVE_SOLID_TO_SPH:
  - Updated temperature transfer (from solid elements to SPH particles) when converting solid elements into SPH particles with ICPL = 1, IOPT = 0.
  - Bug fixed when part ID for newly generated SPH particles is smaller than the original SPH part ID.
  - Introduced a new pure thermal coupling between SPH part and solid parts with ICPL = 3 and IOPT = 0 option (no structural coupling provided).
  - Added a thermal coupling conductivity parameter CPCD. Applies to ICPL = 3 option.
  - Normalized the nodal temperatures for the corner SPH particles with ICPL = 3 and IOPT = 0 option (MPP only).
  - Extended ICPL = 3 and IOPT = 0 option to Lagrangian formulation (form = 7, 8).
- *BOUNDARY_SPH_SYMMETRY_PLANE:
  - Added in an error message if TAIL and HEAD points are at the same location.
- *CONTACT_2D_NODE_TO_SOLID:
  - Added a variable OFFD to specify contact offset.

- Added a new option IEROD = 2 in *CONTROL_SPH in which SPH particles that satisfy a failure criterion are totally eactivated and removed from domain interpolation. This is in contrast to IEROD = 1 option in which particles are partially deactivated and only stress states are set to zero.
- Added *MAT_SPH_VISCOUS (*MAT_SPH_01) for fluid-like material behavior with constant or variable viscosity. Includes a Cross viscosity model.
- Output strain rates for SPH particles to d3plot, d3thdt, and sphout file.
- Added support of *MAT_ADD_EROSION, including GISSMO and DIEM damage, for SPH particles.
- Echo failed SPH particles into d3hsp and messag file.
- *DEFINE_SPH_INJECTION:

  - Changed the method of generating SPH particles. SPH particles will be generated based on the injection volume (injection area*injection velocity*dt)*density from the material model, resulting in more consistent particle masses and particle distribution.
  - Offset injecting distance inside each cycle so that outlet distance will be consistent for different outlet SPH layers.
  - Corrected mass output in d3hsp.

- **Thermal Solver**

  - Modify the thermal solver routines so they return instead of terminating, so that *CASE works properly.
  - *MAT_THERMAL_USER_DEFINED: Fixed bug in element numbering for IHVE = 1.
  - Accept load curve input for dtmin, dtmax and dtemp in *CONTROL_THERMAL_TIMESTEP. As usual if a negative integer number is given its absolute value refers to the load curve id.
  - The temperature results for the virtual nodes of thermal thick shells are now accounted for in *LOAD_THERMAL_D3PLOT. For the mechanics-only simulation thermal thick shells have to be activated.
  - New contact type for thermal solver that models heat transfer from and to a shell edge onto a surface (*CONTACT_..._THERMAL with ALGO > 1):

    - Shells have to be thermal thick shells.
    - Shells are on the slave side.
    - So far only implemented for SMP.
    - Includes support for quads and triangles.

  - New keyword *BOUNDARY_THERMAL_WELD_TRAJECTORY for welding of solid or shell structures.

    - Keyword defines the movement of a heat source on a nodal path (*SET_NODE).
    - Orientation given either by vector or with a second node set.

        - Works for coupled and thermal only analyses.
        - Allows for thermal dumping.
        - Different equivalent heat source descriptions available.
        - Can also be applied to tshells and composite shells.
        - Weld torch motion can be defined relative to the weld trajectory.

    - Solid element formulation 18 now supports thermal analysis.
    - Thermal solver now supports the H8TOH20 option of *ELEMENT_SOLID. This includes support of *INITIAL_TEMPERATURE condition for the extra 12 nodes generated by H8TOH20.
    - Thermal solver now supports the H8TOH27 option of *ELEMENT_SOLID.
    - Explicit Thermal Solver

        - *CONTROL_EXPLICIT_THERMAL_SOLVER: Implement an explicit thermal solver and adapt it to support multi-material ALE cases.
        - *CONTROL_EXPLICIT_THERMAL_PROPERTIES: Enter thermal properties for the explicit thermal solver.
        - *CONTROL_EXPLICIT_THERMAL_CONTACT: Implement a thermal contact for the explicit thermal solver.
        - *CONTROL_EXPLICIT_THERMAL_ALE_COUPLING: Implement a thermal coupling between ALE and Lagrangian structures for use by the explicit thermal solver.
        - *CONSTRAINED_LAGRANGE_IN_SOLID_EDGE: For the explicit thermal ALE coupling, allow the heat transfer through the shell edges if _EDGE is added to *CONSTRAINED_LAGRANGE_IN_SOLID.
        - *CONSTRAINED_LAGRANGE_IN_SOLID: For the explicit thermal solver, add work due to friction to the enthalpies of ALE and structure elements coupled with *CONSTRAINED_LAGRANGE_IN-_SOLID (CTYPE = 4).
        - *CONTROL_EXPLICIT_THERMAL_INITIAL: Initialize the temperatures for the explicit thermal solver.
        - *CONTROL_EXPLICIT_THERMAL_BOUNDARY: Control boundary temperatures for the explicit thermal solver.
        - *CONTROL_EXPLICIT_THERMAL_OUTPUT: Output the temperatures at element centers for the explicit thermal solver.
        - *DATABASE_PROFILE: For the explicit thermal solver, output temperature profiles.

- **Miscellaneous**

    - *INITIAL_LAG_MAPPING: Implement a 3D to 3D lagrangian mapping and map the nodal temperatures.
    - *CONTROL_REFINE_SHELL and *CONTROL_REFINE_SOLID: Add a parameter MASTERSET to call a set of nodes to flag element edges along which new child nodes are constrained.

- *BOUNDARY_PRESCRIBED_MOTION_SET_SEGMENT: Add DOF = 12 to apply velocities in local coordinate systems attached to segments.
- Fixed bug occurring when a part has non-zero *DAMPING_-PART_STIFF-NESS, AND is defined using *PART_COMP-OSITE, AND the MIDs referenced by the different integration points have different material types. Symptoms could include many types of unexpected behavior or error termination, but in other cases it could be harmless.
- *DAMPING_FREQUENCY_RANGE (including _DEFORM option): Improved internal calculation of damping constants such that the level of damping more accurately matches the user-input value across the whole of the frequency range FLOW to FHIGH. As an example, for CDAMP = 0.01, FLOW = 1 Hz and FHIGH = 30 Hz, the actual damping achieved by the previous algorithm varied between 0.008 and 0.012 (different values at different frequencies between FLOW and FHIGH), i.e. there were errors of up to 20% of the target CDAMP. With the new algorithm, the errors are reduced to 1% of the target CDAMP. This change will lead to some small differences in results compared to previous versions of LS-DYNA. Users wishing to retain the old method for compatibility with previous work can do this by setting IFLG (7th field on Card 1) to 1.
- Fixed bug that could cause unpredictable symptoms if Nodal Rigid Bodies were included in the Part Set referenced by *DAMPING_FREQUENCY_-RANGE or *DAMPING_FREQUENCY_RANGE_-DEFORM. Now, the _DEFORM option silently ignores NRBs in the Part Set while *DAMPING_-FREQUENCY_RANGE (non _DEFORM option) damps them.
- Fixed bug in *PART_COMPOSITE: if a layer had a very small thickness defined, such as 1E-9 times the total thickness, that layer would be assigned a weighting factor of 1 (it should be close to zero).
- Fix errors in implementation of *DEFINE_FILTER type CHAIN.
- Fix for *INTERFACE_LINKING_LOCAL when LCID is used. During keyword processing, the LCID value was not properly converted to internal numbering.
- Switch coordinates in keyword reader to double precision.
- Change "Warning" to "Error" for multiply defined materials, boxes, coordinate systems, vectors, and orientation vectors. The check for duplicate section IDs now includes the element type and remains a warning for now, because SPH is still detected as a SOLID. Once that is straightened out, this should be made an error.
- Add "TIMESTEP" as a variable for *DEFINE_CURVE_FUNCTION. This variable holds the current simulation time step.
- Fix a bug for the case of CODE = 5 in *DEFORM-ABLE_TO_RIGID_AUTO-MATIC. (Fields 3 to 8 are now ignored.)
- Issue error message and terminate the simulation when illegal ACTION is used for *DEFINE_TRANSFORM.
- Add option of POS6N for *DEFINE_TRANSFORM to define transformation with 3 reference nodes and 3 target nodes.

- Fix a bug that can occur when adaptive elements are defined in a file included by *INCLUDE_TRANSFORM.
- Merge *DEFORMABLE_TO_RIGID_AUTOMATIC cards if they use the same switch time. This dependency of results on the order of the cards and also gives better performance.
- If *SET_PART_OPTION is used, a "group_file" will be created which can be read into LS-Prepost (Model > Groups > Load) for easy visualization of part sets.
- Forces on *RIGIDWALL_GEOMETRIC_CYLINDER can now be subdivided into sections for output to rwforc. This gives a better idea of the force distribution along the length of the cylinder. See the variable NSEGS.
- Added the keywords *DEFINE_PRESSURE_TUBE and *DATABASE_PRTUBE for simulating pressure tubes in pedestrian crash.
- Fix non-effective OPTIONs DBOX, DVOL, DSOLID, DSHELL, DTSHELL, DSEG for *SET_SEGMENT_GENERAL to delete segments.
- Fix incorrect transformation of valdmp in *DAMPING_GLOBAL with *INCLUDE_TRANSFORM.
- Make *SET_NODE_COLLECT work together with *NODE_SET_MERGE.
- Fixed bug in adaptivity for *INCLUDE_TRANSFORM if jobid is used.
- Bugfix: *INTERFACE_SSI with blank optional card is now read in correctly.

Capabilities added to create LS-DYNA R11:

See release notes (published separately) for further details.

- ***AIRBAG**

  - Fix an airbag bug that can cause MPP to hang when reference geometry is used in a huge model.
  - Limit the EXCP option (excluding application of bag pressure to partition parts) in
    *AIRBAG_INTERACTION to *AIRBAG_WANG_NEFSKE and *AIRBAG_HYBRID only.
  - Allow no-pressure venting parts for *AIRBAG_WANG_NEFSKE.
  - Output material leakage information, including leakage rate, in abstat for *AIRBAG_WANG_NEFSKE.
  - Allow airbag venting holes of *AIRBAG_HYBRID to be represented by a part set.
  - Enhance the robustness of *AIRBAG_INTERACTION to curtail instability, especially in MPP, due to excessive mass exchange owing to numerical noise.
  - Support UP (uniform pressure) airbag in MPP full deck restart. (We use the equivalent terms UP airbag or CV (control volume) airbag for airbags that are NOT based on ALE or particle (CPM) methods.)

- Treat PPOP for *AIRBAG_WANG_NEFSKE similar to other types of UP airbag - once it pops, it stays open.
- *AIRBAG_CPM_INTERACTION:
  - Support slave bag with chamber definition.
  - Support master/slave bags having multiple gas components.
- New CPM (*AIRBAG_PARTICLE) particle-to-fabric and particle-to-particle contact algorithms. These are about 3x faster than the old scheme and it is the default method without any input modifications.
- Support *SENSOR_SWITCH_SHELL_TO_VENT for SMP.
- New variable TSPLIT in *AIRBAG_PARTICLE to activate particle splitting. Particles that exit by porosity leakage or a vent are removed from the system. If TSPLIT is set, the code keeps track of the number of removed particles (A) and active particles (B) every 200 cycles after time TSPLIT. Once A is greater than B, each active particle will be split into A/B + 1 particles for a better particle density in the volume.
- Enable OpenMP. This allows MPP hybrid to scale better.
- Support the exterior air drag force capability while switching from particle to UP formulation.
- Enable *DEFINE_FUNCTION for user to control the airbag particle deflection angle. Three airbag history variables are passed to the user defined function to control the surface roughness: bag pressure, bag volume, current time.
- Support *MAT_ADD_AIRBAG_POROSITY_LEAKAGE
  - Support FAC in the option can be defined as a factor/define_curve/define_function
  - If defined_function, it only works with vopt 7/8. the porosity leakage velocity is a function of current part pressure and time.
- Support molar-fraction-based inflator gas flow rate curve LCMASS (MOLEFRACTION option) even after particle airbag switches to CV airbag via the TSW switch.
- Enhance dm_out calculation to treat trunction error.
- Support specification of inflator orifice area as function of time to control the distribution of mass flowrate.
- Fix bug affecting particle-to-UP switch for multi-chambered airbag. The bug was introduced in r82352 and showed up if gas contains multi-species.
- *AIRBAG_LINEAR_FLUID:
  - Prior to R11, the tangent stiffness matrix resulting from a pressure-volume relationship in simple airbag models was merely an approximation and sometimes resulted in poor convergence. A justification of the approximation is that a rigorous treatment reveals that coupling

terms between all nodes in bag result in a dense stiffness matrix and slow execution. Exploiting the low rank structure of this matrix contribution and the use of Sherman-Morrison formula, the correct (or at least close to correct) stiffness matrix is now implemented without sacrificing speed and at the same time obtaining much better convergence.

- ***ALE**
  - *ALE_STRUCTURED_MESH_CONTROL_POINTS:
    - If the S-ALE mesh has different mesh sizes and transition zones, suggest new lists of control points to create optimal mesh transitions (the new list is output in the messag file).
  - *ALE_STRUCTURED_MESH:
    - 2D version of the S-ALE code.
  - *ALE_INJECTION:
    - Inject materials in an ALE mesh (i.e. add a group to some elements and prescribe the velocities of the related nodes).
  - *ALE_PRESCRIBED_MOTION:
    - Prescribe or initialize velocities by ALE group.
  - *ALE_FSI_TO_LOAD_NODE:
    - Use the 2nd line for a directory path where to read or write the files with the fsi data (alefsi2ldnd*.tmp*). Implement the restart.
  - *CONTROL_ALE and *SECTION_ALE2D_SMOOTHING
    - mpp version of the AFAC, BFAC, CFAC and DFAC mesh smoothings in the 2D ALE code.
  - *CONTROL_ALE (END < 0, 2nd line, 2nd column):
    - If END < 0 in *CONTROL_ALE, remove the ALE mesh after |END|.
  - *CONTROL_ALE (IALEDR = 1, 4th line, 2nd column) and *CONTROL_DYNAMIC_RELAXATION:
    - Exclude ALE computations if IDRFLG = 1 unless IALEDR = 1.
  - *MAT_ADD_EROSION (IDAM > 0, 1st line, 3rd column):

        - Implement the GISSMO damage model in ALE.
    - *DATABASE_ALE_MAT:
        - Output the volume averaged pressures of each ALE groups
    - *INITIAL_DETONATION and *CONTROL_EXPLOSIVE_SHADOW:
        - Compute the lighting times by ALE explosive groups.
    - *DATABASE_ALE_BINARY:
        - Output ALE data in a binary file that can be post-treated.
    - *CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS (ALECOMM = 1, 1st line, 2nd column):
        - Create a MPP communicator for processors involved in ALE applications.
    - Added *ALE_STRUCTURED_MESH_TRIM so initial mesh can be trimmed to better fit the domain of interest.
    - S-ALE mesh starts from a rectangular box and the elements away from the domain of interest are then deleted based on the distance to certain geometry or part/element set. This way S-ALE mesh could be of irregular shape. Fewer elements leads to less running time.
    - Added *ALE_STRUCTURED_MESH_MOTION so S-ALE mesh can be made to follow the motion of the mass center of certain ALE material material(s).
    - Add variable TDEATH in *ALE_STRUCTURED_MESH so S-ALE mesh can be deleted during the run to save running time. All related FSI and nodal coupling cards are automatically deleted too.
    - Fix bugs affecting S-ALE (*ALE_STRUCTURED_MESH):
        - Fix sometimes improper volume fraction initialization by *INITIAL_VOLUME_FRACTION_GEOMETRY when using with ESORT = 1 in *CONTROL_SOLID.
        - Fix sometimes incorrect effect of *INITIAL_HYDROSTATIC_ALE and *ALE_AMBIENT_HYDROSTATIC when using with ESORT = 1 in *CONTROL_SOLID.

- ***BOUNDARY**
    - Speedup the input error checking for *BOUNDARY_PRESCRIBED_MOTION.

- Fix *BOUNDARY_PRESCRIBED_ORIENTATION_RIGID_DIRCOS which did not orientate the rigid body when angles are close to PI when using double precision.
- *BOUNDARY_CYCLIC is supported in selective mass scaling (*CONTROL_TIMESTEP).
- *BOUNDARY_SPC_SYMMETRY_PLANE:
  - Add a new option _SET.
  - Allow more than two definitions of symmetric planes.
  - Fix core dump when *BOUNDARY_SPC_SYMMETRY_PLANE, and *ALE_MULTI-MATERIAL_GROUP are used together.

- **Blast**
  - For *LOAD_BLAST_ENHANCED BLAST = 4, apply blast pressure to segments which reside below the ground plane with new parameter FLOOR. This allows blast pressure to be applied to a deformable floor which deflects below the ground plane, and accommodates floors which are not planar.
  - Implement blast particle-SPH coupling algorithm as invoked by variable NODID in *DEFINE_PARTICLE_BLAST (SMP). The SPH smoothing length is taken into consideration.

- **CESE**
  - Added a new prescribed boundary condition card that allows a normal velocity to be specified in the inlet boundary.
  - Added a positivity-preserving strategy for the spatial derivatives calculations to make the solver more stable (currently only CESE 2D, 3D, and CESE-FSI-IBM 2D solvers are supported).
  - Made it a fatal error to attempt to use *LOAD_BLAST_ENHANCED as a CESE boundary condition with other than a type-2 blast.
  - Improved error tolerances in the conjugate heat transfer method solver used with the immersed-boundary method FSI solver, and corrected other small related errors.
  - Any unused nodes specified in a standalone CESE problem are now treated as a fatal error. This fixes an issue with d3plot output when such orphan nodes were passed on to the d3plot file.
  - Corrected some weightings for the MPP CESE moving mesh solvers. This could have been an issue in cases where an MPP CESE shared node is next to a moving boundary.
  - Adjusted the information output by the implicit ball-vertex mesh motion solver to reflect that the absolute error tolerance is no longer used, only the relative error tolerance. This applies to both the ICFD and CESE solvers that use this mesh motion solver.
  - For the *CESE_DATABASE cards, corrected these issues:

        - Any used segments that are not in the CESE mesh are reported.
        - Element sets look up is now guaranteed to work.
        - Drag calculation output that is attempted for the incorrect FSI method, is now reported. Here are the illegal cases:
            - *CESE_DATABASE_FSIDRAG is only available with the CESE immersed boundary FSI solvers.
            - *CESE_DATABASE_SSETDRAG is not available with the CESE immersed boundary FSI solvers. In other words, it only works with the non-FSI solvers and moving mesh FSI solvers.
        - Failure to find an internal node number in *CESE_DATABASE commands is now reported.
        - Output was not being performed for the initial conditions. Instead, it is now done after the first time step.
        - Warning is given when illegal IDs are given in *CESE_DATABASE... cards. Such IDs are then ignored.

    - Changed behavior for the *CESE_DATABASE cards includes:

        - Output times are now selected in a manner similar to CESE d3plot output, meaning when the CESE simulation time has reached at least the requested output times. Also, output never occurs more than once per CESE time step.
        - Cycle number was reported in some outputs as the 'time step'. The actual solver time step is now used instead.

    - When performing d3plot output for the immersed boundary CESE FSI solver, take into account what structural elements overlay CESE elements for the Schlieren number and chemical species mass fractions.
    - Formation enthalpy concept is added to the CESE Navier-Stokes solver.
    - Many restart-related bugs were fixed in various CESE solvers.

- ***CHEMISTRY**

    - The chemistry solvers are working with the following CESE solvers:

        - 2D, 3D, and axi-symmetric Euler and Navier-Stokes solvers.
        - Immersed boundary method FSI 2D, 3D, and axi-symmetric Euler solvers.

    - Added a restart capability for the CESE chemistry solvers.

- **CONTACT**

    - Fix initialization error related to *CONTACT_GENERAL_INTERIOR that resulted in every subsequent *CONTACT_GENERAL also being treated as INTERIOR.

- Fix for spotwelds improperly deleted due to rigid body conflict when an IPBACK contact interface is in effect. The failure of the constraint-based contact to tie should not cause the weld to be deleted if the penalty-based contact ties.
- Fix reported contact forces and energies for *CONTACT_-TIED_SHELL_-EDGE_TO_SOLID.
- Fix bug for *CONTACT_ERODING_NODES_to_SURFACE. The slave nodes' thickness was ignored, which was problematic for SPH simulations, for example. SST is now taken into account for this contact.
- Fix a problem with SMOOTH contact during implicit springback.
- Enhance implicit processing of tied contact and rotational dofs on slave nodes for special cases with *CONTACT_..._CONSTRAINED_OFFSET.
- Adjust logic for unconstraining rigid body slave nodes for SMOOTH contact to be compatible with deformable to rigid switching.
- Enable the use of *CONTACT_FORMING_..._SMOOTH for implicit.
- Fix the problem of tied contact in MPP having a slave node from another process working with new approach for tied contact.
- Correct initialization of MPP tied contact with implicit mechanics when explicit is running dynamic relaxation followed by implicit.
- Fix a bug in assignment of contact thickness for null beams. This bug was introduced at r89658/R9.
- Add variable 2DBINR (Card F) for surface-to-surface contact involving 2D seatbelt elements initially inside a retractor. This is only available for SOFT = 2 contact.
- Fix a bug for contact involving high order shell elements that occurs when 8-node shell elements are generated by SHL4_TO_SHL8, or when using large part id like 100000001.
- Add an alternative implementation of ICOR (coefficient of restitution) for contact. The algebraic sign of ICOR determines which implementation is used.
- Command line option, soft = 1to2 converts all possible contacts from soft = 1 to soft = 2
- Command line option, soft = 2to1 converts contacts from soft = 2 to soft = 1
- Fix *CONTACT_AUTOMATIC_GENERAL for spotweld beams when using SSID = 0, i.e. all parts included in the contact, and CPARM8 = 2.
- Implement unloading curve, UNLCID, for options FCM = 2/3 in *CONTACT_RIGID_(OPTION).
- Fix zero frictional energy output to glstat and sleout when using *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_ORTHO_FRICTION.
- Implement SMP consistency mode, i.e. ncpu < 0, for:
  - *CONTACT_SURFACE_TO_SURFACE_CONTRACTION_JOINT
  - *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE

- Fix seg fault when using *CONTACT_AUTOMATIC_NODES_TO_-SURFACE for SPH elements impacting on shell plate.
- Fix *DEFORMABLE_TO_RIGID switching using contact forces, CODE = 2 & 4, when the contact type is *CONTACT_FORCE_TRANSDUCER.
- Fix zero contact stiffness if *EOS_TABULATED is using curve LCC instead of tabulated input cards.
- Allow *CONTACT_TIED_SURFACE_TO_SURFACE to use MAXPAR in Optional Card A. It was previously hardcoded to 1.01.
- Implement edge treatment option SRNDE = 1,2 (Optional Card E) for *CONTACT_AUTOMATIC_NODES_TO_SURFACE.
- Fixed MPP segment based (SOFT = 2) contact when used with 2-surface force transducers. Output data may have been incorrect due to errors in accessing arrays. This error occurred when different processors participate in different contact interfaces.
- Fixed MPP implicit SOFT = 2 contact which was failing to properly reset the segment and node data from a converged state leading to possible "floating invalid" error or possible convergence failure.
- Fixed an MPP SOFT = 2 contact problem that could occur if a contact interface used SBOXID to eliminate slave segments. This could cause a segmentation fault during initialization.
- Fixed a possible memory error during initialization of SOFT = 2 contact, when using DEPTH = 5, 15, 25, or 35 or SBOPT = 4 or 5.
- Fixed an error in MPP SOFT = 2 eroding contact that could cause force spikes and inappropriate contact detection after solid elements erode.
- Fixed an error in SOFT = 2 eroding contact that caused the contact to fail to create new segments on interior elements when there were no active segments at the start of the run. This error was most likely to occur in MPP runs when a decomposition can result in a process that has no active segments at the start of the run.
- Changed the behavior of SOFT = 2 contact when the contact keyword indicates a surface to surface contact, but the master surface has no segments. In older versions, the contact would use the slave segments and do a single surface contact with them. Going forward, the contact will not do anything because the slave surface has nothing to contact.
- Fixed some contact options when using linear 3D solids or linear shell elements in implicit analysis. These element were causing a zero explicit time step, and some contacts use this explicit step in the denominator of the stiffness matrix. These include SOFT = 1, SOFT = 2, and tied, penalty-based contacts.
- Added a gap calculation to SOFT = 2 contact. The gaps are written to the binary interface force file. This is supported for both SMP and MPP. Overlaps are reported as negative gaps.
- Fixed a flaw in the support of SSF on *PART_CONTACT when using SOFT = 2 contact with DEPTH = 5. The consequence was to sometimes

choose the wrong part when looking up the scale factor so the stiffness was not predictable.

- Added variable SSFTYP on card F of *CONTACT which affects how SSF on *PART_CONTACT works when used with SOFT = 2 contact. By default, the contact chooses a master segment for each pair of segments in contact. The SSF factor is then taken from the slave segment. When SSFTYP is set to 1 (or any nonzero value), it changes the behavior so the maximum SSF value of the 2 segments in contact will be used. The contact stiffness is therefore independent of how the contact chooses master and slave segments.
- Fixed MPP *PART_CONTACT when used with SSF. It was possible for SSF to be incorrectly taken as zero, so penetration would occur.
- Improved the accuracy of *CONTACT_2D_AUTOMATIC contact. For problems where penetration depths are small, there was an accuracy issue that could cause contact forces to be set to zero in cycles when they should not have been, which could cause a loss of contact history. Because the penetrations are small, the results look about the same, but the contact pressure history is smoother.
- Fixed *CONTACT_2D_AUTOMATIC so it will correctly add new segments when elements using *MAT_081 or*MAT_082 are used for an eroding part.
- Fixed *CONTACT_2D_AUTOMATIC_TIED which could could fail if more than 1 tied contact was used in a model.
- Fixed the sliding option, ISLIDE, of *CONTACT_2D_AUTOMATIC type contacts in both SMP and MPP.
- Fixed MPP *CONTACT_2D_AUTOMATIC_SINGLE_SURFACE and *CONTACT_2D_AUTOMATIC_SURFACE_TO_SURFACE which had a problem in the bucket sort where the buckets may have been too small if segments passed from other processors were larger than the segments already in the processors. This could cause some contact to be missed. It could also cause the job to terminate with Error SOL+1274 if the passed segments are large enough such as can occur with shooting nodes.
- Added an option to offset the contact surface segments attached to 2D solid elements in *CONTACT_2D_AUTOMATIC. The option is controlled by two new parameters, SLDSOS and SLDSOM, on new optional card 5. If the card is omitted, or SLDSOS or SLDSOM are input as zero, then there is zero offset which has always been assumed for surfaces of solids. If input with a value greater than zero, then the offset is set equal to the input value. This behavior is not consistent with the 3D SLDTHK option, which uses half the input value, but it is consistent with the negative value option of SOS and SOM on card 2 which uses the input value, not half of it.
- Add option NEHIS to *USER_INTERFACE_FRICTION:

    - With NEHIS = 0 (default), special choice of element history variables is provided in subroutine usrfrc, see comments there.
    - With NEHIS > 0, plastic strain and element history variables up to NEHIS-1 are provided in original order in subroutine usrfrc.

- Add new *CONTACT_AUTOMATIC_..._TIEBREAK model OPTIONs 13 and 14, which are based on *MAT_240.
- Allow unlimited number of history variables for user-defined tiebreak, *CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIE-BREAK_USER.
- IGNORE = 2 on *CONTACT_..._MORTAR is set to default to avoid unnecessary bad behavior.
- If ISYM < 0 on *CONTROL_CONTACT, this will point to a node set containing all nodes on symmetry planes in the model, which will be picked up by the mortar contact and treat edge treatment accordingly.
- If MPAR1 < 0 on *CONTACT_..._MORTAR for IGNORE = 3, this will govern the penetration reduction as a function of a curve.
- If SLDTHK < 0 on *CONTACT_..._MORTAR, the contact surface is offset "inwards", previously only a positive value was supported.
- Implicit tied contacts (strong objective) supported for groupable option, which is necessary when running a coupled thermal/mechanical simulation.
- User friction supported in mortar contact, see *USER_INTERFACE_-FRICTION. The subroutine is called mortar_usrfrc and found in dyn21.F.
- FS = 2 on *CONTACT_..._MORTAR supported, allowing friction as function of sliding velocity and contact pressure in mortar contact.
- For large penetrations in mortar contact during implicit analysis, the step will be abandoned and a retried with a smaller time step.
- *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED_-WELD supported, SMP and MPP, particularly intended for implicit.
- When mortar tied contact is used with shells, offsets are accounted for to induce a moment from tangential tractions in the interface.
- In mortar contact, let PENMAX and SLDTHK take over the meanings SST and TKSLS have in R9 and earlier, although in a different way. Now PENMAX corresponds to the maximum penetration depth for solids. SLDTHK is used to offset the contact surface from the physical surface of the solid element, instead of playing with SST and TKSLS, which was rather awkward. This update also saves the pain of having to treat shells and solids in separate interfaces if these features are desired.
- A "shooting node logic" algorithm is implemented for mortar contact and is always active. This should reduce the presence of negative sliding energies when run in explicit and also improve initial penetration situations in implicit.
- The MPP groupable contacts include all eroding contacts, i.e.

    - *CONTACT_ERODING_NODES_TO_SURFACE
    - *CONTACT_ERODING_SURFACE_TO_SURFACE
    - *CONTACT_ERODING_SINGLE_SURFACE.

- The MPP groupable contacts include beam-to-beam treatment in *CONTACT_AUTOMATIC_GENERAL. It supports rectangular beams, and output of contact forces to rcforc for an accompanying 2-sided force transducer.
- The MPP *CONTACT_AUTOMATIC_SINGLE_SURFACE_TIED is implemented and supports parameter CLOSE.
- MPP's *CONTACT_AUTOMATIC_BEAMS_TO_SURFACE_ID now supports the contact of slave beam nodes from both sides of the master surface.
- Fix incorrect *INITIAL_STRESS data written to dynain in MPP when np > 1.
- *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE:
  - Improve SOFT = 6 behavior. In guide pin contact, consider the pin's curved edge, therefore overcome the false penetration problem.
- *CONTACT_FORMING_NODES_TO_SURFACE_SMOOTH:
  - Allow segment definition, in addition to part or part set ID.

- ***CONSTRAINED**
  - Fix MPP message passing error that could occur if a node involved in *CONSTRAINED_SHELL_TO_SOLID is shared between more than 2 processors.
  - *CONSTRAINED_GLOBAL: Added user-defined tolerance parameter TOL in length units. If non-zero, the internal mesh-size dependent tolerance gets replaced by this value.
  - Added option to continue a calculation with constrained interpolation after an independent node is deleted using INDSW in *CON-STRAINED_INTERPOLATION.
  - Fix to exactly singular constraint matrix for *CONSTRAINED_-INTERPOLATION running in fast/vector mode.
  - Correct MPP communication during the detection of co-linear nodes in *CONSTRAINED_INTERPOLATION.
  - Add additional error checking on coincident nodes for revolute and screw joints.
  - Promote linear algebra processing for *CONSTRAINED_INTERPOLATION in explicit single precision to real*8 to remove round-off accumulation errors.
  - A more robust solution to the processing of birth and death times for prescribed motion constraints for the LaGrange Multiplier Formulation of joints for explicit.
  - Add implicit capabilities pertaining to UNITCELL:
    - Always unconstrain the last 3 nodes in input and output if they are referenced in *CONSTRAINED_MULTIPLE_GLOBAL constraints. These are control points for UNITCELL.
    - At the end of implicit time step always go through the computation to compute the resultant forces. If there are *CONSTRAINED_-MULTI-

PLE_GLOBAL constraints, output the resultant forces on the last 3 nodes.

- Enhance nodal constraint handling for superelements using explicit to properly handle the implicit/explicit switching case. I also extended the code to recognize *BOUNDARY_SPC definitions instead of just those on the *NODE cards.
- The logic to handle *CONSTRAINED_INTERPOLATION with large number of independent nodes extended for *CONTROL_IMPLICIT_MODES.
- Implicit was enforcing birth and death times on *BOUNDARY_SPC during dynamic relaxation contrary to the manual. So such times are now ignored with dynamic relaxation.
- Added a warning about the combined use of rigid body stoppers and the lagrange multiplier formulation of joints for explicit recommending switching to the penalty formulation.
- Corrections to get implicit to work with *CONSTRAINED_-INTERPOLATION with lots of independent nodes especially in MPP.
- *CONSTRAINED_BEAM_IN_SOLID:
  - Add _PENALTY option to invoke a penalty-based formulation (implicit and explicit).
  - Implicit support for IDIR = 1 (allow sliding along axial direction).
  - Implicit support for AXFOR (user prescribed debonding force between beam and solid).
  - Thermal solver recognizes *CONSTRAINED_BEAM_IN_SOLID by constraining temperature fields between beam and solid nodes.
  - Write LSDA format output file named "cbisfor" containing debonding force (constraint-based) or penalty force (penalty-based).
- Add two keywords *CONSTRAINED_SHELL_IN_SOLID and *CONSTRAINED_SOLID_IN_SOLID, which are similar to *CONSTRAINED_-BEAM_IN_SOLID, but are used to couple shells immersed in solids and solids immersed in solids, respectively.
- Fix a bug affecting *CONSTRAINED_LOCAL whereby z-translation was mistakenly constrained when IRC = 0.
- Fix a bug for PIDCTL of *DEFINE_CURVE_FUNCTION, that occurs when using "0" as sampling rate.
- *CONSTRAINED_SPOTWELD:
  - Enable the normal and shear forces at spotweld failures to be temperature-dependent functions.
- Allow the same part to be used in SPR connections, i.e., MID and SID can now match in *CONSTRAINED_SPR2 or rather PID1 and PID2 can match in *CONSTRAINED_INTERPOLATION_SPOTWELD. The requirement for

this is that the SPR node lies in between the shell elements to be self-connected.

- *CONSTRAINED_INTERPOLATION_SPOTWELD, MODEL = 1, can now be used as connection of two beam nodes.
- Several updates for *CONSTRAINED_INTERPOLATION_SPOTWELD (new card 5):
  - Incorporate torsion mode with GAMMA > 0.
  - Allow definition of separate stiffnesses STIFF, STIFF2, STIFF3, and STIFF4 for tension, shear, bending, and torsion.
  - Optional exponential damage law via LCDEXP.
  - Alternative shear kinematics treatment, parameter SROPT.
  - Output of history variables with NEIPB = 7 on *DATA-BASE_EXTENT_BINARY.
- Add GISSMO damage as failure variable for *CONSTRAINED_TIED_NODES_FAILURE.
- Fixed bug in processing *CONSTRAINED_NODAL_RIGID_BODY and *PART_INERTIA so code does not abort with a bogus "Error 10144".
- *CONSTRAINED_COORDINATE:
  - Fix a machine dependent error when the input coordinates are far away from the part.
  - Extend to rotational constraints. The constraint will be appled to the closest node.
    When IDIR > 10, IR = IDIR-10:
    IR = 1, constrain rx
    IR = 2, constrain ry
    IR = 3, constrain rz
    IR = 4, constrain rx+ry
    IR = 5, constrain ry+rz
    IR = 6, constrain rz+rx
    IR = 7, constrain rx+ry+rz
  - Fix a segmentation fault problem when IDIR is greater than 10.
- *CONSTRAINED_RIGID_BODY_STOPPERS:
  - Extend to have multiple rigid body stoppers for one single rigid body.

- **CONTROL**
  - *CONTROL_REFINE_SOLID and *SECTION_SOLID (ELFORM = 2):
    - Support fully integrated S/R solids.
  - *CONTROL_REFINE_... (IBOX, 1st line, 5th column):

- Implement *DEFINE_BOX_LOCAL and *DEFINE_BOX_ADAPTIVE to refine elements checking criteria provided by these box keywords.

- *CONTROL_ADAPTIVE (ADPTYP = 8, 1st line, 3rd column) and *SECTION_SHELL (ELFORM = 12):

  - Map the plane stress shell thicknesses for the 2D adaptive remeshing

- Staged construction (*CONTROL_STAGED_CONSTRUCTION, *DEFINE_STAGED_CONSTRUCTION_PART):
- Change the behaviour when a part is added.
  Previously, the stiffness and strength of the part would ramp up gradually according to ramp time ATR on *DEFINE_CONSTRUCTION_STAGES. Now, the stiffness and strength have their full value immediately at the start of the stage when the part becomes active (STGA). The reason for the change is to prevent unrealistic deformations occurring while the part is only partially stiff. The same change applies if a part is added using STGA on *LOAD_STIFFEN_PART. There is no change to gravity loading associated with *DEFINE_STAGED_CONSTRUCTION_PART: when a part is added, the gravity load still ramps up according to ramp time ATR. Nor is there any change to the behaviour during part removal.
- Other updates to Staged Construction are:

  - Enabled _TITLE option for *DEFINE_CONSTRUCTION_STAGES and *DEFINE_STAGED_CONSTRUCTION_PART. The titles are ignored by LS-DYNA and are used only in pre-processing. Previously, adding _TITLE would have caused an input error.
  - Fixed bugs in energy balance output for staged construction (*DEFINE_STAGED_CONSTRUCTION_PART and also *LOAD_STIFFEN_PART and *LOAD_GRAVITY_PART). The internal energy was wrongly calculated for dormant parts, and the gravity loading could sometimes be missing from the external work calculation, depending on the other contents of the model. These bugs affected the energy outputs (e.g. in glstat file) but did not affect the other results (stress, displacement, etc).
  - Fix bug in dynain file related to the format of *INITIAL_PWP_NODAL_DATA card, which is written if the model contains *CONTROL_PORE_FLUID. The resulting dynain file could not be read into LS-DYNA.
  - *DEFINE_CONSTRUCTION_STAGES have option inputs RTS, RTE ("real time" at start and end of stage). This is intended to enable an "accelerated analysis" in which processes that really take days or weeks can be modelled in seconds of analysis time. These inputs previously had no effect, they were present only to help the user understand the

times in the input file. Now, "real times" will be written to the output files in place of analysis time.
- Fixed bug that occurred with *CONTROL_STAGED_CONSTRUCTION if Dynamic Relaxation was also switched on. Staged construction is compatible with D.R. only if the analysis begins at the first stage (i.e. time = 0). If not, D.R. now gets switched off automatically and a warning is issued.
- Enabled *LOAD_GRAVITY_PART for tshells.

- **CONTROL_ADAPTIVE**

  - Fix to avoid seg fault in R-adaptivity if 2*(number of shells) > 3*(number of nodes).
  - Fix problem with 2d adaptivity and boundary merging. Some boundary points between materials weren't merged in some cases, depending on where the program thought the 2d contours started. This only applies for *CONTROL_ADAPTIVE ADPTYP = 8 with mmm2d = 1.
  - Fix excessive memory growth during adaptive problems, and reduce memory requirements overall.
  - Generate input error message if *DEFINE_FILTER is used together with *CONTROL_ADAPTIVE, because these filters just won't work with adaptivity.
  - Increase the size of some statically allocated arrays so that larger 2d adaptive problems can be run.
  - *CONTROL_ADAPTIVE:

    - Added Card 5 with only one input variable called INMEMORY.
      EQ.0: conventional out of core shell adaptivity
      EQ.1: in-core shell adaptivity with load rebalancing.

  - *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS:

    - The VID of the rotating axis can now be defined by both *DEFINE_VECTOR and *DEFINE_VECTOR_NODES. It enables the movement of the rotating axis. Previous verion only use *DEFINE_VECTOR to define the VID.
    - The rotational dynamics can work in MPP now.

  - Added variable ICRFILE in *CONTROL_OUTPUT so that nodes sets and element sets associated with *DATABASE_CROSS_SECTION are written to a file to facilitate checking of the cross-section definition.
  - The adaptive element size range defined by RMIN/RMAX in *DEFINE_BOX_ADAPTIVE can now be out of the range defined in *CONTROL_REMESHING for 3D adaptivity.
  - *CONTROL_ADAPTIVE:

        - Add warning message that IREFLG is not supported in 3D adaptivity (ADPTYP = 7).

    - *CONTROL_ADAPTIVE:

        - Add a new feature to the adaptivity of sandwich part: allow multi-layers of solid core to be refined.
        - Support the refinement of 6-node solid elements in the refinement.
        - Fix a bug in the output of 'adapt.msh': an extra blank line was output in *ELEMENT_SHELL_THICKNESS.
        - Fix a duplicate beam error with adaptivity involving beam elements.

    - *CONTROL_ADAPTIVE_CURVE:

        - Fix a bug where the refinement width control was not functioning properly.

- **Discrete Element Method**

    - Switch material type from rigid to elastic for DEM.
    - Enhance MPP's particle-DEM coupling algorithm for *PARTICLE_BLAST.
    - Fix bug if de_massflow_plane and interface force file for *DEFINE_DE_TO_SURFACE_COUPLING are both defined.
    - Fix MPP bug for *PARTICLE_BLAST if some processors contain solid elements while other processors have no solid elements.
    - Fix bug for *PARTICLE_BLAST if solids are used as geometry for HE particles.
    - Implement rebalancing algorithm for *PARTICLE_BLAST, the performance of particle-particle contact and particle-structure contact is increased by ~10~30 times for hundreds of cores----MPP only.
    - Fix bug for *DEFINE_DE_MASSFLOW_PLANE output error if multiple planes are defined.
    - Fix bug for *DEFINE_DE_INJECTION if multiple injection planes are defined.
    - Implement eroding coupling between particle and structure (shell/solid) for *PARTICLE_BLAST. New surfaces of eroded solid parts are taken into consideration.
    - Fix MPP bug when detonation point is defined using a node for *PARTICLE_BLAST.
    - Fix wear depth calculation error for DEM interface force file (MPP only).
    - Skip *PARTICLE_BLAST calculation during dynamic relaxation phase.
    - Fix MPP bug when there are multiple *PARTICLE_BLAST definitions.
    - Add calculation of coordination number for DEM in MPP.
    - Add automatic bucket sort for *DEFINE_DE_TO_SURFACE_COUPLING.
    - Reformulate particle injection algorithm for MPP such that particles with different radii can be injected.

- Add birth time and death time for *CONTROL_DISCRETE_ELEMENT.
- Fix bug in calculation of capillary force for *DEFINE_DE_-TO_SURFACE_-COUPLING.
- Fix bug for capillary force calculation if DEM model has zero potential contact pairs but non-zero capillary force.
- Implement DEM mass output to interface force file for *DEFINE_DE_TO_-SURFACE coupling.
- Add mass output for demrcf.
- Add moment output to demrcf (SMP only).
- Report damping energy for DE non-reflecting boundary conditions.
- Implement explicit thermal modeling of DES (SMP only).
- Implement user defined curve to get mass flow rate for *DEFINE_DE_IN-JECTION.
- Fix minor bug for *DEFINE_DE_TO_SURFACE_COUPLING when the thickness of shell is much larger than segment size.

- **EFG (Element Free Galerkin)**

  - Variables STRFLG and INTOUT in *DATABASE_EXTENT_BINARY can now be turned on when using 3D adaptivity on EFG solid.
  - Special decomposition is implemented for EFG shell formulation 41 to avoid the memory error in MPP.
  - Automatically set IPS = 0 (no pressure smoothing) for EFG solid formulation 41.
  - *MAT_076 is now supported for EFG shell formulation 44.
  - Fixed incorrect stresses in:

    - Plane strain EFG formulation (shell type 43) in both explicit and implicit,
    - Axisymmetric EFG formulation (shell type 44) in explicit.

- ***ELEMENT**

  - *INTEGRATION_BEAM with different PIDs at each integration point: this was an existing capability, but the time step calculation was overly conservative for cases where the section contained a small proportion of a stiffer material, such as the reinforcement in reinforced concrete. For the same reason, LS-DYNA could add large amounts of mass-scaling unnecessarily. The time step calculation has been improved to remedy this.
  - Fixed bug for beam elements ELFORM = 2: with certain combination of inputs only, the output forces and moments could be wrongly rotated about the beam axis. This affected the output forces only, not the solution inside LS-DYNA. The error could take two forms: (a) if IST on *SECTION_BEAM is non-zero, the output forces and moments are supposed to be rotated into the beam's principal axis system, but this rotation could be applied to the

wrong beam elements; and (b) when no ELFORM = 2 elements have IST, but the model also contains beams with ELFORM = 6 and RRCON = 1 on the SECTION_BEAM card, some of the ELFORM = 2 elements could have their output forces and moments rotated by one radian. These bugs are now fixed.

- Speed up keyword for some models that use lots of *ELE-MENT_SOLID_-TET4TOTET10 due to too many memory allocation calls.
- *SECTION_BEAM: Added a flag ITORM for improved representation of torsional modes which can be activated (ITORM = 1) only if an eigenvalue analysis is performed and which applies only to beam type 13. If ITORM is not active (ITORM = 0) the torsional inertia from structural analysis is used which may result in too large eigenvalues related to torsional modes.
- Added cubic solid element formulations 27,28,29 (*SECTION_SOLID).
- Added section forces for higher order solid elements.
- Added element deletion capability for higher order elements.
- Correct issue with prescribed motion on superelements for explicit mechanics.
- Enhance the application of damping to superelements for explicit.
- Tune up output of consistent elemental mass matrices. Needed to capture the lumped mass terms that had been removed. This code changes output of the elemental mass matrices to include the lumped terms.
- Enhance *ELEMENT_DIRECT_MATRIX_INPUT where the matrices for the same superelement do not match in order.
- SEATBELT:

    - Fix MPP bug that could occur where there is more than one type-9 seatbelt pretensioner.
    - Fix misprinted section number and material number for 2D belt error message.
    - Limit the bending stiffness of *MAT_SEATBELT to implicit analysis only. Warning will be issued when non-zero bending stiffness is defined for explicit analysis.
    - Fix a MPP bug whereby incorrect belt material length information is output to d3hsp.
    - Fix a bug that occurs when time-dependent slipring friction is used for 2D belts.
    - Fix erroneous 1d seatbelt slipring message that has been there since version R8.0.0.
    - Enhance *DATABASE_RECOVER_NODE so that it works for shell form 23 with 3x3 integration.

- *USER_NONLOCAL_SEARCH:

    - Provides an interface for gathering the history data of specified elements that surround a “master” element to average (or smooth) the

history data of the master element. The surrounding elements are determined using a user defined strategy. The type of averaging is also user specified. Currently this keyword only works for solid elements.
- supports solid form 1, 16, 19, 21
- supports _ORTHO to search in material local abc directions
- enables user to define a nonlocal search in ellipsoid shape

- Added cohesive element formulation -29 that defines the cohesive midlayer from the average of surrounding shell normals. This fomulation is better suited for simulating normal shear.
- Fixed absence of part mass in d3hsp for cohesive shell element formulation 29.
- Fix memory error when using *NODE_THICKNESS_SET and also fix node set not found error when using *NODE_THICKNESS_SET with *SET_-NODE_INTERSECT.
- Make *TERMINATION_DELETED_SOLIDS work with hex spot weld failures.
- Fix seg fault when using shell type 15, axisymmeteric volume weighted, with *MAT_EROSION and also materials with equation-of-states.
- Fixed implicit accuracy when using higher order shell form 23. The element was
  forced to use 3x3 integration but now supports both 2x2 and 3x3.
- Modified discrete beam form 6 when it is defined with rotational stiffness, but no rotational damping. This change adds a small amount of damping to prevent beams from excessive oscillations which can cause error in the element strain measure.
- Fixed the NREFUP option (*CONTROL_OUTPUT) when the ORIENTATION option of *ELEMENT_BEAM is not active and some other options are not active. The NREFUP option is available for beam types 1, 2, 11, 12, and 14.
- Fixed the thinning of shell form 24 which was thinning about 33% more than it should under large tensile strain.
- Enabled adaptive tet meshing (ADPTYP = 7 on *CONTROL_ADAPTIVE) to work for any reasonable solid element form at the start of the calculation. Previously, the part could only be tet meshed with form 13 elements, or else EFG solid elements 41 or 42, or the job would crash at the first remap step. Now it should work with any structural element form.
- Improved the accuracy of tetrahedral solid form 13. During rigid body translation, some pressure could develop. Now it can translate pressure free, and therefore stress free.
- Fixed beam elements when some elements in a part use *ELE-MENT_-BEAM_ORIENTATION, and some do not. The affected beam forms are 1, 2, 11, and 14.
- Modified the behavior of isotropic materials that are used in composites that are modeled with tshell form 5. The oscillation check on the thickness stress

was modified to directly check stress rather than a total strain. For nonlinear materials, this is more reliable as it was possible to miss the oscillations when checking strains. This should have little effect on most solutions.

- Fixed tshell forms 5 and 7 when used with total Lagrangian material models such as *MAT_002. Large rotations were leading to incorrect strains and stresses.
- Fixed tshell form 3 when used with the hyperelastic materials. The L matrix was in the global coordinate system instead of the local system.
- Changed the d3hsp output of 3-node triangular shell elements so that zero thickness is reported for node 4. The inputted node 4 value was being output and included in the element thickness calculation, but now it is ignored.
- Fixed output of membrane shell form 5 when used with materials other than 34 and 134. The stress and history data written to the d3plot file was wrong.
- Fixed the behavior of CDL and TDL on *SECTION_DISCRETE when one of the nodes is constrained by an SPC. Also, corrected the deforc output when CDL and TDL limits are reached.
- Fixed the strain output in eloutdet for higher order shell form 23.
- Improvement of error checking of cohesive elements (solid forms 19/20/21/22). In addition to checking for adjacent elements (shells, tshells or solids), we now check for being part of a tied contact definition. This eliminates a lot of unnecessary warning messages.
- Fixed adaptivity (and restart) for tetrahedral solid formulation 13 by porting nodal averaged data between adaptive steps. Before this fix, spurious peaks in cross-section forces could occur and be mistaken for contact pressure peaks.
- Add new option for *PART_STACKED_ELEMENTS to deal with in-plane composed reference mesh parts. Must be used together with *NODE_-MERGE_-SET.
- Add cross section forces (*DATABASE_SECFORC) for 20-node and 27-node hexas, i.e., solid forms 23 and 24.
- Add variable ITOFF in *SECTION_BEAM to control torsion behavior for spotweld beams (beam form 9).
- Improve time step estimate for 10-noded tet form 16 with "curved" edges to improve robustness.
- Add variable ICRQ = 2 on *CONTROL_SHELL to only treat thickness continuously across element edges. This is an alternative to ICRQ = 1 which treats both thickness and plastic strain continuously across element edges.
- For higher order shells, the body loads account for shape functions when assembling nodal forces.
- The kinematics for warped/hourglassed cohesive solid forms 20/22 are now corrected so that rigid body motion should give zero eigenvalues.
- If IACC = 1 on *CONTROL_ACCURACY, 6-node quadratic shell form 24 is treated
- According to Martin & Breiner for alleviating membrane and shear locking.

- If IBEAM on *CONTROL_IMPLICIT_EIGENVALUE is set to 2 or 13, then beam formulations 1, 4 and 5 undergo section conversion from an integration rule to resultant properties.
- Fixed error for reading *ELEMENT_SHELL_OFFSET_COMPOSITE.
- Fix parsing errors in *SECTION_BEAM_AISC.
- Fix error in reading of user-defined shell elements.
- A new 4-node tetrahedral solid element (ELFORM = 60) is implemented to support mixed materials. The volumetric locking is eliminated without nodal averaging.

- **EM (Electromagnetic Solver)**

  - EM Analysis developments are divided into several subcategories below.
  - Batteries

    - Added 2 extra variables in d3plot in EM_FEMSTER_NODE for batteries:
      D3PL_RAND_areaCircuit_EM
      D3PL_RAND_areaCell_EM
    - For local area of each Randle circuit and global area of the cell.
    - Added meshless Randle model, see *EM_RANDLES_MESHLESS.
    - Fixed bugs for addition of joule heat rate in meshless Randle.
    - Added battery model with composite tshells.
    - Switched from *EM_CIRCUIT_RANDLE to *EM_BATTERY_-RANDLES for solids and *EM_RANDLE_LAYERED for composite tshells.
    - In *EM_BATTERY_RANDLES, *EM_RANDLE_LAYERED and *EM_-RANDLE_SHORT, Randle Area = 2 is now the default, and the old Randle Area = 0 ("as is") is now Randle Area = 3.
    - Added cylindrical cells.
    - Fix bug in build of the layered (composite tshells) circuits when layers end up by a separator (like in cylindrical cells).
    - Addition of RANDLES instead of RANDLE in *EM_RANDLE_... keywords to be consistent with the manual (the old RANDLE still works).
    - Fix bug in building layer mesh for addition of different composite tshell parts.
    - Addition of Randle Area in randles circuit for composite tshells.
    - Addition of Randles circuit in connection with new LS-PrePost battery packaging for solid elements, in serial and MPP.
    - Addition of optional joule heating from a meshless Randle circuit to a set of parts (uniformly). This is triggered using *EM_ISO-POTENTIAL_CONNECT.

  - II. Electrophysiology

- Added Monodomain (EPMD) and bidomain (EPBD) solvers for electrophysiology.
- Addition of bath loading in EP models, both in EPBD, and in EPMD, using an augmented monodomain approach. This should work both in serial and MPP.
- Added calcium concentration at nodes vector in EP so that it can be used in mechanical models. It can be visualized using d3plot in relative permeability for now.
- Added PCG in EP: the user can choose between MF2 and PCG using EM_SOLVER_FEM.
- Added tetrahedrons in EPMD and EPBD.
- Addition of *EM_EP_TENTUSSCHER_STIMULUS2 to create stimulus on a node set where the amplitude is time dependent given by a load curve (amplitude vs time). Several such stimuli can be created at the same time.
- Addition of user cards for *EM_EP_TENTUSSCHER and *EM_EP_-TENTUSSCHER_STIMULUS for user to input parameters for electrophysiology.
- Addition of beta and Cm in *EM_MAT_003 and *EM_MAT_005.
- Added Godunov method using MF2 in MPP for EP bidomain.
- Addition of *EM_MAT_005 with 2 conductivity tensors for EP bidomain model.
- Addition of implicit and first order operator split (+ combination of the 2) in monodomain method.
- Added activation time as a d3plot output as well as an ASCII file (x,y,z,time) at each node. In d3plot, the activation time is at the ohm heating power for now.
- Added transmembrane potential in d3plot output: it is in the scalar potential for now.

◦ III. Resistive Spot Welding

- Addition of resistive spot welding in 2D (rsw2d).
- Addition of contact Joule Heat Rate for contact resistance in rsw2d.
- Addition of resistive heating solver in 2D, for rsw in 2D.
- Added zero out of em_nodeJHrate in em_zeroEMFieldsOut, so that no more JHR from the contact resistance is added after EM is switched off.

◦ IV. Eddy current

- Added option to use *DEFINE_FUNCTION in LCID for the imposed scalar potential circuit type in EM_CIRCUIT. Allows users to use their own circuit equation as input.

◦ V. Inductive heating

- Added option to define NUMLS, F and A with a Load Curve function of the macro time if a negative integer value is entered.

◦ VI. Miscellaneous EM

- Creation of d3p_bemDecomp file if gmv > 0 in *EM_OUTPUT with BEM face domains.
- Corrected an MPP issue for the EM solver that could occur in problems where some solid and shell elements have the same element number.

- **Forming Analysis**

A special form of shell h-adaptivity called "tube adaptivity" can now be invoked using *DEFINE_BOX_NODES_ADAPTIVE. Here, fission and fusion occurs in shells located inside a "tube", that is, a torus-shaped volume, based on the path of a moving tool. This form of adaptivity can help reduce simulation time for incremental forming or roller hemming.

◦ *ELEMENT_BLANKING:

- Fix corner trimming problem of a flat blank, where the corner elements would not follow the trim line when trimmed.

◦ *CONTROL_FORMING_ONESTEP:

- Improve support to-be-unfolded part as a dynain file, by accepting *ELEMENT_SHELL_THICKNESS, *INITIAL_STRESS_SHELL, and *INITIAL_STRAIN_SHELL.
- Add *MAT_123 to onestep method.
- Fix friction force calculation error.
- Improvements: some of the executables were not able to use multiple CPU in SMP before, now it is possible.

◦ *CONTROL_FORMING_TRIMMING, *DEFINE_CURVE_TRIM:

- Fix bug in trimming: some of the history variables were lost during trimming.
- Fix bug: some SPC nodes were mistakenly removed.
- Fix bug in trimming of sandwiched part with multi-layerd core: shells were wrongly created for every layer of the sandwiched core.
- Improve trimming of solid elements: allow the corner to be trimmed exactly as the trim curve.

◦ *CONTROL_FORMING_TRIMMING, *DEFINE_CURVE_TRIM_3D:

- Improve trimming with _3D option: if a vector id is provided (by mistake) with 3D trimming, the code will set the vector to be zero, as 3D trimming does not require a vector.

- *DEFINE_CURVE_TRIM_2D:
  - Change from *DEFINE_CURVE_TRIM_NEW to *DEFINE_CURVE_-TRIM_2D.
- *CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS:
  - Fix segmentation fault.
  - Extend to triangular elements, in addition to quadrilateral elements.
- *CONTROL_IMPLICIT_FORMING:
  - For IOPTION = 2, fix a truncation error that prevented completion of the analysis.
  - Fix a bug that prevented application in dynamic implicit analysis.
  - Fix kinetic energy calculation error.
- *DEFINE_FORMING_CLAMP:
  - Add error message: "Vector: xx was not defined" in case vector's direction is defined incorrectly.
  - Check constraints for the rigid clamper and free the constraints from the moving clampers.
- *DEFINE_PART_FROM_LAYER:
  - Add *CONTACT_SURFACE_TO_SURFACE between layers generated by this keyword.
- *CONTROL_FORMING_AUTOPOSITION_PARAMETER:
  - If separation distance cannot be found, for example, when the MPID is not found when calculating the separation distance, or out of position, or the DIR is not input correctly, instead of returning a very large number, the value of PREMOVE will be returned.
  - Support *DEFINE_COORDINATE_VECTOR.
- *CONTROL_FORMING_OUTPUT:
  - Enable the variable NOUT not only in punch drawing but also in binder closing.
  - Allow output of d3plot according to NOUT without using y1,y2,... or a curve id.
  - Fix a bug to prevent excessive intfor output.
- *DEFINE_FORMING_BLANKMESH:

- Fix X, Y shifting problem (not working) for NPLANE = 2 and 3.

- *INCLUDE_AUTO_OFFSET:
  - Extend to beam and solid elements.
  - Fix a beam offset problem when adaptivity is turned on.
  - Add a new option _USER. User can now control how much the node and shell element are offset. In addition, with this option, the offset can be used for sheet blank part. Without this option, the auto offset can only be used for rigid bodies.
  - Extend _USER option to include solid and beam parts.

- *CONTROL_FORMING_TIPPING:
  - Improve this keyword to allow three source coordinates and the corresponding three target coordinates for the tipping. Use NMOVE = -6 to activate this feature.

- *CONTROL_FORMING_UNFLANGING:
  - Fix a segmentation fault when writing the result file 'unflanginfo.out'.

- *DEFINE_MULTI_DRAWBEADS_IGES:
  - Fix a duplicate node set issue when SMOOTH contact is used with automatic draw beads generation from IGES.

- *CONTROL_FORMING_PRE_BENDING:
  - Add a warning message that the keyword must be placed at the end of the input file.

- *DEFINE_BOX_DRAWBEAD:
  - Fix a bug in calculating the box size.

- *CONTROL_FORMING_STONING:
  - Allow the element set to be defined with *SET_SHELL_GENERAL and *DEFINE_BOX.

- *DEFINE_FIBERS, and *CONTROL_FORMING_ONESTEP:
  - Add a new keyword (*DEFINE_FIBERS) to define carbon fibers and their related properties for one-step inverse forming simulations of carbon fiber reinforced composites.
  - Output initial flat part, which is re-orientated by aligning node N1 and N2 in x-direction.

- Output element fiber information, including orientation and width.
- When N1 or N2 is zero or undefined, another line of input is required for the target coordinates for N1 and N2, and the code will find the nearest nodes.
- Outputs:
  1st history variable: the angle between two fibers.
  2nd history variable: the angle between the first fiber with respect to the element direction.
  3rd history variable: the angle between the second fiber with respect to the element direction.
- Up to three major fiber reorientations are allowed.

◦ *CONTROL_FORMING_STRAIN_RATIO_SMOOTH:

- Change keyword name from *CONTROL_FORMING_TOLERANCE to *CONTROL_FORMING_STRAIN_RATIO_SMOOTH.

◦ *CONTROL_FORMING_AUTOCHECK:

- Add a new variable IFSHARP. When IFSHARP = 0, check the sharp edge and delete the elements; when IFSHARP = 1, ignore sharp edge.
- Check for and fix for triangular elements incorrectly defined with four unique nodes.

◦ *INTERFACE_COMPENSATION_FLANGE:

- Added this new keyword to handle flanging die compensation.

◦ *DEFINE_PART_FROM_LAYER:

- Add a new keyword to generate multi-layers of shells for a composite structure (carbon fiber material, for example).

◦ *ELEMENT_LANCING:

- Add lancing of multi-layered solid elements.

◦ *CONTROL_FORMING_BESTFIT:

- Add some bestfit statistics in the "messag" file after best fit. The first column is the percentage of nodes within the range indicated; the second column is the percentage of nodes within the upper limit of the range indicated.
- Output the maximum gap.

◦ *DEFINE_CURVE_FLD_FROM_TRIAXIAL_LIMIT, and *DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_FLD:

        - Added these two new keywords.

    - *INCLUDE_COMPENSATION_TRIM_NODE:

        - Add a new option which will include a file containing all the nodes near the trimming line. This file should be generated from *INTERFACE_COMPENSATION_NEW_REFINE_RIGID (bndnd0.tmp).

    - *INTERFACE_COMPENSATION_3D:

        - Add a new option _REFINE_RIGID to automatically identify the nodes near the trimming curves, so surface compensation later will be smoother.
        - Add a new variable TANGENT. TANGENT = 1 maintains the boundary tangency from the addendum to the binder.
        - Combine rigid refinement along trimming curve into the main compensation in one single run.
        - Change keyword *INTERFACE_COMPENSATION_NEW to *INTERFACE_COMPENSATION_3D.

- ***FREQUENCY_DOMAIN**

    - Added logic for some frequency response computations so that the amount of drilling rotation control is the same as for eigenvalue computations.
    - *DATABASE_FREQUENCY_ASCII_OPTION:

        - Added modal contribution fraction output to NODOUT_SSD and ELOUT_SSD.
        - Added option NODFOR_SSD as an option to *DATABASE_FREQUENCY_ASCII.
        - Added options NODOUT_PSD and ELOUT_PSD for random vibration analysis.

    - *DATABASE_FREQUENCY_BINARY_OPTION:

        - Added D3ZCF binary database to fringe plot zero-crossing frequencies (with positive slope) in random vibration analyis.
        - Added D3ACC binary database to fringe plot acoustic pressure contribution from boundary elements in BEM acoustic computation.

    - *FATIGUE:

        - Implemented this keyword to run time domain fatigue analysis based on stress or strain.
        - Added new mean stress correction methods: Goodman-tension-only and Gerber-tension-only to provide conservative analysis for compression mean stress.

- Added an option EN to *MAT_ADD_FATIGUE to define material's EN curve.
- Improvement to skip fatigue computation if the local strain is less than 1.e-6.
- Improved strain based fatigue analysis when using Maximum Shear Strain and using Signed Von Mises strain.
- Added restart option RESTRT to fatigue analysis, if the stress/strain time history has been precomputed.

◦ *FREQUENCY_DOMAIN_ACOUSTIC_BEM:

- Added half space option to dual BEM based on Burton-Miller formulation.
- Implemented incident wave to Kirchhoff method.
- Implemented incident wave to Burton-Miller BEM.
- Implemented incident wave to Rayleigh method.
- Fixed bug in running acoustic analysis with multiple boundary conditions in MPP.
- Enabled running BEM restart (ibemrest = 6) based on atv matrix computed previously. SMP only.
- Generating D3ACS database for collocation bem (method = 3) and dual collocation bem (method = 4) only.

◦ *FREQUENCY_DOMAIN_PATH:

- Added option _NOJOBID, so that users can run restart based on the same eigenvector database for each CASE (otherwise, LS-DYNA will add different prefix to the file name in each CASE automatically).

◦ *FREQUENCY_DOMAIN_RANDOM_VIBRATION:

- Added the following new load types for random vibration analysis:
  VAFLAG = 9 base velocity
  10 base displacement
  11 enforced acceleration by large mass method
  12 enforced velocity by large mass method
  13 enforced displacement by large mass method
- Fixed a bug in running PSD interpolation when log-log interpolation is used and the PSD includes both magnitude and phase delay (for cross PSD).
- Implemented Lalanne method for frequency domain fatigue analysis.
- Added option LCTYP2 to define phase difference in cross psd by degrees and radians.

◦ *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM:

- Changed the rule of dumping nodal displacement results to D3SPCM. Now the state variable of nodal displacement results is the displacement peak value itself, without adding original nodal coordinates. LS-PrePost has been updated to accomodate this change.
- Added xyplot file spectrum_curve_print. This file saves the intermediate base acceleration spectrum, converted from base motion time history (LCTYP = 10, 11, 12).
- Added the following new load types for response spectrum analysis:
  LCTYP = 5 (base velocity vs natural period)
  6 (base acceleration vs natural period)
  7 (base displacement vs natural period)
  8 (nodal force vs natural period)
  9 (pressure vs natural period)
  10 (base velocity time history)
  11 (base acceleration time history)
  12 base displacement time history)
- For group force computation, now we calculate the group force for each mode first and then run the mode combination on them. Previously we calculate the group force as the sum of the nodal force, after the mode combination.
- Added Von Mises stress output for beams for response spectrum analysis.
- Updated response spectrum analysis so that it can work with intermittent eigenvalue analysis.
- Added nodal force and group force output (NODFOR_SPCM) for response spectrum analysis.

◦ *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM_DDAM:

- Implemented DDAM for navy ship shock response analysis.
- Implemented CSM (Closely Spaced Modes) treatment for DDAM.
- Added mcomb = -14 to run DDAM with user defined CSM.
- Added a new parameter EFFMASS, to define a required minimum percentage for total modal mass, to decide the modes to be used in DDAM analysis. The default value is 80 (80%).
- Added a parameter UNIT, to define the unit system in the input deck, as the NAVSEA constants are only valid with the BIN unit system.

◦ *FREQUENCY_DOMAIN_SSD:

- Implemented mean stress correction for SSD fatigue analysis.
- Implemented option restmd = 2 to restart with old scratch file modeshp to save time.
- Implemented an option _FRF to *FREQUENCY_DOMAIN_SSD, to provide FRF results in SSD.

- Added option _MODAL_CONTRIBUTION to output modal contribution fraction for nodes and elements.
- Enabled combination of modal damping and local damping in SSD computation.
- Added the following new load types with rotational degree-of-freedom.
  VAD = 9 (base angular velocity)
  10 (base angular acceleration)
  11 (base angular displacement)

- **ICFD (Incompressible Fluid Solver)**

  - New ICFD features and major modifications:

    - Major rework of the boundary layer mesh generation capabilities. See MESH_BL keyword.
    - Major rework on the wave generation capabilities. Added 2D and 3D solitary waves as well as a irregular wave model (JONSWAP spectra). See *ICFD_BOUNDARY_FSWAVE keyword.
    - Added an option so that pmin and pmax in MESH_SIZE_SHAPE can be defined using *ICFD_DEFINE_POINT. The big advantage is that *ICFD_DEFINE_POINT can move and therefore *MESH_-SIZE_-SHAPE as well. Also added new Shape name: SPOL as well as birth and death times.
    - Added keyword to define a volumetric heat source. See *ICFD_-DEFINE_HEATSOURCE.
    - Added keyword which allows the user to define an initial plane for level set rather than building the initial interface mesh. See *ICFD_-INITIAL_LEVELSET.
    - Added keyword to control gap size in embed shell cases. See *ICFD_-CONTROL_EMBEDSHELL.

  - Small feature additions in ICFD and modifications to existing keywords:

    - Output frequency of d3plot in steady state in now controlled by sixth flag of *ICFD_CONTROL_OUTPUT.
    - Added output frequency for *ICFD_DATABASE_TEMP.
    - *ICFD_DEFINE_POINT : can now be made to follow a surface part's displacements.
    - Allowing *DEFINE_FUNCTION to be used for R and LCID in *ICFD_-DEFINE_NONINERTIAL.
    - *ICFD_CONTROL_GENERAL : potential flow solver can now be used in transient analysis (which can for example be useful in cases like conjugate heat transfer).

        - *ICFD_MAT : added the option to scale the surface tension by using a load curve or a *DEFINE_FUNCTION.

    - Bug fixes in ICFD:

        - Fix temperature SUPG stabilizing parameter for Conjugate Heat Transfer problem.
        - Fix the viscosity as a function of temperature when using Non-Newtonian fluids with NNID = 6,7,8 in Free-Surface problems.
        - Fix the surface tension force term.
        - Fix issue in multiphase algorithm. Stability and accuracy greatly improved.
        - Improved stability of k-epsilon model in steady state solver.

    - Minor ICFD improvements:

        - Change the nodal assembly by an element integration and assembly in anisotropic porous media solver. This way, forces at porous media interfaces are better described for coarse meshes and thin porous media domains.
        - Split the mesh statistics in icfd_mstats.xxx.dat into bulk mesh and boundary layer.
        - Added 'wetness' variable for free surface cases that shows how much a surface has touched the water.
        - Support of icfd_timeiter.dat for the steady state solver.
        - Added timer for potential flow solver.
        - Added a spatial smoothing for the surface shear stress calculation.
        - Added ICFD endtime in d3hsp initial keyword reading.
        - Accelerated heat transfer calculation.
        - Added a small warning that ICFD does not scale in SMP with ncpu higher than 1.
        - Change in unreference node detection criteria. Before, it was stopping with an error message, now it proceeds with a warning.
        - Added warning when porous media model or non newt model not detected.
        - Added avg pre and avg flux to ICFD_DATABASE_FLUX output.

- **Implicit (Mechanical) Solver**

    - *CONTROL_IMPLICIT_MODAL_DYNAMIC:

        - Support output to elout for modal dynamic analysis.
        - Make performance enhancements for transient modal analysis including the implicit Newmark scheme for time integration and a node set for loads in *CONTROL_IMPLICIT_MODAL_DYNAMIC.

- When using a direct solver for implicit (LSOLVR = 2 or 6 in *CONTROL_-IMPLICIT_SOLVER), the system of linear equations is reordered (permuted) to reduce the solution cost. Up to and including version R10, two options were available: MMD (Multiple Minimum Degree) and METIS (external package from University of Minnesota). These two options are serial algorithms, and they might run out of memory or consume a large fraction of total run time for very large models. As of version R11, a parallel, distributed-memory algorithm, LS-GPart, is introduced. It scales in both memory and time and should only be attempted for very large MPP implicit models and with the guidance of implicit support staff at LSTC (support@lstc.com). LS-GPart can be used by setting variable ORDER to 4 in *CONTROL_IMPLICIT_SOLVER). A new keyword *CON-TROL_IMPLICIT_ORDERING was also introduced to fine-tune ordering options.
- Correct a number of minor issues with memory access for implicit.
- Enhance implicit handling of nodal inertia for special cases.
- Fix divide by zero in power iteration for buckling problems with inertia relief.
- Enhancements for matrix dumping left out the special case of matrix dumping for intermittent eigenvalue problems which is now corrected.
- Adjust output to d3hsp for implicit linear equation solver options to match keyword manual.
- Correct the MPP implementation for the new stiffness control on implicit rotational dynamics.
- Adjust shift logic for lanczos eigensolver for a special case.
- Enhance implicit treatment of sense switch sw1 to avoid issues in SMP.
- Apply correction for a problem that computed way too many eigenmodes in one iteration of the MPP eigensolver.
- Add the new feature for the specification of stiffness types for *CONTROL_-IMPLICIT_ROTATIONAL_DYNAMICS.
- Lower the implicit dynamic memory greed factor from 0.85 to 0.80.
- Put end-of-file tag on d3eigv after writing stresses for MCMS eigensolver. Normalize MCMS computed eigenvectors to have unit norm.
- Correct the dumping of matrices from implicit when MTXDMP > 1.
- Correct a misalignment of statements in flxinit causing SMP to fail with *PART_MODES and the use of the PARTM feature from *CONTROL_-RIGID.
- Implement handling of failure of *CONSTRAINED_TIEBREAK for implicit. Required saving of the failure flag to use when constraint matrix structure has to be the same as the last analyze phase.
- Fix long standing potential memory clobber for single precision SMP implicit. (Nevertheless, single precision is not recommended for implicit.)
- Improve implicit logic for determining which DOFs are active for 2D problems in MPP.
- Enhance the modal stress scaling to be more responsive to model features to make the computation more robust.

- Enhance implicit key point logic during dynamic relaxation phase. We were using the incorrect end time which led to a zero time step.
- Add error test for when dynamic relaxation is using implicit (idrflg = 5 or 6) but DRTERM is not specified.
- Correct the MPP implementation of *PART_MODES so that the part can now be distributed across processes.
- Enable implicit to collect resultant forces for SPC constraints in local coordinates.
- Fix *CONTROL_IMPLICIT_MODES to correctly build superelements in MPP.
- For implicit MPP, correct tagging of the end of each d3eigv* file.
- Reduce memory requirements for MPP Lanczos.
- Enhance implicit to recognize superelement mass in the mass matrix multiplication.
- Enhance *CONTROL_IMPLICIT_MODES to be able to create a superelement for a model that was already using a superelement.
- Enhance the specification of IMFLAG < 0 for *CONTROL_IMPLICIT_GENERAL. The old approach only allowed toggling between 0 and 1 using a curve. This was extended to allow ordinate values of 0, 1, 4, or 5 in the curve controlling implicit/explicit switching.
- Enhance processing of rotational inertias for implicit, especially discrete elements and rigid body inertias.
- Add additional implicit debugging by checking the ends of beam elements and what they are or are not connected to.
- Reset more arrays to get the nonlinear elements to work correctly for implicit linear multi-step (NSOLVR = -1).
- If two independent nodes for *CONSTRAINED_INTERPOLATION have the same coordinates then a divide by zero could happen. That has been corrected.
- Add INTERFACE_SPRINGBACK to the cases where implicit collects and processes damping terms instead of suppressing them as is the case for implicit statics.
- Update implicit's collection of damping terms for discrete elements to account for user specified coordinate system and the case of no second node.
- Patch up reporting of bad pivots in MPP.
- For MPP, implement the checking for rigid body node replication, which was already in SMP. Also added proper error termination for analyzing the constraint matrix in MPP.
- Correct dynamic memory allocation for the implicit case for cases with *CONSTRAINED_INTERPOLATION with a large number of independent nodes but no inertia relief.
- Adjust the output value for IMFLAG to be the user specified value instead of the internally adjusted value. Also make sure the user cannot input IMFLAG = 3 which is an internal value that should not be used as input.
- Enhance linear implicit for prescribed motion data on deformable nodes.

- Correct reading of constraint mode aux vectors used by modal dynamics to impose prescribed motion constraints.
- Change typing of integer back to integer*4 to get proper output to binary format of superelement file generated by *CONTROL_IMPLICIT_STATIC_-CON-DENSATION.
- Finish migration of implicit eigensolver using Power Method for Inertia Relief + Buckling to dynamic memory.
- Enhance the separable component report to include rigid bodies.
- Enhance Implicit Usage Alert message so that for SMP it only outputs the recommended value for memory and not the memory2 setting.
- Correct the memory management for the stiffness matrix in SMP as it was freed too early when using implicit modes.
- Made a number of enhancements and corrections for the feature to apply boundary prescribed motion constraints during implicit modal dynamics.
- Added a user supplied linear equation solver capability for implicit mechanics.
- Extend changes for implicit nodal rotational inertia scaling to include *CONTROL_IMPLICIT_MODES.
- Account for rotational nodal inertia terms in implicit computations for discrete nodal inertia matrices.
- Remove double booking of rotational nodal inertia and discrete inertias that have had the rotational nodal inertia added. This will change the eigenvalues computed by implicit by making them more correct.
- Adjust implicit's treatment of inertia's so that implicit intermittent eigenvalue computations will match an implicit non-intermittent eigenvalue computation.
- Add control for the accuracy output to eigout.
- Apply patches to implicit mechanics from the development code as part of our work on solving very large implicit problems.
- Fix a deep and long hidden bug in implicit inertia relief.
- Correct underallocation of storage for inertia relief workspace for problems with more than 6 rigid body modes.
- Convert all of the implicit linear algebra to F95 dynamic memory. Added additional tracking statistics on the use of dynamic memory used by implicit. Enhance output with an Implicit Usage Alert to tell users how to set memory and memory2 for this model.
- Fix the marking of the end of the d3mode file when writing stresses.
- Fix a problem marking the end of the d3eigv file when modal stresses are written.
- Add detection of bad beam input for implicit.
- Enhance keypoint logic to enforce keypoints at the initial time step and on restart from explicit.
- Fix a problem with the timing for when "dead" nodes are incorporated into implicit.
- Added features to allow tighter coupling between implicit and USA.

- Correct output of eigenvalues and frequencies for the nonsymmetric eigenvalue problem to match that of rotational dynamics.
- Improve error message for reading d3eigv file for *PART_MODES for the case when the user input a d3eigv file from a different model than intended.
- Enhance implicit inertia relief to be optional for any explicit phase.
- Added MCMS (LSTC's name for AMLS) approximate eigensolver.
- Corrected the computation of reduced mass matrix for *PART_MODES when the nodes for the part are distributed in MPP. Also corrected the computation of total kinetic energy for the same situation.
- Earlier enhancements to adjust for poorly scaled implicit mechanical problems negatively affected the direct linear equation solver used in the implicit treatment of joints for explicit in single precision. The pivot tolerance was inappropriately being reduced. Now fixed.
- Migration of shell arrays to dynamic storage broke d3iter. Also had to reset i/o address for d3iter on restart.
- When implicit springback was following an explicit transient step the implicit keywords with the _SPR were not properly handled. Now corrected.
- Added the feature of resetting implicit geometry at the start of each time step to enable implicit linear multiple load analysis.
- Fix an implicit problem where a linear implicit analysis follows inertia relief computation.
- Enabled tshells to use the consistent mass matrix option of *CONTROL_IMPLICIT_CONSISTENT_MASS.
- Sense switch sw4 is now supported in implicit, in both SMP and MPP.
- If IMASS = 0 on *CONTROL_IMPLICIT_DYNAMICS, i.e., the analysis is static, the kinetic energy is output as zero to make physical sense.
- If HGEN = 1 on *CONTROL_ENERGY, energy from drilling constraint is included in hourglass energy and thus accounted for in energy balance.
- A nonzero start time of initial velocities is supported in implicit dynamics, see *INITIAL_VELOCITY_GENERATION_START_TIME.
- If IRATE = -1 on *CONTROL_IMPLICIT_DYNAMICS, rate effects are active even in implicit statics, which is sensible if viscoelastic effects are used in quasi-static analyses.
- Rotational prescribed motion on rigid bodies induce a fictitious residual force to avoid initial zero loads, which is a way to prevent stalling of the analysis. Translational prescribed motion was supported in this way in prior versions.
- Four variables (GJADSTF/GJADVSC/TJADSTF/TJADVSC) in *CONTROL_-RIGID automatically add joint stiffnesses to all joints, primarily for use in implicit in models with many joints (e.g., dummies) to stabilize the overall behavior.
- NLNORM = 4 on *CONTROL_IMPLICIT_SOLUTION mixes rotational and translational degrees of freedom for computing residual forces by weighing the rotational contribution with a length scale internally calculated to avoid

a units problem. The length scale used is reported in the output and can be overwritten by NLNORM < 0.

- Implemented a chained *CASE treatment for implicit, for splitting a "complicated" process into several "simple" simulations allowing for transfer of state between each such simulation. This makes use of writing and reading dynain.lsda (*INTERFACE_SPRINGBACK_LSDYNA/*INCLUDE) between cases, and in addition to element stresses, etc. that are the common state constituents, we support
  - mortar contact friction, tied, tiebreak and tied weld
  - tied contact slave node and master segment pairs
  - stabilization history of elements (hourglass and drilling)
- This is an ongoing project which is intended to be supported in an Implicit GUI in LS-PrePost for facilitating its setup.
- If IACC = 1 on *CONTROL_ACCURACY, "bad" implicit element formulations are automatically switched to type 2 (solid) or type 16 (shells).
- Fixed bug in d3eigv output for model with *CONTROL_IMPLICIT_-INERTIA_RELIEF.

- ***INITIAL**
  - *INITIAL_LAG_MAPPING (NELANGL = -1, 2nd line, 6th column):
    If NELANGL = -1, no mesh are generated or projected; just map the data if the 2nd run mesh geometry matches the 1st run one at its final cycle
  - *INITIAL_SOLID_VOLUME:
    Recalculate and reset initial volume of solid elements using material models with EOS before analysis if the original nodal position has been moved by nodal projections in contact initialization. This option eliminates calculation of non-physical initial hydrostatic pressure due to the nodal repositioning.
  - Fix ineffective *INITIAL_VELOCITY_GENERATION for part defined with *PART_INERTIA when ID = 0, STYPE = 0 and IRIGID = 1.
  - Fix incorrect initial velocity when ICID.ne.0 in *INITIAL_VELOCITY_-GENERATION, and rotational velocity, omega, is not zero and *PART_INERTIA is also present.
  - Fixed *INITIAL_STRESS_SHELL and *INITIAL_STRESS_TSHELL when used with *INTEGRATION_SHELL. The integration rule was getting lost leading to unnecessary interpolation of data.
  - Fixed dynain writing and reading of *INITIAL_STRESS_SHELL for the fully integrated C0 shell (shell form 20).
  - Enable multiple *INITIAL_VELOCITY_GENERATION keywords to be used with *ELEMENT_SHELL_COMPOSITE or *ELEMENT_TSHELL_-COMP-OSITE. Only one velocity generation keyword was supported previously.

- With the new parameter IVADD, the *INITIAL_VEHCILE_KINEMATICS velocity field can be superimposed on pre-defined nodal velocities.
- IZSHEAR = 2 on *INITIAL_STRESS_SECTION gets a special treatment for preloading bolts, each bolt is seen as an entity and the constraint is to prescribe the mean stress in the section and not in each element, meaning that the bolt is capable of take bending resistance, currently applies to the common low order elements (type -2,-1,1,2,10,13,15) for explicit and implicit

- Isogeometric Elements

  - Addition of HAZ (heat affected zone) features to IGA shells. No new keywords were added. The capability works identically to standard FE shells.
  - Reduce unit system sensitivity of *CONSTRAINED_NODE_TO_-NURBS_-PATCH_SET.
  - *ELEMENT_SOLID_NURBS_PATCH:

    - Isogeometric solid analysis is now available in both SMP (with consistency flag turned on) and MPP.
    - Activate user defined materials for isogeometric solids.

  - *ELEMENT_SHELL_NURBS_PATCH:

    - Isogeometric shell analysis is now available in SMP with consistency flag turned on. (MPP was already available.)
    - Add a power iteration method to get the maximum eigen-frequency for each isogeometric shell element. This will be used to set a reasonable scale factor for trimmed element.
    - Modify the time step scale factor for IGA trimmed shell element so that the overall time step will not be unreasonably small due to very small trimmed elements.

  - *ELEMENT_SHELL_NURBS_PATCH:

    - Fixed problem in decomposition for heavily trimmed NURBS-patches.
    - The problem may have occurred if most parts of a NURBS-patch are trimmed (not actually part of the geometry). In such cases the MPP decomposition could have gone wrong.

  - Write element information for NURBS shells to d3hsp (Element ID, Part ID, number of nodes and connectivity). This is invoked by setting NPOPT = 1 in *CONTROL_OUTPUT.
  - Fix bug when NURBS shells are present in a model and extra DOFs where assigned (*NODE_SCALAR, or shell forms 24/25/26).
  - Fix for *ELEMENT_SHELL_NURBS_PATCH when applying body force using *LOAD_BODY_GENERALIZED or *LOAD_BODY_PARTS. Apply the body force on the control points.

- Add support for initialization of shell thickness at in-plane int. point for *ELEMENT_SHELL_NURBS_PATCH(_TRIMMED) using *INITIAL_-STRAIN_SHELL_NURBS_PATCH (SMP and MPP).
- Add support for initialization of stresses, plastic strain, history variables and strains for *ELEMENT_SHELL_NURBS_PATCH(_TRIMMED) using *INITIAL_STRESS/STRAIN_SHELL_NURBS_PATCH.
- Add support of writing *ELEMENT_SHELL_NURBS_PATCH_TRIMMED to ASCII dynain file.
- Add support of *PART_COMPOSITE for isogeometric shells.
- *ELEMENT_SHELL_NURBS_PATCH & *CONTROL_ADAPTIVE:
  Allow adaptivity for regular shell elements if NURBS patches are in the model.
- Add possibility to define a negative real value for NISR and NISS to define a desired size of the automatically created interpolation shell elements. This may be especially useful when using NURBS elements for rigid tools in forming applications.
- Allow *CONTROL_FORMING_AUTOPOSITION_PARAMETER with IGA NURBS shells.
- Allow *PART_MOVE with IGA NURBS shells.
- Write stresses of interpolation elements to elout for *ELEMENT_SOLID_-NURBS_PATCH.
- Add support for initialization of stresses, plastic strain, history variables and strains for *ELEMENT_SOLID_NURBS_PATCH using *INITIAL_-STRESS/STRAIN_SOLID_NURBS_PATCH.

- ***LOAD**

  - Fix INCLUDE_TRANSFORM offset for coordinate systems in *LOAD_-NODE_POINT which were using the wrong offset.
  - Added support for *LOAD_THERMAL for higher order (quadratic and cubic) solids. The temperature at the nodes is interpolated from the nodes to the integration points so that the temperature is not the same at all integration points.
  - Correct the issue where use of *LOAD_BODY_ is applied to a model with rigid bodies to avoid null elemental stiffness matrices.
  - Fixed *LOAD_THERMAL_VARIABLE_ELEMENT_TSHELL. It was not working.
  - Fixed *LOAD_THERMAL_VARIABLE when used with shell form 2 that has default warping stiffness (BWC = 2 on *CONTROL_SHELL), and when OSU = 1 on *CONTROL_ACCURACY to activate an objective stress update.
  - *LOAD_SEGMENT_CONTACT_MASK is now supported for mortar contact in both SMP and MPP, in both implicit and explicit.
  - Fix bug in accessing ground motion ID (*DEFINE_GROUND_MOTION) from *LOAD_SEISMIC_SSI.
  - Allow *DEFINE_FUNCTION for *LOAD_THERMAL_LOAD_CURVE.

- *LOAD_SURFACE_STRESS:
  Fix a bug when there are more than 1 contact on one side of the blank, the area calculation was larger and the pressure smaller than it should be.
- *LOAD_BODY_VECTOR:
  Fix a bug in writing binary dynain (MPP).

- ***MAT and *EOS**

  - Fix bug in *MAT_079. The equations giving the influence of pressure on stiffness and strength were not exactly as written in the manual. PREF had been used in place of (PREF-P0). The code has been corrected to match the manual. This change will affect results from existing models, but usually P0 would be given a very small value so in most cases the difference should not be significant.
  - *MAT_089:

    - Now works with Tetrahedron ELFORM = 13. Previously the behavior was the same as for ELFORM = 10 (volumetric locking could occur).
    - Fixed bug affecting solid elements only - timestep calculation was wrong. Response could be unstable especially for higher values of Poisson's ratio, e.g. 0.4. Workaround was to reduce the timestep.

  - *MAT_119 unload option 3 - very small displacements followed by unloading could result in excessive unload stiffness and unexpectedly large mass-scaling, arising from small numerical rounding errors in the interpolated version of the load curves LCIDTR, LCIDTS, etc. Now fixed.
  - *MAT_169 (*MAT_ARUP_ADHESIVE) - enabled for implicit analysis.
  - *MAT_172 (*MAT_CONCRETE_EC2):

    - Fixed bug for combination of MAT_172 with Staged Construction (*CONTROL_STAGED_CONSTRUCTION). While elements were dormant, crushing damage could occur that persisted after the element became active.
    - Through-thickness strain was wrongly calculated when cracks are opening or closing. This strain is only an output parameter and does not affect the other results, but could potentially have led to unexpected element deletion if used with *MAT_ADD_EROSION.
    - Enabled CMPFLG (*DATABASE_EXTENT_BINARY).

  - *MAT_197 (MAT_SEISMIC_ISOLATOR)
  - Added optional rotational stiffness (new optional Card 7). This is useful when multiple isolator elements are stacked on top of each other.
  - Added new TYPE = 3 Lead Rubber Bearing. Includes cavitation in tension, buckling in compression, degradation of shear strength due to heating of the lead core.
  - *MAT_203

- Added history variables 10 & 11 for post-processing: high-tide tensile strains in the two local reinforcement directions.
- Enabled CMPFLG (*DATABASE_EXTENT_BINARY).
- Fix bug in hysteresis behaviour that could occasionally cause error terminations.
- Elements are now eroded when existing input parameter EPDAM2 is reached. If EPDAM2 and EPDAM1 are not defined, or if DRESID is non-zero, the element never erodes.

◦ *MAT_208 (*MAT_BOLT_BEAM):

- Now the element erodes when it reaches failure criteria. Previously, the forces and moments were set to zero but the element did not get deleted.
- Added new input field AXSHFL to control whether shear displacements (excluding sliding within the clearance gap) are capable of lengthening the bolt and increasing axial tension. By default (as in R10 and previous versions), shear displacements can increase axial tension. This is reasonable if the shear deformation is associated with rotation or bending of the bolt itself while the plates remain a fixed distance apart. But if the shearing is largely due to deformation of the bearing surfaces then (in real life) the bolt length does not increase and the tension is unaffected. This latter behaviour can now be invoked by setting AXSHFL to 1. This effect will be more significant as shear displacements become large. Note that displacements associated with sliding across the clearance gap are always ignored for purposes of calculating the axial load.

◦ *MAT_211 (*MAT_SPR_JLR) fixed bug - load curve IDs 8 or 9 digits long not read correctly from *MAT card in single precision version
◦ GISSMO and DIEM damage models now work with higher order solid elements.
◦ Instead of using a number or percentage of failed integration points to trigger erosion of higher order solids in *MAT_ADD_EROSION, use volume fraction of failed material. The reason for this approach is that the volume associated with each integration point varies widely within the higher order solids.
◦ Fix a bogus error message for *MAT_ADD_PORE_AIR that occurs when PERMX is defined as "0", while PERMY and/or PERMZ are not zero
◦ *MODULE feature for user-defined materials is built in with "sharelib" binaries.
◦ *MAT_153/*MAT_DAMAGE_3: Extended to up to 10 backstresses that can be determined
◦ from stress-strain data for solid/shell elements. This extension also supports implicit dynamics.

- Added support for cohesive shells in *MAT_240.
- Fix the convergence issue with plasticity algorithm in *MAT_260A/*MAT_-STOUGHTON_NON_ASSOCIATED_FLOW.
- Fix incorrect stress initialization when using *MAT_005/*MAT_-SOIL_-AND_FOAM with vol. strain vs pressure defined using load curve LCID, together with *LOAD_DENSITY_DEPTH.
- Fix seg fault or incorrect stresses when initializing stresses using *INITIAL_-STRESS_SOLID for *MAT_107/*MAT_MODIFIED_JOHN-SON_COOK.
- Fix strain rate effects on *MAT_157/*MAT_ANISOTROPIC_ELASTIC_-PLASTIC for implicit static analysis.
- Fix seg fault when using *MAT_157/*MAT_ANISOTROPIC_ELASTIC_-PLASTIC and MAT_ANISOTROPIC_ELASTIC_PLASTIC for 2D analysis.
- Fix incorrect results when using *MAT_TABULATED_JOHN-SON_-COOK/*MAT_224 with table LCKT defined and the first abscissa value set to a negative temperature.
- Increase robustness of *MAT_BARLAT_ANISOTROPIC_PLAS-TICI-TY/*MAT_033 for solids.
- Fix input error when using *MAT_ELASTIC_WITH_VISCOSITY_-CURVE/*MAT_060c when LCID = 0.
- Store computed yield strength as history variable #6 for *MAT_-PIECE-WISE_LINEAR_PLASTIC_THERMAL/*MAT_255.
- Fix to work-energy sums when *MAT_090/*MAT_ACOUSTIC material is used. If kinematics are not requested via *CONTROL_ACOUSTIC (the default), then acoustic element energies are not included in the solution sums. If kinematics are requested, then the energies are included. Previously, acoustic elements contributed to the work, but not the kinetic energy. Note - computing acoustic element kinematics is not required to calculate acoustic pressures and roughly doubles the cost of the acoustic elements. This correction has no affect upon the computed solution.
- Fixed brick material *MAT_089 with hourglass form 6. The hourglass scaling was bad which caused bad results.
- Modified shell *MAT_214 to prevent dividing by zero in some cases.
- Fixed *MAT_091 when used with shell form 16. The material directions were being calculated incorrectly causing the stress to be wrong.
- Fixed conflict between RTCL damage in *MAT_123 and heat affected zones. The TRIAX parameter was being overwritten causing possibly excessive damage.
- Node and connectivity data for elements that use *MAT_FABRIC/*MAT_-034 has been restored to the the dynain file. It had been removed along with stress and strain data which was removed because the fabric material cannot be prestressed by dynain.
- Fixed laminated shell theory with shell *MAT_172. It was calculating wrong shear strain.

- Fixed *MAT_023, *MAT_072, *MAT_083, *MAT_153, *MAT_223, *MAT_-229, and *MAT_259 when used with linear solid form 18. Some material constants were not passed correctly.
- Fixed *MAT_024 plastic strain failure for beam element forms 4 and 5. The elements were not failing.
- Fixed *MAT_077 for tetrahedral solid form 13. Some newer options were not working.
- Fixed *MAT_244 (shells and solids) when LCY1, LCY2, LCY3, LCY4, or LCY5 was left equal to zero. In this case, the code attempted to use nonexistent data, and it's unclear how this affected results. This caused the Windows executable, to error terminate with a "Program Exception - access violation" message.
- Fixed a problem where *MAT_172 was used to model 100% reinforcement and TYPEC was 3 or 6, and Young's Modulus of concrete (ET36) was input as zero. For steel only, this value should not matter, but it caused a divide by zero error during initialization. This is now prevented by having ET36 default to a small positive number for 100% reinforcement.
- Fixed shell form 17 when used with *MAT_077. There was no call to the stress update routine.
- Fixed ESORT > 0 when used with shell form 18 and *MAT_077. The triangular elements are now switched to triangular shell form 17 to be compatible with quad shell form 18.
- Fixed various errors in *MAT_NONLOCAL:

  - When both incremental and non-incremental data are requested for smoothing.
  - When used with shell materials with more than one in-plane point, or brick elements with more than 1 integration point. When coefficients are calculated for nearby points, the sum of coefficients was excluding other points within the same element or layer. The sum is used to normalize the weighted average. Since it was too small, the function that is smoothed would tend to grow.
  - Fixed an MPP only bug in the *MAT_NONLOCAL material averaging. A buffer could be overwritten if there was more than one nonlocal definition in the model. The same error was fixed for *MAT_CODAM2, *MAT_GURSON_RCDC, or *MAT_PLASTICITY_-WITH_DAMAGE with the RCDC option and the characteristic element length or nonlocal radius was defined.
  - The nonlocal search messages to screen and message files were modified to be more informative. During the search, a progress message is printed with each 100 million points added to the lists. This is intended to combat the perception that the code has hung.

- Added new options for *MAT_SPOTWELD_DAMAGE-FAILURE failure by OPT 6. There are 2 new TFLAG options:

  - TFLAG = 2 causes the max sheet thickness to be used.
  - TFLAG = 3 causes the sum of thicknesses to be used.

- A sheet thickness scale factor was added which scales the sheet thickness calculated by the TFLAG options.
- Fixed *MAT_126 when used with tshell forms 3, 5, or 7. The angle initialization was incorrect leading to bad stress values.
- Enabled *MAT_025, *MAT_173, *MAT_193, and *MAT_198 to be used with tshell forms 3, 5, and 7.
- Enabled *MAT_123 to work with tshell forms 3, 5, and 7 using options EPSTHIN and LCTSRF, and also NUMINT < 0 option. All 3 options were previously only available for thin shells and tshell forms 1, 2, and 6.
- Fixed the stress from *EOS_GASKET model when used with tshell form 2. Also fixed the compressive failure strain (CFS) for *EOS_GASKET with bricks or tshell forms 5 and 7. It was setting CFS = TFS.
- Fixed the behavior of NUMFIP on *MAT_ADD_EROSION when used with shell or tshell composite sections. When counting failed layers, the zeroing of the counter did not happen unless the first layer of the composite used *MAT_ADD_EROSION. This would have caused elements to fail that should not have failed. All 3 options, NUMFIP > 0, -100 < NUMFIP < 0, and NUMFIP < -100 are now working.
- Fixed the eigenvalue calculation in the first cycle when the model has solids or tshell forms 3, 5, or 7, and these elements are used with *MAT_022, *MAT_054, *MAT_055, *MAT_059, or *MAT_213.
- Fixed thermal strains in *MAT_021 tshells. The material directions were not processed correctly causing incorrect thermal strains. Both deformation and output were fixed and are now working for all tshell forms.
- Fixed the material direction when using shell *MAT_091. Also enabled output in the material direction when CMPFLG = 1 on *DATA-BASE_EXTENT_BINARY.
- If EPSR and EPSF are defined in *MAT_054, correct computation of transverse shears strains for solids and tshells.
- Add possibility to use failure criterion in *MAT_054 for solids in a transversal isotropic manner. It is assumed that the material 1-direction is the main axis and that the behavior in the 2-3 plane is isotropic. This feature is invoked by setting TI = 1 (card 2, column 7) in *MAT_054.
- Fix bug for shear stiffness behavior in *MAT_058 when using a table definition for GAB and only providing stress-strain-curves for positive shear.
- Fix bug for strain-rate dependent stiffness behavior in *MAT_058 when using a table definition for EA, EB or GAB under compressive loading.
- Add history variable for ellipsoidal failure surface for *MAT_059 (SC.lt.0), shells only. Now history variable 8 is the failure surface "f", which is similar to the Tsai-Wu criterion.

- *MAT_100_DA(*MAT_SPOTWELD_DAIMLERCHRYSLER),*DEFINE_-CONNECTION_PROPERTIES:Add the possibility to define a yield curve or table for DSIGY, SIGY in case of using PRUL = 1.
- For *MAT_100 (*MAT_SPOTWELD), OPT = 0/-1: Add the possibility to define force/moment resultants as a function of the effective strain rate, by defining an appropriate load curve. This will be indicated by a negative value for NRR, NRS, NRT, MRR, MSS, MTT. This functionality is implemented for beam, hex and hex-assembly spotwelds.
- Extended capability of initializing *MAT_157 solids using input variable IHIS and *INITIAL_STRESS_SOLID.
  Blocks of material parameters that can be initialized in this way are:

  - material direction (q11,...,q33) - 6 values
  - stiffnesses (c11,c12,c13,...,c66) - 21 values
  - anisotropic constants (F,G,H,L,M,N) - 6 values
  - curve/table - ID (LCSS) - 1 value
  - strengths (XC,XT,YC,YT,ZT,ZC,SXY,SYZ,SZX) - 9 values << NEW

  See User's Manual for details.
- Add value of failure criterion (Tsai-Wu or Tsai-Hill) to history variable 10 for postprocessing in *MAT_157. Shell elements only.
- Fix bug in *MAT_261 when table definition is used to define the non-linear in-plane shear behavior (LCSS). This applies to shells/tshells/solids.
- Add default strength limits (XC, ... 1.e+16) if they are not defined for *MAT_-261/*MAT_262.
- Added a criterion to avoid possible snapback behavior in *MAT_262 by only allowing certain "softening" modulus with respect to the elastic stiffness. This might be important when using rate-dependent strength limits (*DEFINE_CURVE) and the values for the fracture toughnesses are not properly set.
- *MAT_4A_MICROMEC /*MAT_215:
  Add strength limit XC for failure in fiber compression.
- *MAT_GENERALIZED_PHASE_CHANGE/*MAT_254:

  - Added latent heat algorithm to *MAT_254. Input follows the same phase transformation matrix as the other transformation related parameters.
  - Added user-defined history variables. Up to 8 history variables can be defined using *DEFINE_FUNCTION. Parameter list for functions includes time, user histories, phase distribution, temperature, peak temperature, temperature rate, stresses, and plastic strains.

- New parameter dtemp in *MAT_CWM/*MAT_270 that can invoke a subcycling in the material formulation, if the temperature jump within a time step exceeds the limit defined by dtemp.

- Activated latent heat algorithm for thermal material *MAT_THERMAL_-CWM/*MAT_T07.
- *MAT_CORUS_VEGTER/*MAT_136:
  - Renamed material from *MAT_CORUS_VEGTER to *MAT_VEGTER, as Corus no longer exists.
  - Tabular input for strain rate dependency implemented.
  - New option *MAT_VEGTER_STANDARD:
    --expects a parameter alpha_ps,theta instead of the second component of the plane strain point
    --input consistent with most literature data
    --material routine itself is unchanged
    --temperature dependent strain rate dependency also possible with the Bergstrom equation
  - New option *MAT_VEGTER_2017:
    --uses a simplified input
    --Input of tensile strengths (Rm0,Rm45,Rm90), uniform elongation (Ag0,Ag45,Ag90) and plastic anisotropy (R0,R45,R90)
    --based on a model provided by TATA steel, the standard input data for MAT_VEGTER is reproduced
    --material routine itself is unchanged
    --temperature dependent strain rate dependency also possible with the Bergstrom equation

- *MAT_REINFORCED_THERMOPLASTIC/MAT_249:
  Changed handling of post-processing data in history values. User can define the post-processing data to be written into histories.
- Add new failure criterion DTMIN (minimum time step) to *MAT_ADD_-EROSION.
- Add new failure criterion MXTMP (maximum temperature) to *MAT_-ADD_EROSION for solid and shell elements.
- Add new option LCFLD < 0 to *MAT_ADD_EROSION. In this case, |LCFLD| refers to a table where FLD curves are shell thickness dependent (in contrast, existing LCFLD > 0 provides rate dependence).
- Fix for combination of *MAT_ADD_EROSION and tshell form 2 or beams. Strain-based criteria (e.g. MXEPS) did not work correctly before.
- Several changes for *MAT_ADD_EROSION with IDAM = 1 (GISSMO):
  - If LCSDG < 0, then |LCSDG| refers to a *DEFINE_FUNCTION with arguments triaxiality and Lode parameter.
  - New option LCREGD < 0, where |LCREGD| refers to a table that contains regularization factor vs. element size curves for different triaxialities.
  - Compute element size for LCREGD not only once at t = 0 (SIZFLG = 0), but also after each adaptive refinement step.

- New option REFSZ < 0 of GISSMO provides a plastic strain value, that corresponds to that reference size, written to history variable ND+17.
- Improve robustness if DMGEXP < 1 is used.

- Fix for GISSMO to be used in adaptive remeshing. Evaluation of damage coupling flag could go wrong due to averaging of history variables in rezone step.
- Support DIEM (*MAT_ADD_EROSION with IDAM < 0) for beam form 1.
- Add new keywords *MAT_ADD_DAMAGE_GISSMO and *MAT_ADD_-DAMAGE_DIEM. The idea is to separate these damage models from *MAT_ADD_EROSION, where only pure element erosion criteria remain. That should simplify understanding the manual. Old input is still supported.
- Add new MIDFAIL options 2 and 3 to *MAT_ADD_DAMAGE_GISSMO.
- Add new keyword option _STOCHASTIC for *MAT_ADD_DAM-AGE_-GISSMO to allow spatially varying failure behavior when used together with *DEFINE_STOCHASTIC_VARIATION.
- Add new option HISVN to *MAT_ADD_DAMAGE_GISSMO: A user-defined history variable (e.g. hardness) can be used to modify the failure. LCSDG is in this case a TABLE_3D with the arguments triaxiality, Lode parameter, and that history variable.
- Add option to *MAT_ADD_GENERALIZED_DAMAGE that allows defining the number of failed integration points (to trigger element erosion) for each history variable separately. Only applicable for shells.
- Add new MIDFAIL options 2, 3, and 4 to *MAT_ADD_GENERAL-IZED_-DAMAGE.
- Add NUMINT option to *MAT_089 for shell elements.
- Add two nonlocal failure criteria to *MAT_280. The first one works similar to the ENGCRT/RADCRT criterion of *MAT_ADD_EROSION. The second is similar to the SOFT option of *MAT_054, where tensile strength is reduced in next-to-failed (crashfront) elements.
- Add new material model *MAT_BARLAT_YLD2004/*MAT_199 for solid elements in explicit analysis.
- New option ITERS < 0 in *MAT_143 invokes an alternative plasticity algorithm. It also comes with a new logarithmic rate dependence option, IRATE = 2.
- Allow initial temperatures for *MAT_224_GYS to be set via *INITIAL_-STRESS_SOLID. Also, enable *MAT_224_GYS to be used in coupled thermal-mechanical analyses.
- Modified materials *MAT_234 and *MAT_235 so that they work with angles on the *SECTION_SHELL card to define material directions for layers.
- Add option HISOUT = 1 to store principal strains as history variables 25-27 for *MAT_181 (available for solid elements).
- New options for *MAT_240/*MAT_COHESIVE_MIXED_MODE_ELASTO-PLASTIC_RATE, invoked by adding _THERMAL, _3MODES, or _THER-

MAL_3MODES to the keyword. Allows temperature dependent material data and/or inclusion of a third deformation and fracture mode.

- Add new optional hardening rule HR = 10 to *MAT_036 and *MAT_243: table with pre-strain dependence.
- Variable BETA < 0 of *MAT_224 can now refer to a *DEFINE_TABLE_3D, where the dissipation factor can be defined as a function of temperature (TABLE_3D), strain rate (TABLE), and plastic strain (CURVE).
- Add principal strains as new history variables 18-20 in *MAT_083.
- Add new option IHYPO to *MAT_023. IHYPO = 1 switches the material model formulation from hyperelastic to hypoelastic for solids, which allows stress initialization through *INITIAL_STRESS_SOLID.
- Fix for *MAT_190 when used together with *DAMPING_PART_STIFFNESS and RYLEN = 2. Premature failure due to FLD was likely to occur.
- Allow the Material Model Driver (Appendix K) to be used in batch mode: If a file called "mmd.bat" exists in the working directory, then the commands contained therein get executed sequentially. Supported commands are print, cross, time, and quit.
- Algorithmically consistent tangent modulus implemented for *MAT_024 and *MAT_123.
- Add flag for allowing nonsymmetric tangent moduli in user materials. Currently supports only solid forms -1, -2 and 2.
- Fix bug whereby encrypted *MAT_075 data was echoed in d3hsp.
- For the case where an encrypted material model referenced load curve(s), the material type was revealed by d3hsp in the load curve usage summary. That is no longer the case.
- Change the stress update in *MAT_PML_ELASTIC_FLUID to make it more stable. Viscous damping is still needed to achieve stability.
- *MAT_232/*MAT_BIOT_HYSTERETIC is now supported in all solid element formulations.
- Added *MAT_293/*MAT_COMPRF.
  This material model simulates the behavior of pre-impregnated (prepreg) composite fibers during the high temperature preforming process. In addition to providing stress and strain, it also provides warp and weft yarn directions and stretch ratios after the forming process. The major applications of the model are for materials used in light weight automobile parts.
- Added *MAT_296/*MAT_ANAND_VISCOPLASTICITY. This visco-plastic model uses a set of evolution equations instead of loading-unloading criterion to describe dislocation motion and hardening or softening behavior of materials. This model can be applied to simulate solders used in electronic packaging.
- Added 2-way option for tshell formulation 5 in *MAT_054.
- *MAT_260B:

  - Set default value for P12 = -0.5, p22 = 1.0, p33 = 3.0.
  - Set default value for G12 = -0.5, G22 = 1.0, G33 = 3.0.

        - Correct shear stress calculation error.
        - Set default value for DEPSO to 0.001, so as to avoid division by zero.

    - *MAT_260A:

        - Set default value for R00 = 1.0, R45 = r00, R90 = r00
        - Set default value for sig0, sig45, sig90, and sigb to make sure that the non-associated flow will be degenerated to associated flow.
        - Add Equation of State.
        - If SIGB is zero, then it is assumed that all the SIGS' are equal.
        - Add Xue's failure model.

    - *MAT_123:
      When major strain is used as a failure criteria, the equivalent major strain is added to the current step by using the current strain ratio and the previous deformation strain carried over from the previous (forming) stage.
    - *DEFINE_CURVE_STRESS:
      Add new options to *DEFINE_CURVE_STRESS, add ITYPE = 1,2,3,4,5,11.
    - *MAT_036:

        - Output the optimized material parameters when using both R-values and the A, C, H, P parameters as input.
        - Add an error message when the material model is attempted for tshells.
        - Turn off the output of material parameters if encryption is used.

- **MPP**

    - Suppress output of pfile information to d3hsp and messag if the *CONTROL_MPP_PFILE line itself is encrypted.
    - Fix automatic setting of "decomp { bagref }" in the pfile, which was broken in revision 101313.
    - Add MPP contact timing calls around force transducer initialization and net force calculations, so they are better represented in the contact timing table.
    - Adjusted initialization of *PART_MODE for MPP so that certain processing only happens on processor 0.
    - Correct problems with using MPP predecomposition when using jobid specifications. There were issues with initializing the file ids.
    - Enhance MPP eigensolver to have a kinder, gentler termination when no eigenmodes found.
    - For MPP, register the part in *PART_MODES to not be split across processes.
    - Allow parameter expression in keyword *CONTROL_MPP_PFILE.
    - MPP load balancing profiles are output to both .csv and .xy files.
    - pfile directive "decomp { defgeo }" in a full deck restart causes decomposition to be re-done using the current deformed geometry. This can keep

elements in contact in the same processor and helps the MPP performance of models with large deformation for example SPH bird strike, car wading, etc.

- **Output**
  - *DATABASE_ELOUT, *DATABASE_HISTORY_BEAM: Fix old typo in beam history collection routine, the effect of which is not obvious but could possibly have led to incorrect output of some beam data in elout.
  - Do a better job deleting scratch LSDA files from the file system, including removing all %XXX extensions. This may help with some customer file system issues during large adaptive problems.
  - Add rotations (moments) for nodes in the bndout file if the problem has 6 degrees of freedom.
  - Implement new force collection routines for bndout data, which should do a better job of reporting just those forces/energies that are due to the boundary conditions applied.
  - Adjust implicit logic for secforc output for arclength method.
  - Fix a problem in reporting tied contact resultant forces in implicit.
  - Enhance logic that determines when to write out the last state to d3plot for implicit.
  - Fix an MPP bug for nodal stress/strain output, which could occur when more than one part share a node where stress/strain output is requested.
  - Fix missing plastic strain tensors in d3plot when STRFLG in *DATA-BASE_-EXTENT_BINARY is set and INTSTRN = 1 in *INTERFACE_-SPRING-BACK.
  - Fixed stress output for shell forms 13, 14 and 15 when NIP = 4 on *SEC-TION_SHELL and MAXINT < 0 on *DATABASE_EXTENT_BINARY. The stress outputted was incorrect.
  - Fixed issues when reporting eroded hourglass energy to glstat and matsum. For brick elements, the eroded energy was counted twice, and for thick shells, eroded energy was not counted at all. Both issues caused energy to not balance in the glstat data.
  - Enabled secfor output for higher order shell forms 23 and 24.
  - The 'Effective Stress' option is supported in *DEFINE_MATERIAL_HIS-TO-RIES, meaning that the material dependent equivalent stress is output to the d3plot database. Currently only *MAT_036 and *MAT_133 honors this option, all other materials will output the von Mises stress for now.
  - Variable NTIED of *DATABASE_EXTENT_INTFOR can be used to output the tied status on the slave side of tied mortar contact, including tiebreak and tied weld.
  - Fixed intfor database for SMP if the file size is larger than 7M byte.
  - Echo *DATABASE_EXTENT_COMP flags in d3hsp.
  - Fixed bug for d3plot if both DECOMP = 5 or 6 in *DATABASE_-EXTENT_-BINARY, and PSETID is specified in *DATABASE_BINARY_-D3PLOT.

- Fixed bug for *DATABASE_HISTORY_DISCRETE if BEAM = 1 in *DATABASE_BINARY_D3PLOT.
- Fixed bug in writing *SECTION_*_TITLE to d3prop file.
- Fixed legend of ssstat in binout.
- Fixed wrong cross-section ID in secforc if dyna.str is used.
- Write dynain file in I10 or long format is keyword input includes those formats.
- Fixed bug affecting d3plot when analysis includes dynamic relaxation.
- Added one additional significant digit to *NODE coordinates in dynain.
- Fix bug in d3plot when both higher order and ALE elements are in the model.
- *DATABASE_HISTORY_NODE_LOCAL_ID:
  Fix bad node labels in nodout when long input format is used.
- Fixed bug where HEADING in *DATABASE_HISTORY_NODE_ID was limited to 10 characters in free (comma-delimited) format.

- **Restarts**

  - Correct implicit memory pointers to work correctly on restart.
  - Also corrected dump and restart lengths for implicit restart.
  - Properly start up explicit LaGrange Multiplier treatment of joints at restart.
  - Fix seg fault when using *DELETE_CONTACT for restart when running with SMP.
  - Fix error termination for full deck restart that includes *DEFINE_ELEMENT_DEATH.
  - Fix input error during structured input when using *INITIAL_VELOCITY_GENERATION and *CHANGE_VELOCITY_GEN-ERATION together in a full deck restart.
  - Fix incorrect full deck restart analysis if initial run was implicit and the full deck restart run is explicit. This affects MPP only.
  - *CHANGE_CURVE_DEFINITION in a restart was not working properly to modify curve LCDT in *DATABASE_BINARY_D3PLOT.
  - Fix corrupt d3part database affect a small restart with *DELETE_PART and *DELETE_FSI.
  - Fix bug in full deck restart of tied contact that resulted in force discontinuities across the restart.

- ***SENSOR**

  - Fix a bug for *SENSOR_DEFINE_FORCE when FTYPE = JOINT, which occurs when joint id > 9999999.
  - Fix a bug for *SENSOR_CONTROL for type = SPC, that was introduced in r115453 & r115457.
  - The bug occurs when a SPC boundary condition was initially off and then turned on later.

- Fix a bug for *SENSOR_DEFINE_..._SET that can occur if sensor command is defined before the definition of the related *SET commands.
- Add MTYPE = BNDOUT to *SENSOR_DEFINE_MISC to trace energy reported in bndout.
- Add MTYPE = MATSUM to *SENSOR_DEFINE_MISC to trace energy reported in matsum.
- Fix a sensor bug that occurs when *SENSOR_DEFINE_FORCE is used to trace the force associated with prescribed motion.
- Fix a sensor bug that occurs when *SENSOR_DEFINE_MISC has MTYPE = CURVE that refers to a load curve of *DEFINE_CURVE_FUNCTION. The bug was introduced in r115362.
- Couple thermal-only analysis with general sensor so that sensor can be used to terminate the analysis.
- Add TYPE = ELESET to *SENSOR_CONTROL to erode elements.
- Add MTYPE = NFAILE to *SENSOR_DEFINE_MISC to trace number of failed elements.
- Fix a bug affecting *SENSOR_DEFINE_FORCE with FTYPE = JOINTSTIF.
- Fix bug in spotweld-constraint handling in MPP when TYPE = SPOTWELD in *SENSOR_CONTROL.
- Fix a bug in *SENSOR_CONTROL when TYPE = SPC. The bug occurs when a node is involved in more than one SPC definition.
- Add MTYPE = CURVE for *SENSOR_DEFINE_MISC so that sensor can trace the value of a time-dependent curve or a *DEFINE_CURVE_-FUNCTION.

- **SPG (Smooth Particle Galerkin)**

  - Added *CONSTRAINED_IMMERSED_IN_SPG for composite analysis.
    This keyword applies to immersion of beam or shell elements in SPG solids. This is a new feature of SPG method for failure analysis of some particular composites such as rebar in concrete and fiber-reinforced composites.
  - Added *CONTACT_SPG for self-contact in SPG method.
    This feature is useful in studying some self-contact for high velocity impact/penetration applications where failed particles may still interact in compression modes.
  - Optimize the SMP performance of SPG solid formulation 47.
  - *CONSTRAINED_IMMERSED_IN_SPG now works in MPP.
  - *MAT_181 is now supported for SPG solid formulation 47.

- **SPH (Smooth Particle Hydrodynamics)**

  - *CONTACT_2D_NODE_TO_SOLID:

- Contact was not robust when using master surface with sharp angles or thin structure, like a needle. It has been revised to handle better this kind of geometry, and take into account the thickness of slave nodes.
- Add a maximum parametric coordinate parameter MAXPAR for segment search (default = 1.05).
- Change format when transferring values from keyword format to structured format and pfile format. This change greatly improves precision.

- *CONTROL_SPH:
  Add IEROD = 3 to enforce zero deviatoric stress to eroded SPH particles, and conserve volumetric response if an EOS is defined.
- *DEFINE_SPH_To_SPH_COUPLING:
- Allow negative values of SRAD. In that case, we compute a contact distance based on volume instead of smoothing length.
- Fix bug for SPH parts using *MAT_147 (*MAT_FHWA_SOIL). History variables were improperly initialized, rendering the whole damage evolution aspect of the material inoperative with SPH.
- Implement enhancement for fluid formulations in SPH so that interaction between multiple SPH fluid parts is more robust.
- Fix bug affecting *DEFINE_BOX_SPH. This feature was broken.
- Fix bug for *DEFINE_ADAPTIVE SOLID_TO_SPH. History variables were not transmitted to the SPH particles properly for some materials.
- Add option to automatically compute the contact thickness of slave SPH particles using ITHK in *CONTROL_SPH. The thickness calculated by ITHK = 1 is used only if SST (*CONTACT_-AUTOMATIC_NODES_TO_-SURFACE) or OFFD (*CONTACT_2D_NODE_-TO_SOLID) are set to zero. All default behaviors remain unchanged.
- Enable SPH in full deck restart.
- SPH part is now identified as element type 4 in d3hsp like this, "element type ..... = 4"
- *ELEMENT_SPH_VOLUME specifies volume instead of mass for SPH particle.
- Added in adaptive SPH formulation (ASPH) with anisotropic smoothing tensor (FORM = 9, SMP only) and renormalization approximation for adaptive SPH formulation with anisotropic smoothing tensor (FORM = 10, SMP only). The axes of those forms evolve automatically to follow the mean particles spacing as it varies in time, space and direction based on the strain rate tensors. These forms have better accuracy and stability than the standard SPH formulation.
- Added a new function ISHOW = 1 in *CONTROL_SPH whereby SPH particles generated by *DEFINE_ADAPTIVE_SOLID_TO_SPH will be shown as points instead of spheres before activating them, i.e., before erosion of parent solid.

- Added in a new variable ISPHKERN in *SECTION_SPH for higher order kernel option: with ISPHKERN = 1, a quintic spline kernel function (smoother and more accurate) will be used instead (supported in FORM = 0,1,4,9,10 in *CONTROL_SPH for both SMP and MPP version of executables).
- Added binout support for SMP rcforc file with *CONTACT_2D_NODE_-TO_SOLID. Write user id (instead of internal ID) in rcforc for *CONTACT_-2D_NODE_TO_SOLID. Fixed rcforc output in the case of multiple *CON-TACT_2D_NODE_TO_SOLID contacts (bug 13454).
- Added in _SPH_VARIATION option (SMP only) for *DEFINE_STOCHAS-TIC_ELEMENT with SPH particles.
- Supported *DEFINE_STOCHASTIC_VARIATION option for SPH particles (combined with *MAT....._STOCHASTIC option) with material models *MAT_010, *MAT_015, and *MAT_024.
- Support conventional mass scaling (negative DT2MS in *CONTROL_-TIMESTEP) for SPH.
  This feature works for both SPH 3D and 2D cases.

- **Thermal Solver**

  - *CONTROL_EXPLICIT_THERMAL_... and *CONTROL_ADAPTIVE (ADPTYP = 8, 1st line, 3rd column): For the explicit thermal solver, map the temperatures after each adaptive restart for ADPTYP = 7,8.
  - *CONTROL_EXPLICIT_THERMAL_...: Implement the explicit thermal solver in ALE 2D.
  - *CONTROL_EXPLICIT_THERMAL_CONTACT:
  - If part id < 0 in the *SET_PART called by *CONTROL_-EXPLICIT_THER-MAL_CONTACT, the autocontact is activated for that part. Otherwise, by default, the thermal contact is only searched between different parts.
  - Added death time TDEATH and birth time TBIRTH to *BOUNDARY_TEM-PERATURE_NODE/SET in order to activate and deactivate temperature constraints.
  - Add iterative solution option for the thermal radiation boundary condition linear equation solver. Correct logic used in determining the acceptable thermal solver options for SMP.
  - Fixed bug where dynain was missing thermal history variables.
  - New keyword *BOUNDARY_TEMPERATURE_RSW:

    - Prescribe nodal temperatures within a (possibly moving) ellipsoidal region of the structure.
    - Temperatures for the center and the boundary of the ellipsoid have to be input. In between there is a quadratic approximation.
    - Outside of the ellipsoid, no temperature values are prescribed.
    - Position and axis of symmetry are defined by to nodes.
    - Applicable to solid and thermal thick shell elements.

- Modification of variable time stepping in thermal solver:
  - Ensure that the step size of the last time step before a breakpoint is not smaller than half the previous step. If necessary, the last two time step sizes are averaged. The breakpoints are still hit exactly. This avoids drastic step size reductions (sometimes by some orders of magnitude) that slow down the further simulation.
  - Accept load curve input for dtmin, dtmax and dtemp in *CONTROL_-THERMAL_TIMESTEP. As usual if a negative integer number is given its absolute value refers to the load curve ID.

- *LOAD_HEAT_CONTROLLER now implemented for MPP.
- *BOUNDARY_THERMAL_WELD_TRAJECTORY:
  - Additional option for heat source definition. With IFORM.eq.5, the energy rate distribution does not have a pre-defined form, but can be given as a function of the local coordinates r,s,t, the time and the current weld velocity. The formula is input with a *DEFINE_-FUNCTION keyword.
  - Fixed d3hsp output. Now external IDs of used load curves instead of internal ids are written and cross reference for the curves' usage is output

- Thermal edge contact:
  - Models heat transfer from and to a shell edge onto a surface (solid facet or shell).
  - Shell edges have to belong to thermal thick shells (THSHEL = 1 in *CONTROL_SHELL).
  - Shell edges are on the slave side.
  - Activated if parameter ALGO is larger than 1 (2 = two_way, 3 = one_way).
  - Available in both MPP and SMP.

- Composite thick shells:
  - Added composite thick shell functionality to the structure heat transfer solver.
  - Up to this point the lay-up defined by *PART_COMPOSITE_TSHELL has not been taken into account by the thermal solver. Now, additional degrees of freedom are generated and the element is split into virtual elements in the element routine. The implementation is very similar to what has been done for thin composite shells defined by *PART_-COMPOSITE.

- Fixed error in reading *SECTION_SHELL_THERMAL

- *DEFINE_CURVE_FUNCTION is now supported in the thermal solver.

- **XFEM (eXtended Finite Element Method)**

  - Added erosion option for XFEM shells.

- **Miscellaneous**

  - *DAMPING_FREQUENCY_RANGE_DEFORM can now be applied to tshells.
    In previous versions it worked only for solids, beams, shells, and discrete elements.
  - Fixed bug in *DAMPING_FREQUENCY_RANGE_DEFORM.
  - The damping's contribution to internal energy was wrongly calculated. This did not affect the solution (stresses, displacements, etc), only the output values of internal energy.
  - Update to external case driver to support LSDA based includes in the keyword input.
  - Fix incorrect handling of symmetric load curves when checking discretization errors, which resulted in incorrect and misleading error messages.
  - Fix initialization problem that could arise if more than one *INTERFACE_-LINKING instance references the same *INTERFACE_-COMPONENT.
  - Fix some memory allocation and initialization related to *INTER-FACE_-LINKING.
  - Improve reporting of seatbelt input errors in nastran reader.
  - Added *DEFINE_DRIFT_REMOVE to provide a correction to curves to compensate for errors in accelerometers.
  - Fix a bug that occurs when *INCLUDE_TRANSFORM is used together with *DEFINE_BOX and/or *PART_INERTIA.
  - Fix a bug that generates duplicate parts even when *PART_DUPLICATE is never defined.
  - Automatically merge DEFORMABLE_TO_RIGID_AUTOMATIC cards with the same options for better performance.
  - *DEFINE_PRESSURE_TUBE: Added 2nd keyword input line for viscosity, step size, and damping. Added support for automatically generated shell/solid tubes.
  - Fix wrong pointer used for section id/properties when generating part for visualization of rigid wall during adaptivity. This caused error message, KEY+137, during adaptivity.
  - Fix input error for duplicate part if multiple *RIGIDWALL_-GEOMET-RIC_..._DISPLAY keywords are used, some with part id PID specified and some not.
  - Fix non-effective OPTIONs DBOX, DVOL, DSOLID, DSHELL, DTSHELL, DSEG for for deleting segments in *SET_SEGMENT_GENERAL.

- Add failure function terms (normal, bending, shear) as arguments of functions in *DEFINE_CONNECTION_PROPERTIES with PRUL.ge.2
- Add new option DGTYP = 5 to *DEFINE_CONNECTION_PROPERTIES.
- Fix for *DEFINE_CONNECTION_PROPERTIES with PRUL.ge.2 (*DEFINE_FUNCTION). The function IDs were not working with *INCLUDE_TRANSFORM and large IDs > 2**24 also failed.
- Add warning message for *DEFINE_FUNCTION: if the function name starts with i, j, k, l, m, n, I, J, K, L, M, or N, it will return an integer value.
- Removed the echo of each *CONTACT data and load curve data in d3hsp if that data are encrypted.
- Fixed bug in reading *CONTROL_REQUIRE_REVISION if free format is used.
- Fix a number of issues related to long format input:

  - Corrected legend that is written in nodout if long format is used.
  - Fixed bug in reading long format if *KEYWORD long = yes is used in include file.
  - Fixed bug in reading long format input for *INCLUDE_STAMPED_PART if optional 4 or 5 card doesn't exist
  - Fixed bug for long format bug for the following keywords:
    *MAT_EMMI
    *PERTURBATION_NODE
    *DEFINE_TABLE_MATRIX
    *INTERFACE_SPRINGBACK_LSDYNA
    *EOS_013
    *ALE_FSI_TO_LOAD_NODE
    *INITIAL_VOLUME_FRACTION
    *CONTROL_ADAPTIVE (3d)
    *CONTACT_AUTOMATIC_GENERAL_INTERIOR_MPP
    *CONTACT_ERODING_SINGLE_SURFACE_MPP
    *PERTURBATION_NODE
    *CONSTRAINED_GENERALIZED_WELD_FILLET
    *AIRBAG_HYBRID_ID
    *MAT_ADD_EROSION
    *FREQUENCY_DOMAIN_FRF
    *PARAMETER
    *DEFINE_HEX_SPOTWELD_ASSEMBLY
    *DEFINE_SPH_TO_SPH_COUPLING
    *ELEMENT_BEARING
    *SPH_COUPLING
    *ALE_2D
    *ELEMENT_BEAM_PIPE
    *BOUNDARY_PRESCRIBED_FINAL_GEOMETRY
    *PARAMETER_EXPRESSION
    *NODE_MERGE

*NODE_MERGE_SET
*ELEMENT_NURB_SOLID_PATH
*INITIAL_STRAIN_*
*BOUNDARY_THERMAL_BULKNODE
*CONSTRAINED_SHELL_TO_SOLID
*MAT_RIGID if geometry contact entity is used
*MAT_VISCOELASTIC_HILL_FOAM
*MAT_002

- Fixed bug in reading multiple entries of any of the following commands:
  *CONTROL_REQUIRE_REVISION
  *SECTION_SHELL_EFG
- Fixed bug for *INCLUDE_TRANSFORM in the case where the material ID is in alpha (non-numeric).
- Fixed bug in reading *SECTION_SHELL and *SECTION_SOLID if multiple sections are entered under one keyword.
- Correct the application of reaction forces from *INTERFACE_SSI_AUX_EMBEDDED in *INTERFACE_SSI (MPP only).
- Fix bug in running SSI problems in single precision.
- 3D adaptivity now partially supports *INCLUDE_TRANSFORM: a transformation of model geometry can now be specified using TRANID.
- *PART_MOVE:
  Extend to support *SET_PART_COLLECT and *SET_PART_ADD.
- Enable support of parametric filename in *INCLUDE when adaptivity is used.
- *INTERFACE_SPRINGBACK_LSDYNA, and *INCLUDE_BINARY:
  Fix missing SPCs output in binary dynain files in both SMP and MPP

Capabilities added to create LS-DYNA R12:

See release notes (published separately) for further details.

- ***AIRBAG**

  - Fix a bug for the output of area and leakage information for all parts constituting a control volume airbag. The bug occurred when an airbag was comprised of more than 10 parts.
  - *AIRBAG_HYBRID_CHEMKIN: fix an MPP bug introduced in r117881 that results in incorrect airbag pressure.
  - Remove the interaction between *AIRBAG_REFERENCE_GEOMETRY and *INITIAL_FOAM_REFERENCE_GEOMETRY so that they can only be used to define the reference geometry of shell and solid elements respectively.
  - *AIRBAG_PARTICLE (CPM):

        - New keyword *DEFINE_CPM_NPDATA to support more part-specific input for *AIRBAG_PARTICLE. Invoked by NPDATA > 0 and STYPEH = 2 or 3. Among other things, this new feature allows user to control smoothing algorithm for applying particle to fabric impulse.
        - Add new limit checking and self adjusting algorithm for 4th order polynomial nonmonotonic function of nonlinear CP to avoid incorrect result.
        - Support inflator mass flowrate curve (LCTi) using *DEFINE_CURVE, *DEFINE_CURVE_FUNCTION and *DEFINE_FUNCTION.
        - New feature for *DEFINE_CPM_VENT for pushout vent to allow user to apply the ambient pressure when internal parts extend out from the external vent.
        - Support C23 (discharge coefficient) as function of vent area.
        - Support Autoliv porous leakage model (FVOPT = -1,-2) under CPM and CPM+UP switch capabilities.
        - Add tire inflation capability under CPM method to maintain the target tire pressure during the initial setup.
        - Support inflator volume evaluated from current geometry while using user-defined inflator chamber.
        - Fix bug for airbag with solid parts inside. The volume from those parts are excluded from bag volume.

    - New keyword *CONTROL_AIRBAG for CV (Control Volume) closed volume check.
    - Support multiple airbags when not all airbags have a reference geometry (*AIRBAG_REREFERENCE_GEOMETRY).

- ***ALE**

    - *INITIAL_ALE_MAPPING: Add a parameter SYM to apply specific mapping rules to elements and nodes outside the mesh of the previous run that wrote the mapping file to be used in the current run.
    - *ALE_MAPPING: Add new keyword to map data during a run (not just initially like with *INITIAL_ALE_MAPPING). A particular state from the mapping file is read.
    - *INITIAL_VOLUME_FRACTION_GEOMETRY and CNTTYP = 7: Allow the user to define the geometry using *DEFINE_FUNCTION.
    - *INITIAL_DETONATION: If PID < -1, |PID| is the ID of a *SET_PART.
    - *INITIAL_HYDROSTATIC_ALE, *ALE_AMBIENT_HYDROSTATIC: Account for the compaction to compute the initial and ambient hydrostatic pressure for *EOS_MIE_GRUNEISEN.
    - *ALE_PRESCRIBED_MOTION and *BOUNDARY_AMBIENT: Add a new parameter SIDR in both keywords to control their use during the dynamic relaxation phases (similar to SIDR in *DEFINE_CURVE).

- *BOUNDARY_AMBIENT: Set birth and death times with the first and last abscissa of *DEFINE_CURVE for the internal energy and relative volume time curves (LCID1 and LCID2).
- *CONTROL_ALE: Add a new variable BNDFLX to select only the ALE groups (*SET_MULTI-MATERIAL_GROUP_LIST) that can flux in when these groups are in the ALE elements along the mesh boundaries. Set BNDFLX = -1 to forbid any influx along free mesh boundaries.
- *ALE_BURN_SWITCH_MMG: Implement this keyword to allow the user to implement his own burn models.
- *ALE_MESH_INTERFACE: Implement this new keyword to mesh material interfaces with triangular shells (the material volume can also be meshed with tetrahedra).
- *CONSTRAINED_LAGRANGE_IN_SOLID: Implement a 2D version of the implicit thermal ALE coupling.
- *CONSTRAINED_LAGRANGE_IN_SOLID_EDGE with CTYPE = 5: Renew the coupling interface along the outside shell edges involved in the ALE coupling after shell erosion.
- *CONSTRAINED_LAGRANGE_IN_SOLID with 4 =< CTYPE <= 6 : Write in and read back from d3full the FSI relative displacements and other coupling arrays.
- *CONTROL_SEGMENTS_IN_ALE_COUPLING: New keyword that deactivates segments in the ALE penalty coupling (CTYPE = 4,5,6 in *CONSTRAINED_LAGRANGE_IN_SOLID) if the segments are face to face. If variable SYM = 1, a segment normally constrained is excluded from the coupling (similar to ISYM in *CONTROL_CONTACT).
- *DATABASE_BINARY_FSIFOR and *CONTROL_MPP_DECOMPOSITION_NUMPROC: Prevent fsifor from being corrupted by a MPP job starting right after MPP decomposition.
- *DATABASE_BINARY_FSIFOR and *SECTION_ALE2D: Output the interface force file (fsifor) in 2D (meshfringe in LS-PrePost should be used to see the coupling force and pressure distributions along the segment edges).
- *DATABASE_FSI: Output the center of pressure at fx-lc,fy-lc,fz-lc (or gx,gy,gz) in dbfsi if a node set (NDSETID) is provided in *DATABASE_FSI.
- *SECTION_ALE1D and *DATABASE_EXTENT_BINARY BEAMIP > 0: Access auxiliary and history variables in d3plot for ALE 1D elements (like NEIPH for solids and NEIPS for shells).
- *DATABASE_TRACER and TRACK = 2: The tracer moves with the ALE mesh.
- *DATABASE_TRACER and ALE mapping: Output the tracer final locations in a keyword format (to be included in the next run initialized by a mapping).
- A new ellipsoid geometry option is added in *INITIAL_VOLUME_FRACTION_GEOMETRY.
- *INITIAL_HYDROSTATIC_ALE now supports ALE single material with void formulation, that is, ELFORM 12 in *SECTION_SOLID.

- Structured ALE (S-ALE):
  - Support *EOS_MURNAGHAN to model weakly incompressible water.
  - *ALE_STRUCTURED_MESH_VOLUME_FILLING implementeded to fill ALE fluids into the initial S-ALE mesh and sidestep *INITIAL_VOLUME_FRACTION_GEOMETRY.
  - *ALE_STRUCTURED_FSI implemented to perform ALE fluid-structure interaction with structured ALE mesh. It could effectively stop the leakage often observed when using *CONSTRAINED_LAGRANGE_IN_SOLID.

- ***BATTERY (Electrochemistry Solver)**

  (See also "Battery module Release" in the "EM (Electromagnetic Solver)" section.)

  The *BATTERY family of keywords invokes the new electrochemistry solver, which is only available in double precision executables. *BATTERY keywords are documented in Volume III of the LS-DYNA Keyword User's Manual.

  The keywords starting with *BATTERY refer to and control the problem set up for detailed one-dimensional electrochemistry modeling of battery cells. This is intended to be used for battery-thermal-structure-interaction problems.

  Two Lithium Ion Battery (LIB) models are available via two one-dimensional full cell models.

  - Single insertion which has this structure: Anode Lithium metal electrode/Separator/Cathode composite electrode
  - Dual insertion which has this structure: Anode composite electrode/Separator/Cathode Composite electrode

  These two cell models are based on an electrochemical model with various anode, separator, and cathode material properties. Therefore, the user can select a material property for their anode and cathode including an open-circuit potential (OCP). For the separator, the user can also select a material property among the different electrolytes which include all of the transport properties necessitated in battery simulation. Note that all transport properties are uniquely designed for each battery type.

  The selection of the single insertion or dual insertion model is done in keyword *BATTERY_ECHEM_CONTROL_SOLVER. Here, users can select the

battery run mode such as Galvanostatic or Potentiostatic, the number of cycles, and termination controls. Please see Vol III of the keyword manual for more details.

With the *BATTERY_ECHEM_CELL_GEOMETRY keyword, users can choose the length of each layer in a single cell including both sides of the current collectors, and the corresponding total number of mesh elements can also be input.

After this is done, then users need to set up their own values of all variables in the material cards for all layers: anode, cathode, and separator. The three main keywords are,

*BATTERY_ECHEM_MAT_ANODE

*BATTERY_ECHEM_MAT_CATHODE

*BATTERY_ECHEM_MAT_ELECTROLYTE

Here, first the OCP ID must be carefully selected (please refer to the keyword manual) and then Coulomb capacity, initial state of charge, thermodynamics data, and porous media data, respectively.

In the *BATTERY_ECHEM_THERMAL keyword card, the user can set the initial temperature and other thermal properties. However, please note that this card is ignored when the battery solver is coupled to the thermal and thermal-mechanical solvers.

Please note that the purpose of these 1D solvers is give the user a tool to test their OCP, material properties, and transport battery properties. For coupling the electrochemical battery solver to the thermal-mechanical solver, users must design their own mesh system with a different part number and must assign the individual part numbers for each solver. For example, if the electrochemical battery solver needs to solve within a specific part (all battery regions), then the user must assign that part number in the first variable column in each of these material keywords: *BATTERY_ECHEM_MAT_-ANODE, *BATTERY_ECHEM_MAT_CATHODE, and *BATTERY_-ECHEM_MAT_ELECTROLYTE. The same part number should be assigned for use by the structural thermal and mechanical solvers. Structural parts outside of the battery would have other part numbers. Please note that the 1D electrochemistry solver solves all elements assigned to the part numbers for the battery, and also note that this battery solver has its own 1D mesh system in each battery structural element, while the thermal-mechanical solver has its own 3D mesh system.

Currently however, only the dual insertion model is coupled with the LS-DYNA thermal-mechanical solvers. This model covers all Lithium-Ion batteries used by the battery manufacturers of cellular phones, and automotive industries batteries.

So, in version R12.0.0, users can simulate how the Lithium-Ion battery responds thermally and/or mechanically when external heat sources or impacting forces are applied to the battery pack. The battery model can be a single battery cell, a cell stack which has multiple cells connected (over 100 cells), and even multiple cell stacks connected in parallel. By controlling the solution scheme of the structural solvers, users can calculate the structural effects for a certain period of time of battery operation such as initial, middle, and end of state of battery discharging or charging.

For more detail about electrochemical battery theory, please refer to the LS-DYNA multiphysics theory manual. For keyword setup, please refer to the multiphysics (Vol III) keyword manual. In addition, for interested users, please request some sample keyword decks from LS-DYNA technical support, or email directly to kyoungsu.im@ansys.com.

- ***BOUNDARY**
  - Fix bug in *BOUNDARY_PRESCRIBED_MOTION which could cause velocity boundary conditions to be incorrectly handled for dof=+/- 4 or 8 if the node has more than one velocity boundary condition, e.g., one during dynamic relaxation and another during transient analysis.
  - Corrected MPP communication error associated with Implicit loading constraints for *BOUNDARY_PRESCRIBED_MOTION_FINAL_GEOMETRY.
  - For convection, flux and radiation boundary conditions, the parameter PSEROD on *BOUNDARY_... specifies a part set for which new segments exposed to the environment due to solid element erosion will inherit the boundary condition data.
  - New option BNDOUT2DYNAIN on *BOUNDARY_PRESCRIBED_MOTION_RIGID, allows for output of reaction force to dynain for use in subsequent simulations.
  - SPC2BND on *CONTROL_OUTPUT which will convert constraints on *MAT_RIGID to prescribed motions, for access to reaction forces in bndout.
  - Fix *BOUNDARY_NON_REFLECTING to have the correct velocity averaging and force redistribution for triangular segments.
  - *CONTROL_ADAPTIVE: Boundary conditions *BOUNDARY_RADIATION_SET and *BOUNDARY_CONVECTION_SET are updated in accordance with mesh changes due to 3D tet adaptivity under the condition that the segment set(s) are defined using *SET_SEGMENT_GENERAL with OPTION = PART.

- ***BLAST**
    - Fix blast wind velocity field for *LOAD_BLAST_ENHANCED BLAST = 4. Previously velocity was always the ground-reflected wave. Now, if a segment is not in the Mach stem region, the blast wind comes from both the incident-only and ground-reflected waves.
    - Fix a bug which could potentially affect results for models which contained both *LOAD_BLAST and *LOAD_BLAST_ENHANCED with TBO.ne.0. The time offsets could get mixed up or ignored.

- ***CESE** (Compressible Fluid Solver)
    - For the solvers not performing chemistry calculations, a switch has been made to a positivity-preserving method. For most problems, this will lead to a smaller time step, depending upon the shape of the smallest mesh element. Note that this method guarantees that density and internal energy remain positive.
    - Several bug fixes were made that make these solvers more stable.
    - For the CESE moving mesh conjugate heat transfer solver, fixed several bugs, including some MPP bugs that depended upon the mesh decomposition.

- ***CHEMISTRY**
    - The inflator model can be extended up to a 5 chamber model. Previously, it was limited to only a 3 chamber model which consists of the combustion chamber, gas chamber, and tank. In this version, we added two more gas plenum chambers to control both the speed of the gas and pressure into the tank.
    - Resolved issues with back flows when the pressure of the downstream gas chamber is higher than the upper chamber by correcting the pressure equation.
    - Updated the procedure for computing/loading an initial blast profile as an initial condition of a detonation or deflagration.
    - Fixed some problems in the axisymmetric combustion solver.
    - Chemically-reacting FSI flow problems using the Euler equation solver now work for any problem up to 60 species in the combustion gas. FSI with this Euler solver is strong and stabilized.

- ***DUALCESE (Dual CESE Compressible Fluid Solver)**
    - The *DUALCESE family of keywords invokes the new dual Conservation Element/Solution Element (dual CESE) compressible fluid solver. This solver is only available in double precision executables.
    - The *UNIT family of keywords is introduced which provides a coherent way to specify units in an LS-DYNA problem. For now, these keywords only

work with the dual CESE solver but may be extended to other solvers in the future.

- Dual CESE Solver Characteristics:
  - Explicit
  - Double precision
  - Dynamic memory handling
  - SMP and MPP
  - 3D solver / special case 2D solver and 2D axisymmetric solver. Note: 2D axisymmetric solver is only done for the CFD solver, not yet for FSI-ibm or FSI-mmm solver cases.
  - Automatic coupling with the LS-DYNA structural solver
  - Eulerian fixed mesh or moving mesh (Either type input with *DUALCESE_ELE2D or *DUALCESE_ELE3D cards, or via *MESH cards for a tetrahedral mesh)

- Dual CESE Solver Main Features:
  - CFD solver is based upon the dual CESE (Conservation Element / Solution Element) method that is a new version of CESE technology with enhanced accuracy and robustness
  - Highly accurate shock wave capturing
  - Embedded (immersed) boundary approach or moving mesh (fitted) approach for FSI problems
  - Mesh can be broken up into regions/parts with a different solver per region, with the intent to minimize the region where a moving mesh FSI technique is used, or where an immersed boundary FSI method is used. This capability is intended to help optimize solver performance.
  - Complex fluid equations of state (EOS) are now available through the REFPROP and COOLPROP EOS libraries. This is only supported with *DUALCESE models.
  - Bi-cubic table look-up systems are now available for the REFPROP and COOLPROP libraries to dramatically speed up the evaluation of thermodynamic quantities.

- Dual CESE Solver Applications (Non-exhaustive):
  - FSI problems
  - Shock wave capturing
  - Shock/acoustic wave interaction

- Planned future features of Dual CESE Solver:
  - FSI with material erosion (as done in the *CESE immersed boundary FSI solver)
  - Coupled stochastic fuel spray solver (See *STOCHASTIC keywords)

- Coupling with chemistry (See *CHEMISTRY keywords) solver
- Multi-species transport
- Conjugate heat transfer
- Cavitation model

- ***CONTACT**

  - Fix old issue for *CONTACT_ENTITY type 9 to properly handle rotations when the entity is not centered on (0,0,0).
  - Fix for spotwelds improperly deleted due to rigid body conflict when an IPBACK *CONTACT interface is in effect. The inability of the constraint-based contact interface to tie should not cause the weld to be deleted if the penalty side ties. (MPP only.)
  - Fix SMP *CONTACT_2D_NODE_TO_SOLID. Friction was being ignored.
  - Added a warning that *CONTACT_TIEBREAK_NODES_TO_SURFACE_-ID is not supported for implicit computations.
  - *CONTACT_2D_..._THERMAL is now available in MPP.
  - Fix an SMP bug for ERODING contact of solid elements, which could result in erroneous contact thickness.
  - Fix seg fault or incorrect frictional behavior when using _THERMAL_FRIC-TION option for *CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_-SURFACE and *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE (SMP only).
  - Fix instability when using *CONTACT_TIEBREAK_NODES_TO_SUR-FACE (SMP only).
  - Fix incorrect *CONSTRAINED_TIE-BREAK behavior when the master node is the last node in the input after sorting.
  - Fix failure to detect contact between beam and shell edge when using *CON-TACT_AUTOMATIC_BEAMS_TO_SURFACE and the beam diameter is large compared to the segment size (SMP only).
  - Fix *CONTACT_AUTOMATIC_GENERAL for spot weld beams when us-ing SSID = 0, i.e. all parts included in the contact, and CPARM8 = 2.
  - Implement unloading curve, UNLCID, for options FCM = 2/3 in *CON-TACT_RIGID_(OPTION).
  - FTORQ = 1 and 2 (Opt. Card E in *CONTACT), affecting transmittal of mo-ments, are now implemented in SMP for *CONTACT_AUTOMATIC_SUR-FACE_TO_SURFACE_TIEBREAK and *CONTACT_AUTOMATIC_ONE_-WAY_SURFACE_TO_SURFACE_TIEBREAK.
  - FTORQ = 2 is now implemented in SMP for *CONTACT_AUTOMATIC_-SURFACE_TO_SURFACE and *CONTACT_ONE_WAY_SURFACE_TO_-SURFACE.
  - Added contact energy density to the intfor database for segment-based (SOFT = 2) contact. This option is invoked by NENG = 1 on *DATABASE_-EXTENT_INTFOR.

- Added support for ORTHO_FRICTION option for segment-based (SOFT = 2) contact. This option is available for all versions, SMP, Hybrid, and MPP with and without the groupable option, and for all supported CONTACT keywords.
- Added options to limit the scope the DPRFAC option of segment-based (SOFT = 2) contact. When DPRFAC is set to a value greater than or equal to 1.0, then DPRFAC is the ID of either a shell set, a segment set, or a part set. When both segments are in the set, then the DPRFAC option is active for that segment pair. If either segment is not in then set, the DPRFAC is inactive. When searching for the set, the search will proceed in the order of shell set, then segment set, and finally part set. The first set that is found with a matching ID will be used. The set attribute DA1 will be read and used as the DPRFAC value.
- Added support for VSF parameter on Card 3 of *CONTACT when using SOFT = 2 contact.
- Added support for FTORQ = 2 (*CONTACT, Card E) for SMP and MPP segment-based (SOFT = 2) contact. Setting FTORQ = 2 adds torque to master nodes to counteract the moment that is created by frictional contact forces due to the contact surface being offset from the mid-plane of the element. This added torque reduces the net moment from contact to zero.
- Added keyword *DEFINE_FRICTION_SCALING which allows shell segments to have friction coefficients on the inner and outer face (SOFT = 2 contact only). Independent scale factors on the inner and outer face scale the nominal friction coefficient. This was motivated by airbags which have liners on the inside of the bag thus creating a smoother (slicker) surface on the inside as compared to the outside.
- Added a new variable EDGEK on Card C of *CONTACT which scales edge-to-edge contact stiffness for SOFT = 2 contact having DEPTH = 5, 15, 25, or 35.
- Added control of thick segment checking using the SNLOG flag on *CONTACT. A thick segment check has been done quietly throughout the history of segment-based (SOFT = 2) contact. A search along the contact surface is made looking for thick pairs that are too close, and those pairs are removed from contact. This is done to prevent possible unstable behavior that could occur if the offsets of nearby thick segments come into contact at a bend in the mesh. When SNLOG = 0 or 1, the thick segment check will be done quietly as before. Set to 2, the check will be done and a warning 21465 will be written to report the segment pairs that are removed from contact. Set to 3, the thick segment check will be omitted.
- Enabled segment-based contact (SOFT = 2) to work with segments attached to cohesive elements with zero volume.
- Enabled *CONTACT_ENTITY to work with thick shell elements. It was previously error terminating in the input phase.

- Enabled SINGLE_SURFACE segment-based (SOFT = 2) contact to output gap values to the intfor output database. Previously, output of the gap values worked only for the non-SINGLE_SURFACE contact types.
- Enabled the SPOTHIN and SWRADF variables on *CONTROL_CONTACT to work when spot welds are modeled by sharing nodes with the shell parts that are welded. This has not worked until now because the search for affected shells was based on tied contact, and tied contact is not needed when nodes are shared.
- Enabled *CONTACT_2D_AUTOMATIC to work with erosion of 2D solid elements that use *MAT_081 or *MAT_082.
- Enabled multiple instances of *CONTACT_2D_AUTOMATIC_TIED to be reliably used in a model.
- Improved the PSTIFF option of segment-based contact to make the segment mass better match the nodal masses. The main effect is to increase the stiffness for segments on the edge of the mesh. For a regular mesh with no mass scaling or lumped nodal masses, the PSTIFF option now calculates the same segment mass as the default method based on segment volume and density.
- Fixed segment-based contact for adaptive remeshing and full deck restarts.
- Fixed the coefficient of restitution option (ICOR) in segment-based contact when it is used along with DPRFAC (Depth of Penetration Reduction FACtor).
- Fixed *CONTACT_2D_FORCE_TRANSDUCER with both slave and master sides defined when they are used with *CONTACT_2D_AUTOMATIC in MPP. A memory clobber was likely and also output was incorrect.
- Fixed MPP wear calculation in segment-based contact. A mix-up of ID numbers means that the wear was only working when contact interfaces were numbered sequentially from 1.
- Fixed the THERMAL option of *CONTACT in MPP when some MPP partitions do not participate in all the contact definitions.
- Fixed *CONTACT_2D_AUTOMATIC_TIED when used with selective mass scaling, which is made active by setting IMSL = 1 and DT2MS < 0 on *CONTROL_CONTACT. The problem occurred when selective mass scaling was applied to the nodes that are tied. Prior to the change, kinematic constraints were failing.
- Fixed MPP *CONTACT_2D_AUTOMATIC when beams were in contact, or when solids were used with nonzero values of solid surface offset(s) (variables SLDSOS or SLDSOM). Both cases could lead to using invalid memory space and a segmentation fault.
- Add new option ICNEP = 1 to *DEFINE_FRICTION with which those lines with non-existent parts or part sets will be ignored.
- Variable ENGOUT on *CONTROL_OUTPUT added to write minimum (wherever occurring) contact energy densities to d3plot (MORTAR contact only).
- Variable NENG on *DATABASE_EXTENT_INTFOR added to write contact energy densities to intfor (MORTAR contact only).

- Variable PENOUT on *CONTROL_OUTPUT added to write maximum (wherever occurring) penetrations across interfaces to d3plot (MORTAR contact only).
- Variable NPEN on *DATABASE_EXTENT_INTFOR to write absolute and relative penetrations to intfor (MORTAR contact only).
- Variable NTWELD on *DATABASE_EXTENT_INTFOR writes user tied weld history variables to intfor file (MORTAR contact only).
- Variable VC on *CONTACT_... supported for MORTAR contact.
- Added variable TDPEN on *CONTACT_2D_AUTOMATIC_SURFACE_-TO_SURFACE_MORTAR. This is the analogue to interference option MPAR1 for IGNORE = 3 in 3D automatic MORTAR contact.
- Mortar contact supports erosion, that is, any solid segment or shell edge segment belonging to parts involved in the contact definition will become active in the contact when exposed to the environment due to erosion.
- IGNORE = 4 on *CONTACT_..._MORTAR supports a curve of relative interference reduction vs. time. When MPAR1 < -1.0, |MPAR1| is the curve ID.
- *LOAD_THERMAL_CONSTANT is supported in mortar contact, in those situations where the contact properties depend on temperature.
- In eigenvalue analysis, nonzero mortar contact stress will not influence rigid body modes, i.e., you should see the proper rigid body modes among the available output.
- TEMP < 0 on *CONTACT_..._MORTAR_TIED_WELD calls a user tied weld interface, allowing implementation of "arbitrary" tied condition in weld simulations.
- *CONTACT_TIED_..._THERMAL supported for (spotweld) beams.
- The THERMAL option can now be used for the following tied contacts:

  *CONTACT_TIED_NODES_TO_SURFACE_OFFSET_THERMAL

  *CONTACT_TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET_THERMAL

  *CONTACT_TIED_NODES_TO_SURFACE_THERMAL

  *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_OFFSET_THERMAL

  *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET_THERMAL

  *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET_THERMAL

  *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_THERMAL

  BUT the slave set type can NOT be a node set, at least not yet.
- Now implemented for MPP are *CONTACT_AUTOMATIC_SURFACE_-TO_SURFACE_TIEBREAK and *CONTACT_AUTOMATIC_SURFACE_-TO_SURFACE_TIED_WELD.

- Segment-based contact (SOFT = 2) now supports groupable and non-blocking features.
- Option FTORQ for transmittal of moments across the contact interface is implemented in the groupable node-to-surface contacts, surface-to-surface contacts, and tied contacts.
- *CONTACT_AUTOMATIC_GENERAL supports options SOFT = 1 and SRNDE for shell edge treatment.
- *CONTACT_AUTOMATIC_GENERAL supports options SOFT = 1 and SRNDE for the shell edge treatment.
- A new orthogonal friction model is developed for the constraint and penalty *RIGIDWALL_ORTHO. A RIGIDWALL_ORTHO with equal friction coefficients in all directions now behaves as a regular RIGIDWALL with a single friction coefficient.
- *CONTACT_RIGID_TO_RIGID is enhanced to support unloading curve, FCM = 1/2/3.
- *CONTACT_CONSTRAINT_NODES_TO_SURFACE is enhanced to support variable PENSF on MPP Card 2 of *CONTACT and recover contact forces that existed before solid remeshing associated with 3D solid adaptivity.
- *CONTACT_AUTOMATIC_GENERAL is improved to stabilize short beams with small rotational inertias.
- ERODING contacts are now supported in full deck restarts.
- *CONTACT_AUTOMATIC_BEAMS_TO_SURFACE supports beam to shell_edge contact and detects the contact between shell edges and phantom nodes.
- A new built-in variable IDRFLG added to *DEFINE_CURVE_FUNCTION, *DEFINE_FUNCTION. IDRFLG carries a value of 1 during dynamic relaxation and 0 in the transient phase.
- Multiple *DEFINE_FRICTION tables with the same ID are merged.
- Effect of variable I2D3D is correctd in MPP to avoid duplicated contact segments when the attached shell and solid are not assigned to the same processor.
- ERODING contact in MPP is fixed to compute the stiffness of newly-exposed solid nodes and segments and apply the optional thickness SLDTHK to the newly-exposed solid segments.
- The contact thickness in *PART_CONTACT is applied to *CONTACT_AUTOMATIC_GENERAL.
- *CONTACT_AUTOMATIC_GENERAL is corrected by reducing the contact stiffness to half.

- ***CONSTRAINED**

  - Fix ordering issue during initialization of *CONSTRAINED_INTERPOLATION in MPP that could have resulted in incorrect "colinear" warnings or possibly deadlock

- Fix MPP message passing error that could occur if a node involved in *CONSTRAINED_SHELL_TO_SOLID is shared between more than 2 processors.
- Enhanced processing of an exactly singular constraint matrix for *CONSTRAINED_INTERPOLATION.
- Corrected error in the checking for massless *CONSTRAINED_INTERPOLATION constraints in single precision.
- Enhance *CONSTRAINED_INTERPOLATION in the explicit solver so the linear algebra of the constraint equation processing always operates in double precision to reduce loss of precision in single precision executables.
- Add implicit support to *CONSTRAINED_SHELL_IN_SOLID.
- *CONSTRAINED_BEAM_IN_SOLID:
  - Add support for IDIR = 1. This means the beam is allowed to freely slip longitudinally inside solid elements.
  - Add implicit support for the AXFOR option. Debonding between beams and solids may now be simulated using the implicit solver.
  - Add implicit support for CONSTRAINED_BEAM_IN_SOLID_PENALTY.
- Fix incorrect *CONSTRAINED_INTERPOLATION motion of the dependent nodes if the number of constrained dof is less then the total dof, i.e., DDOF.ne.123456.
- Fix ineffective PNODE in *CONSTRAINED_NODAL_RIGID_BODY when PNODE < 0.
- Fix kinetic energy dependence on MPP core count when using *CONSTRAINED_JOINT_(OPTION) with LMF = 1 in *CONTROL_RIGID.
- Added swforc file output for welds modeled by *CONSTRAINED_GENERALIZED_WELD. Output values are the brittle failure forces, axial and shear, and the failure function.
- Improved temperature-dependent failure curve option of spot welds defined by * CONSTRAINWED_SPOTWELD. Curve interpolation was using the wrong segment of the curve. Also, changed swforc output to show zero axial and shear forces after a weld has failed.
- Add correct torsion to SPR3 (*CONSTRAINED_INTERPOLATION_SPOTWELD with STIFF4 > 0).
- Small modification for PIDVB < 0 of *CONSTRAINED_SPR2/SPR3: Beams get deleted after failure (they were just separated before).
- Fix for *CONSTRAINED_SPR2/SPR3: part id of internally generated beams has to take Nodal Rigid Bodies into account.
- Ctl. Add error trap if INTPERR = 1 is set on *CONTROL_SHELL and data interpolation would be done with *INITIAL_STRESS_SHELL.
- *CONSTRAINED_JOINT_STIFFNESS_CYLINDRICAL now available to model cylindrical/revolute connections with play
- *CONSTRAINED_COORDINATE now supports part set ID in addition to part ID.

- *CONSTRAINED_COORDINATE: When IDIR is negative, the constraint is applied at the nearest node, not at the coordinate.
- For nodes which are not attached to any elements, turn off the automatic global constraint normally applied to such nodes if those nodes are involved in *CONSTRAINED_MULTIPLE_GLOBAL.

- ***CONTROL**

  - *CONTROL_REFINE_... with NTOTRF = -1: Refine solids or shells with elements dynamically added during the run (as opposed to NTOTRF > 0 for which the user must estimate the number of child elements required for the refinement and these elements are added during the initialization).
  - *CONTROL_REFINE_SHELL: Support contact for the case of NTOTRF = -1, i.e., dynamically add contact segments due to mesh refinement.
  - In *CONTROL_REFINE_... keywords:

    - If CRITRF < 0, reverse the refinement conditions
    - If CRITRM < 0, reverse the coarsening conditions

  - *CONTROL_STAGED_CONSTRUCTION:

    - Added support for *ELEMENT_SOLID ELFORMs 4, 13, 16, 17.
    -
    - Added new input field IDYNAIN to suppress output of dynain file at end of each stage. This can be done for all stages using IDYNAIN on *CONTROL_STAGED_CONSTRUCTION, or for individual stages using IDYNAIN on *DEFINE_CONSTRUCTION_STAGES.
    - Fix bug affecting *MAT_021 and *MAT_021 solids with Staged Construction (*DEFINE_STAGED_CONSTRUCTION_PART). When a part became active, the stresses were wrongly initialized. Bug affected solid elements only.

  - *CONTROL_PORE_FLUID: Fixed bug in internal energy calculation for solid element types with more than one integration point. This affected only models containing *CONTROL_PORE_FLUID.
  - *CONTROL_ADAPTIVE: Small change to 2d adaptive remeshing that should improve behavior for meshes with bad boundaries.
  - Fix problem with 2d adaptivity and boundary merging. Some boundary points between materials weren't merged in some cases, depending on where the program thought the 2d contours started. This only applies for *CONTROL_ADAPTIVE adpopt = 8 with mmm2d = 1.
  - Implement idrflg = -3 for *CONTROL_DYNAMIC_RELAXATION which invokes the version R7 (and earlier) implementation of idrflg = 3. Now idrflg = 3 means the parts not included in the part set are eliminated from the computation, and idrflg = -3 means all parts are included in the explicit

computation but only those in the part set are included in the distortional-kinetic-energy-based convergence check in the dynamic relaxation phase.

- Ignore *CONTROL_SPOTWELD between parts that are inactive during implicit dynamic relaxation phase.
- Fix a bug whereby SHLTRW in *CONTROL_SHELL was ignored when ISTUPD > 0.
- Add variable MLKBAG to *CONTROL_OUTPUT to include accumulated airbag gas leakage in abstat.
- Fix wrong pointer used for section id/properties when generating part for visualization of rigidwall during adaptivity. This cause error message KEY+137.
- *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS:

  - Add a stiffness option (ISTFNS) to *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS.
    EQ.1: no stiffness is added.
    EQ.2: only spin softening is added which keeps stiffness symmetric.
    EQ.3: both spin softening and gyroscopic effects are added which makes the matrix skew-symmetric.

    Default is 3.
  - Add the whirling direction to the output eigout.

- DISPLAY option in *RIGIDWALL_PLANAR will be ignored for stationary rigid walls when SKIPRWG = 1 in *CONTROL_CONTACT.
- Added options to the ERODE parameter on *CONTROL_TIMESTEP. The original options were 0 and 1 where 1 causes solid and thick shell elements to be eroded if their time step dropped below TSMIN*DT0 where TSMIN is specified on *CONTROL_TERMINATION, and DT0 is the solution time step that is calculated in the first cycle. The new options are to check and delete shell elements and beam elements having small time steps. The control of shells is in the 'tens' place and the beams is in the 'hundreds' place. The valid options for ERODE are now 1, 10, 11, 100, 101, 110, and 111. For ERODE = 111, beams, shells, and solids are all checked. For ERODE = 11, shells and solids are checked. The original options of 0 and 1 still behave exactly as they did.
- Added support of *CONTROL_SUBCYCLE for segment-based (SOFT = 2) contact. Contact forces are calculated for all nodes of a segment pair only if any node of the pair is due to have a force calculated.
- Fixed some issues so that solid elements and thick shell elements will work as advertised with respect to DTMIN on *CONTROL_TERMINATION, ERODE on *CONTROL_TIMESTEP, and PSFAIL on *CONTROL_SOLID. These variables control whether we terminate the analysis or fail elements when their time step becomes small or their volume becomes negative.

- Add an option TET13V on *CONTROL_SOLID to choose between the efficient and a more accurate version of the tet type 13 implementation.
- Lagrangian multiplier joints, LMF = 1 on *CONTROL_RIGID, are supported for consistent mass scaling (RBSMS = 1 on *CONTROL_RIGID).
- *CONTROL_ADAPTIVE:Fix bug in 3D tet adaptivity when there are rigid parts with a tetrahedral mesh and ELFORM = 10. Before the bug fix, the rigid parts with ELFORM = 10 were not properly written out during adaptive restart, resulting in an error termination.
- *CONTROL_REMESHING: 3D tet adaptivity is able to handle the remeshing of multiple bodies in a single adaptive parts. Preserving of feature lines is controlled by the variable SEGANG.

- **Discrete Element Method**

  - New keyword *DEFINE_DE_FLOW_DRAG to support DES interacting with external flow field.
  - Add a moving system with birth/death times in *DEFINE_DE_ACTIVE_-REGION.
  - Fix bug of generation failure and uneven distribution of injected DES particles (*DEFINE_DE_INJECTION) when the radius of injected DES particles is very small.
  - Fix segmentation fault when using simple and small restarts in DEM.
  - Add the keyword *DEFINE_DE_COHESIVE to enable the capillary force in specified DES node set or parts.
  - Add the keyword *INITIAL_STRESS_DES to initialize stress in discrete element spheres (DES).
  - Output the DES initial stress and geometry information into dynain_ini and dynain_geo files (FSPLIT = 1 in *INTEFACE_SPRINGBACK_LS_DYNA).
  - Fix incorrect behavior when using *DEFINE_DE_ACTIVE_REGION with itype = 1, i.e. BOX, with *DEFINE_TRANSFORMATION. The box was not rotated.

- **Element Free Galerkin (EFG)**

  - *SECTION_SOLID_EFG: Volume of 6-noded solid elements using element formulation 41 (EFG) was calculated incorrectly. That bug is now fixed.

- ***ELEMENT**

  - For triangular shells with thickness stretch (*SECTION_SHELL, ELFORM = 27) the option IDOF = 1 is made available allowing a continuous thickness field between them and quadrilateral shells with thickness stretch, ELFORM = 25,26.
  - For the Belytschko-Tsay shell with thickness stretch (*SECTION_SHELL, ELFORM = 25) an improved representation of stresses over the thickness may be obtained with the new option IDOF = 11 instead of IDOF = 1 (for

continuous thickness fields) or with the new option IDOF = 12 instead of IDOF = 2 (for discontinuous thickness fields).

- Changed *DEFINE_ELEMENT_EROSION to be more compatible with the original FE options and added support for IGA elements via *DEFINE_ELEMENT_EROSION_IGA.
- Added calculation of implicit tangent for shell formulation 46, a 2D cohesive element.
- *ELEMENT_DIRECT_MATRIX_INPUT:
  - Enhanced the reading of superelement files to support connections using nodes, scalar nodes, internal dofs, and the label of "0" for scalar nodes.
  - Added MPP logic for collecting nodes associated with superelements for use in decomposition for the reading of binary formatted files for superelements.
  - Fix an error associated with using superelements in MPP explicit. Two arrays should have been initialized to zero, but the bug did not manifest itself until the user used 12 processes. The arrays are now properly initialized to zero.
  - Corrected the initialization of superelements involving nodes with SPC constraints.
- *ELEMENT_SEATBELT_PRETENSIONER: make type-8 pretensioner available for 2D belt.
- *ELEMENT_SEATBELT_SENSOR: add type-5 sensor for tracing retractor payout.
- Restructure nodal volume data communication for solid tet formulation 13 for better parallel performance.
- Added new more conservative time step calculation (independent of shell thickness) for improved stability in cohesive shell formulation 29.
- Fix incorrect stress output to d3plot and ASCII files when using tetrahedron solid types 10 and 13 with orthotropic materials and when CMPFLG = 1 in *DATABASE_EXTENT_BINARY.
- Fixed a bug that caused axisymmetric elements (shell ELFORMs 14,15) to instantly fail when DTMIN > 0 in *MAT_ADD_EROSION.
- Shell elements with *MAT_NULL can be deleted by TIME in *DEFINE_ELEMENT_DEATH_SHELL.
- Added a check for invalid ELFORM values on *SECTION cards. Invalid values were leading to terminations without clear error termination messages.
-
- Added an optional modification to the time step calculation that is done for solid elements. See variable IHDO in *CONTROL_TIMESTEP. Setting IHDO to 1 causes the time step calculation to be modified such that it does not have a discontinuity between expansion and compression. With the default

time step calculation, the solution time step can jump up or down about 6% when an element controlling the time step switches from compression to expansion. The size of the jump depends on the linear bulk viscosity coefficient. These jumps may not occur if multiple elements are controlling the time step as it is unlikely for all to be expanding at the same time, so the smaller time step is used consistently. Setting IHDO to 1 prevents this discontinuity from occurring.

- Added support for shell form 23 in output file eloutdet (*DATABASE_ELOUT). When shell form 23 is used, the code can now output stress and strain at 2x2 or 3x3 integration points and extrapolate to all 8 nodes.
- Added a bulk viscosity option for thick shell (TSHELL) elements (See variable TSTYPE on *CONTROL_BULK_VISCOSITY).
- Enabled plane strain and axisymmetric elements (*SECTION_SHELL ELFORM 13, 14, 15) to include stress initialization by *INITIAL_STRESS_SHELL for materials that use equations of state.
- Enabled solid spot weld assemblies to use 8 node and 6 node cohesive elements with *MAT_240. There is no assembly failure calculation, but the resultant forces and moments are calculated and can be output to the swforc file. This works with solid element formulations 19, 20, 21, and 22.
- Enabled variables CDL and TDL on *SECTION_DISCRETE to be used when one of the discrete beam nodes is constrained by an SPC.
- Improved the accuracy of pressure loading on fully integrated, volume-weighted axisymmetric solid elements (ELFORM = 15/NIP = 4 on *CONTROL_SHELL), particularly along the axis of symmetry.
- Improved the post-failure behavior of integrated beam elements by adding an option (NAUDP on *SECTION_BEAM) to update the neutral axis when one or more integration points fail so that nodal moments are correctly calculated. This option is available for the integrated beam formulations 1, 9, 11, and 14 when used with *MAT_003, *MAT_098, *MAT_100, *MAT_124, and *MAT_158.
- Improved mass scaling of beam element formulations 4 and 5. A dimensional problem was causing erratic behavior which could cause unstable behavior.
- Fixed bug affecting simultaneous use of ERODING contact with cohesive pentahedral solids, which through use of ESORT = 1, are changed to formulations 21/22. Prior to this fix, this combination of keywords could have lead to error termination due to negative volume.
- Add check if two or more parts use solid element formulation 13 (tetrahedron with pressure averaging) and those parts share some nodes. Write a warning in that case because such a condition could lead to instabilities.
- COHOFF on *SECTION_SOLID allows for adjusting the reference layer for cohesive element formulations 20 and 22.
- Fixed bug in plane strain, 8-node shell formulation 55 to show correct stresses when 4 in-plane points are required for d3plot output. Restored writing out stress components at all integration points in elout. Shell

formulation 55 is for implicit analysis only and requires 4, 9 or 16 integration points (default 4) since it is meant to simulate the singular stress field around a crack tip.
- For thick shell (TSHELL) formulations 1 and 2, fixed incorrect stress output in elout when used for composite materials and CMPFLG = 1.

- ***EM (Electromagnetic Solver)**

  - *EM_SOLVER_BEM: Option PRECON = 4 (line 1 field 4), LLT factorization of the BEM matrix, can now be used in MPP. It could only be used in serial/SMP up to now.
  - Eddy current / inductive heating solver:

    - Addition of monolithic solver for FEM+BEM solve (higher time steps and improved stability for ferromagnetic materials) (*EM_SOLVER_FEMBEM).
    - Optimization of memory/CPU cost in BEM solve.
    - Improved accuracy of solution on prismatic elements.
    - Addition of LLT factorization as a preconditioner for BEM solver in MPP (*EM_SOLVER_BEM).
    - Added 2D and 2D axi capability for RSW and other applications (*EM_CONTROL and *EM_MAT_004).
    - Support of eroding conductors (*EM_CONTROL_EROSION and *EM_MAT_...).
    - Improved accuracy of thermal coupling for tetrahedral elements.

  - EM contact:

    - Optimize contact search and robustness => reduced calculation times and memory cost
    - Added capability in RSW to specify electric contact law (*EM_CONTACT_RESISTANCE).

  - Battery Module Release:

    - Equivalent circuit model (ECM): Implementation of Randles circuits as simplified electro-chemistry model
    - Dynamic EM-mechanical-thermal coupling for external and internal short circuit (battery crush) (see in particular *EM_RANDLES_SHORT).
    - Bath loading, coupling mono-bi and bath in same domain.
    - Purkinje network (*EM_EP_MECACOUPLING).
    - Current defined stimuli (*EM_EP_TENTUSSCHER_STIMULUS).

- ***FATIGUE**

- Added using *DEFINE_TABLE to define SN curves for multiple mean stress.
- Added new mean stress correction method (METHOD = 5): Morrow equation (SN curve).
- Extended thick shell element fatigue analysis (including multiaxial) to MPP.
- Extended solid element fatigue analysis (including multiaxial) to MPP.
- Extended shell element fatigue analysis (including multiaxial) to MPP.
- Fixed a bug in running *FATIGUE_SUMMATION. This function is now working as a standalone module. Previously it was used inside fatigue analysis.
- Added warning message MSG_FTG+5 in case the calculated stress value is higher than the stress of the first point on SN curve, to remind user to check the SN curve or stress calculation. If the stress value is higher than the stress of the first point on SN curve, extrapolation is used and this may not be accurate, especially if the first point is the UTS (Ultimate Tensile Strength).
- Added a parameter (DMGMIN in *FATIGUE) to allow user to define base damage ratio for elements with zero stress or strain.
- Added a new option _D3PLOT to allow running time domain fatigue analysis based on d3plot.
- Implemented *FATIGUE_FAILURE to remove failed elements from model.
- Implemented *FATIGUE_MULTIAXIAL to run multiaxial fatigue analysis.
- Implemented *FATIGUE_LOADSTEP to run fatigue analysis with multiple load steps.
- Added warning message MSG_FTG+4 if fatigue properties are not defined for the current element under fatigue analysis.
- Implemented definition of initial fatigue damage ratio by part or part set (PID/SID in *INITIAL_FATIGUE_DAMAGE_RATIO).
- Added semi-log SN curve definition. This can be used both in time domain and frequency domain fatigue analysis.
- Added exposure time TEXPOS for fatigue analysis, in case it is different from endtim in *CONTROL_TERMINATION.
- Added an option D3PLOT to *INITIAL_FATIGUE_DAMAGE_RATIO to allow using damage variables from transient preload cases as initial fatigue damage ratio.
- Fixed a bug in defining mean stress correction methods. The UTS/yield strength should be defined on material model, not part ID, according to the Manual.

- **Forming Analysis**

  - *DEFINE_BOX_NODES_ADAPTIVE now supports more than 2 levels of mesh refinement for tube adaptivity. Maximum level is specified by variable LEVEL.

- ONESTEP simulations with triangular elements used to produce misleading effective plastic strain results in regions with high curvatures. This is now fixed.
- Fix bug in ONESTEP method that produces false minor and major strain in the output file onestepresult
- Fix bug in *CONTROL_FORMING_ONESTEP, OPTION = 6.
- Fix MPP bug for ONESTEP output files repositioned.k and onestepresult.
- Fixed the problem of lacking temperatures at the lancing line in MPP lancing simulations.
- Fixed a memory access problem which resulted in error termination of simulations of unflanging process.
- *CONTROL_FORMING_MAXID is supported by 3d tet adaptivity (*CONTROL_ADAPTIVE).
- *INCLUDE_STAMP_PART: Improvement to support *DEFINE_TRANSFORMATION.
- *DEFINE_CURVE_TRIM_3D: Projection of curve to blank is now allowed when the trimming curve is far away from the blank. This was previously disallowed.
- *CONTROL_FORMING_ONESTEP:
  - Add a CPU time report for onestep method to record the time spent on initial unfolding.
  - Allow user to output specific part to the file of 'onestepresult'.
  - The fracture curve LCSDG can be automatically found from *MAT_ADD_EROSION.
- *CONTROL_FORMING_PRE_BENDING(_LOCAL): New option allows user to define a vector based on a local coordinate system. If the LOCAL option is used, then coordinate system ID is input on Card 2.
- *ELEMENT_LANCING: Allow multi-curve lancing with different starting times. Previously, only one starting time was allowed. With this new feature, all the lancing curves will be used to cut at the same time, but the nodes will not be able to separate until the lancing time is reached.
- *CONTROL_FORMING_AUTOPOSITION_PARAMETER
  - Support 2D elements.
  - When CID is a negative value, its absolute value refers to a vector ID (*DEFINE_VECTOR) rather than a coordinate system ID.
- *DEFINE_FORMING_ONESTEP_MASTER: This new keyword allows a second blank to be welded to a master blank. The two blanks are connected by using spot welds.
- Add keyword *CONTROL_FORMING_TRIM_SOLID_REFINEMENT to homogeneously refine elements along a trim curve. Supports *DEFINE_CURVE_TRIM_2D and *DEFINE_CURVE_TRIM_3D.

- *DEFINE_CURVE_TRIM_2D allows refinement of a sandwiched part along trimming curves. The sandwich core comprised of multiple layers of solid elements can now be refined. In addition, the trimming curve can be in any direction (2D trimming only).
- *CONTROL_FORMING_ONESTEP_ORTHO uses 2 nodes in the final part to define the rolling direction of the anisotropic material.
- *DEFINE_FIBERS:
  - User can perform carbon fiber simulation without the matrix (fibers only).
  - No need to define material property for the matrix.

- ***FREQUENCY_DOMAIN**
  - Correction to rotational inertia for *FREQUENCY_DOMAIN_SSD.
  - *con Extend logic for implicit constraint handling to tied contact with _CONSTRAINED_OFFSET to skip any SPC constraints on the rotational dofs of the SPC slave nodes. We already have such logic to skip the constraints on the translational dofs for the slave nodes for all tied contacts.
  - *loa: Correct the issue where use of *LOAD_BODY is applied to a model with rigid bodies. If the vector of nodes having the load applied belong to a rigid body, then the vector of elemental stiffness matrices is null. So logic was added to skip the call to the implicit matrix assembly subroutine for that case.
  - *DATABASE_FREQUENCY_BINARY_*{OPTION}*:
    - Added a new option SUMMATION to *DATABASE_FREQUENCY_BINARY_D3PSD and *DATABASE_FREQUENCY_BINARY_D3RMS to get total PSD and RMS response from multiple loading resources.
    - Update to support *INCLUDE_TRANSFORM.
    - Added NFSPACE = 3 to define output frequency as eigenfrequencies.
    - Added an option (BINARY = 2) to dump out individual mode response to d3spcm (*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM).
  - *DATABASE_FREQUENCY_ASCII_*{OPTION}*:
    - Update to support *INCLUDE_TRANSFORM.
    - Added NFSPACE = 3 to define output frequency as eigenfrequencies.
  - *FREQUENCY_DOMAIN_ACOUSTIC_BEM:
    - Update to support *INCLUDE_TRANSFORM.
    - Added radiated power computation to Rayleigh method.
  - *FREQUENCY_DOMAIN_ACOUSTIC_FEM:

- Update to support *INCLUDE_TRANSFORM.
- Update to use mixed elements (tet, hex and pentahedrons).
- Fixed a bug in running with combined boundary conditions (transient vibration and impedance).
- Added a flag (RESTRT) to run restart from transient analysis results. LS-DYNA will read an existing binary database to extract the velocity boundary condition.
- Added defining impedence boundary condition by *DEFINE_CURVE.
- Added a flag "idump" in *FREQUENCY_DOMAIN_ACOUSTIC_-FEM_EIGENVALUE to dump out acoustic stiffness and mass matrices.

◦ *FREQUENCY_DOMAIN_FRF:

- Update to support *INCLUDE_TRANSFORM.
- Enabled using structural damping (*DAMPING_STRUCTURAL).

◦ *FREQUENCY_DOMAIN_LOCAL:

- Implemented this keyword to run frequency domain analysis on part of model.
- Implemented options SOLID_SET, SHELL_SET, BEAM_SET and TSHELL_SET.
- Implemented option EXCLUDE to exclude specified sets of elements or nodes.
- Enabled using this feature on response spectrum analysis (*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM).

◦ *FREQUENCY_DOMAIN_MODE:

- Implemented option LOAD_PROJECTION, to select the normal modes based on load projection ratio.
- Implemented option MODAL_COEFFICIENT to select the participating modes based on relative significance of the modal coefficients.
- Enabled this keyword to FRF (*FREQUENCY_DOMAIN_FRF) and SSD (*FREQUENCY_DOMAIN_SSD).

◦ *FREQUENCY_DOMAIN_RANDOM_VIBRATION:

- Update to support *INCLUDE_TRANSFORM.
- Added parameters ICOARSE and TCOARSE in *FREQUENCY_DOMAIN_RANDOM_VIBRATION to reduce points in PSD curve to save CPU time.
- Added parameter TOL_INTG to allow user to define the relative tolerance for integration. The default is 1.e-5.
- Updated PSD curve input so that loading types 9 (base velocity), 10 (base displacement), 11 (enforced acceleration), 12 (enforced velocity),

and 13 (enforced displacement) can be defined through time history curve.
- Added structural damping (*DAMPING_STRUCTURAL) to random vibration analysis.

◦ *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM

- Enabled running response spectrum analysis with CASE.
- Added beam elements stress/strain components in elout_spcm when applicable. Previously only resultant forces and moments are provided.
- Fixed a bug in reading d3eigv family files for multipoint response spectrum analysis.
- Update to support *INCLUDE_TRANSFORM.
- Fixed a bug in running response spectrum with base excitation defined in time domain.

◦ *FREQUENCY_DOMAIN_SSD

- Fixed a bug in reading d3eigv for restart option (RESTMD = 1) when both 4-noded shells and 8-noded shells (like ELFORM 23) are present in the model.
- Update on SSD to use the current geometry of model if SSD is performed after dynamic analysis. Previously SSD is always performed based on the initial geometry.
- Implemented radiated acoustic power computation with Rayleigh integral.
- Added radiation efficiency computation for ERP (*FREQUENCY_DOMAIN_SSD_ERP).
- Enabled sense switch for direct SSD (ctrl-c).
- Updated *FREQUENCY_DOMAIN_SSD_FATIGUE by adding scale factor for each load. Added MSG_KEY+1232, MSG_KEY+1233 to remind user about the changes.
- Added torque and rotational dof excitation for SSD (VAD = 8, 9, 10, 11), with local damping (DMPFLG = 1).
- Update to support *INCLUDE_TRANSFORM.
- Extended *FREQUENCY_DOMAIN_MODE_LOAD_PROJECTION to the computation of nodfor_ssd, nodfor_ssd and elout_ssd.
- Added option _FREQUENCY_DEPENDENT to run direct SSD with frequency dependent material properties.
- Added option _FRF to run SSD with FRF setting. Changed the output filename to d3frf. With this option, one can get fringe plot for FRF (previously only xyplot results were given by FRF).
- Added rotational dof output in nodout_ssd, when applicable.

- Added new loading options vad = 12, 13, 14 for direct SSD (*FREQUENCY_DOMAIN_SSD_DIRECT), for prescribed velocity, acceleration, and displacement.

- ***ICFD (Incompressible Fluid Solver)**

  - Navier Stokes solve/free surface flows:

    - *ICFD_BOUNDARY_FSWAVE: Added Stokes wave of fifth order, solitary wave model and irregular waves using JONSWAP spectrum. Add a complete check of wave generation settings and the validity of the parameters within each wave theory. *ICFD_BOUNDARY_PRESCRIBED_VEL can be used jointly to model the interaction of any wave theory and any prescribed velocity at inlets.
    - *ICFD_BOUNDARY_PERIODIC: Addition of periodic, reflective and sliding mesh boundary conditions.
    - *ICFD_CONTROL_TIME: Maximum and minimum time step definitions can now be made functions of time.
    - *ICFD_CONTROL_OUTPUT_SUBDOM: Supports output to d3plot (before only supported VTK and VTU formats).
    - *ICFD_DEFINE_SOURCE: Allows the user to specify a generic volumetric force entering the velocity solve. Can be useful in various applications, for example, when coupling ICFD with the EM solver.
    - *ICFD_INITIAL: *DEFINE_FUNCTION can now be used to define an initial velocity profile.
    - *ICFD_INITIAL_LEVELSET: Extended the number of initial surfaces which can be generated; boxes and spheres are now possible on top of surface plane.
    - *ICFD_MAT: Variable CA specifies the contact angle for the liquid-vapor interface, for use in surface tension problems.
    - *ICFD_MODEL_POROUS: Added new porous models, pmid = 11 and pmid = 10, for parachute fluid flow. To be used with thin shells (See *MESH_EMBEDSHELL).
    - Bug Fix: Removed pressure outlet smoothing technique after noticing it could cause disruption of results. Results of internal flow problems might be especially affected.
    - Bug fix: Added more digits to icfd_fluidmesh.key output to avoid round off errors.

  - Turbulence models:

    - *ICFD_CONTROL_TURBULENCE, SUBMOD = 2: Added LES Dynamic turbulence model originally proposed by DK Lilly (1991) with localization on coefficient CS by Piomelli and Liu (1995).

- *ICFD_DEFINE_TURBSOURCE: new keyword which allows addition of turbulent source terms for RANS models.
- *ICFD_CONTROL_TURB_SYNTHESIS: now supports *DEFINE_-FUNCTIONs.
- Bug fix: Fixed incorrect calculation of stresses in laws of the wall.

◦ FSI, DEM coupling and remeshing:

- *ICFD_CONTROL_ADAPT: Variable KIS may be used to diminish the excessive coarsening that can potentially occur when remeshing by keeping the nodal mesh size distribution of the initial volume mesh.
- *ICFD_CONTROL_ADAPT: Variable VAR gives the user more control on the error calculation triggering a remesh.
- *ICFD_CONTROL_DEM_COUPLING: Variable MAXVEL is a 'max velocity' term for the DEM force calculation intended to avoid spurious high velocities which could cause the fluid solver to diverge.
- *ICFD_CONTROL_DEM_COUPLING: Variable DTYPE = 1 invokes option to use Morrison formula for Drag calculation to apply on DEM particles.
- *ICFD_CONTROL_FSI: Added control on the number of FSI fluid subiterations. This avoids the sometimes unneeded excessive number of FSI subiterations when the fluid and very light structures (like parachutes) develop a resonance-like mode inside the FSI subiterations (coupling iterations). See variable NSUB.

◦ Thermal and Conjugate heat transfer (CHT):

- *ICFD_BOUNDARY_FLUX_TEMP: Modified boundary condition so that the prescribed value is equivalent to quantity prescribed in BOUNDARY_HEAT_FLUX.
- *ICFD_BOUNDARY_CONJ_HEAT: Added option to impose a temperature drop between solid and fluid (see variables VAL and SFLCID). Also added an alternative penalty based (mortar) coupling method (CTYPE = 1). This can be beneficial in cases with large mesh size discrepancies but is typically associated with higher solve times.
- *ICFD_BOUNDARY_CONVECTION_TEMP: Added keyword to prescribe a convection boundary condition for thermal problems.
- *ICFD_CONTROL_CONJ: TSF feature added in ICFD solver for thermal and CHT problems.
- *ICFD_CONTROL_TURBULENCE: Added thermal law of the wall; see variable TWLAW.
- Bug fix: Previously, all FSI surfaces were taken into account for the thermal and fluid coupling. Now only the boundaries to which the keyword

        - *ICFD_BOUNDARY_CONJ_HEAT is applied will be coupled to the solid.
        - Bug fix: HCLCID and TCLCID in *ICFD_MAT were incorrectly read in MPP.
        - Bug fix: icfd_thermal.dat now keeps getting output in CHT cases after the Navier-Stokes equations have reached steady state.

- **Implicit (Mechanical Solver)**

    - New LSOLVR options in *CONTROL_IMPLICIT_SOLVER:

        - 17 (SMP only): iterative solver with a preconditioner based on a local incomplete factorization.
        - 27 (MPP only): ditto.

    - *CONTROL_IMPLICIT_ORDERING: Made various improvements to LS-GPart (ORDER = 4).

        - Significant improvements in MPP;
        - Added hybrid parallelism (OpenMP);
        - Added a progress report ("heartbeat") for large problems.

    - New option where ILIMIT in *CONTROL_IMPLICIT_SOLUTION can vary with time. If ILIMIT < 0, |ILIMIT| is taken as the curve ID which defines the iteration limit between automatic stiffness reformations (ordinate) vs. time (abscissa).
    - Implemented an approximate constitutive matrix for implicit analysis of *MAT_187/*MAT_SAMP-1.
    - New implicit feature for automatic generation of solution key points through examination of load curves for points of particular interest, e.g., times at which maximum load occurs. Set DTMIN < 0 in *CONTROL_IMPLICIT_AUTO to invoke this option.
    - Enhanced the performance of the output to d3* files for *CONTROL_IMPLICIT_EIGENVALUE and *CONTROL_IMPLICIT_MODES. This saves 15% of the wall clock time for large eigenvalue problems where hundreds of eigenmodes are written to d3eigv.
    - Corrected the use of load curve specifications for the following implicit control variables:

        - IAUTO < 0 in *CONTROL_IMPLICIT_AUTOMATIC_DYN,_SPR
        - IMASS < 0 in *CONTROL_IMPLICIT_DYNAMICS_DYN,_SPR
        - ISTIF < 0 in *CONTROL_IMPLICIT_SOLUTION_DYN,_SPR

    - *CONTROL_IMPLICIT_INERTIA_RELIEF:

- Corrected the processing of some kinds of contact during the implicit inertia relief computation.
- Disabled use of iterative linear equation solvers in *CONTROL_IMPLICIT_SOLVER when *CONSTRAINED_IMPLICIT_INERTIA_RELIEF is active.
- Needed to account of inertia relief modes in MPP for the computation of resulting forces from constraints.
- Enhance explicit's use of *CONTROL_IMPLICIT_INERTIA_RELIEF by adding a second orthogonalization step for the accelerations.

◦ *CONTROL_IMPLICIT_MODES:

- Fixed application of implicit constraint modes in MPP for the case where a constraint mode is also a shared node.
- Added a switch on *CONTROL_IMPLICIT_MODES to not write d3mode file when generating superelements.
- Improved the creation of superelement files in MPP.
- Correct dynamic memory allocation issue that was causing problems with computing the reduced stiffness matrix in MPP for implicit modes.
- Corrected output to d3mode in MPP.
- Added error checking on nodes for *CONTROL_IMPLICIT_MODES to make sure they are deformable.

◦ *CONTROL_IMPLICIT_MODAL_DYNAMICS:

- Allow the use of shell formulation 18 in modal dynamics when reading from d3eigv.
- Correct a typo in the computation of the explicit time step for modal superposition. Also applies to *CONTROL_IMPLICIT_DYNAMICS with imass = 2 or 3.
- Adjusted reading of d3eigv for modal dynamics for the case where the modes are computed and then immediately used -- all with stresses included.
- Correct the implicit modal dynamics stress output for beams using non-default value for BEAMIP.

◦ *CONTROL_IMPLICIT_EIGENVALUE:

- Freed allocated memory that was not free at the end of subroutine im_mode_stress.
- Adjusted the logic for intermittent eigenvalue computation to deal with explicit not quite getting to the end time.
- Properly handle the output of 10 noded tets to the d3eigv database.
- Reduce size of d3eigv by not adding rotational dofs for solid element only models.

- The computation of the i/o address for writing the d3eigv database in MPP for models with 8 noded shell elements was wrong. The d3eigv database would probably fill the i/o system causing an abnormal termiation. The i/o address computation is now correct.
- Enable the computation of Campbell diagrams for Implicit Rotational Dynamics as part of the Intermittent Eigenvalue capabilities.
- Enable the use of intermittent eigenvalues during both dynamic relaxation and regular transient phase.
- For eigenvalue analyses where there are lots and lots of zero eigenmodes where the eigensolver fails we will now output the computed modes to the d3eigv database for model debugging purposes.
- Extend the cost logic for Lanczos Eigensolvr, both SMP and MPP, to be more aggressive about staying with the current shift. Improves performance for large problems as the cost function is not necessarily monotonic.
- Added shift and scaling to the standard eigenvalue problem to make the solution methodology more robust. This is the same shift and scaling as used for MCMS.
- Corrected output to d3eigv for implicit MPP eigensolver for large number of integration points.
- Add logic so that for some frequency response computations the amount of drilling rotation control is the same as for eigenvalue computations.
- Added Sectoral Symmetry to *CONTROL_IMPLICIT_EIGENVALUE as an implicit eigensolver option for SMP DP. Sectoral Symmetry is an eigensolver for models with a large degree of rotational symmetry such as fan blades.
- Adjust zero shift for Lanczos eigensolver.
- Fix up MPP storage issues in MPP implementation of Lanczos. Increased minimum storage of eigenvalues from 500 to 2500. Restricted eigenvector checking due to MPI buffer length.
- Add additional tag at end of d3eigv for MCMS that was missing. Normalize MCMS computed eigenvectors to have unit norm.

- Correct joint constraint processing in implicit mechanics which should improve the reporting of joint forces (*DATABASE_JNTFORC) in implicit.
- Enhance implicit's processing of nodal mass matrices to include the local coordinate transformation.
- *CONTROL_IMPLICIT_SOLUTION with NSOLVR = 1: Fixed problem in which Implicit Linear Analysis used the forces at the end of the linear step to compute resultant forces. This did not match expectations of the users. Computation of resultant forces, such as those in bndout, were changed to use the force at the beginning of the linear step.
- *CONTROL_IMPLICIT_SOLVER:

- Correct the removal of ordering reuse logic for the MPP implementation of implicit mechanics. This is an important time saving feature for very large models with multiple time steps.
- For intermittent eigenvalue analysis the direct solvers used by the eigenvalue analysis were forced for the entire execution. We now save and restore the solver option selected by the user to use during the non-eigenvalue execution of the run. This enables the use of iterative solvers iterative solvers.
- Fix the implicit logic to properly deal with NEGEV = 1 in *CONTROL_IMPLICIT_SOLVER and IAUTO.ne.0 in *CONTROL_IMPLICIT_AUTO.
- Disabled the use of ORDER = 1 in *CONTROL_IMPLICIT_SOLVER for MPP as this is a really bad choice in MPP.
- Adjusted output to d3hsp for implicit linear equation solver options to match keyword manual. (*CONTROL_IMPLICIT_SOLVER)
- Adjust implicit logic for dumping matrices (MTXDMP in *CONTROL_IMPLICIT_SOLVER). Correct the dumping of matrices from implicit when MTXDMP > 1.

- *CONTROL_IMPLICIT_STATIC_CONDENSATION: Correct indexing of array involved in implicit static condensation.
- Echo the input for *CONTROL_IMPLICIT_RESIDUAL_VECTOR to d3hsp.
- Fix an issue of the implicit linear algebra filenames being inconsistent between Linux and Windows.
- Add an implicit-only option for processing *CONSTRAINED_INTERPOLATION which is referred to as force-geometry-mass (fgm). Forces are moved from the dependent node to the independent nodes, geometry of the dependent node is computed from the independent nodes, and the mass on the dependent node is moved to the independent nodes. All without using a constraint matrix or LaGrange Multiplier approach.
- No longer automatically increase nodal DOF from 3 to 6 due to the presence of *CONSTRAINED_INTERPOLATION. This is unnecessary for solid element models and increases the cost of large solid element models for implicit by 15% due to huge increase in constraint processing to constrain out all of the rotational dofs.
- Correct implicit's handling of SPC constraints on shared nodes in MPP that are also involved in tied contact.
- Enhance Implicit constraint processing to accept function definitions for motor joint movement (*CONSTRAINED_JOINT_***_MOTOR).
- *CONTROL_IMPLICIT_SOLUTION: Added NSOLVR = -1 so that the initial geometry is used for each step of a multistep linear analysis.
- Added implicit time stamps and additional performance tracking for MPP Lanczos eigensolver. Extended command Line Option impcon = 1.
- Added a command line directive of impcon = 1 which tracks events during implicit simulation to analyze performance for large implicit jobs. This is

also a useful debugging tool as it helps identify the particular part of implicit is being executed at an abnormal termination.

- Single precision executables are now disallowed for implicit mechanics (both MPP and SMP).
- Restored final memory report for implicit SMP runs. Logic was based on old memory allocation scheme instead of new dynamic memory allocation.
- Turned on Mumps, a new linear solver option for implicit mechanics, to be included for double precision SMP on XEON64 and AMD64 systems.
- Corrected an inefficiency for implicit when there are many many AutoSPC constraints in MPP. A linear search through the AutoSPC constraints was replaced with a binary search.
- Corrected the collection of resultant forces when using implicit mechanics for tied contact using _OFFSET. Other tied contact such as _CONSTRAINED_OFFSET were OK.
- Added SMP implementation of new keyword *CONTROL_IMPLICIT_RESIDUAL_VECTORS.
- Apply logic that was in R9 and R10 but had not made it into R11 and R12 to check for incorrect values in nodal inertia array for implicit dynamics and eigenvalue analysis.
- Fixed divide by zero in power iteration for buckling problems with inertia relief. (*CONTROL_IMPLICIT_BUCKLING, *CONTROL_IMPLICIT_INERTIA_RELIEF)
- Correct problem where solution control was inappropriately returned to implicit after issuing sense switch sw1.
- Added *CONTROL_IMPLICIT_RESIDUAL_VECTOR. Purpose: Activate and control the computations of residual vectors. Residual vectors are the linear response of a model to a specified load which is then orthogonalized to a set of eigenmodes and any previously computed residual vectors. The eigenmodes can either be computed during the execution or read from an input file. The computation of residual vectors is the same as multi-step linear (NSOLVR = -1 on *CONTROL_IMPLICIT_SOLUTION) but has the additional step of orthogonalization.
- Add a stiffness option to *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS. The stiffness terms are suppressed to keep the stiffness matrix symmetric but the force terms are added.
- *MAT_ELASTIC_FLUID/*MAT_001_FLUID usage with *CONTROL_IMPLICIT_EIGENVALUE: Warning added to highlight the absence of enforcement of irrotationality in solid elements that use this fluid material model. In eigenvalue analyses this absence will often lead to the prediction of numerous spurious rotational fluid modes. This material model should therefore be used with caution in modal analyses.
- Fixed implicit element stiffness of shell elements when used with laminated shell theory, and when also using anisotropic materials *MAT_022, *MAT_034, *MAT_040, *MAT_054, *MAT_055, or *MAT_058. In certain cases, the shear stiffness was too small.

- Lagrangian multiplier joints, LMF = 1 on *CONTROL_RIGID, are supported for implicit analysis.
- Fixed bug with d3eigv when using TET10S8 = 1 on *CONTROL_OUTPUT.

- ***INITIAL**

  - *INITIAL_LAG_MAPPING:

    - Thick shell data from a previous run can be mapped to a new thick shell mesh.
    - *INITIAL_LAG_MAPPING with NELANGL = -3: For the Lagrangian mapping, project the boundary nodes of the current mesh onto the boundary faces of the previous mesh.
    - In a solid-to-solid mapping, map the strain array output by STRFLG = 1.
    - Map the nodal pressures for ELFORM = 13 tetrahedra.
    - If NELANGL = -2 in a 3D to 3D mapping, the data from the previous run are rotated around the axis of an angle ANGLE before mapping.
    - *INITIAL_LAG_MAPPING_WRITE3DAXI: For a 3D axisymmetric Lagrangian model, add the option WRITE3DAXI to *INITIAL_LAG_MAPPING to output the mapping file as if it came from a 2D axisymmetric model.

  - Fix bug in the *INITIAL_STRESS_TSHELL module of the dynain output file. For Thick Shells with 6-10 integration points through the thickness, the z-coordinates of the integration points were incorrect.
  - Added new option ISTIFF to *INITIAL_STRESS_SECTION for enhanced stability.
  - Added new ISTIFF option for artificial stiffness in *INITIAL_STRESS_SECTION.
  - Fix *INITIAL_STRESS_TSHELL output to dynain for thick shell form 7. Output stresses for all 4 in-plane integration points instead of just 1.
  - Fix incorrect rotational axis when using *PART_INERTIA and *INITIAL_VELOCITY_GENERATION with IRIGID = 1 and the rotational axis is defined using nodes by setting NX = -999 and defining the nodes with NY and NZ.
  - Make *MAT_GENERAL_NONLINEAR_1DOF_DISCRETE_BEAM / *MAT_121 work with *INITIAL_STRESS_BEAM.
  - Disable *INITIAL_AXIAL_FORCE_BEAM and *INITIAL_STRESS_SECTION when IDRFLG = -999 in *CONTROL_DYNAMIC_RELAXATION.
  - Fix seg fault or incorrect stresses when initializing stresses using *INITIAL_STRESS_SOLID with *MAT_107/*MAT_MODIFIED_JOHNSON_COOK.
  - Fixed a bug that made *INITIAL_VELOCITY_GENERATION_START_TIME not work for rigid parts.

- STRFLG in *DATABASE_EXTENT_BINARY will be automatically set to unity when using *INITIAL_STRAIN_SOLID.
- Fixed the writing of dynain files that use beam element formulations 7 or 8. They were writing the wrong number of integration points.
- Fixed a problem that occurred when when initializing stress in thick shells using a dynain file created by a previous run that used a different number of integration points per layer than the new run. In this case, when reading the dynain data to initialize the element, the stress tensor was being properly averaged or extrapolated as needed, but the material extra history variables were not, which for some materials caused wrong material behavior.
- Fixed the reading of initial strain data for thick shell formulations 2, 3, 5, 6, and 7. A strain transformation was missing.
- Add new keyword *INITIAL_HISTORY_NODE(_SET) to initialize select history variables at node locations. These nodal values are internally interpolated to the element integration points using the finite element shape functions. A main difference to using *INTIAL_STRESS_SHELL/SOLID, where you would need to initalize ALL history variables up the ones you really need/want to initialize is that *INITIAL_HISTORY_NODE(_SET) allows you to just pick the few variables you would like to initialize without touching the others.
- *INITIAL_AXIAL_FORCE_BEAM now supported for beam formulation 1 (H-L beam) and any material. Previously, only beam formulation 9 with *MAT_SPOTWELD was allowed.
- Selective mass scaling is not applicable to beam formulation 1 (Hughes-Liu beam).
- *CONTROL_ADAPTIVE: Users are able to use *INITIAL_VELOCITY_GENERATION for the 3D adaptive parts as long as STYP = 3 (node set) and the corresponding node set is defined using *SET_NODE_GENERAL with OPTION = PART. (This would apply only to the unusual situation in which initial velocities were invoked AFTER the mesh is adapted.)

- **Isogeometric Analysis (IGA)**

  - *ELEMENT_SHELL_NURBS_PATCH:

    - Include initial parameterization check for isogeometric shells.
    - Include automatic removal of trimming loops fully contained within a single element.

  - Updated *DEFINE_SPOTWELD_RUPTURE to work with isogeometric shell elements.
  - Added IGA shell and IGA solid element erosion due to the material failure strain, *MAT_ADD_DAMAGE, and *MAT_ADD_EROSION.

- Contact for IGA in implicit is now available using the automatically generated interpolation elements. The IGACTC option in *CONTROL_CONTACT, however, is not currently supported for implicit.
- Added *INITIAL_VELOCITY_GENERATION support for IGA elements using parts and part sets.
- Added support for checking element stiffness matrices for IGA through the LPRINT option in *CONTROL_IMPLICIT_SOLVER.
- Both IGA solids and IGA shells now work with *MAT_RIGID. Previously, only IGA shells were supported for rigid material.
- IGA now works for implicit springback.
- Added the ISTUPD thickness change options (*CONTROL_SHELL) to ELFORM = 3 IGA shell elements.
- Added drilling stiffness type 4 (see DRCM in *CONTROL_IMPLICIT; DRCPSID and DRCPRM in *CONTROL_SHELL) to the IGA shells.
- Improved the time centering for IGA shells for improved convergence rates in implicit analysis.
- *ELEMENT_SHELL_NURBS_PATCH: Enabled keyword *LOAD_NURBS_SHELL for trimmed elements.
- *ELEMENT_SOLID_NURBS_PATCH:

  - IINT defines the integration rule.
    - EQ.0: reduced Gauss integration rule (default)
    - EQ.1: full Gauss integration rule.
    - EQ.2: patch-wise reduced integration rule.
    - EQ.3: non-uniform Gauss integration rule uses full integration for exterior elements and reduced integration for interior elements.
  - Add conventional mass-scaling for isogeometric solids

- These material models are now supported for shells in isogeometric analysis:

  - *MAT_224/*MAT_TABULATED_JOHNSON_COOK
  - *MAT_054-055/*MAT_ENHANCED_COMPOSITE_DAMAGE

- Laminated shell theory is now supported for shells in isogeometric analysis. This option can be activated for shell formulations that allow for transverse shear deformability by:

  - Setting LAMSHT = 1,3 or 5 in *CONTROL_SHELL, and
  - Using *INTEGRATION_SHELL to specify an integration rule, which is referenced in *SECTION_SHELL.

- *INITIAL_STRAIN_SHELL_NURBS_PATCH: Fixed output for thickness in case ISTUPD = 0 in *CONTROL_SHELL.

- *INITIAL_(STRESS/STRAIN)_SHELL_NURBS_PATCH: Fixed error in writing of *DEFINE_NURBS_CURVE to dynain file (*INTERFACE_SPRINGBACK_LSDYNA).
- Fixed bugs that arose from improper merging of interpolation nodes at patch boundaries. This could have lead to errors such as "Zero Normal Vector".
- Remove the limitation to have NURBS patches and related trimming curves defined in a normalized parameter space as this is not common in CAD-files.
- Fix bug when using IGA-shells (*ELEMENT_SHELL_NURBS_PATCH) together with user defined elements (ELFORM = 100-105).

- ***LOAD**

  - There were unexpected error terminations when *LOAD_THERMAL_LOAD_CURVE was used in an adaptive analysis. This has been fixed.
  - *LOAD_THERMAL_D3PLOT reads temperature data from a file in d3plot format. The file name is specified as an option on the command line. If this file does not exist in the working directory, LS-DYNA now gives a reasonable message and exits gracefully.
  - *LOAD_BODY_GENERALIZED: fix a bug that happens when CID > 0 and the CID doesn't have its origin at (0,0,0).
  - Fix incorrect results when using *DEFINE_CURVE_FUNCTION with AX/AY/AZ for *LOAD_SEGMENT.
  - Fix output to nodfor when using *LOAD_SSA.
  - Fix ineffective *LOAD_THERMAL_CONSTANT when used with *MAT_ELASTIC_VISCOPLASTIC_THERMAL/*MAT_106.
  - Fix seg fault when using *LOAD_MOVING_PRESSURE due to uninitialized variable.
  - Fixed a bug preventing *LOAD_BLAST_ENHANCED from working for 2D analysis.
  - Added line-of-sight feature (LSFLG) to *LOAD_MOVING_PRESSURE: When LSFLG.EQ.1, the pressure will be applied to the first-hit segments from the nozzle.
  - When IDIR = 3 in *LOAD_MOVING_PRESSURE, the pressure is in the direction from NODE1 to NODE2, but only the normal component of the pressure is applied on the segments.
  - Enabled *LOAD_THERMAL to work for beam nodes when nodal releases are defined at those nodes. The thermal loading was previously incorrect.
  - New keyword *LOAD_THERMAL_RSW: Prescribes nodal temperatures within a (possibly moving) ellipsoidal region of the structure in mechanical-only simulations. So there is no heat transfer available to the surroundings. Input is comparable to thermal boundary condition *BOUNDARY_TEMPERATURE_RSW. Features include:

        - Temperatures for the center and the boundary of the ellipsoid have to be input. Inbetween, there is a quadratic approximation.
        - Outside of the ellipsoid, no temperature values are prescribed.
        - Position and axis of symmetry are defined by two nodes
        - Optional definition of a heat affected zone (HAZ) around the weld nugget. Temperatures at its boundary are to be given and a linear interpolation from the boundary of the nugget to the boundary of the HAZ is used. Outside the HAZ a default temperature is applied.

    - Implement *LOAD_SPCFORC in MPP.
    - Reduce memory usage and computation time for *LOAD_SEISMIC_SSI_-NODE when ground motions are specified at every node on the soil-structure interface.
    - New keyword *LOAD_EXPANSION_PRESSURE applies a uniform pressure (e.g., from explosive detonation) on the internal surfaces of a chamber while one wall of the chamber moves (as a piston) so that the loaded area increases.

- ***MAT and *EOS**

    - Added *MAT_295/*MAT_ANISOTROPIC_HYPERELASTIC.

        - This is a collection of (nearly-in)compressible, (an)isotropic, hyperelastic material models primarily aimed at describing the mechanical behavior of biological soft tissues. Some of the material models may also be used to analyze a wider class of materials including fiber-reinforced elastomers and stretchable fabrics.
        - Modules include isotropic, anisotropic, and active anisotropic.
        - Allows for electrophysiology-fluid-structure interaction (EFSI).
        - Available for solid element formulations 1, 2, and 10.

    - *MAT_079/*MAT_HYSTERETIC_SOIL: New option for cyclic degradation. See input fields SIGTH, SIGR and CHI on Card 5.
    - *MAT_119/*MAT_GENERAL_NONLINEAR_6DOF_DISCRETE_BEAM: New failure option FCRIT (Field 7, Card 6, see also Remark 1). The pre-existing failure algorithm uses two separate criteria: one that considers only negative displacements and rotations of all six degrees of freedom, and a second criterion considering only positive displacements and rotations. Thus, positive x-displacement and positive y-displacement could combine to cause failure while positive x-displacement and negative y-displacement could not. The new option FCRIT = 1 offers a single criterion that considers both positive and negative displacements/rotations.
    - *MAT_169/*MAT_ARUP_ADHESIVE bugfix: The "interfacial failure" option (TMAXE, SMAXE) was not propagating from element to element as it should.

- Fixed bugs in *MAT_172/*MAT_CONCRETE_EC shear capacity calculation (TYPESC > 0):
  - This capability did not work with fully-integrated shell elements such as ELFORM = 16.
  - Incorrect results or error terminations could occur if material types other than 172 were mixed with MAT_172 in PART_COMPOSITE and the MAT_172 definition had TYPESC > 0.
- *MAT_197/*MAT_SEISMIC_ISOLATOR: New option for sliding isolator with rim failure (TYPE = 5).
- *MAT_203/*MAT_HYSTERETIC_REINFORCEMENT: Fixed bug in MAT_203 affecting beam elements with EPDAM1, EPDAM2 non-zero, and DRESID equal to zero. When the beam reached its erosion limit, error termination or unexpected erosion of neighboring elements could occur.
- *MAT_209/*MAT_HYSTERETIC_BEAM: New material model for modelling buildings in earthquakes. A significantly extended version of *MAT_191 (*MAT_SEISMIC_BEAM) with lumped plastic hinges at both ends of the beam and many additional capabilities.
- Add heat generation factor QCURE to *MAT_ADHESIVE_CURING_VISCOELASTIC/*MAT_277 which relates the heat generated in one time step with the increment of the degree of cure in that step.
- *MAT_296/*MAT_ANAND_VISCOPLASTICITY:
  - Correct the deformation resistance S calculation.
  - Fix floating point exception related bug when compiling with FPE option active.
- Add keyword *MAT_ADD_CHEM_SHRINKAGE.
  - The inputs and features are same as those of *MAT_ADD_THERMAL_EXPANSION except for the usage of curve LCID.
  - The physical chemical shrinkage rate is calculated as

    $$\beta = (1.0 - \alpha)^3 - 1.0$$

    where curve LCID defines alpha (ordinate) vs. temperature (abscissa).
  - When defining curve LCID in *DEFINE_CURVE, the variable DATTYP should be set to -100.
- Extension of THERML variable in *MAT_077_H to allow C11, C20, C02, and C30 to also be specified as a function of temperature.
- Add calculation of internal energy to *EOS_019/*EOS_MURNAGHAN.
- Make *EOS_019/*EOS_MURNAGHAN compatible with *INCLUDE_TRANSFORM.

- Add support for optional EOS in *MAT_COMPOSITE_FAILURE_SPH_-MODEL/*MAT_059.
- Variable PHEL in *MAT_JOHNSON_HOLMQUIST_CERAMICS/*MAT_-110 can be specified equal to 0.0 by default or by user input. In this case PHEL is calculated internally. This value is now echoed as "phel" in d3hsp.
- *MAT_ADD_PERMEABILITY: allow the permeability to be a function of volumetric strain, plastic strain rate, or pressure.
- *MAT_ADD_PORE_AIR: fix an MPP bug triggered when input format is long = s, write standard keyword input and write long structured input deck.
- *MAT_FABRIC: make strain restoration factor TSRFAC available for linear liner.
- *MAT_GENERAL_NONLINEAR_6DOF_DISCRETE_BEAM/*MAT_119:
  - Allow unloading option of "2" for crushable beam, IFLAG = 2.
  - Add the stiffness matrix to enable implicit for crushable beam.
- *MAT_SEATBELT:
  - Enhance 2D belt by offering choice of element formulations and material properties along transverse direction for 2D belt.
  - Enable 2D belt material to have loading curve, LLCID, defined as a table.
  - Issue error message when the edge nodal set of 2D belt part is not on the edge of the related 2D belt part
- Added isotropic plasticity to *MAT_030/*MAT_SHAPE_MEMORY.
- Added new material model *MAT_291/*MAT_SHAPE_MEMORY_ALLOY for shape memory alloys. This is a micromechanics-inspired constitutive model for shape-memory alloys that accounts for initiation and saturation of phase. This model is based on Kelly, Stebner, Bhattacharya (2016) and is available for solid elements only.
- Added support for curves defining thermal properties vs. mechanical history variables in *MAT_THERMAL_ISOTROPIC_TD_LC.
- Fixed the problem in *MAT_244/*MAT_UHS_STEEL where redundant table look-up operations in the plasticity algorithm increased the computational cost.
- Added history variable (at the 8th position) in *MAT_133/*MAT_BARLET_YLD2000 to allow user input of a effective plastic strain shift which can be used to account for stored energy in sub-structures after heat treatment. This option is only applicable to the case of HARD < 0 (curve/table input of hardening curves).
- Added 3D table option to hardening curve input of *MAT_233/*MAT_CAZACU_BARLAT so that the material model is applicable to simulations of hot/warm forming of materials such as magnesium.

- Fix incorrect initial strains when using *MAT_MOONEY-RIVLIN_RUBBER/*MAT_027 with *INITIAL_FOAM_REFERENCE_GEOMETRY and running in single precision.
- Fix problem of solution hanging when using *MAT_PIECEWISE_LINEAR_PLASTICITY_STOCHASTIC and *DEFINE_HAZ_PROPERTIES (MPP only).
- Fix incorrect behavior for *MAT_GENERAL_JOINT_DISCRETE_BEAM/*MAT_097 when BEGTIM in *CONTROL_START is set > 0.0.
- Fix incorrect stresses for *MAT_ANISOTROPIC_VISCOPLASTIC/*MAT_103 and *MAT_ANISOTROPIC_PLASTIC/*MAT_103P when using BETA for *ELEMENT_SHELL_BETA or BETA in the *MAT input or material integration point angles input on *SECTION_SHELL.
- Fix seg fault when using *MAT_PIECEWISE_LINEAR_PLASTICITY/*MAT_024 for Hughes-Liu beam with implicit analysis.
- Output fitted data to curveplot for *MAT_HYPERELASTIC_RUBBER/MAT_77_H when N > 1.
- Fix effect of FBRT on tensile strength, XT, for *MAT_ENHANCED_COMPOSITE_DAMAGE/*MAT_055. FBRT should only affect the tensile strength upon compressive maxtrix mode failure.
- If FAILTM < 0.0 in *MAT_ADD_EROSION, skip erosion failure evaluation during dynamic relaxation phase.
- The calculation of Q12 for *MAT_SPECIAL_ORTHOTROPIC/MAT_130 and *MAT_RESULTANT_ANISOTROPIC/MAT_170 is incorrect. It should be Q12=(v21p*E11p)/(1-v12p*v21p).
- Fix ineffective curves LCSRA, LCSRB, LCSRC, LCSRAB, LCSRBC, LCSCA in *MAT_MODIFIED_HONEYCOMB/*MAT_126.
- Error terminate with message, INI+484, if LOG_INTERPOLATION is invoked for *MAT_PIECEWISE_LINEAR_PLASTICITY/*MAT_024 and the strain rate for the table is defined by LCSS is negative.
- Fix ineffective user-define failure subroutine (FAIL < 0) for *MAT_MODIFIED_PIECEWISE_LINEAR_PLASTICITY/*MAT_123.
- Fix output to elout for solids using *MAT_PAPER/*MAT_274 and CMPFLG > 0 in *DATABASE_EXTENT_BINARY.
- Allow *MAT_065/*MAT_MODIFIED_ZERILLI_ARMSTRONG to compute the flow stress using a more general form by using power exponent, m, of the effective plastic strain instead of the square root of the effective plastic strain in the case of an FCC metal.
- Fix seg fault when using *MAT_ADD_EROSION for *PART_COMPOSITE when the first layer of the *PART_COMPOSITE is not using *MAT_ADD_EROSION.
- Fix incorrect direction of thermal expansion when using *MAT_NONLINEAR_ORTHOTROPIC / *MAT_040 with *MAT_ADD_THERMAL_EXPANSION.
- Fix contact penetration after element erosion when using *MAT_110 / *MAT_JOHNSON_HOLMQUIST_CERAMICS with eroding contact.

- Fix incorrect stress/strain output to d3plot when using *MAT_023 / *MAT_TEMPERATURE_DEPENDENT_ORTHOTROPIC and running 2D analysis with MAXINT > 3 in *DATABASE_EXTENT_BINARY.
- Fix incorrect stress initialization when using *LOAD_DENSITY_DEPTH in combination with *MAT_005/*MAT_SOIL_AND_FOAM where the material's volumetric strain vs. pressure is input using curve LCID.
- Fix strain rate effects on *MAT_157/*MAT_ANISOTROPIC_ELASTIC_-PLASTIC for implicit static analysis.
- Fix seg fault when using *MAT_278/*MAT_CF_MICROMECHANICS.
- Fixed a bug causing 2WAY = 1 in *MAT_ENHANCED_COMPOSITE_-DAMAGE/*MAT_054 not to be recognized.
- Added rate sensitivity to *MAT_S04/*MAT_SPRING_NONLINEAR_-ELASTIC: LCD can point to a table (*DEFINE_TABLE) which gives loading rates (relative velocity values). There is then defined a corresponding curve of force vs. displacement for each loading rate.
- Added PRESTRAIN keyword option for *MAT_MODIFIED_PIECEWISE_-LINEAR_PLASTICITY/*MAT_123. When PRESTRAIN is used, the variable IPS on Card 5 is used to control the prestrain option. If IPS is set to 1, then the strain tensor defined by *INITIAL_STRAIN_SHELL will be used as prestrain when checking major strain failure, EPSMAJ.
- Added a new option called TTOPT to *MAT_SPOTWELD which is used by solid and solid assembly spot welds. The option controls the behavior of TRUE_T making it possible to revert the TRUE_T behavior back to how it worked in R8 and previously. If TTOPT = 0 or if the TTOPT is not input, then the behavior is unchanged with this update. If TTOPT = 1, then R8 and earlier behavior happens. TTOPT = 2 is like TTOPT = 1, but the average top and bottom values of forces and moments are used for failure making failure invariant with respect to element numbering.
- Added support for user-defined spot weld failure in *MAT_100_DAMAGE-FAILURE with individual solid elements. Previously, only beam elements and solid weld assemblies were supported.
- Enabled *MAT_ADD_THERMAL_EXPANSION to work in thick shell elements when the solutions is mechanical only. Some code was missing causing bad behavior.
- Enabled *MAT_128/*MAT_HEART_TISSUE for thick shell forms 3, 5, and 7. Initialization of the F tensor was missing causing a 1st cycle termination.
- Enabled *MAT_145, *MAT_244, and *MAT_254 to work with thick shell formulations 3, 5, and 7.
- Enabled multiple instances of nonlocal material averaging to be used in MPP. This change affects *MAT_NONLOCAL, and also *MAT_CODAM2, *MAT_GURSON_RCDC, or *MAT_PLASTICITY_WITH_DAMAGE with the RCDC option.
- Enabled the variable EC of *MAT_058/*MAT_LAMINATED_COMPOS-ITE_FABRIC to be used with thick shell formulations 1, 2 and 6. If a positive value is input, then it will be used. f EC is left blank or set to zero or a

negative number, then the minimum of EA and EB will continue to be used. Also, *MAT_058 will now echo to d3hsp EC = 0 when used with thin shells because the EC value is ignored.

- Improved hourglass form 6 when use with *MAT_089/*MAT_PLASTICITY_POLYMER. A poor choice of the material parameter used to scale the hourglass stiffness made it difficult to use.
- Fixed spot weld assemblies that use user defined failure option 12 in *MAT_100. An error was causing OPT = 12 to not work at all.
- Fixed the use of *MAT_ADD_DAMAGE_DIEM with *MAT_024 and tetrahedral element formulation 13. The damage calculation used bad data leading to a wrong result.
- Fixed the initialization of the material direction for orthotropic materials when used with AOPT = 2 in 2D plane strain, plane stress, and axisymmetric elements.
- Fixed implicit beam spot weld elements (*SECTION_BEAM ELFORM = 9) when used with NF > 0 on *MAT_100/*MAT_SPOTWELD. The averaging of the stress was causing the model to converge to the wrong solution with beam forces too low.
- *MAT_296/*MAT_ANAND_VISCOPLASTICITY: Solved a problem that caused discrepancies between material parameters provided in the input keyword and the values reported in the d3hsp file. Analysis results were not impacted.
- *MAT_097/*MAT_GENERAL_JOINT_DISCRETE_BEAM: Solved a problem that may have caused analysis termination due to fatal error when a simulation switched from implicit to explicit.
- *MAT_261/*MAT_LAMINATED_FRACTURE_DAIMLER_PINHO: Solved a problem that may have prevented accumulation of back stress for in-plane shear plasticity model.
- *MAT_LAMINATED_FRACTURE_DAIMLER_CAMANHO/*MAT_262:

    - Fixed bug in in-plane shear behavior for tshells if curves are used to define strain rate dependency of SIGY and ETAN.
    - Corrected warning messages in case of tshell formulations 1/2 and adjusted the computation of element length for fracture toughness to match with thin shells.
    - Added flag MSG in *MAT_262 to control the writing of warning messages.
    - Added transverse shear damage similar to *MAT_054/*MAT_058.
    - Added shell-only option to use nonlinear (elastic) stress-strain curves instead of constant moduli (EA, EB, GAB). This allows asymmetric tension-compression.
    - Added flag DSF that controls the failure of an integration point based on in-plane shear failure. This was added to be able to reinstate the in-plane shear failure behavior prior to revision 117876 (DSF = 1).

- Fixed incorrect behavior of *MAT_262 when first layer in *PART_-COMPOSITE used a non-262 material model (shells only).
- Fixed bug when using table (2D) for defining strain rate dependent fracture toughnesses values with respect to characteristic element length.

◦ *MAT_LAMINATED_COMPOSITE_FABRIC/*MAT_058:

- Fixed an error in calculation of the initial time step when curves define the stiffness values for EA, EB, GAB (shells only).
- Add the SOLID option to support *MAT_058 for solid elements.
- Fixed bug when using curve LCEFS to define ERODS failure strain as a function of strain rate (shells only).
- Add option to define direction-dependent failure strains. For this, define a curve LCDFAIL that has 5 (for shells) or 9 (for solids) data points and has LCINT set to the number of data points. The ordinate values define the failure strains in the different directions as follows: 1: ef_11t ; 2: ef_11c ; 3: ef_22t ; 4: ef_22c ; 5: ef_12 ; (and additionally for solids) 6: ef_33t ; 7: ef_33c ; 8: ef_23 ; 9: ef_31, The curve's abscissa values must be ascending to meet input requirements but are otherwise ignored.

◦ *MAT_ANISOTROPIC_ELASTIC_PLASTIC/*MAT_157: Output the history variable that tracks failure (according to the Tsai-Wu or Tsai-Hill criterion) as history variable #10.
◦ *MAT_ENHANCED_COMPOSITE_DAMAGE/*MAT_054: Correct the computation of transverse shear strains for solids and tshells when EPSR and EPSF are defined.
◦ *MAT_VISCOPLASTIC_MIXED_HARDENING/*MAT_225: Fix issues when a table to account for strain-rate dependency is used together with kinematic hardening (BETA < 1).
◦ Add support of IORTHO = 1 for cohesive user-define materials.
◦ *MAT_106/*MAT_ELASTIC_VISCOPLASTIC_THERMAL: Added possibility to specify up to 8 user-defined history variables for output. The values of these history variables are computed via *DEFINE_FUNCTION. These values can, for example, be used to evaluate the hardness of the material.
◦ *MAT_254/*MAT_GENERALIZED_PHASE_CHANGE:

- New transformation laws (PTLAW = 5,6,7,8,9) which are tailored for titanium (Ti-6Al-4V).
  - EQ.5: Recovery of alpha martensite based on JMAK, using Koistinen-Marburger equation to estimate equilibrium concentration. Transformation of recovered phase partially to to-phase/
  - EQ.6: Associated with 5; transformation of remaining recovered phase to this to-phase.

EQ.7: Parabolic growth rate calculating the step-wise dissolution of two or more phases into one other phase. This law calculates the sum of remaining phase concentrations. The from-phase is the first to be dissolved.

EQ.8: Associated with 7; additional phases to be dissolved according to 7. Phases are dissolved one after the other.

EQ.9: Extensions of Koistinen-Marburger; threshold of fr-phase and incomplete transformation within a certain range of temperature rates.

EQ.12: JMAK transformation law (same as 2), but active within the temperature range from start and end temperature, i.e. it is active in heating and cooling.

- Read PTTAB6 for JMAK transformation law to account for accelerated phase transformation due to plastic strain in the material.
- Account for TSF (*CONTROL_THERMAL_SOLVER) in phase kinetics.
- History data only written for phases that are actually defined.
- Added variable POSTV which allows user to specify additional history variables (accumulated thermal strain, accumulated strain tensor, plastic strain tensor, equivalent strain) for output. When the temperature is in the annealing range, these values are reset.
- New input variable for initial plastic strain to initialize a constant nonzero equivalent plastic strain value in the whole part.
- Added enhanced annealing option that controls the reset of plastic strains above a certain critical temperature. The anneal process can be described by a time and temperature dependent approach (JMAK, tabular input, load curve inputs).
- Add a cut-off temperature for the thermal expansion.

◦ *MAT_270/*MAT_CWM:

- New option ANOPT (annealing option) that allows cutting the thermal expansion above a temperature limit (e.g. annealing temperature).
- Added variable POSTV which allows user to specify additional history variables (accumulated thermal strain, accumulated strain tensor, plastic strain tensor, equivalent strain) for output. When the temperature is in the annealing range, these values are reset.

◦ *MAT_277/*MAT_ADHESIVE_CURING_VISCOELASTIC:

- Implemented Arrhenius shift function as alternative to the WLF shift function.
- Heat generation due to curing can be accounted for.

◦ *MAT_278/*MAT_CF_MICROMECHANICS:

  - Added curing induced heating.
  - Fixed the solid formulation to give comparable results as the shell formulations.
  - For the solid formulation, added input for the elastic properties of the fiber contribution in the direction that is orthogonal to the fiber plane.

- Fix for combination of *MAT_ADD_PORE_AIR and *MAT_ADD_EROSION with solid element formulations 1, 10, 43, 60, 61, and 99. History variables could have been mixed up.
- Increase accuracy for rate curves SIGVM < 0 and MXEPS < 0 of *MAT_ADD_EROSION. The original input curve is used instead of the rediscretized one.
- Add new variable P4 for *MAT_ADD_DAMAGE_DIEM to define if transverse shear stresses are included in the stress invariant computations (e.g triaxiality) or not. Useful for shells with corresponding “plane stress” material models, such as *MAT_024_2D, *MAT_036, ...
- Add new option MIDFAIL = 4 to GISSMO (*MAT_ADD_DAMAGE_GISSMO) and "eGISSMO" (*MAT_ADD_GENERALIZED_DAMAGE).
- Add new option LP2BI to *MAT_ADD_DAMAGE_GISSMO (shells only): failure becomes a function of triaxiality and a new bending indicator (adopted from *MAT_258).
- Fixes for GISSMO damage when used together with an equation-of-state (*EOS).
- Take IRATE flag from *CONTROL_IMPLICIT_DYNAMICS into account for GISSMO.
- Add new keyword *MAT_ADD_SOC_EXPANSION for geometric expansion due to State Of Charge from EM solver (currently only available for isotropic, hypoelastic materials with solid element formulations -2, -1, 1, 2, and 10).
- Add option to *MAT_024 with VP = 3, where yield stress can now be a function of plastic strain, strain rate, and up to five history variables that can be set using *INITIAL_HISTORY_NODE. That means LCSS refers to *DEFINE_TABLE_XD up to a level of 7.
- Add *MAT_024's option VP = 3 (filtered total strain rate) for solid elements.
- Enhancement when using log interpolation of strain rate data with VP = 1 in shell elements in *MAT_024, *MAT_123, or *MAT_251: eliminate numerical noise for very low strain rates by internally adding lower limits for strain rate and yield stress.
- Add effective strain as history variable #28 for *MAT_034/*MAT_FABRIC with FORM = 14, 24, or -14.
- Fix problem with internal computation of E0 for *MAT_036/*MAT_3-PARAMETER_BARLAT if hardening rule HR = 2 or HR = 5 is used AND E0 is zero in the input AND E is lessthan zero (curve for Young's modulus).
- Add new options to *MAT_068/*MAT_NONLINEAR_PLASTIC_DISCRETE_BEAM: translational and rotational stiffnesses TK{R,S,T} and

RK{R,S,T} can now be assigned negative values referring to curve IDs for load/moment vs. displacement/twist to get nonlinear elastic behavior.

- Fix for *MAT_105/*MAT_DAMAGE_2 with large curve IDs for LCSS (only solids).
- Make *MAT_133/*MAT_BARLAT_YLD2000 available for solid elements (currently only for explicit analysis and without C-R kinematic hardening).
- Fix for *MAT_169/*MAT_ARUP_ADHESIVE with variables referring to *DEFINE_FUNCTIONs: function ID offsets from *INCLUDE_TRANSFORM (IDFOFF) did not work.
- *MAT_187/*MAT_SAMP-1: instability criterion now available as history variable #28.
- Modifications for *MAT_187L/*MAT_SAMP_LIGHT:
  - log interpolation for table (optional) and no extrapolation
  - improved tolerance for plane stress iteration procedure
  - add viscoelasticity (defined by LCEMOD and BETA)
  - add flag to control curve treatment, either using discretized (0, default) or original (1).
- Fix thermal expansion in *MAT_188/*MAT_THERMO_ELASTO_VISCOPLASTIC_CREEP for solid elements: double precision version could show no expansion at all.
- Enable *MAT_199/*MAT_BARLAT_YLD2004 to be used with shell element formulations 25,26,27 and tshell element formulations 3,5,7. Set default values (1.0) for the transformation matrix coefficients.
- Add option to *MAT_224/*MAT_TABULATED_JOHNSON_COOK: variable BETA < 0 can now refer to a *DEFINE_TABLE_4D for triaxiality (TABLE_4D), temperature (TABLE_3D), strain rate (TABLE) and plastic strain (CURVE) dependence.
- Allow *MAT_224 to be used with Lode parameter dependent tables even for shell elements, i.e., in cases where LCF refers to *DEFINE_TABLE and/or LCI refers to *DEFINE_TABLE_3D.
- Allow *MAT_224 implicit to be used with EFG and MEFEM (ELFORMs 41, 42, 43, 45; see *SECTION_SOLID).
- Change behavior of NUMINT > 1 for *MAT_224 if used with multi-integration point solid elements. Integration points that reach damage = 1 still update stresses and therefore carry load. Only if NUMINT integration points reach damage = 1, the element gets eroded. The old way (single IPs got zero stresses before the element was deleted) just led to severe deformations and instabilities.
- Fix floating invalid problem in *MAT_224 (solids and LCG > 0): plastic strain rate could have become negative in rare cases.
- Make *MAT_251/*MAT_TAILORED_PROPERTIES available for solid elements.

- Enable numerical tangent for *MAT_252/*MAT_TOUGHENED_ADHESIVE_POLYMER allowing implicit analysis i^{th} moderate nonlinearities.
- Fixed problem using *MAT_258/*MAT_NON_QUADRATIC_FAILURE and *DAMPING_PART_STIFFNESS together with RYLEN = 2 in *CONTROL_ENERGY.
- Modification for *MAT_280/*MAT_GLASS: Parameter NIPF now also applies to the EPSCR failure criteria, i.e., elements get deleted when NIPF integration points failed due to reaching EPSCR.
- *MAT_ADD_INELASTICITY introduced for low order solids and shells. This keyword can be used to add inelastic behaviors, including plasticity, viscoelasticity, and creep, to any material model whose stress is calculated incrementally.
- *MAT_ADD_DAMAGE_DIEM now supports Hughes-Liu beams, thick shell type 2, and axisymmetric elements.
- All parameters on first two cards of *MAT_ADD_EROSION are supported for Instability on *DEFINE_MATERIAL_HISTORIES.
- *MAT_EXTENDED_3-PARAMETER_BARLAT/*MAT_036E now supports tables to define stress vs. plastic strain and strain rate. Viscoplastic consistency is mandatory.
- Enchanced message SOL+737. “Strain range of stress-strain input for *MAT_031 has been exceeded.”
- In checking for fixed nodes for PML boundary (*MAT_PML_...), correctly merge translational constraints from *NODE and *BOUNDARY_SPC, in case both are used at the same node.
- Calculate internal energy correctly for PML elements. *MAT_PML_NULL may have issues.
- Implement *DAMPING_FREQUENCY_RANGE_DEFORM for solid PML elements.
- Introduce new label “shl2d” for user materials in plane stress elements, distinguishing them from shells.
- Add keyword "*MAT_ADD_CHEM_SHRINKAGE_TITLE", with which the chemical shrinkage effect can be simulated.
- *MAT_242/*MAT_KINEMATIC_HARDENING_BARLAT2000:
  - Define Young's modulus with a curve. If EA is less than zero, the absolute value of EA is the curve ID defining Young's modulus as a function of plastic strain.
  - Add option for modified Yoshida formulation with two variables SC1, SC2 (same as *MAT_125).

- **MPP**
  - When MPP predecomposition is done for more processors than the calculation, and the model contains airbags, the airbag ownership was not being

properly handled leading to problems including possibly segfault. This has been fixed.

- Improvements and fixes for MPP handling of decomposition regions and processor subsets.
- Fix MPP contact initialization issue that was segfaulting in some models if parts were being excluded from the d3plot output.
- Add new pfile option "transform_keyword" to the "decomp" section. Including this new flag will cause the global decomposition transformations to apply to all the decomposition regions created by these keywords:

  *PARTS_DISTRIBUTE

  *PARTSET_DISTRIBUTE

  *ARRANGE_PARTS

  *CONTACT_DISTRIBUTE

  Before r86885, and after that was reverted in r102224, these regions were decomposed without any coordinate transformation at all. Between those versions, the global transformation applied to all these regions. I think this behavior makes sense, but someone complained that r86885 changed behavior, so it was removed in r102224. Then someone ELSE complained that r10224 changed the behavior that THEY wanted, so this option is now added.
- Fix missing initialization that broke d3plot files when MPP predecomposition was used with models having *DEFINE_CURVE_FUNCTIONs that used the PIDCTL function.
- Corrected MPP processing of *PART_MODES option of ANSID .ne. 0. The node list for this option needed to be sorted and logic adjusted so that all processes, even those with no such nodes, are involved in the communication.

- **Output**

  - *DATABASE_TRACER_GENERAL: In this keyword, tracers are nodes that output histories for any solids or beams or shells or thick shells with the tracers in their volumes.
  - *DATABASE_TRACER_GENERAL and *CONTROL_REFINE_SHELL: Implement the tracers in the shell refinement.
  - *DATABASE_TRACER_GENERAL and *CONTROL_REFINE_SOLID: Implement the tracers in the solid refinement.
  - *DATABASE_PROFILE: Implement TYPE = 6 to output the distribution (along a given direction) of thick shell data.
  - *DATABASE_PROFILE: Output strain profiles if STRFLG = 1 in *DATABASE_EXTENT_BINARY.

- MPP fix for frictional work output to intfor (*DATABASE_BINARY_INTFOR) when soft = 2 contact is being used
- "Interface Pressure" output to intfor is now returned to always being positive. There is no clear way to distinguish between compression and tension at this point in the code, so restoring the old behavior seems best.
- Fix issue with reported time step controlling element in glstat.
- Reaction forces for superelements (*ELEMENT_DIRECT_MATRIX_INPUT) using explicit time integration are now available in spcforc.
- SPCs in *CONSTRAINED_NODAL_RIGID_BODY_SPC are now internally converted to equivalent *BOUNDARY_PRESCRIBED_ MOTION_RIGID if SPC2BND is set to 1 on *CONTROL_OUTPUT. This allows output of reactions to bndout.
- New stress extrapolation from gauss points to nodes for shells (similar to the SOLSIG option for solids). Enable by setting the SHLSIG variable to 1 in *CONTROL_OUTPUT. Applies to shell element formulations 16, 20, and 21.
- *DATABASE_RECOVER_NODE:
  - Fix an MPP bug for the nodal stress calculation of 10-node tetrahedron element.
  - Add 3 more nodal stress components to be recovered by using the velocity vector.
  - Add elemental extrapolation method for stress recovery.
- Error terminate with KEY+501 message if different BEAMIP settings are used in duplicate *DATABASE_EXTENT_BINARY.
- *ele: Eliminate order dependency of IDOF in *SECTION_SHELL in which the IDOF in the first shell part became the default for the other shell parts with the same formulation instead of being independent.
- Fix incorrect rigid body velocities output to matsum for axisymmetric analysis with eroding elements.
- Fix incorrect rigidwall forces & stonewall energy reported for *RIGIDWALL_PLANAR_ORTHO.
- Shell reference surface offset NLOC in *SECTION_SHELL will now properly affect moment calculation for output to secforc (*DATABASE_SECFORC).
- Eliminated output of nodes having zero mass to massout (*DATABASE_MASSOUT).
- Fixed a bug that caused inconsistencies in strains output to d3plot and d3part.
- Fixed a bug that caused discrepancies in elout and disbout for discrete beams using *MAT_121/*MAT_GENERAL_NONLINEAR_1DOF_DISCRETE_BEAM.
- Fixed a bug that caused zero forces in nodfor for discrete beams (beam ELFORM 6).

- Fixed a bug that caused incorrect stresses in d3plot and elout for shell ELFORM 23.
- Fixed a bug that caused discrepancy in rigid body velocity output to glstat and matsum.
- Fixed a bug that caused segfault when using LCUR in *DATABASE_DISBOUT.
- Added missing report of triggered failure criterion when solid elements fail due to *MAT_ADD_EROSION.
- Fixed a bug that caused incorrect filtered results in ncforc when using *DATABASE_NCFORC_FILTER in MPP.
- Enabled the change in beam length to be viewed when fringe plotting discrete beams.
- Enabled variable ENGFLG on *DATABASE_EXTENT_BINARY to apply to thick shells as well as thin shells. This allows internal energy density of tshells to be written to d3plot.
- Fixed a thick shell output problem affecting d3plot that occurred when MAXINT on *DATABASE_EXTENT_BINARY was larger than the number of layers in the element. For the nonexistent layers, we were reading data from the next element in memory. Now data for the last existing point is repeated.
- Fixed two issues when reporting eroded hourglass energy to glstat and matsum. For solid elements, the eroded energy was not removed from the regular hourglass energy, so it was counted twice causing a poor energy balance in the glstat data. Also, thick shells were not reporting eroded hourglass energy.
- Fixed bug in which DECOMP = 2 or 4 in *DATABASE_EXTENT_BINARY corrupts d3drlf.
- Fixed bug whereby a simple restart corrupted the d3part database.
- Echo *DATABASE_EXTENT_BINARY_COMP variables to d3hsp.

- **Restarts**

  - *DAMPING_FREQUENCY_RANGE_DEFORM now works smoothly with full deck restarts (*STRESS_INITIALIZATION). Applies to solids, beams, shells, thick shells. Previously, a stress-jump would occur at the beginning of the restart run.
  - Fix conditional so that node rotational masses are properly synchronized when initializing *CONTACT_TIED_..._BEAM_OFFSET during MPP full deck restart.
  - Fix MPP full deck restart issue that caused hanging if the new input file contained encrypted input.
  - Correct the handling of restarting implicit after full restart.
  - Fix bug affecting restarts in implicit.
  - Support the following for full deck restart and consequently, *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION.

- SPH active region with local coordinate system (*DEFINE_SPH_ACTIVE_REGION, ICID).
- SPH injection (*DEFINE_SPH_INJECTION)
- Explicit joints (*CONSTRAINED_JOINT).
- LMF joint formulation (*CONTROL_RIGID, LMF = 1).

◦ Fix seg fault when using *DELETE_CONTACT for restart when running with SMP.
◦ Fix error termination for full deck restart using *DEFINE_ELEMENT_DEATH.
◦ Added new option COMP to *DELETE_PART for simple restart.

*DELETE_PART_COMP

This option eliminates data pertaining to deleted parts from d3plot thereby reducing the size of the d3plot database.

- ***SENSOR**

◦ Limit the usage of *SENSOR to transient analysis only, not dynamic relaxation.
◦ *SENSOR_DEFINE_CALC-MATH: fix a bug triggered when CALC = SUM and any of involved sensors, SENSi, are negative for subtraction.
◦ *SENSOR_DEFINE_ELEMENT: fix a bug triggered when trying to trace the stress/strain of a failed element.
◦ *SENSOR_DEFINE_FORCE: fix an MPP bug for TYPE = CONTACT2D.
◦ *SENSOR_DEFINE_MISC: fix a bug in which incorrect kinetic energy of a rigid body was sensed.
◦ *SENSOR_SWITCH_CALC-LOGIC: allow as many switches as needed for logic calculation.
◦ *SENSOR_CONTROL:

- Fix a bug for TYPE = BAGVENTPOP in which venting is not turned off for *AIRBAG_HYBRID.
- Fix a bug for TYPE = PRESC-MOT triggered when the referenced node or rigid part has a big ID.
- Fix a bug for TYPE = BELTPRET that fails to turn on the pretensioner.
- Fix a bug for TYPE = BELTRETRA that fails to lock the retractor when NEWLEG of *CONTROL_OUTPUT > 0 and no SBTOUT output is requested.
- Add CNRB option to control *CONSTRAINED_NODAL_RIGID_BODY.
- Add TYPE = LOADTHM to control *LOAD_THERMAL_VARIABLE and *LOAD_THERMAL_VARIABLE_NODE
- Add optional time delay.

- Fix order dependency problem when using *DEFORMABLE_TO_RIGID activated by *SENSOR_CONTROL TYPE = "DEF2RIG".

- **SPG (Smooth Particle Galerkin)**

  - Volume of 6-noded solid elements using element formulation 42 (SPG) was calculated incorrectly. That bug is now fixed.
  - Fix SPG bug in MPP whereby the time step size varies with different number of MPP processes.
  - *CONSTRAINED_IMMERSED_IN_SPG: Fix fatal memory issue if the number of immersed nodes is much larger than the number of nodes of the SPG part.
  - Enabled additional material models for SPG application, some of which include orthotropic behavior: *MAT_110, *MAT_122, *MAT_123, *MAT_143, *MAT_199, *MAT_260A, *MAT_269.
  - *SECTION_SOLID_SPG now automatically sets element formulation to 47.
  - If FAIL is set greater than 0 in *MAT_003 or *MAT_024, it is used in the SPG bond failure.
  - Added SPG bond failure criteria corresponding to *MAT_ADD_DAMAGE_GISSMO and *MAT_ADD_EROSION (1st principal stress, max shear strain, 3rd principal strain).

- **SPH (Smooth Particle Hydrodynamics)**

  - Incompressible SPH (FORM = 13) is available in beta form.
  - Add two material models for implicit SPH formulation (FORM = 13):

    - *MAT_300/*MAT_SPH_02/*MAT_SPH_IMPLICIT_FLUID: Input includes surface tension coefficient and surface area minimization coefficient.
    - *MAT_301/*MAT_SPH_03/*MAT_SPH_IMPLICIT_STRUCTURE: Used for boundary particles. Input includes surface adhesion coefficient.

  - Add *DATABASE_SPHMASSFLOW and *DEFINE_SPH_MASSFLOW_PLANE to measure SPH mass flow rate across a defined plane. This feature is to SPH as *DATABASE_DEMASSFLOW is to DEM.
  - Add option to move *DEFINE_SPH_ACTIVE_REGION around by following a moving coordinate system.
  - Add option for buffer zone in *DEFINE_SPH_ACTIVE_REGION.
  - Add an error message when a *DEFINE_BOX_SPH is referenced in *CONTROL_SPH boxid. This should point to a regular *DEFINE_BOX.
  - Add option to reactivate particles, and add an optional buffer zone in *DEFINE_BOX_SPH.
  - *DEFINE_BOX_SPH gets a special treatment with FORM = 13. Particles outside of the activation box get frozen in space, and deactivated particles can

become active when entering the box. This allows calculating only particles surrounding the region of interest.

- Reduce d3plot size for incompressible SPH by trimming down irrelevant variables.
- *DEFINE_ADAPTIVE_SOLID_TO_SPH:
  - Zero out increment of internal energy for eroded particles.
  - Bug fix in SMP: Results were inconsistent in SMP when ncpu was greater than one.
  - Bug fix in stress calculation of embedded SPH particles when ICPL = 1. Internal energy of these particles was also inconsistent.
  - When elements erode, distribute their internal energy to their child particles.
  - Reset IOPT = 0 if ICPL = 0.
  - Fix the MPP 3D strain calculation when adaptive with pure thermal analysis is used. Remove contribution of adaptive particles to strain rate calculation.
  - Properly update density of embedded particles based on element current volume. When particles became active, they used to still carry the rest density.
  - Enforce embedded SPH particles to output zero internal energy, for energy conservation.
- *DEFINE_SPH_INJECTION:
  - Speed-up initial smoothing length calculation when *DEFINE_SPH_-INJECTION is present. It used to be very slow for models containing a large number of injected particles.
  - Add option to define a variable speed of injection. If the injection speed scale factor is defined as negative, it refers to a *DEFINE_-CURVE defining the velocity magnitude vs. time.
  - Use yet-to-be injected particles as ghost particles on the other side of the injection plane.
  - If the coordinate system is updated in time (IFLAG = 1 in *DEFINE_-COORDINATE_NODES), we also update the injection direction.
  - If the injection plane follows a node (NID in *DEFINE_SPH_INJEC-TION), and IFLAG = 1 in *DEFINE_COORDINATE_NODES, we also continously rotate the whole stack of particles to be injected with the plane.
- Added SPH support for *MAT_193/*MAT_DRUCKER_PRAGER.
- Bug fix in psetid option of *DATABASE_BINARY_D3PLOT if psetid contains SPH parts.
- Fix ASCII output of SPH neighbor count, which was always zero.

- Bug fix in SPH neighbor search when variable smoothing length is employed. The neighbor list was created based on minimum smoothing length instead of maximum. As a result, some neighbors were potentially omitted.
- Add option in *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION to remove dead SPH particles from the model at each redecomposition step.
- Improve MPP load balancing when a *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION is present and a moving box defines the activation region of SPH.
- Create MPP variants of 2D plane-strain and 2D axisymetric SPH formulations with thermal coupling.
- Add support for *LOAD_THERMAL_LOAD_CURVE in SPH.
- Fix bug for SPH thermal formulation in MPP. Artificial viscosity calculation was incorrect.
- Fix bug in MPP with ERODING contact and SPH. In some MPP decompositions, some processors were ignoring contact between SPH and Lagrangian parts.
- Restored TEROD and CEROD parameters for *MAT_NULL with SPH particles (these two parameters were not applied from R9).
- Material models added for SPH:

  - *MAT_COMPOSITE_DAMAGE/MAT_022 for SPH plane strain.
  - *MAT_MODIFIED_CRUSHABLE_FOAM/*MAT_163 for SPH axisymmetric, plane strain, and 3D.
  - *MAT_ENHANCED_COMPOSITE/*MAT_054,_055 for SPH axisymmetric and 3D.

- *CONTROL_SPH:

  - Improvements in adaptive smoothed particle hydrodynamics formulations (FORMs 9,10) with an anisotropic kernel. Speed up SMP operationsr; optimize bucket cycle; update principle direction and scale factor.
  - Renamed MEMORY variable to NMNEIGH and added error/warning messages to help clarify meaning for users.

- New kernel functions (SPHKERN in *SECTION_SPH) for 3D SPH.

  - SPHKERN = 1 (quintic spline kernel function) now supports FORMs 5,6. (Previously only supported for FORMs 0,1,9,10.
  - Added SPHKERN = 2 for FORMs 0,1,5,6. This quadratic spline kernel function mainly aims for high velocity impact to relieve the problem of compressive instability.

- Added SPHKERN = 3 for FORMs 0,1,5,6. This quartic smoothing kernel function (G.R.Liu 2003) is similar to the standard cubic spline kernel function but should be more stable and accurate.

- *DEFINE_SPH_INJECTION: (both SMP and MPP):

  - Add injection plane moving with a node NID.
  - Add injection velocity based on a local coordinate system CID.

- Fix inexplicable penetration when using *CONTACT_AUTOMATIC_NODES_TO_SURFACE with SPH elements. This affects SMP only.

- **Thermal Solver**

  - *CONTROL_EXPLICIT_THERMAL_SOLVER and *ELEMENT_TSHELL: Implement heat conduction for thick shells in the explicit thermal solver.
  - *CONTROL_EXPLICIT_THERMAL_SOLVER can use *INITIAL_TEMPERATURE to initialize the element temperatures (it averages the initial nodal temperatures at element centers).
  - *CONTROL_EXPLICIT_THERMAL_INITIAL with ID < 0 allows initialization of temperature by element instead of element set (ID > 0).
  - New SOLVER options in *CONTROL_THERMAL_SOLVER:
    EQ.18: preconditioner based on a local incomplete factorization;
    EQ.19: preconditioner based on a block low-rank factorization of the global problem.
  - *CONTROL_THERMAL_SOLVER:

    - Fixed the echo of thermal solver options to d3hsp.
    - Enhanced the setting of default for GMRES parameters when nonsymmetric iterative was internally chosen over the input value for thermal linear equation solver.
    - Migrated the linear algebra data structures in thermal mechanics to Fortran 95 dynamic storage as part of the storage modernization effort. Reduces requirement of user on setting the memory command line parameter.
    - Declare all old SMP linear equation solvers for thermal as obsolete. Automatically switch to the new solvers with an appropriate warning. This is a step towards modernization of linear algebra for thermal.
    - Added nonsymmetric problem support to the modern linear algebra path in thermal. This is mostly the THERMAL_BULK feature.
    - Turned off support for direct linear equation solvers for thermal in single precision. In single precision change direct solver options to appropriate iterative solvers. Symmetric (11) goes to modified Incomplete Choleski PCG (16) which is the strongest symmetric iterative solver. Nonsymmetric (20) goes to GMRES (17).

- Correct the case where cgtol is specified on *CONTROL_THERMAL_-SOLVER but reltol is not.
- Corrected testing of linear equation solver option for thermal. The addition of GMRES option for conjugate heat transfer problems had to be properly dealt with.

- Turn off all implicit mechanics if a thermal-only solution.
- *CONTROL_THERMAL_SOLVER: Fix a conflict associated with using the nonsymmetric direct solver (SOLVER = 30) for the thermal side of a coupled thermal/mechanical analysis when the mechanical analysis is implicit.
- *BOUNDARY_RADIATION_<option>_VF_READ requires a view factor file. An input trap was added if this view factor file is missing.
- There was a limit on *BOUNDARY_TEMPERATURE_NODE and *INITIAL_TEMPERATURE_NODE keyword appearances. This limit was removed, and the speed of the keyword reader was improved.
- Added new keyword *CONTROL_THERMAL_FORMING for easy setup of hot/warm forming simulations. This keyword sets default values for thermal control keywords so that there is no need to include *CONTROL_SOLUTION, *CONTROL_THERMAL_SOLVER and *CONTROL_THERMAL_TIMESTEP in the input deck unless fine tuning of simulation parameters is needed. This keyword can also be used to set up default thermal parameters for forming contact pairs.
- Added new keyword *BOUNDARY_TEMPERATURE_TRAJECTORY to apply a temperature boundary condition on nodes enclosed in either a cylindrical or rectangular prism volume moving along a trajectory. The center of the volume moves along the trajectory defined by a nodal path at a prescribed velocity. The geometry of the enclosing volume can be time-variant. This keyword applies only to solid elements.
- New keyword *BOUNDARY_FLUX_TRAJECTORY defines a moving flux surface boundary condition along a nodal path. Based on the shape of the heat source and its aiming direction, the projection onto the surface is calculated and used for application of the flux. Additional features include:

  - Propagation to newly exposed surfaces after element erosion.
  - Option to balance the change in projected area by modifying the surface density.
  - Total amount of heat input as resulting from the numerical integration of the surface densities can be enforced to coincide with the input.
  - Various heat distributions are available:
    - Constant distribution within a double elliptic region
    - Gaussian distribution within a double elliptic region
    - User-defined function (*DEFINE_FUNCTION) based on local coordinates, time, and temperature

- *LOAD_HEAT_GENERATION: Added variable REFNODE to define a reference node for the function definition. The current nodal coordinates can be referred to in the function definition as “xref”, “yref”, and “zref”. It allows defining a heat source motion associated with the motion of the reference node.
- Thermal contact for 'edge' contact of composite tshell elements now considers the composite stacking sequence by way of an internally constructed mesh of stacked tshell elements.

- **XFEM (eXtended Finite Element Method)**

  - *BOUNDARY_PRECRACK: Adjusted the location of pre-crack to avoid passing through nodal points.
  - Recoded neighbors list to handle triangular mesh better.
  - Added support of GISSMO damage model for XFEM. Set FAILCR = 0 in *SECTION_SHELL_XFEM to activate GISSMO model.
  - 2D XFEM shell (ELFORM = 52 in *SECTION_SHELL_XFEM) is a fully integrated form and so NIP = 4 is now hardwired.
  - Second order stress update can now be specified for 2D XFEM shell form 52 (OSU = 1 in *CONTROL_ACCURACY).

- **Miscellaneous**

  - *DAMPING_FREQUENCY_RANGE and *DAMPING_FREQUENCY_-RANGE_DEFORM had an error trap related to the time step being too large to compute damping at frequency FHIGH. The trap was unnecessary for *DAMPING_FREQUENCY_RANGE_DEFORM and has been removed. The trap was previously necessary for *DAMPING_FREQUENCY_RANGE to avoid instability, but now the code has been modified to remain unconditionally stable and the error trap has been removed. The code modification will result in some change of results for *DAMPING_FREQUENCY_-RANGE (but not for the _DEFORM option). The difference will be most noticeable when the product of the timestep with FHIGH is large, such as in Implicit calculations.
  - Added new command line option to run only a subset of the cases defined in the input deck via *CASE. As before, if all cases in a model should be run, the single word “CASE” should appear as a command line argument. But if only some cases should be run, those case ids can be listed on the command line like:

    ```
    CASE=17,22,147
    ```

    There should be NO spaces before or after the =, and the list following the = should be a comma delimited list of case ids, with no spaces.
  - LS-DYNA now returns an exit code of 1 to the system in case of non-normal termination. Previously, we always returned 0 (no error) no matter what, which could be misleading if a user's job script made use of the exit code.

- Suppress d3hsp output of any *PARAMETER that was encrypted in the input.
- Fix old bug in *INTERFACE_LINKING that would cause incorrect behavior for displacements scaled via a *DEFINE_FUNCTION if (and only if) the function was defined in terms of 3 variables (spatial displacement dependence only).
- The third variable in *PART_ADAPTIVE_FAILURE now does this:
  EQ.1: disable adaptivity of THIS part after it is split into two pieces
  EQ.2: disable ALL adaptivity after this part is split into two pieces
  The default value is 0, which does nothing.
- Add keyword *DEFINE_DRIFT_REMOVE to restore periodicity to specified curve(s) that physically should exhibit periodicity. For example, accelerometer data collected as a car drives one circuit around a track should exhibit periodicity such that when integrated to get displacement, that displacement is zero. However, noise in the data leads to nonphysical drift. This keyword activates an algorithm that approximately restores periodicity.
- *INTERFACE_LINKING_NODES: Interface nodes that were shared between processes were not properly dealt with for implicit.
- *DEFINE_CURVE_FUNCTION:
  - Fix a bug for SENSOR or SENSORD that may result in erroneous information in CURVOUT.
  - Fix a bug for function PIDCTL that occurred in a thermal-only analysis.
- *PART_DUPLICATE:
  - Fix a bug in which 10-noded tet elements were not duplicated.
  - Add option of BOXID to assure that the transformed configuration falls inside the box
- *SET_PART_TREE: Define a branch in a tree structure. A branch is a part set that can be defined using parts and/or sub-branches. With this keyword, the whole model can be modeled as a hierarchical tree structure.
- *SET_SHELL: allow *SET_SHELL_LIST_GENERATE to generate a set of 2D belts.
- New keyword *DEFINE_QUASAR_COUPLING to enable coupling of LS-DYNA and Cadlm's QUASAR ROM (Reduced Order Model). Supports multiple ROMs attached to LS-DYNA FEM model.
- Madymo coupling:
  - Support TASS Madymo coupling using Intel MPI.
  - Support Madymo 7.7 and above.
- Fixed outdated argument lists for user subroutines. Before this fix user-defined features using *MODULE may not work correctly.

- *DEFINE_PRESSURE_TUBE now supports decomposition of automatically generated solid/shell tubes in MPP.
- Added support for user element history variables in *USER_NONLOCAL_-SEARCH.
- Implement *DEFINE_DRIFT_REMOVE for MPP.
- Fix empty *SET_SEGMENT error, KEY+140, when using *SET_SEGMENT_-GENERAL with OPTION set to SET_SOL-ID/SET_SLDI0/SET_SLDF#/SET_TSHELL/SET_TSHIO.
- Fix seg fault when using *INTERFACE_LINKING_NODE_SET.
- New functions AX2, AY2, and AZ2 for *DEFINE_CURVE_FUNCTION. The difference compared to AX, AY, and AZ is that if n2 = 0, then return component relative to axes fixed in n1.
- New function DIST() in *DEFINE_CURVE_FUNCTION, which is used to calculate the distance traveled by a node
- Add feature to *DEFINE_MATERIAL_HISTORIES, LABEL="History" whereby if the first attribute A1 is negative, its absolute value points to a curve. The first ordinate value of that curve defines an operation to be performed on a list of history variables. The subsequent ordinate values give the history variable numbers in the list. See the User's Manual for details and an example.
- Add keyword *DEFINE_ELEMENT_EROSION_(SHELL/TSHELL) to allow more flexible control of when to actually delete an element.
- Add two new OPTIONs for *DEFINE_TRANSFORMATION.

  - TRANSL2ND: translation given by two nodes and a distance
  - ROTATE3NA: rotation given by three nodes and an angle (two nodes provide rotation axis, going through the third node)

- Add new option DTYPE = 1 to *PERTURBATION_NODE to allow uniform distribution between SCL × [−AMPL, AMPL] for TYPE = 8.
- Add runtime output to "Open include file: ..." instance.
- Removed the echo of encrypted data to d3hsp for the following keywords: *PARAMETER, *DEFINE_FRICTION, *ELEMENT_DISCRETE, *CONTACT, and *DEFINE_CURVE.
- Added three sense switches swb, swc and swd, all of which will create dynain data. *INTERFACE_SPRINGBACK_LSDYNA must be defined in the input deck.

  swb: a dynain is written and the analysis continues.
  swc: a restart and dynain are written and the analysis continues.
  swd: a restart and dynain are written and the analysis terminates.

  The first dynain file created by one of these sense switches is named dynain.1, and the filename is incremented by 1 (dynain.2, dynain.3, etc.) each time one of the sense switches is submitted.
- Replace keyword *SET_SPRING with *SET_DISCRETE.

- Changed the default of plabel to plabel = no for faster input processing. If the input contains non-numeric (character) ID(s), an error message will be issued, in which case the user would need to include "plabel = yes" on the execution line so that the non-numeric ID(s) could be properly read.
- Fix bug in which message announcing loading of an include files appeared twice.
- Fixed bug in madymo coupling in which the restart dump was called twice.
- Improve check for empty or duplicate *SETs and issue appropriate warnings/errors.
- Can now use an environment variable to set the default memory: setenv LSTC_DEFMEM n where n is the memory size
- Can now suppress almost all output by including "benchmark = y" on either the execution line or on the *KEYWORD line in the input deck. It will also remove part, node and element output in d3hsp.
- Fixed bug in reading long format if *KEYWORD long = yes is used in include file.
- A new flag TOL in *INCLUDE_UNITCELL is introduced for users to define the tolerance for identifying the pairs of nodes in the periodic positions.
- New keywords: *COSIM_FMI_CONTROL and *COSIM_FMI_INTERFACE: Adds capability to remotely co-simulate with other software supporting FMI standard.
- Deliver model order reduction (MOR) *CONTROLLER_PLANT for the linear structural and piezo-electric system. The state space matrices are exported based on the modal truncation or Krylov subspace.
- *PART_DUPLICATE: Add a new optional variable ZMIN. Transformed part(s) will have a minimum z-coordinate equal to ZMIN.
- *INCLUDE_STAMPED_PART_SOLID_TO_SOLID: This keyword extends the idea of *INCLUDE_STAMPED_PART to solid elements. Information about a solid part is mapped from one analysis (e.g., stamping) to another analysis (e.g., crash).

    - Maps the stress tensor, effective plastic strain, and strain tensor.
    - Calculates the thickness from the first analysis and modifies the coordinates of the nodes in the second model to conform to that thickness.
    - When PID is a negative number, it means that the top surface of the solid needs to be reversed to match the forming result. Otherwise, the mapping will be wrong.
    - When the element orientation is defined in *ELEMENT_SOLID_ORTHO, the angles are mapped to the new mesh.

- *PART_MOVE:

    - Maximum number of parts moved in one input increased from 400 to 1000.

- When CID is a negative value, it defines a vector id or VID pointing to the direction of the part move and the variable ZMOV is the move distance.

Capabilities added to create LS-DYNA R13:

See release notes located at https://lsdyna.ansys.com/wp-content/uploads/2026/10/Release_Notes_LS-DYNA_R13_0_0_rev2.pdf.

Capabilities added to create LS-DYNA R14:

See release notes located at https://lsdyna.ansys.com/wp-content/uploads/2026/10/Release_Notes_LS-DYNA_R14_0_0_rev1.html.

Capabilities added to create LS-DYNA R15:

See release notes located at https://lsdyna.ansys.com/wp-content/uploads/2026/10/Release_Notes_LS-DYNA_R15_0_2_rev0.html.

Capabilities added to create LS-DYNA R16:

See release notes located at https://lsdyna.ansys.com/wp-content/uploads/2026/10/Release_Notes_LS-DYNA_R16_0_0_rev1.html.

Capabilities added to create LS-DYNA R17:

See release notes located at https://lsdyna.ansys.com/wp-content/uploads/2026/10/Release_Notes_LS-DYNA_R17_0_0_rev0.html.

MATERIAL MODELS

Some of the material models presently implemented are:

- elastic,
- orthotropic elastic,
- kinematic/isotropic plasticity [Krieg and Key 1976],
- thermoelastoplastic [Hallquist 1979],
- soil and crushable/non-crushable foam [Key 1974],
- linear viscoelastic [Key 1974],
- Blatz-Ko rubber [Key 1974],

- high explosive burn,
- hydrodynamic without deviatoric stresses,
- elastoplastic hydrodynamic,
- temperature dependent elastoplastic [Steinberg and Guinan 1978],
- isotropic elastoplastic,
- isotropic elastoplastic with failure,
- soil and crushable foam with failure,
- Johnson/Cook plasticity model [Johnson and Cook 1983],
- pseudo TENSOR geological model [Sackett 1987],
- elastoplastic with fracture,
- power law isotropic plasticity,
- strain rate dependent plasticity,
- rigid,
- thermal orthotropic,
- composite damage model [Chang and Chang 1987a 1987b],
- thermal orthotropic with 12 curves,
- piecewise linear isotropic plasticity,
- inviscid, two invariant geologic cap [Sandler and Rubin 1979, Simo et al, 1988a
- 1988b],
- orthotropic crushable model,
- Mooney-Rivlin rubber,
- resultant plasticity,
- force limited resultant formulation,
- closed form update shell plasticity,
- Frazer-Nash rubber model,
- laminated glass model,
- fabric,
- unified creep plasticity,
- temperature and rate dependent plasticity,
- elastic with viscosity,
- anisotropic plasticity,
- user defined,

- crushable cellular foams [Neilsen, Morgan, and Krieg 1987],
- urethane foam model with hysteresis,

and some more foam and rubber models, as well as many materials models for springs and dampers. The hydrodynamic material models determine only the deviatoric stresses. Pressure is determined by one of ten equations of state including:

- linear polynomial [Woodruff 1973],
- JWL high explosive [Dobratz 1981],
- Sack "Tuesday" high explosive [Woodruff 1973],
- Gruneisen [Woodruff 1973],
- ratio of polynomials [Woodruff 1973],
- linear polynomial with energy deposition,
- ignition and growth of reaction in HE [Lee and Tarver 1980, Cochran and Chan 1979],
- tabulated compaction,
- tabulated,
- TENSOR pore collapse [Burton et al. 1982].

The ignition and growth EOS was adapted from KOVEC [Woodruff 1973]; the other subroutines, programmed by the authors, are based in part on the cited references and are nearly 100 percent vectorized. The forms of the first five equations of state are also given in the KOVEC user's manual and are retained in this manual. The high explosive programmed burn model is described by Giroux [Simo et al. 1988].

The orthotropic elastic and the rubber material subroutines use Green-St. Venant strains to compute second Piola-Kirchhoff stresses, which transform to Cauchy stresses. The Jaumann stress rate formulation is used with all other materials with the exception of one plasticity model which uses the Green-Naghdi rate.

SPATIAL DISCRETIZATION

The elements shown in Figure 1-1 are presently available. Currently springs, dampers, beams, membranes, shells, bricks, thick shells and seatbelt elements are included.

The first shell element in DYNA3D was that of Hughes and Liu [Hughes and Liu 1981a, 1981b, 1981c], implemented as described in [Hallquist et al. 1985, Hallquist and Benson 1986]. This element [designated as HL] was selected from among a substantial body of shell element literature because the element formulation has several desirable qualities:

- It is incrementally objective (rigid body rotations do not generate strains), allowing for the treatment of finite strains that occur in many practical applications.

- It is compatible with brick elements, because the element is based on a degenerated brick element formulation. This compatibility allows many of the efficient and effective techniques developed for the DYNA3D brick elements to be used with this shell element;
- It includes finite transverse shear strains;
- A through-the-thickness thinning option (see [Hughes and Carnoy 1981]) is also available.

All shells in our current LS-DYNA code must satisfy these desirable traits to at least some extent to be useful in metalforming and crash simulations.

The major disadvantage of the HL element turned out to be cost related and, for this reason, within a year of its implementation we looked at the Belytschko-Tsay [BT] shell [Belytschko and Tsay 1981, 1983, 1984] as a more cost effective, but possibly less accurate alternative. In the BT shell the geometry of the shell is assumed to be perfectly flat, the local coordinate system originates at the first node of the connectivity, and the co-rotational stress update does not use the costly Jaumann stress rotation. With these and other simplifications, a very cost effective shell was derived which today has become perhaps the most widely used shell elements in both metalforming and crash applications. Results generated by the BT shell usually compare favorably with those of the more costly HL shell. Triangular shell elements are implemented, based on work by Belytschko and co-workers [Belytschko and Marchertas 1974, Bazeley et al. 1965, Belytschko et al. 1984], and are frequently used since collapsed quadrilateral shell elements tend to lock and give very bad results. LS-DYNA automatically treats collapsed quadrilateral shell elements as C^0 triangular elements.

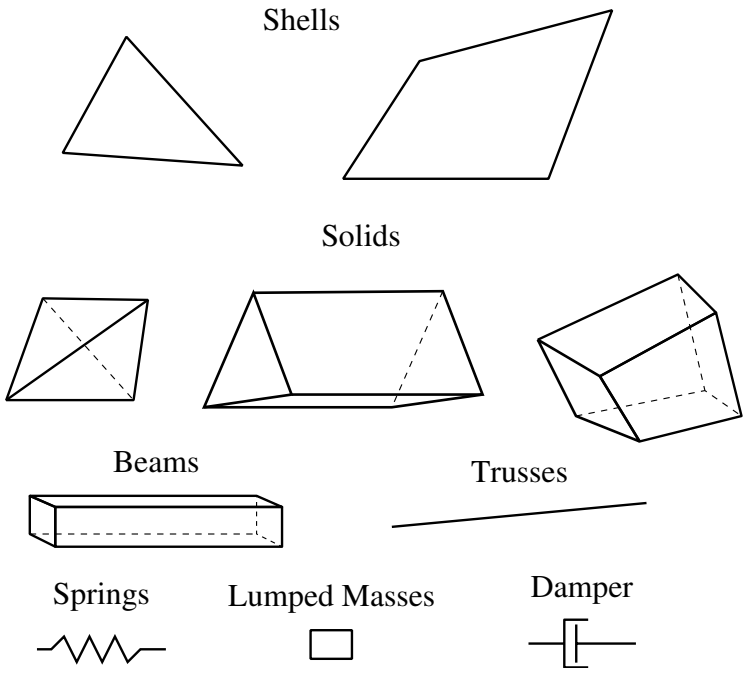


Figure 1-1. Elements in LS-DYNA. Three-dimensional plane stress constitutive subroutines are implemented for the shell elements which iteratively update the stress tensor such that the stress component normal to the shell midsurface is zero. An iterative update is necessary to accurately determine the normal strain component which is necessary to predict thinning. One constitutive evaluation is made for each integration point through the shell thickness.

Since the Belytschko-Tsay element is based on a perfectly flat geometry, warpage is not considered. Although this generally poses no major difficulties and provides for an efficient element, incorrect results in the twisted beam problem and similar situations are obtained where the nodal points of the elements used in the discretization are not coplanar. The Hughes-Liu shell element considers non-planar geometries and gives good results on the twisted beam. The effect of neglecting warpage in a typical application cannot be predicted beforehand and may lead to less than accurate results, but the latter is only speculation and is difficult to verify in practice. Obviously, it would be better to use shells that consider warpage if the added costs are reasonable and if this unknown effect is eliminated. Another shell published by Belytschko, Wong, and Chiang [Belytschko, Wong, and Chiang 1989, 1992] proposes inexpensive modifications to include the warping stiffness in the Belytschko-Tsay shell. An improved transverse shear treatment also allows the element to pass the Kirchhoff patch test. This element is now available in LS-DYNA. Also, two fully integrated shell elements, based on the Hughes and Liu formulation, are available in LS-DYNA, but are rather expensive. A much faster fully integrated element which is essentially a fully integrated version of the Belytschko, Wong, and Chiang element, type 16, is a more recent addition and is recommended if fully integrated elements are needed due to its cost effectiveness.

Zero energy modes in the shell and solid elements are controlled by either an hourglass viscosity or stiffness. Eight node thick shell elements are implemented and have been found to perform well in many applications. All elements are nearly 100% vectorized. All element classes can be included as parts of a rigid body. The rigid body formulation is documented in [Benson and Hallquist 1986]. Rigid body point nodes, as well as concentrated masses, springs and dashpots can be added to this rigid body.

Membrane elements can be either defined directly as shell elements with a membrane formulation option or as shell elements with only one point for through thickness integration. The latter choice includes transverse shear stiffness and may be inappropriate. For airbag material a special fully integrated three and four node membrane element is available.

Two different beam types are available: a stress resultant beam and a beam with cross section integration at one point along the axis. The cross section integration allows for a more general definition of arbitrarily shaped cross sections taking into account material nonlinearities.

Spring and damper elements can be translational or rotational. Many behavior options can be defined, e.g., arbitrary nonlinear behavior including locking and separation.

Solid elements in LS-DYNA may be defined using from 4 to 8 nodes. The standard elements are based on linear shape functions and use one point integration and hourglass control. A selective-reduced integrated (called fully integrated) 8 node solid element is available for situations when the hourglass control fails. Also, two additional solid elements, a 4 noded tetrahedron and an 8 noded hexahedron, with nodal rotational degrees of freedom, are implemented based on the idea of Allman [1984] to replace the nodal midside translational degrees of freedom of the elements with quadratic shape functions by corresponding nodal rotations at the corner nodes. The latter elements, which do not need hourglass control, require many numerical operations compared to the hourglass controlled elements and should be used at places where the hourglass elements fail. However, it is well known that the elements using more than one point integration are more sensitive to large distortions than one point integrated elements.

The thick shell element is a shell element with only nodal translations for the eight nodes. The assumptions of shell theory are included in a non-standard fashion. It also uses hourglass control or selective-reduced integration. This element can be used in place of any four node shell element. It is favorably used for shell-brick transitions, as no additional constraint conditions are necessary. However, care has to be taken to know in which direction the shell assumptions are made; therefore, the numbering of the element is important.

Seatbelt elements can be separately defined to model seatbelt actions combined with dummy models. Separate definitions of seatbelts, which are one-dimensional elements,

with accelerometers, sensors, pretensioners, retractors, and sliprings are possible. The actions of the various seatbelt definitions can also be arbitrarily combined.

CONTACT-IMPACT INTERFACES

The three-dimensional contact-impact algorithm was originally an extension of the NIKE2D [Hallquist 1979] two-dimensional algorithm. As currently implemented, one surface of the interface is identified as a reference surface and the other as a tracked surface. Each surface is defined by a set of three or four node quadrilateral segments, called reference and tracked segments, on which the nodes of the tracked and reference surfaces, respectively, must slide. In general, an input for the contact-impact algorithm requires that a list of segments be identified for two surfaces. In nonsymmetric contact, these are the tracked and reference surfaces. For symmetric contact, the algorithm is run twice so that each surface acts as both the tracked and reference surfaces. For the single surface algorithm only one surface is defined and each node in the surface is checked each time step to ensure that it does not penetrate through the surface. Internal logic [Hallquist 1977, Hallquist et al. 1985] identifies a reference segment for each tracked node and a tracked segment for each reference node and updates this information every time step as the tracked and reference nodes slide along their respective surfaces. Note that for general automatic definitions only parts/materials or three-dimensional boxes must be given. Then the possible contacting outer surfaces are identified by the internal logic in LS-DYNA. More than 20 types of interfaces can presently be defined including:

- sliding only for fluid/structure or gas/structure interfaces
- tied
- sliding, impact, friction
- single surface contact
- discrete nodes impacting surface
- discrete nodes tied to surface
- shell edge tied to shell surface
- nodes spot welded to surface
- tiebreak interface
- one way treatment of sliding, impact, friction
- box/material limited automatic contact for shells
- automatic contact for shells (no additional input required)
- automatic single surface with beams and arbitrary orientations
- surface to surface eroding contact

- node to surface eroding contact
- single surface eroding contact
- surface to surface symmetric constraint method [Taylor and Flanagan 1989]
- node to surface constraint method [Taylor and Flanagan 1989]
- rigid body to rigid body contact with arbitrary force/deflection curve
- rigid nodes to rigid body contact with arbitrary force/deflection curve
- edge-to-edge
- draw beads

Interface friction can be used with most interface types. The tied and sliding only interface options are similar to the two-dimensional algorithm used in LS-DYNA2D [Hallquist 1976, 1978, 1980]. Unlike the general option, the tied treatments are not symmetric; therefore, the surface which is more coarsely zoned should be chosen as the reference surface. When using the one-way slide surface with rigid materials, the rigid material should be chosen as the reference surface.

For geometric contact entities, contact has to be separately defined. Note that that for the contact of a rigid body with a flexible body, either the sliding interface definitions as explained above or the geometric contact entity contact can be used. Currently, the geometric contact entity definition is recommended for metal forming problems due to high accuracy and computational efficiency.

INTERFACE DEFINITIONS FOR COMPONENT ANALYSIS

Interface definitions for component analyses are used to define surfaces, nodal lines, or nodal points (*INTERFACE_COMPONENTS) for which the time histories of nodal coordinates and if applicable (*INTERFACE_LINKING_NODE/EDGE) and available (beams, shells), nodal rotations, are saved at some user specified frequency (*CONTROL_OUTPUT). This data may then be used to drive interfaces (*INTERFACE_LINKING) in subsequent analyses. This capability is especially useful for studying the detailed response of a small member in a large structure. For the first analysis, the member of interest need only be discretized sufficiently that the displacements and velocities on its boundaries are reasonably accurate. After the first analysis is completed, the member can be finely discretized and interfaces defined to correspond with the first analysis. Finally, the second analysis is performed to obtain highly detailed information in the local region of interest.

When starting the analysis, specify a name for the interface segment file using the Z = parameter on the LS-DYNA command line. When starting the second analysis, the name of

the interface segment file (created in the first run) should be specified using the L = parameter on the LS-DYNA command line.

Following the above procedure, multiple levels of sub-modeling are easily accommodated. The interface file may contain a multitude of interface definitions so that a single run of a full model can provide enough interface data for many component analyses. The interface feature represents a powerful extension of LS-DYNA's analysis capability.

PRECISION

The explicit time integration algorithms used in LS-DYNA are in general much less sensitive to machine precision than other finite element solution methods. Consequently, double precision is not generally required. The benefits of this are greatly improved utilization of memory and disk. When problems have been found we have usually been able to overcome them by reorganizing the algorithm or by converting to double precision locally in the subroutine where the problem occurs. Particularly sensitive problems (e.g. some buckling problems, which can be sensitive to small imperfections) may require the fully double precision version, which is available on all platforms. Very large problems requiring more than 2 billion words of memory will also need to be run in double precision, due to the array indexing limitation of single precision integers.

GETTING STARTED

DESCRIPTION OF KEYWORD INPUT

The keyword input provides a flexible and logically organized database that is simple to understand. Similar functions are grouped together under the same keyword. For example, under the keyword *ELEMENT are included solid, beam, shell elements, spring elements, discrete dampers, seat belts, and lumped masses. Many keywords have options that are identified as follows: "*OPTIONS*" and "*{OPTIONS}*". The difference is that "*OPTIONS*" requires that one of the options must be selected to complete the keyword command. The option <BLANK> is included when {} are used to further indicate that these particular options are not necessary to complete the keyword.

The LS-DYNA User's Manual is alphabetically organized in logical sections of input data. Each logical section relates to a particular input. There is a control section for resetting LS-DYNA defaults, a material section for defining constitutive constants, an equation-of-state section, an element section where element part identifiers and nodal connectivities are defined, a section for defining parts, and so on. Nearly all model data can be input in block form. For example, consider the following where two nodal points with their respective coordinates and shell elements with their part identity and nodal connectivity's are defined:

```
$define two nodes
$
*NODE
10101x y z
10201x y z
$    define two shell elements
$
*ELEMENT_SHELL
10201pidn1n2n3n4
10301pidn1n2n3n4
Alternatively, acceptable input could also be of the form:
$     define one node
$
*NODE
10101x y z
$    define one shell element
$
*ELEMENT_SHELL
10201pidn1n2n3n4
$
$     define one more node
$
*NODE
10201x y z
$ define one more shell element
$
*ELEMENT_SHELL
10301pidn1n2n3n4
```

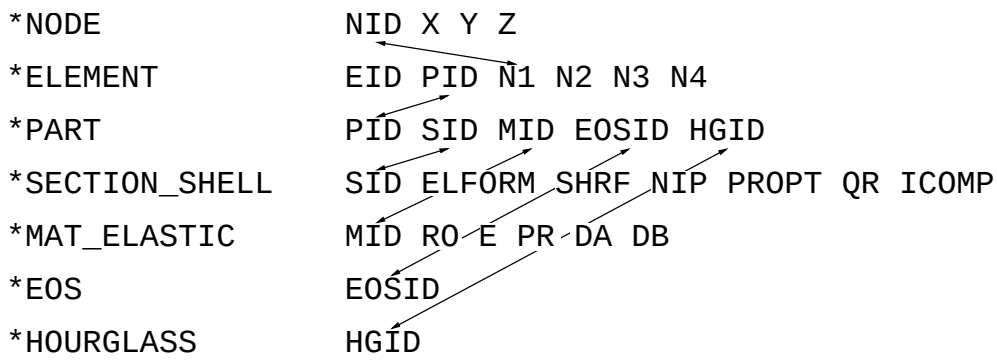


Figure 2-1. Organization of the keyword input.

A data block begins with a keyword followed by the data pertaining to the keyword. The next keyword encountered during the reading of the block data defines the end of the block and the beginning of a new block. A keyword must be left justified with the "*" contained in column one. A dollar sign "$" in column one precedes a comment and causes the input line to be ignored. Data blocks are not a requirement for LS-DYNA, but they can be used to group nodes and elements for user convenience. Multiple blocks can be defined with each keyword if desired as shown above. It would be possible to put all nodal points definitions under one keyword *NODE, or to define one *NODE keyword prior to each node definition. The entire LS-DYNA input is order independent with the exception of the optional keyword, *END, which defines the end of input stream. Without the *END termination is assumed to occur when an end-of-file is encountered during the reading.

Figure 2-1 highlights how various entities relate to each other in LS-DYNA input. In this figure the data included for the keyword, *ELEMENT, is the element identifier, EID, the part identifier, PID, and the nodal points identifiers, the NID's, defining the *element connectivity*: N1, N2, N3, and N4. The nodal point identifiers are defined in the *NODE section where each NID should be defined just once. A part defined with the *PART keyword has a unique part identifier, PID, a section identifier, SID, a material or constitutive model identifier, MID, an equation of state identifier, EOSID, and the hourglass control identifier, HGID. The *SECTION keyword defines the section identifier, SID, where a section has an element formulation specified, a shear factor, SHRF, a numerical integration rule, NIP, among other parameters.

Constitutive constants are defined in the *MAT section where constitutive data is defined for all element types including solids, beams, shells, thick shells, seat belts, springs, and dampers. Equations of state, which are used only with certain *MAT materials for solid elements, are defined in the *EOS section. Since many elements in LS-DYNA use uniformly reduced numerical integration, zero energy deformation modes may develop. These modes are controlled numerically by either an artificial stiffness or viscosity which resists the formation of these undesirable modes. The hourglass control can optionally be user specified using the input in the *HOURGLASS section.

During the keyword input phase where data is read, only limited checking is performed on the data since the data must first be counted for the array allocations and then reordered. Considerably more checking is done during the second phase where the input data is printed out. Since LS-DYNA has retained the option of reading older non-keyword input files, we print out the data into the output file d3hsp (default name) as in previous versions of LS-DYNA. An attempt is made to complete the input phase before error terminating if errors are encountered in the input. Unfortunately, this is not always possible and the code may terminate with an error message. The user should always check either output file, d3hsp or messag, for the word "Error".

The input data following each keyword can be input in free format. In the case of free format input the data is separated by commas, i.e.,

```
*NODE
10101,x ,y ,z
10201,x ,y ,z

*ELEMENT_SHELL
10201,pid,n1,n2,n3,n4
10301,pid,n1,n2,n3,n4
```

When using commas, the formats *must not* be violated. An I8 integer is limited to a maximum positive value of 99999999, and larger numbers having more than eight characters are unacceptable. The format of the input can change from free to fixed anywhere in the input file. The input is case insensitive and keywords can be given in either upper or lower case. *The asterisks "*" preceding each keyword must be in column one.*

To provide a better understanding behind the keyword philosophy and how the options work, a brief review of the keywords is given below.

*AIRBAG

The geometric definition of airbags and the thermodynamic properties for the airbag inflator models can be made in this section. This capability is not necessarily limited to the modeling of automotive airbags, but it can also be used for many other applications, such as tires and pneumatic dampers.

*ALE

This keyword provides a way of defining input data pertaining to the Arbitrary-Lagrangian-Eulerian capability.

*BATTERY

These keywords invoke a one-dimensional electrochemistry solver that can be coupled to the structural and thermal solvers for modeling battery cells. See Volume III.

*BOUNDARY

This section applies to various methods of specifying either fixed or prescribed boundary conditions. For compatibility with older versions of LS-DYNA it is still possible to specify some nodal boundary conditions in the *NODE card section.

*BOUNDARY_ELEMENT_METHOD

This family of keywords sets parameters for boundary method analysis of incompressible fluid dynamics or fluid-structure interaction problems.

*CASE

This keyword option provides a way of running multiple load cases sequentially. Within each case, the input parameters, which include loads, boundary conditions, control cards, contact definitions, initial conditions, etc., can change. If desired, the results from a previous case can be used during initialization. Each case creates unique file names for all output results files by appending CID*n* to the default file name.

*CESE

These keywords invoke the Conservation Element/Solution Element (CESE) compressible fluid solver. See Volume III.

*CHEMISTRY

This section gives the ability to access chemistry databases and select a method for solving a chemical model. These keywords can currenlty only be used with the CESE solver. See Volume III.

*COMPONENT

This section contains analytical rigid body dummies that can be placed within vehicle and integrated implicitly.

*CONSTRAINED

This section applies constraints within the structure between structural parts. For example, nodal rigid bodies, rivets, spot welds, linear constraints, tying a shell edge to a shell edge with failure, merging rigid bodies, adding extra nodes to rigid bodies and defining rigid body joints are all options in this section.

*CONTACT

There are numerous options in the *CONTACT section for treating interaction between disjoint parts.

*CONTROL

Options available in the *CONTROL section allow the resetting of default global parameters, such as the hourglass type, the contact penalty scale factor, shell element formulation, numerical damping, and termination time.

*DAMPING

Defines damping either globally or by part identifier.

*DATABASE

This keyword with a combination of options can be used for controlling the output of ASCII databases and binary files output by LS-DYNA. With this keyword the frequency of writing the various databases can be determined.

*DEFINE

This section allows the user to define curves for loading, constitutive behaviors, etc.; boxes to limit the geometric extent of certain inputs; local coordinate systems; vectors; and orientation vectors specific to spring and damper elements. Items defined in this section are referenced by their identifiers throughout the input. For example, a coordinate system identifier is sometimes used on the *BOUNDARY cards, and load curves are used on the *AIRBAG cards.

*DEFORMABLE_TO_RIGID

This section allows the user to switch parts that are defined as deformable to rigid at the start of the analysis. This capability provides a cost efficient method for simulating events, such as rollover events. While the vehicle is rotating the computation cost can be reduced significantly by switching deformable parts that are not expected to deform to rigid parts. Just before the vehicle comes in contact with ground, the analysis can be stopped and restarted with the part switched back to deformable.

*DUALCESE

These keywords invoke the dual Conservation Element/Solution Element (dual CESE) compressible fluid solver. Compared to the CESE solver, this solver is more accurate and robust. See Volume III.

*ELEMENT

Define identifiers and connectivities for all elements which include shells, beams, solids, thick shells, springs, dampers, seat belts, and concentrated masses in LS-DYNA.

*EM

This section provides input for the electromagnetism module. This module is for solving 3D eddy-current, inductive heating, or resistive heating problems. It can be coupled with the mechanical and thermal solvers. Typical applications include magnetic metal forming and welding. See Volume III.

*EOS

This section reads the equations of state parameters. The equation of state identifier, EOSID, points to the equation of state identifier on the *PART card. See Volume II.

*FATIGUE

This section is for defining and solving fatigue and durability analysis problems.

*FREQUENCY_DOMAIN

This family of keywords is for defining and solving frequency domain and vibration acoustic problems.

*HASH

These keywords request generating a checksum for a portion of the input.

*HOURGLASS

Defines hourglass and bulk viscosity properties. The identifier, HGID, on the *HOURGLASS card refers to HGID on *PART card.

*ICFD

This section probides the options available for the incompressible fluid solver. See Volume III.

*IGA

These keywords set up and control the available isogeometric-related capabilities. Some of the keywords describe the basic geometric objects, others create isogeometric elements.

*INCLUDE

To make the input file easy to maintain, this keyword allows the input file to be split into sub-files. Each sub-file can again be split into sub-sub-files and so on. This option is beneficial when the input data deck is very large.

*INITIAL

Initial velocity and initial momentum for the structure can be specified in this section. The initial velocity specification can be made by *INITIAL_VELOCITY_NODE card or *INITIAL_VELOCITY cards. In the case of *INITIAL_VELOCITY_NODE nodal identifiers are used to specify the velocity components for the node. Since all the nodes in the system are initialized to zero, only the nodes with non-zero velocities need to be specified. The *INITIAL_VELOCITY card provides the capability of being able to specify velocities using the set concept or boxes.

*INTEGRATION

In this section the user defined integration rules for beam and shell elements are specified. IRID refers to integration rule number IRID on *SECTION_BEAM and *SECTION_SHELL cards, respectively. Quadrature rules in the *SECTION_SHELL and *SECTION_BEAM cards need to be specified as a negative number. The absolute value of the negative number refers to user defined integration rule number. Positive rule numbers refer to the built in quadrature rules within LS-DYNA.

*INTERFACE

Interface definitions are used to define surfaces, nodal lines, and nodal points for which the displacement and velocity time histories are saved at some user specified frequency. This data may then be used in subsequent analyses as an interface ID in the *INTERFACE_LINKING_DISCRETE_NODE as constraining nodes, in *INTERFACE_LINKING_SEGMENT as constraining segments, and in *INTERFACE_LINKING_EDGE as the constraining edge for a series of nodes.

This capability is especially useful for studying the detailed response of a small member in a large structure. For the first analysis, the member of interest need only be discretized sufficiently that the displacements and velocities on its boundaries are reasonably accurate. After the first analysis is completed, the member can be finely discretized in the region bounded by the interfaces. Finally, the second analysis is performed to obtain highly detailed information in the local region of interest.

When beginning the first analysis, specify a name for the interface segment file using the Z=parameter on the LS-DYNA execution line. When starting the second analysis, the name of the interface segment file created in the first run should be specified using the L=parameter on the LS-DYNA command line. Following the above procedure, multiple levels of sub-modeling are easily accommodated. The interface file may contain a multitude of interface definitions so that a single run of a full model can provide enough interface data for many component analyses. The interface feature represents a powerful extension of LS-DYNA's analysis capabilities. A similar capability using *INTERFACE_SSI may be used for soil-structure interaction analysis under earthquake excitation.

*ISPG

These keywords enable the creation of a model for incompressible smooth particle Galerkin (ISPG) parts. They can simulate incompressible fluids that can couple with structural parts. This group of keywords cannot be in the normal deck and must be in a file included with *INCLUDE_ISPG.

*KEYWORD

Flags LS-DYNA that the input deck is a keyword deck. To have an effect this must be the very first card in the input deck. Alternatively, by typing "keyword" on the execution line, keyword input formats are assumed and the "*KEYWORD" is not required. If a number is specified on this card after the word KEYWORD, it defines the memory size to be used in words. The memory size can also be set on the command line.

> **NOTE:** The memory specified on the execution line overrides memory specified on the *KEYWORD card.

*LOAD

This section provides various methods of loading the structure with concentrated point loads, distributed pressures, body force loads, and a variety of thermal loadings.

*LSO

These keywords enable the output of LSDA files with a limited set of data for specific domains of the model. See Volume III.

*MAT

This section allows the definition of constitutive constants for all material models available in LS-DYNA, including springs, dampers, and seat belts. The material identifier, MID, points to the MID on the *PART card. See Volume II.

*MESH

The *MESH keywords create mesh for certain kinds of analyses. These keywords are currently only used to produce meshes for the CESE, dual CESE, and ICFD solvers. See Volume III.

*MODULE

This section provides a way to load user-compiled libraries at runtime for user-defined capabilities, such as material models, equations of state, and element formulations.

*NODE

Define nodal point identifiers and their coordinates.

*PARAMETER

This option provides a way of specifying numerical values of parameter names that are referenced throughout the input file. The parameter definitions, if used, should be placed at the beginning of the input file following *KEYWORD. *PARAMETER_EXPRESSION permits general algebraic expressions to be used to set the values.

*PART

This keyword serves two purposes:

1. Relates part ID to *SECTION, *MATERIAL, *EOS and *HOURGLASS sections.

2. Optionally, in the case of a rigid material, rigid body inertia properties and initial conditions can be specified. Deformable material repositioning data can also be specified in this section if the reposition option is invoked on the *PART card, that is, *PART_REPOSITION.

*PERTURBATION

This keyword provides a way of defining deviations from the designed structure, such as buckling imperfections.

*RAIL

This keyword provides a way of defining a wheel-rail contact algorithm intended for railway applications but can also be used for other purposes. The wheel nodes (defined on *RAIL_TRAIN) represent the contact patch between wheel and rail.

*RVE

This keyword controls Representative Volume Element (RVE) analysis.

*RIGIDWALL

Rigid wall definitions have been divided into two separate sections, PLANAR and GEOMETRIC. Planar walls can be either stationary or moving in translational motion with mass and initial velocity. The planar wall can be either finite or infinite. Geometric walls can be planar as well as have the geometric shapes, such as rectangular prism, cylindrical prism and sphere. By default, these walls are stationary unless the option MOTION is invoked for either prescribed translational velocity or displacement. Unlike the planar walls, the motion of the geometric wall is governed by a load curve. Multiple geometric walls can be defined to model combinations of geometric shapes available. For example, a wall defined with the CYLINDER option can be combined with two walls defined with the SPHERICAL option to model hemispherical surface caps on the two ends of a cylinder. Contact entities are also analytical surfaces but have the significant advantage that the motion can be influenced by the contact to other bodies, or prescribed with six full degrees-of-freedom.

*SECTION

In this section, the element formulation, integration rule, nodal thicknesses, and cross sectional properties are defined. All section identifiers (SECID's) defined in this section must be unique; that is, if a number is used as a section ID for a beam element, then this number cannot be used again as a section ID for a solid element.

*SENSOR

This keyword provides a convenient way of activating and deactivating boundary conditions, airbags, discrete elements, joints, contact, rigid walls, single point constraints, and constrained nodes. The sensor capability is new in the second release of version 971

and will evolve in later releases to encompass many more LS-DYNA capabilities and replace some of the existing capabilities, such as the airbag sensor logic.

*SET

A concept of grouping nodes, elements, materials, etc., in sets is employed throughout the LS-DYNA input deck. Sets of data entities can be used for output. So-called surfa nodes used in contact definitions, surfa segment sets, surfb segment sets, pressure segment sets, and so on can also be defined. The keyword, *SET, can be defined in two ways:

1. Option LIST requires a list of entities in which eight entities are defined per card and as many cards as needed are used to define all the entities.

2. Option COLUMN, where applicable, requires an input of one entity per line along with up to four attribute values which are used by other keywords to specify, for example, the failure criterion input that is needed for *CONTACT_CONSTRAINT_NODES_TO_SURFACE.

*STOCHASTIC

These keywords are for modeling spray and embedded particles in TBX explosives using stochastic PDEs. They can only be used with the CESE solver. See Volume III.

*TERMINATION

This keyword provides an alternative way of stopping the calculation before the termination time is reached. The termination time is specified on the *CONTROL_TERMINATION input and will terminate the calculation whether or not the options available in this section are active.

*TITLE

In this section a title for the analysis is defined.

*UNIT

This section provides a way to specify units in a problem. Currently, it is only used by the dual CESE (*DUALCESE) solvers.

*USER

This section provides a method for specifying input to user-defined subroutines and for smoothing history data using user-defined subroutines.

RESTART

This section of the input is intended to allow the user to restart the simulation by providing a restart file and optionally a restart input defining changes to the model, such as deleting contacts, materials, elements, switching materials from rigid to deformable, and deformable to rigid.

*RIGID_DEFORMABLE

This section switches rigid parts back to deformable in a restart to continue the event of a part impacting the ground which may have been modeled with a rigid wall.

*STRESS_INITIALIZATION

This is an option available for restart runs. In some cases there may be a need for the user to add contacts, elements, etc., which are not available options for standard restart runs. A full input containing the additions is needed if this option is invoked upon restart.

SUMMARY OF COMMONLY USED OPTIONS

The following table gives a list of the commonly used keywords related by topic.

Table 2.1. Keywords for the most commonly used options.

Topic	Component	Keywords
Geometry	Nodes	*NODE
	Elements	*ELEMENT_BEAM *ELEMENT_SHELL *ELEMENT_SOLID *ELEMENT_TSHELL
	Discrete Elements	*ELEMENT_DISCRETE *ELEMENT_SEATBELT *ELEMENT_MASS

Topic	Component	Keywords
Materials	Part	PART cards glues the model together: *PART → { *MAT, *SECTION, *EOS, *HOURGLASS
	Material	*MAT
	Sections	*SECTION_BEAM *SECTION_SHELL *SECTION_SOLID *SECTION_TSHELL
	Discrete sections	*SECTION_DISCRETE *SECTION_SEATBELT
	Equation of state	*EOS
	Hourglass	*CONTROL_HOURGLASS *HOURGLASS
Contacts & Rigid walls	Defaults for contacts	*CONTROL_CONTACT
	Definition of contacts	*CONTACT_*OPTION*
	Definition of rigid walls	*RIGIDWALL_*OPTION*
Boundary Conditions & Loadings	Restraints	*NODE *BOUNDARY_SPC_*OPTION*
	Gravity (body) load	*LOAD_BODY_*OPTION*
	Point load	*LOAD_NODE_*OPTION*
	Pressure load	*LOAD_SEGMENT_*OPTION* *LOAD_SHELL_*OPTION*
	Thermal load	*LOAD_THERMAL_*OPTION*
	Load curves	*DEFINE_CURVE
Constraints and spot welds	Constrained nodes	*CONSTRAINED_NODE_SET
	Welds	*CONSTRAINED_GENERALIZED_WELD *CONSTRAINED_SPOT_WELD
	Rivet	*CONSTRAINED_RIVET

Topic	Component	Keywords
Output Control	Items in time history blocks	*DATABASE_HISTORY_*OPTION*
	Default	*CONTROL_OUTPUT
	ASCII time history files	*DATABASE_OPTION
	Binary plot/time history/restart files	*DATABASE_BINARY_OPTION
	Nodal reaction output	*DATABASE_NODAL_FORCE_GROUP
Termination	Termination time	*CONTROL_TERMINATION
	Termination cycle	*CONTROL_TERMINATION
	CPU termination	*CONTROL_CPU
	Degree of freedom	*TERMINATION_NODE

EXECUTION SYNTAX

The execution line for LS-DYNA, sometimes referred to as the command line, is as follows:

LS-DYNA I=**inf** O=**otf** G=**ptf** D3PART=**d3part** D=**dpf** R=**rtf** N=**rtff** F=**thf** T=**tpf** A=**rrd** M=**sif** S=**iff** H=**iff** Z=**isf1** L=**isf2** B=**rlf** W=**root** E=**efl** X=**scl** C=**cpu** K=**kill** V=**vda** Y=**c3d** P=**pfile** BEM=**bof** CPM=**cmpof** DEM=**demof** PBM=**pbmof** ISPH=**isphof** FSILNK=**fsilof** {**KEYWORD**} {**THERMAL**} {**COUPLE**} {**CASE**} {**PGPKEY**} {**FLUIDS**} {**MICROSCALE**} {**HASH**} {**SHARECRV**} MEMORY=**nwds** MODULE=**dll** NCPU=**ncpu** PARA=**para** ENDTIME=**time** NCYCLE=**ncycle** JOBID=**jobid** D3PROP=**d3prop** GMINP=**gminp** GMOUT=**gmout** MCHECK=**y** MAP=**map** MAP1=**map1** MAP2=**map2** LAGMAP=**lagmap** LAGMAP1=**lagmap1** SHELL=**shefch** ND3KIL=**ncyclek** STDOUT=**stdout** SRTFLG=**srtflg** HEARTBEAT=**ncycleh**

where,

inf = Input file (user-specified)

otf = High speed printer file (default = d3hsp)

ptf = Binary plot file for postprocessing (default = d3plot)

d3part = Binary plot file for subset of parts; see *DATABASE_BINARY_D3PART (default = d3part)

dpf = Dump file to write for purposes of restarting (default = d3dump). This file is written at the end of every run and during the run as requested by *DATABASE_BINARY_D3DUMP. To stop the generation of this dump file,

specify "d=nodump" (case insensitive). Sense switch "sw1" triggers writing a dump file and overrides "d=nodump".

thf = Binary plot file for time histories of selected data (default = d3thdt)

rtf = Binary file from which to restart the analysis for restarts (see RESTART ANALYSIS). This applies to simple and small deck restarts. It also applies to full deck restarts for SMP.

rtff = d3full file from which to restart the analysis for full deck restarts in MPP (see RESTART ANALYSIS).

tpf = Optional temperature file

rrd = Running restart dump file (default = runrsf)

sif = Stress initialization file (user-specified)

iff = Interface force file (user-specified). See *DATBASE_BINARY_INTFOR and *DATABASE_BINARY_FSIFOR.

isf1 = Interface segment save file to be created (default = infmak)

isf2 = Existing interface segment save file to be used (user-specified)

rlf = Binary plot file for dynamic relaxation (default = d3drlf)

efl = Echo file containing optional input echo with or without node/element data

root = Root file name for general print option

scl = Scale factor for binary file sizes (default =70)

cpu = Cumulative cpu time limit in seconds for the entire simulation, including all restarts, if cpu is positive. If cpu is negative, the absolute value of cpu is the cpu time limit in seconds for the first run and for each subsequent restart run.

kill = If LS-DYNA encounters this file name it will terminate with a restart file (default = d3kil)

vda = VDA/IGES database for geometrical surfaces

c3d = CAL3D input file

pfile = File containing MPP specific parameters (see Appendix O for details)

bof = *FREQUENCY_DOMAIN_ACOUSTIC_BEM output file

cpmof = CPM interface force file (user-specified). See *DATABASE_BINARY_*OPTION*.

demof = DEM interface force file (user-specified). See *DATABASE_BINARY_*OPTION*.

pbmof = Particle blast method interface force file (user-specified). See *DATABASE_BINARY_*OPTION.*

isphof = Incompressible SPH interface force file (user-specified). See *DATABASE_BINARY_*OPTION.*

fsilof = ALE interface linking database file (user-specified). See *DATABASE_BINARY_*OPTION.*

nwds = Number of words to be allocated. A word is 4 bytes in single precision and 8 bytes in double precision. This number overwrites the memory size specified on the *KEYWORD card at the beginning of the input deck. This option is necessary if the default memory allocation is insufficient, in which case LS-DYNA will write an error message stating such.

dll = The dynamic library for user subroutines. Only one dynamic library can be loaded via "module=dll". See *MODULE_LOAD command for loading multiple dynamic libraries.

ncpu = Overrides NCPU and CONST defined in *CONTROL_PARALLEL. A positive value sets CONST = 2 and a negative value sets CONST = 1. See the *CONTROL_PARALLEL command for an explanation of these parameters. The *KEYWORD command provides an alternative way to set the number of CPUs.

para = Overrides PARA defined in *CONTROL_PARALLEL.

time = Overrides ENDTIM defined in *CONTROL_TERMINATION.

ncycle = Overrides ENDCYC defined in *CONTROL_TERMINATION.

jobid = Character string which acts as a prefix for all output files. Maximum length is 72 characters. Do not include the following characters:) (* / ? \.

d3prop = See *DATABASE_BINARY_D3PROP input parameter IFILE for options.

gminp = Input file for reading recorded motions in *INTERFACE_SSI (default = gmbin).

gmout = Output file for writing recorded motions in *INTERFACE_SSI_AUX (default = gmbin).

map = Output/Input file for writing/reading data in ALE mappings controlled by *INITIAL_ALE_MAPPING or *BOUNDARY_ALE_MAPPING.

map1 = Output file for writing data in ALE mappings controlled by *INITIAL_ALE_MAPPING or *BOUNDARY_ALE_MAPPING if MAP= is used to read data from a different file.

map2 = Input .txt file that lists multiple mapping files for reading data from multiple ALE mapping files controlled by *INITIAL_ALE_MAPPING.

lagmap = Output/Input file for writing/reading data in Lagrangian mappings controlled by *INITIAL_LAG_MAPPING.

lagmap1 = Output file for writing data in Lagrangian mappings controlled by *INITIAL_LAG_MAPPING if LAGMAP= is used to read data from a different file.

shefch = Change the shell element formulation from 16 to 30 or 30 to 16 without changing the input deck. To do this, use either "shell=16to30" or "shell=30to16".

ncyclek = Overrides the default frequency of checking for the d3kil file. The default is every 100 cycles in explicit and every cycle in implicit.

stdout = Control the messages printed to stdout. The available values and their meanings are in the following table. All means that all of that particular type of message is printed, while none means none of that kind of message is printed. The default value of stdout is 0.

stdout	Information	Warning	Error
0	all	all	all
1	none	all	all
2	all	none	all
3	none	none	all
4	all	all	none
5	none	all	none
6	all	none	none
7	none	none	none

srtflg = Flag to process parts, contacts, nodal rigid bodies, and elements in a sequence sorted by their respective user IDs, regardless of the order they appear in the input files. When turned on (srtflg = 1), it ensures consistent results if the user modifies the order of these entities in the input files, such as by changing the order of include files between runs of the same model. srtflg = overrides SRTFLG on *CONTROL_ACCURACY. Thus, srtflg = 0 deactivates this sorting.

ncycleh = Flag to write a status line to an ASCII file named heartbeat. A nonzero ncycleh causes this file to be written. If ncycleh > 0, a line is appended to the file every ncycleh cycles. If ncycleh < 0, the file contains a single line that is overwritten every |ncycleh| cycles.

These command line options can be *in any order*.

To avoid undesirable or confusing results, each LS-DYNA run should be performed in a separate directory, unless using the command line parameter "jobid" described above. If rerunning a job in the same directory, old files should first be removed or renamed to

avoid confusion since the possibility exists that the binary database may contain results from both the old and new run.

By including "keyword" anywhere on the execute line or instead if *KEYWORD is the first card in the input file, the keyword formats are expected; otherwise, the older structured input file will be expected.

To run a coupled thermal analysis the command "couple" must be in the execute line. A thermal-only analysis may be run by including the word "thermal" in the execution line.

The execution line option "pgpkey" will output the current public PGP key used by LS-DYNA for encrypting the input. LS-DYNA writes the public key and some instructions on how to use the key are to the screen and to a file named "lstc_pgpkey.asc".

For licenses restricted to the fluids capabilities of LS-DYNA, you need to include "fluids" as an execution line option.

The execution line option "microscale" is needed for running the microscale process in a multiscale simulation with either *INCLUDE_MULTISCALE or *INCLUDE_MULTISCALE_SPOTWELD. See those keywords for details.

Setting "ncycle=1" on the execution line causes the calculation to run just one cycle followed by termination with restart files. No editing of the input deck is required. The calculation can then be restarted with or without any additional input.

If the word "case" appears on the command line, then *CASE statements will be handled by the built-in driver routines. Otherwise, they should be processed by the external "lscasedriver" program, and if any *CASE statements are encountered, it will cause an error.

If "mcheck=y" is given on the command line, the program switches to "model check" mode. In this mode, the program will run only 10 cycles – just enough to verify that the model will start. For implicit problems, all initialization is performed, but execution halts before the first cycle. If the network license is being used, the program will attempt to check out a license under the program name "LS-DYNAMC" so as not to use up one of the normal DYNA licenses. If this fails, a normal execution license will be used.

If "hash" is given on the command line, then SHA256 hashes are computed and reported at the bottom of the d3hsp file for the following: any *HASH/*HASH_END blocks, all input files, and some of the generated output files (currently glstat, d3plot, and binout files). Additionally, the final d3hsp file is digitally signed by LS-DYNA, generating a d3hsp.sig file, which can be verified using GPG with the 2048-bit LS-DYNA public encryption key.

Including "sharecrv" on the command line causes the raw load curve data, which is the same on all processors, to be put into shared memory on each compute node. Doing this reduces the amount of memory required for execution. We recommend this command line option only if necessary, as it may have a slight impact on speed. This feature is only available in the MPP version and is not supported with PlatformMPI.

SENSE SWITCH CONTROLS

The status of an in-progress LS-DYNA simulation can be determined by using the sense switch. On UNIX versions, this is accomplished by first typing a "^C" (Control-C). This sends an interrupt to LS-DYNA, which is trapped, and the user is prompted to input the sense switch code. LS-DYNA has a number of terminal sense switch controls that are tabulated below:

Type	Response
SW1	A restart (dump) file is written, and LS-DYNA terminates.
SW2	LS-DYNA responds with time and cycle numbers.
SW3	A restart file is written, and LS-DYNA continues.
SW4	A plot state is written, and LS-DYNA continues. Additionally, a d3dat file is output if using the CPG method or if the input deck contains *DATABASE_BINARY_D3DAT.
SWA	Flush ASCII file buffers.
SWB	A dynain file is written, and LS-DYNA continues. For SWB, SWC, and SWD (see below), *INTERFACE_SPRINGBACK_LSDYNA must be defined in the input deck. The file name is dynain.#, where # is incremented by 1 each time SWB, SWC, or SWD is issued.
SWC	A restart file and a dynain file are written, and LS-DYNA continues.
SWD	A restart file and a dynain file are written, and LS-DYNA terminates.
SWE	Terminate explicit dynamic relaxation and proceed with the transient analysis.
conv	Temporarily override nonlinear convergence tolerances.
endtime=*t*	Change the termination time to *t* (*t* is a real number).
iter	Enable/Disable output of binary plot database "d3iter" showing mesh after each equilibrium iteration. Useful for debugging convergence problems.
lprint	Enable/Disable printing of equation solver memory, cpu requirements.

Type	Response
nlprint	Enable/Disable printing of nonlinear equilibrium iteration information.
prof	Output current timing information to messag (SMP) or prof.out (MPP).
stop	Halt execution immediately, closing open files.

On UNIX/LINUX and Windows systems the sense switches can still be used if the job is running in the background or in batch mode. To interrupt LS-DYNA simply create a file called d3kil containing the desired sense switch, for example, "sw1". LS-DYNA periodically looks for this file, and if found, the sense switch contained therein is invoked and the d3kil file is deleted. Command line option nd3kil=ncyclek changes the frequency of checking for this file. A null d3kil file is equivalent to a "sw1".

When LS-DYNA terminates, all scratch files are destroyed: the restart file, plot files, and high-speed printer files remain on disk. Of these, only the restart file is needed to continue the interrupted analysis.

PROCEDURE FOR LS-DYNA/MPP

As described above the serial/SMP code supports the use of the SIGINT signal (usually Ctrl-C) to interrupt the execution and prompt the user for a "sense switch." The MPP code also supports this capability. However, on many systems a shell script or front end program (generally "mpirun") is required to start MPI applications. Pressing Ctrl-C on some systems will kill this process, and thus kill the running MPP-DYNA executable. On UNIX/LINUX systems, as workaround, when the MPP code begins execution it creates a file named, "bg_switch", in the current working directory. This file contains the following single line:

rsh **<machine name>** kill -INT **<PID>**

where < machine name > is the hostname of the machine on which the root MPP-DYNA process is running, and <PID> is its process id. (on HP systems, "rsh" is replaced by "remsh"). Thus, simply executing this file will send the appropriate signal. For Windows, usually the d3kil file is used to interrupt the execution.

For more information about running the LS-DYNA/MPP Version see Appendix O.

Files: Input and Output

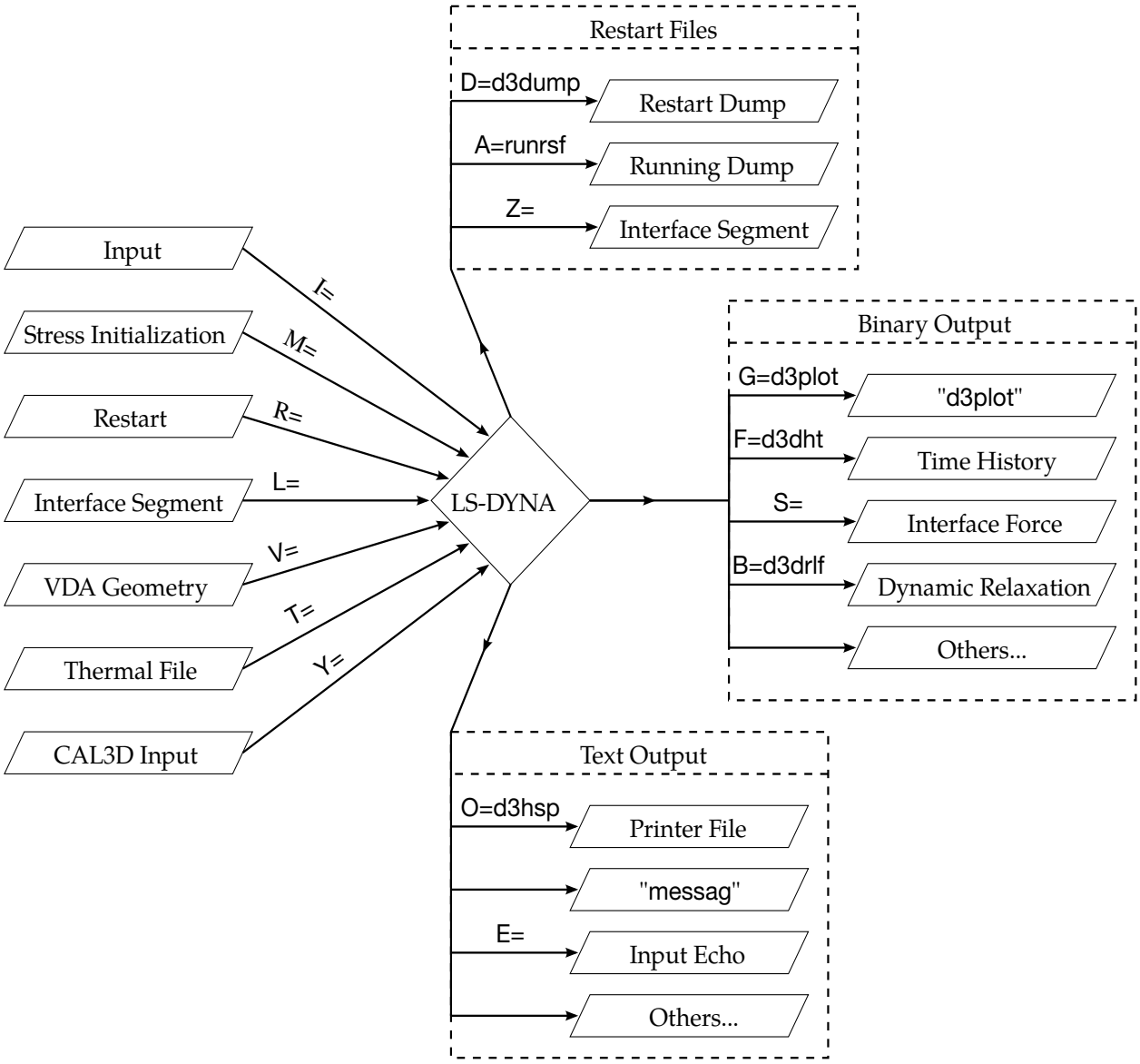


Figure 2-2. Files Input and Output.

FILES

1. **Uniqueness.** File names must be unique.

2. **Interface forces.** The interface force file is created only if it is specified on the execution line "S=**iff**".

3. **File size limits.** For very large models, the default size limits for binary output files may not be large enough for a single file to hold even a single plot state, in

which case the file size limit may be increased by specifying "X=**scl**" on the execution line. The default file size limit (X=70) is 70 times one-million octal words or 18.35 Mwords. That translates into 73.4 Mbytes (for 32-bit output) or 146.8 Mbytes (for 64-bit output).

4. **CPU limits.** Using "C=**cpu**" defines the maximum CPU usage allowed. When the CPU usage limit is exceeded, LS-DYNA will terminate with a restart file. During a restart, **cpu** should be set to the total CPU used up to the current restart plus whatever amount of additional time is wanted.

5. **File usage in restart.** When restarting from a dump file, the execution line (except of a full deck restart in MPP) becomes

   LS-DYNA I=**inf** O=**otf** G=**ptf** D=**dpf** R=**rtf** F=**thf** T=**tpf** A=**rrd** S=**iff** Z=**isf1** L=**isf2** B=**rlf** W=**root** E=**efl** X=**scl** C=**cpu** K=**kill** {KEYWORD}

   where

   rtf=[name of dump file written by LS-DYNA]

   The root names of the dump files written by LS-DYNA are controlled by dpf (default = d3dump) and rrd (default = runrsf). A two-digit number follows the root name, for example, d3dump01, d3dump02, etc., to distinguish one dump file from another. Typically, each dump file corresponds to a different simulation time.

   The adaptive dump files contain all information required to successfully restart so that no other files are needed except when CAD surface data is used. When restarting a problem that uses VDA/IGES surface data, the vda input file must be specified, e.g.:

   LS-DYNA R=**d3dump01** V=**vda**

   No stress initialization is possible at restart. Also the VDA files and the CAL3D files must not be changed.

6. **Default file names.** File name dropouts are permitted; for example, the following execution lines are acceptable:

   LS-DYNA I=**inf**

   and

   LS-DYNA R=**rtf**

7. **Interface segments.** For an analysis using interface segments, the execution line in the first analysis is given by:

LS-DYNA I=**inf** Z=**isf1**

and in the second by:

LS-DYNA I=**inf** L=**isf1**

8. **Batch execution.** In some installations (e.g., GM) calculations are controlled by a file called "names" on unit 88. The names files consists of two lines in which the second line is blank. The first line of names contains the execution line, for instance:

I=**inf**

For a restart the execution line becomes:

I=**inf** R=**rtf**

RESTART ANALYSIS

The LS-DYNA restart capability allows analyses to be broken down into stages. After the completion of each stage in the calculation, a "restart dump" is written that contains all information necessary to continue the analysis. The size of this "dump" file is roughly the same size as the memory required for the calculation. Results can be checked at each stage by post-processing the output databases in the normal way, so the chance of wasting computer time on incorrect analyses is reduced.

The restart capability is frequently used to modify models by deleting excessively distorted elements, materials that are no longer important, and contact surfaces that are no longer needed. Output frequencies of the various databases can also be altered. Often, these simple modifications permit a calculation that might otherwise not continue on to a successful completion. Restarting can also help diagnose why a model is giving problems. By restarting from a dump that is written before the occurrence of a numerical problem and obtaining output at more frequent intervals, it is often possible to identify where the first symptoms appear and what aspect of the model is causing them.

The format of the restart input file is described in this manual. If, for example, you wish to restart the analysis from dump state *nn*, contained in file D3DUMP*nn*, then:

1. Create the restart input deck, if required, as described in the Restart Section of this manual. Call this file restartinput.

2. Start LS-DYNA from the command line by invoking:

LS-DYNA I=**restartinput** R=**D3DUMP*nn***

If no alterations to the model are made, then the execution line:

LS-DYNA R=**D3DUMP*nn***

will suffice. Of course, the other output files should be assigned names from the command line if the defaults have been changed in the original run.

The full deck restart option allows the user to begin a new analysis, with deformed shapes and stresses carried forward from a previous analysis for selected materials. The new analysis can be different from the original, such as more contact surfaces and different geometry (of parts which are not carried forward). Examples of applications include:

- Crash analysis continued with extra contact surfaces.
- Sheet metal forming continued with different tools for modeling a multi-stage forming process.

To perform a full-deck restart:

1. Create the restart input deck using *STRESS_INITIALIZATION, as described in the Restart Section of this manual. Call this file restartinput.

2. For SMP, start LS-DYNA from the command line by invoking:

   LS-DYNA I=**restartinput** R=**D3DUMP*nn***

   For MPP, start LS-DYNA from the command line by invoking:

   LS-DYNA I=**restartinput** N=**D3FULL*nn***

A typical restart file scenario:

LS-DYNA is run using an input file named "job1.inf", and a restart dump named "d3dump01" is created. A new input file, "job2.inf", is generated and submitted as a restart with, "R=d3dump01", as the dump file. The input file job2.inf contains the entire model in its original *undeformed* state but with more contact surfaces, new output databases, and so on.

Since this is a restart job, information must be given to tell LS-DYNA which parts of the model should be initialized in the full deck restart. When the calculation begins, the restart database contained in the file d3dump01 is read, and a new database is created to initialize the model in the input file, job2.inf. The data in file job2.inf is read, and then LS-DYNA proceeds through the entire input deck and initialization. At the end of the initialization process, all the parts selected are initialized from the data saved from d3dump01. This means that the deformed position and velocities of the nodes on the elements of each part, and the stresses and strains in the elements (and, if the material of the part is rigid, the rigid body properties) will be assigned.

It is assumed during this process that any initialized part has the same elements, in the same order, with the same topology, in job1 and job2. If this is not the case, the parts cannot be initialized. However, the parts may have different identifying numbers.

For discrete elements and seat belts, the choice is all or nothing. All discrete and belt elements, retractors, slip rings, pretensioners, and sensors must exist in both files and will be initialized.

Materials which are not initialized will have no initial deformations or stresses. However, if initialized and non-initialized materials have nodes in common, the nodes will be moved by the initialized material causing a sudden strain in the non-initialized material. *This effect can give rise to sudden spikes in loading.*

Points to note are:

- Time and output intervals are continuous with job1; that is, the time is not reset to zero.
- Don't try to use the restart part of the input to change anything since this will be overwritten by the new input file.
- Usually, the complete input file part of job2.inf will be copied from job1.inf, with the required alterations. We again mention that there is no need to update the nodal coordinates since the deformed shapes of the initialized materials will be carried forward from job1.
- Completely new databases will be generated with the time offset.

VDA/IGES DATABASES

VDA surfaces are surfaces of geometric entities which are given in the form of polynomials. The format of these surfaces is as defined by the German automobile and supplier industry in the VDA guidelines, [VDA 1987].

The advantage of using VDA surfaces is twofold. First, the problem of meshing the surface of the geometric entities is avoided and, second, smooth surfaces can be achieved which are very important in metal forming. With smooth surfaces, artificial friction introduced by standard faceted meshes with corners and edges can be avoided. This is a big advantage in springback calculations.

A very simple and general handling of VDA surfaces is possible allowing arbitrary motion and generation of surfaces. For a detailed description, see Appendix L.

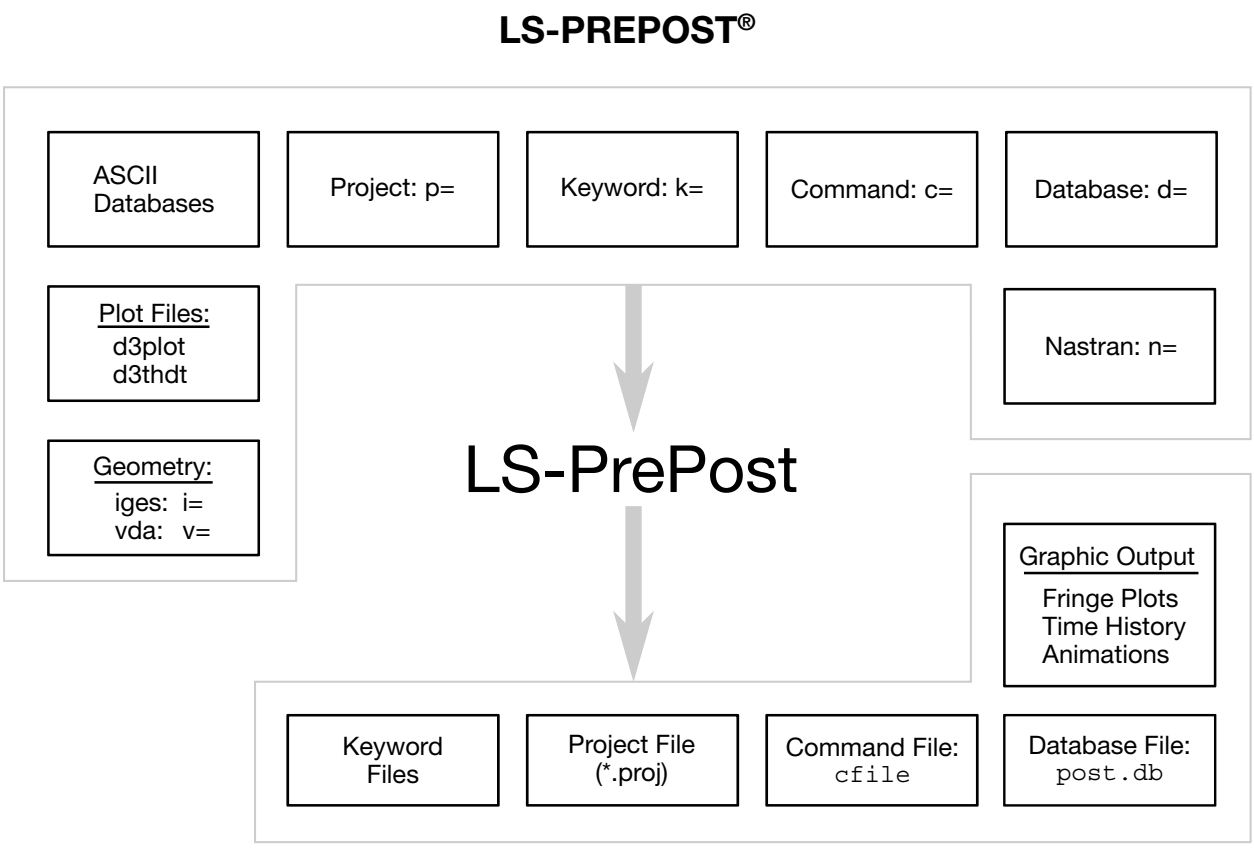


Figure 2-3. File Organization

LS-DYNA is designed to operate with a variety of commercial pre- and post-processing packages. Currently, direct support is available from TRUEGRID, PATRAN, eta/VPG, HYPERMESH, EASi-CRASH DYNA and FEMAP. Several third-party translation programs are available for PATRAN and IDEAS.

Alternately, the pre- and post-processor LS-PrePost is available from Ansys and is specialized for LS-DYNA. LS-PrePost is an advanced pre- and post-processor that is delivered free with LS-DYNA. The user interface is designed to be both efficient and intuitive. LS-PrePost runs on Windows, Linux, and Unix, utilizing OpenGL graphics to achieve fast model rendering and XY plotting.

Some of the capabilities available in LS-PrePost are:

- Complete support for all LS-DYNA keyword data.
- Importing and combining multiple models from many sources (LS-DYNA keyword, IDEAS neutral file, NASTRAN bulk data, STL ASCII, and STL binary formats).
- Improved renumbering of model entities.
- Model Manipulation: Translate, Rotate, Scale, Project, Offset, Reflect

- LS-DYNA Entity Creation: Coordinate Systems, Sets, Parts, Masses, CNRBs, Boxes, Spot welds, SPCs, Rigidwalls, Rivets, Initial Velocity, Accelerometers, Cross Sections, etc.
- Mesh Generation: 2Dmesh Sketchboard, nLine Meshing, Line sweep into shell, Shell sweep into solid, Tet-Meshing, Automatic surface meshing of IGES and VDA data, Meshing of simple geometric objects (Plate, Sphere, Cylinder)
- Special Applications: Airbag folding, Dummy positioning, Seatbelt fitting, Initial penetration check, Spot weld generation using MAT_100
- Complete support of LS-DYNA results data file: d3plot file, d3thdt file, All ASCII time history data file, Interface force file

LS-PrePost processes output from LS-DYNA. LS-PrePost reads the binary plot-files generated by LS-DYNA and plots contours, fringes, time histories, and deformed shapes. Color contours and fringes of a large number of quantities may be interactively plotted on meshes consisting of plate, shell, and solid type elements. LS-PrePost can compute a variety of strain measures and reaction forces along constrained boundaries.

LS-DYNA generates three binary databases. One contains information for complete states at infrequent intervals; 50 to 100 states of this sort is typical in a LS-DYNA calculation. The second contains information for a subset of nodes and elements at frequent intervals; 1000 to 10,000 states is typical. The third contains interface data for contact surfaces.

EXECUTION SPEEDS

The relative execution speeds for various elements in LS-DYNA are tabulated below:

Element Type	Relative Cost
8 node solid with 1 point integration and default hourglass control	4
as above but with Flanagan-Belytschko hourglass control	5
constant stress and Flanagan-Belytschko hourglass control, i.e., the Flanagan-Belytschko element	7
4 node Belytschko-Tsay shell with four thickness integration points	4
4 node Belytschko-Tsay shell with resultant plasticity	3
BCIZ triangular shell with four thickness integration points	7
C^0 triangular shell with four thickness integration points	4
2 node Hughes-Liu beam with four integration points	9
2 node Belytschko-Schwer beam	2
2 node simple truss elements	1

Element Type	Relative Cost
8 node solid-shell with four thickness integration points	11

These relative timings are very approximate. Each interface node of the sliding interfaces is roughly equivalent to one-half zone cycle in cost. Figure 2-4 illustrates the relative cost of the various shell formulations in LS-DYNA.

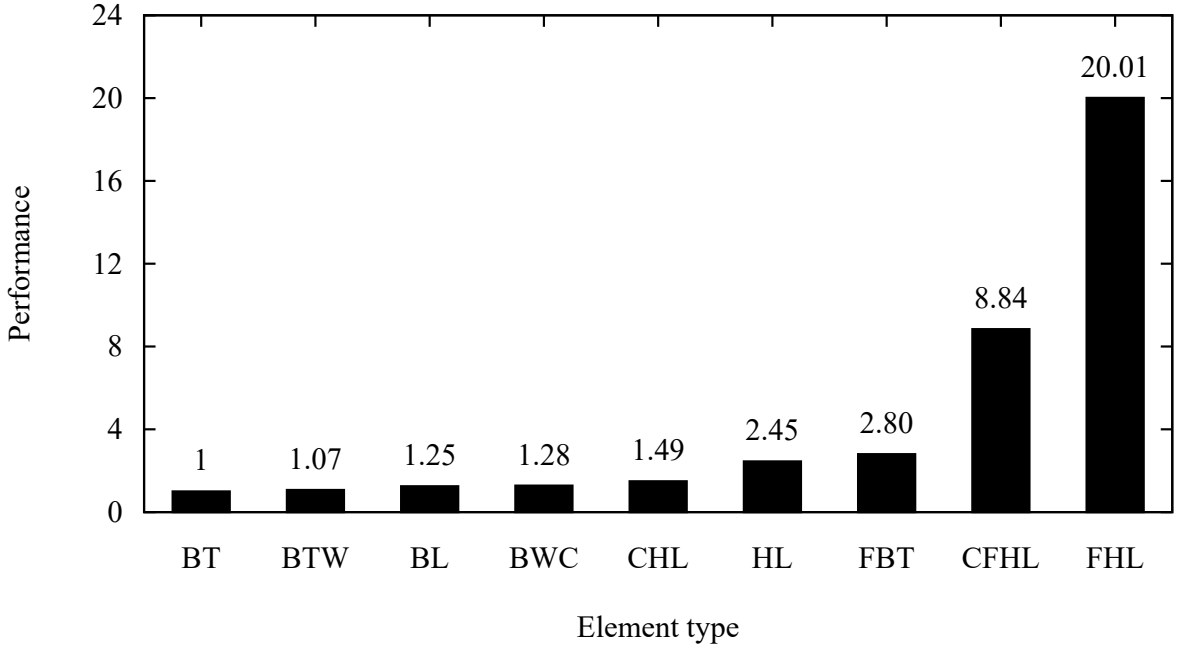


Figure 2-4. Relative cost of the four noded shells available in LS-DYNA where BT is the Belytschko-Tsay shell, BTW is the Belytschko-Tsay shell with the warping stiffness taken from the Belytschko-Wong-Chiang, BWC, shell. The BL shell is the Belytschko-Leviathan shell. CHL denotes the Hughes-Liu shell, HL, with one point quadrature and a co-rotational formulation. FBT is a Belytschko-Tsay like shell with full integration, FHL is the fully integrated Hughes-Liu shell, and the CFHL shell is its co-rotational version.

UNITS

The units in LS-DYNA must be consistent. One way of testing whether a set of units is consistent is to check that:

$$[\text{force unit}] = [\text{mass unit}] \times [\text{acceleration unit}]$$

and that

$$[\text{acceleration unit}] = \frac{[\text{length unit}]}{[\text{time unit}]^2}.$$

Examples of sets of consistent units are tabulated below. For a more comprehensive table, see https://lsdyna.ansys.com/consistent-units/.

	(a)	(b)	(c)
Length unit	meter	millimeter	millimeter
Time unit	second	second	millisecond
Mass unit	kilogram	tonne	kilogram
Force unit	Newton	Newton	kiloNewton
Young's Modulus of Steel	210.0E+09	210.0E+03	210.0
Density of Steel	7.85E+03	7.85E–09	7.85E–06
Yield stress of Mild Steel	200.0E+06	200.0	0.200
Acceleration due to gravity	9.81	9.81E+03	9.81E-03
Velocity equivalent to 30 mph	13.4	13.4E+03	13.4

GENERAL CARD FORMAT

The following sections specify, for each keyword command, the cards that must be defined and those that are optional. Each card is described in its fixed format form and is shown as a number of fields in an 80 character string. With the exception of "long format input" as described later in this section, most input cards consist of 8 fields with a field length of 10 characters. A sample card is shown below. The card format is clearly stated if the format is different than 8 fields of 10 characters. See also notes on a special "I10 format" at the end of this section.

As an alternative to fixed format, a card may be in free format with the values of the variables separated by commas. When using comma-delimited values on a card, the number of characters used to specify a value must not exceed the field length for fixed format. For example, an I8 number is limited to a value of 99999999, and a larger number with more than 8 characters is unacceptable. A further restriction is that characters beyond column 80 of each line are ignored by the code. Fixed format and free, comma-delimited format can be mixed throughout the deck and even within different cards of a single command but not within a card.

The limits on the number of characters per variable and the number of characters per line, as stated above, are raised in the case of long format input. See the description of long format input below.

Example Card.

Card [N]	1	2	3	4	5	6	7	8
Variable	NSID	PSID	A1	A2	A3	KAT	CTYPE	
Type	I	I	F	F	F	I	A	
Default	none	none	1.0	1.0	0	1	none	
Remarks	1			2		3		

In the example shown above, the row labeled "Type" gives the variable type. Type can be F for floating point, I for an integer, or A for alphanumeric. The row labeled "Default" reveals the default value set for a variable if zero is specified, the field is left blank, or the card is not defined. The "Remarks" row refers to enumerated remarks at the end of the section.

Optional Cards:

Each keyword card (line beginning with "*") is followed by a set of data cards. Data cards are either,

1. *Required Cards.* Unless otherwise indicated, cards are required.

2. *Conditional Cards.* Conditional cards are required, provided some condition is satisfied. The following is a typical conditional card:

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ABID	HEADING						
Type	I	A70						

3. *Optional Cards.* An optional card is one that may be replaced by the next keyword card. The fields in the omitted optional data cards are assigned their default values.

Example. Suppose the data set for *KEYWORD consists of 2 required cards and 3 optional cards. Then, the fourth card may be replaced by the next keyword card. All the fields in the omitted fourth and fifth cards are assigned their default values.

WARNING: In this example, even though the fourth card is optional, the input deck may *not* jump from the third to fifth card. The *only* card that card 4 may be replaced with is the next keyword card.

Long Format Input:

To accommodate larger or more precise values for input variables than are allowed by the standard format input as described above, a "long format" input option is available. One way of invoking long format keyword input is by adding "long=**y**" to the execution line. A second way is to add "long=**y**" to the *KEYWORD command in the input deck.

long=**y**: read long keyword input deck; write long structured input deck.
long=**s**: read standard keyword input deck; write long structured input deck.
long=**k**: read long keyword input deck; write standard structured input deck.

The "long=**s**" option may be helpful in the rare event that the keyword input is of standard format, but LS-DYNA reports an input error and the dyna.str file (see *CONTROL_STRUCTURED) reveals that one or more variables are incorrectly written to dyna.str as a series of asterisks due to inadequate field length(s) in dyna.str.

The "long=**k**" option really serves no practical purpose.

When long format is invoked for keyword input, all variables that are less than or equal to 20 characters in standard format are expanded to or remain at 20 characters. There is one exception to the previous rule for alphanumeric variables of 20 characters in length. A20 in standard format becomes A40 in long format. Alphanumeric variables that are longer than A20 in standard format are extended to A160 in long format. The number of input lines in long format remains unchanged from the standard format; instead, only the length of each line changes. For example, long format for the card corresponding to the _ID option of a keyword becomes (I20, A160), whereas standard format is (I10, A70).

Allowing a 20-character field for a numerical variable does not mean that it will allow for 20 significant digits. There are more fundamental limitations at work, particularly when running in single precision. The absolute value of an integer in single precision cannot be larger than $2^{31} - 1 = 2{,}147{,}483{,}647$. The mantissa of a floating point value cannot exceed 2^{23} in single precision and cannot exceed 2^{52} in double precision.

To convert a standard format input deck to a long format input deck, run LS-DYNA with "newformat = long" on the execution line. For example, if standard.k is a standard format input deck,

LS-DYNA i=**standard.k** newformat=**long**

will create a long format input deck called standard.k.long.

To mix the long and standard formats within one input deck, use "+" or "-" signs within the deck. If the execution line indicates standard format, appending "+" to the end of a keyword invokes long format just for that keyword. For example, "*NODE +" in place of "*NODE" invokes a read format of (I20, 3F20, 2I20).

Similarly, if the execution line indicates long format, appending "-" to the end of a keyword invokes standard format for that keyword. For example, "*NODE –" in place of "*NODE" invokes the standard read format of one line per node (I8, 3F16, 2I8).

To take this idea a step further, adding a "-" or "+" to the end of the *INCLUDE keyword command signals to LS-DYNA that all the commands in the included file are standard format or long format, respectively.

I10 Format Input:

In some cases, the user may be satisfied in extending 8-character input fields where they exist in standard format input to 10-character fields. If that is all that is required, then invoking the so-called "I10 format" input would suffice.

I10 format input is invoked by appending "i10=**y**" to the execution line or else by adding "i10=**y**" to the *KEYWORD command in the input deck. Or you can invoke I10 format for specific keywords in an input deck by adding the percent sign "%" to the end of the keyword. For example, "*NODE %" in place of "*NODE" invokes a read format of (I10, 3F16, 2I10) instead of the standard (I8, 3F16, 2I8).

To convert a standard format input deck to an I10 format input deck, run LS-DYNA with "newformat = i10" on the execution line. For example, if standard.k is a standard format input deck,

LS-DYNA i=**standard.k** newformat=**i10**

will create an I10 format input deck called standard.k.i10.

*AIRBAG

Purpose: Define an airbag or control volume.

The keyword *AIRBAG provides a way of defining thermodynamic behavior of the gas flow into the airbag as well as a reference configuration for the fully inflated bag. The keyword cards in this section are defined in alphabetical order:

AIRBAG_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}

*AIRBAG_ALE

*AIRBAG_CPG

*AIRBAG_INTERACTION

*AIRBAG_PARTICLE

*AIRBAG_REFERENCE_GEOMETRY

*AIRBAG_SALE

*AIRBAG_SHELL_REFERENCE_GEOMETRY

***AIRBAG**_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}*

OPTION1 specifies one of the following thermodynamic models for modeling airbags using a control volume (CV) approach:

SIMPLE_PRESSURE_VOLUME

SIMPLE_AIRBAG_MODEL

ADIABATIC_GAS_MODEL

WANG_NEFSKE

WANG_NEFSKE_JETTING

WANG_NEFSKE_MULTIPLE_JETTING

LOAD_CURVE

LINEAR_FLUID

HYBRID

HYBRID_JETTING

HYBRID_CHEMKIN

FLUID_AND_GAS

OPTION2 specifies that an additional line of data is read for the WANG_NEFSKE type thermodynamic relationships. The additional data controls the initiation of exit flow from the airbag. *OPTION2* takes the single option:

POP

OPTION3 specifies that a constant momentum formulation is used to calculate the jetting load on the airbag an additional line of data is read in. *OPTION3* takes the single option:

CM

OPTION4, given by:

ID

specifies that an airbag ID and heading information will be the first card of the airbag definition. This ID is a unique number that is necessary for the identification of the airbags in the definition of airbag interaction using *AIRBAG_INTERACTION keyword. The numeric IDs and heading are written into the abstat and d3hsp files.

Core Cards: Common to all airbags

ID Card. Additional card for the ID keyword option. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

Card ID	1	2	3	4	5	6	7	8
Variable	ABID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF
Type	I	I	I	F	F	F	F	F
Default	none	0	0	1.	1.	0.	0.	0.
Remark			optional					

VARIABLE	DESCRIPTION
ABID	Airbag ID. This must be a unique number.
HEADING	Airbag descriptor. It is suggested that unique descriptions be used.
SID	Set ID
SIDTYP	Set type: EQ.0: segment NE.0: part set ID
RBID	Rigid body part ID for user defined activation subroutine: LT.0: -RBID is taken as the rigid body part ID. Built in sensor subroutine initiates the inflator. Load curves are offset by initiation time. EQ.0: the control volume is active from time zero.

VARIABLE	DESCRIPTION
	GT.0: RBID is taken as the rigid body part ID. User sensor subroutine initiates the inflator. Load curves are offset by initiation time. See Appendix D.
VSCA	Volume scale factor (default = 1.0)
PSCA	Pressure scale factor (default = 1.0)
VINI	Initial filled volume
MWD	Mass weighted damping factor, D
SPSF	Stagnation pressure scale factor, $0 \le \gamma \le 1$

Remarks:

Card 1 is necessary for all airbag options. The option dependent cards follow.

Lumped parameter control volumes are a mechanism for determining volumes of closed surfaces and applying a pressure based on some thermodynamic relation. The volume is specified by a list of polygons similar to the pressure boundary condition cards or by specifying a material subset which represents shell elements that form the closed boundary. All polygon normal vectors must be oriented to face *outwards* from the control volume; however for *AIRBAG_PARTICLE, which does not rely on control volumes, all polygon normal vectors must be oriented to face *inwards* to get proper volume (see *AIRBAG_PARTICLE). If holes are detected, they are assumed to be covered by planar surfaces.

There are two sets of volume and pressure variables used for each control volume model. First, the finite element model computes a volume V_{femodel} and applies a pressure P_{femodel}. The thermodynamics of a control volume may be computed in a different unit system with its own set of variables: V_{cvolume} and P_{cvolume} which are used for integrating the differential equations for the control volume. The conversion is as follows:

$$V_{\text{cvolume}} = (\text{VSCA} \times V_{\text{femodel}}) - \text{VINI}$$
$$P_{\text{femodel}} = \text{PSCA} \times P_{\text{cvolume}}$$

where VSCA, PSCA, and VINI are input parameters. Damping can be applied to the structure enclosing a control volume by using a mass weighted damping formula:

$$F_i^d = m_i D\left(v_i - v_{\text{cg}}\right) \ ,$$

where F_i^d is the damping force, m_i is the nodal mass, v_i is the velocity for a node, v_{cg} is the mass weighted average velocity of the structure enclosing the control volume, and D is the damping factor.

An alternative, separate damping is based on the stagnation pressure. The stagnation pressure is roughly the maximum pressure on a flat plate oriented normal to a steady state flow field. The stagnation pressure is defined as $p = \gamma\rho V^2$, where V is the normal velocity of the control volume relative to the ambient velocity, ρ is the ambient air density, and γ is a factor which varies from 0 to 1 that must be chosen by the user. Small values are recommended to avoid excessive damping.

Sensor Input:

The sensor is mounted on a rigid body which is attached to the structure. *The motion of the sensor is evaluated in the local coordinate system of the rigid body. See *MAT_RIGID.* This local system rotates and translates with the rigid material. The default local system for a rigid body is taken as the principal axes of the inertia tensor.

When the user defined criterion for airbag deployment is satisfied, a flag is set and deployment begins. All load curves relating to the mass flow rate as a function of time are then shifted by the initiation time.

RBID = 0: No Rigid Body

For this case there is no rigid body, and the control volume is active from time zero. There are no additional sensor cards.

RBID > 0: User Supplied Sensor Subroutine

The value of RBID is taken as a rigid body part ID, and a user supplied sensor subroutine will be called to determine the flag that initiates deployment. See Appendix D for details regarding the user supplied subroutine. For RBID > 0 the additional cards are specified below:

User Subroutine Control Card. This card is read in when RBID > 0.

Card 2a	1	2	3	4	5	6	7	8
Variable	N							
Type	I							
Default	none							

User Subroutine Constant Cards. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

Card 2a.1	1	2	3	4	5	6	7	8
Variable	C1	C2	C3	C4	C5			
Type	F	F	F	F	F			
Default	0.	0.	0.	0.	0.			

VARIABLE	DESCRIPTION
N	Number of input parameters (not to exceed 25)
C1, …, CN	Up to 25 constants for the user subroutine

RBID < 0: User supplied sensor subroutine

The value of –RBID is taken as rigid body part ID and a built in sensor subroutine is called. For RBID < 0, there are three additional cards.

Acceleration Sensor Card.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	AX	AY	AZ	AMAG	TDUR			
Type	F	F	F	F	F			
Default	0.	0.	0.	0.	0.			

Velocity Sensor Card.

Card 2b.2	1	2	3	4	5	6	7	8
Variable	DVX	DVY	DVZ	DVMAG				
Type	F	F	F	F				
Default	0.	0.	0.	0.				

Displacement Sensor Card.

Card 2b.3	1	2	3	4	5	6	7	8
Variable	UX	UY	UZ	UMAG				
Type	F	F	F	F				
Default	0.	0.	0.	0.				

VARIABLE	DESCRIPTION
AX	Acceleration level in the local *x*-direction to activate inflator. The absolute value of the *x*-acceleration is used. EQ.0: inactive
AY	Acceleration level in local the *y*-direction to activate inflator. The absolute value of the *y*-acceleration is used. EQ.0: inactive
AZ	Acceleration level in the local *z*-direction to activate inflator. The absolute value of the *z*-acceleration is used. EQ.0: inactive
AMAG	Acceleration magnitude required to activate inflator. EQ.0: inactive
TDUR	Time duration acceleration must be exceeded before the inflator activates. This is the cumulative time from the beginning of the calculation, meaning it is not continuous.

VARIABLE	DESCRIPTION
DVX	Velocity change in the local x-direction to activate the inflator. (The absolute value of the velocity change is used.) EQ.0: inactive
DVY	Velocity change in the local y-direction to activate the inflator. (The absolute value of the velocity change is used.) EQ.0: inactive
DVZ	Velocity change in the local z-direction to activate the inflator. (The absolute value of the velocity change is used.) EQ.0: inactive
DVMAG	Velocity change magnitude required to activate the inflator. EQ.0: inactive
UX	Displacement increment in the local x-direction to activate the inflator. (The absolute value of the x-displacement is used.) EQ.0: inactive
UY	Displacement increment in the local y-direction to activate the inflator. (The absolute value of the y-displacement is used.) EQ.0: inactive
UZ	Displacement increment in the local z-direction to activate the inflator. (The absolute value of the z-displacement is used.) EQ.0: inactive
UMAG	Displacement magnitude required to activate the inflator. EQ.0: inactive

*AIRBAG_SIMPLE_PRESSURE_VOLUME_{OPTION}

Card Summary:

The additional card required for the SIMPLE_PRESSURE_VOLUME keyword is Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

CN	BETA	LCID	LCIDDR				

Data Card Definitions:

Additional card for SIMPLE_PRESSURE_VOLUME option. (For card 1 see the "core cards" section of *AIRBAG.)

Card 3	1	2	3	4	5	6	7	8
Variable	CN	BETA	LCID	LCIDDR				
Type	F	F	I	I				
Default	none	none	none	0				

VARIABLE	DESCRIPTION
CN	Coefficient. Define if the load curve ID, LCID, is unspecified. LT.0.0: |CN| is the load curve ID, which defines the coefficient as a function of time.
BETA	Scale factor, β. Define if a load curve ID is not specified.
LCID	Optional load curve ID defining pressure as a function of relative volume.
LCIDDR	Optional load curve ID defining the coefficient, CN, as a function of time during the dynamic relaxation phase.

Remarks:

Pressure is calculated using the following:

$$\text{Pressure} = \frac{\beta \times \text{CN}}{\text{Relative Volume}}$$

$$\text{Relative Volume} = \frac{\text{Current Volume}}{\text{Initial Volume}}$$

The pressure, therefore, is a function of the ratio of current volume to the initial volume. The constant, CN, is used to establish a relationship known from the literature. The scale factor β is simply used to scale the given values. This simple model can be used when an initial pressure is given and no leakage, no temperature, and no input mass flow is assumed. A typical application is the modeling of air in automobile tires.

The load curve, LCIDDR, can be used to ramp up the pressure during the dynamic relaxation phase to avoid oscillations after the desired gas pressure is reached. In the DE-

FINE_CURVE section this load curve must be flagged for dynamic relaxation. After initialization either the constant or load curve ID, |CN|, is used to determine the pressure.

*AIRBAG_SIMPLE_AIRBAG_MODEL_{OPTION}

Card Summary:

The additional cards required for the SIMPLE_AIRBAG_MODEL option begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

CV	CP	T	LCID	MU	AREA	PE	RO

Card 4a. This card is included if CV = 0.

LOU	T_EXT	A	B	MW	GASC		

Card 4b. This card is included if CV ≠ 0.

LOU							

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	CV	CP	T	LCID	MU	AREA	PE	RO
Type	F	F	F	I	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
CV	Heat capacity at constant volume (e.g., Joules/kg/°K). See Remark 3.
CP	Heat capacity at constant pressure (e.g., Joules/kg/°K). See Remark 3.
T	Temperature of input gas
LCID	Load curve ID specifying input mass flow rate. See *DEFINE_CURVE. See Remark 2.
MU	Shape factor for exit hole, μ (see Remark 2): LT.0.0: $\lvert\mu\rvert$ is the load curve number defining the shape factor as a function of absolute pressure.
AREA	Exit area, A (see Remark 2): GE.0.0: A is the exit area and is constant in time. LT.0.0: $\lvert A\rvert$ is the load curve number defining the exit area as a function of absolute pressure.
PE	Ambient pressure, p_e
RO	Ambient density, ρ

Card 4a	1	2	3	4	5	6	7	8
Variable	LOU	T_EXT	A	B	MW	GASC		
Type	I	F	F	F	F	F		
Default	0	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
LOU	Optional load curve ID giving mass flow out as a function of gauge pressure in bag. See *DEFINE_CURVE. See Remark 2.
T_EXT	Ambient temperature
A	First heat capacity coefficient of inflator gas (e.g., Joules/mole/^oK)
B	Second heat capacity coefficient of inflator gas, (e.g., Joules/mole/$^oK^2$)
MW	Molecular weight of inflator gas (e.g., kg/mole)
GASC	Universal gas constant of inflator gas (e.g., 8.314 Joules/mole/^oK)

Card 4b	1	2	3	4	5	6	7	8
Variable	LOU							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
LOU	Optional load curve ID giving mass flow out as a function of gauge pressure in bag. See *DEFINE_CURVE. See Remark 2.

Remarks:

1. **Airbag pressure.** The gamma law equation of state used to determine the pressure in the airbag is given by:

$$p = (\gamma - 1)\rho e \ ,$$

where p is the pressure, ρ is the density, e is the specific internal energy of the gas, and γ is the ratio of the specific heats, such that

$$\gamma = \frac{c_p}{c_v} \ .$$

Here c_v is the specific heat at constant volume, and c_p is the specific heat at constant pressure. See Remark 3. A pressure relation is defined as:

$$Q = \frac{p_e}{p} \ ,$$

where p_e is the external pressure and p is the internal pressure in the bag. A critical pressure relationship is defined as:

$$Q_{\text{crit}} = \left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma}{\gamma - 1}},$$

where γ is the ratio of specific heats as defined above and

$$Q \le Q_{\text{crit}} \Rightarrow Q = Q_{\text{crit}} \ .$$

2. **Leakage.** From conservation of mass, the time rate of change of mass flowing into the bag is given as:

$$\frac{dM}{dt} = \frac{dM_{\text{in}}}{dt} - \frac{dM_{\text{out}}}{dt} \ .$$

The inflow mass flow rate is given by the load curve ID, LCID. Leakage, the mass flow rate out of the bag, can be modeled in two alternative ways. One is to give an exit area with the corresponding shape factor. In this case, the load curve ID, LOU, must be set to zero. The mass flow rate is given by:

$$\frac{dM_{\text{out}}}{dt} = \mu A \sqrt{2p\rho} \sqrt{\frac{\gamma\left(Q^{\frac{2}{\gamma}} - Q^{\frac{\gamma+1}{\gamma}}\right)}{\gamma - 1}}$$

where μ is the shape factor for the exit hole, A is the exit area, p is the pressure, ρ is the density, γ is the ratio of specific heats (see Remark 1), and Q is the ratio of the external pressure to the internal pressure (see Remark 1). The other method is to define a mass flow out by a load curve, LOU. In this case, μ and A must both be set to zero.

3. **Specific heats.** If CV = 0. then the constant-pressure specific heat is given by:

$$c_p = \frac{(a + bT)}{\text{MW}} \ ,$$

and the constant-volume specific heat is then found from:

$$c_v = c_p - \frac{R}{\mathrm{MW}} \ .$$

*AIRBAG_ADIABATIC_GAS_MODEL_{OPTION}

Card Summary:

The additional cards required for the ADIABATIC_GAS_MODEL option begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

PSF	LCID	GAMMA	P0	PE	R0		

Card 3	1	2	3	4	5	6	7	8
Variable	PSF	LCID	GAMMA	P0	PE	R0		
Type	F	I	F	F	F	F		
Default	1.0	none	none	none	none	none		

VARIABLE	DESCRIPTION
PSF	Pressure scale factor
LCID	Optional load curve for preload flag. See *DEFINE_CURVE.
GAMMA	Ratio of specific heats
P0	Initial pressure (gauge)
PE	Ambient pressure
RO	Initial density of gas

Remarks:

The optional load curve ID, LCID, defines a preload flag. During the preload phase the function value of the load curve versus time is zero, and the pressure in the control volume is given as:

$$p = \text{PSF} \times p_0$$

When the *first nonzero* function value is encountered, the preload phase stops and the ideal gas law applies for the rest of the analysis. If LCID is zero, no preload is performed.

The gamma law equation of state for the adiabatic expansion of an ideal gas is used to determine the pressure after preload:

$$p = (\gamma - 1)\rho e \ ,$$

where p is the pressure, ρ is the density, e is the specific internal energy of the gas, and γ is the ratio of the specific heats:

$$\gamma = \frac{c_p}{c_v} \ .$$

The pressure above is the absolute pressure. The resultant pressure acting on the control volume is:

$$p_s = \text{PSF} \times (p - p_e)$$

where PSF is the pressure scale factor. Starting from the initial pressure p_0 an initial internal energy is calculated:

$$e_0 = \frac{p_0 + p_e}{\rho(\gamma - 1)} \ .$$

***AIRBAG_WANG_NEFSKE**_{OPTIONS}

Card Summary:

The additional cards required for WANG_NEFSKE, WANG_NEFSKE_JETTING, and WANG_NEFSKE_MULTIPLE_JETTING option begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if the ID keyword option is used. To use the *AIRBAG_-INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

CV	CP	T	LCT	LCMT	TVOL	LCDT	IABT

Card 4. This card is required.

C23	LCC23	A23	LCA23	CP23	LCCP23	AP23	LCAP23

Card 5. This card is required.

PE	RO	GC	LCEFR	POVER	PPOP	OPT	KNKDN

Card 6. If the inflator is modeled (LCMT = 0), fill in the following card. If not, include but leave blank.

IOC	IOA	IVOL	IRO	IT	LCBF		

Card 7. Include this card when CV = 0.

TEXT	A	B	MW	GASC	HCONV		

Card 8. This card is included if and only if the POP keyword option is used.

TDP	AXP	AYP	AZP	AMAGP	TDURP	TDA	RBIDP

Card 9a. This card is included if the WANG_NEFKSE_JETTING keyword option is used.

XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	CA	BETA

Card 9b. This card is included if the WANG_NEFSKE_MULTIPLE_JETTING keyword option is used.

XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	LCJRV	BETA

Card 10. This card is included if either the WANG_NEFKSE_JETTING or the WANG_-NEFKSE_MULTIPLE_JETTING keyword option is used.

XSJFP	YSJFP	ZSJFP	PSID	ANGLE	NODE1	NODE2	NODE3

Card 11. The is card is included if the CM keyword option is used.

NREACT							

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	CV	CP	T	LCT	LCMT	TVOL	LCDT	IABT
Type	F	F	F	I	I	F	I	F
Default	none	none	0.	0	none	0.	0.	not used

VARIABLE	DESCRIPTION
CV	Specific heat at constant volume, e.g., Joules/kg/°K
CP	Specific heat at constant pressure, e.g., Joules/kg/°K
T	Temperature of input gas. For temperature variations a load curve, LCT, may be defined.
LCT	Optional load curve ID defining temperature of input gas as a function of time. This overrides T.
LCMT	Load curve specifying input mass flow rate or tank pressure versus time. If the tank volume, TVOL, is nonzero the curve ID is assumed to be tank pressure as a function of time. If LCMT = 0, then the inflator must be modeled; see Card 6. During the dynamic relaxation phase the airbag is ignored unless the curve is flagged to act during dynamic relaxation.
TVOL	Tank volume which is required only for the tank pressure as a function of time curve, LCMT.
LCDT	Load curve for time rate of change of temperature (dT/dt) as a function of time.
IABT	Initial airbag temperature. (Optional, generally not defined.)

Card 4	1	2	3	4	5	6	7	8
Variable	C23	LCC23	A23	LCA23	CP23	LCCP23	AP23	LCAP23
Type	F	I	F	I	F	I	F	I
Default	none	0	none	0	none	0	0.0	0

VARIABLE	DESCRIPTION
C23	Vent orifice coefficient which applies to exit hole. Set to zero if LCC23 is defined below.
LCC23	The absolute value, |LCC23|, is a load curve ID. If the ID is positive, the load curve defines the vent orifice coefficient which applies to exit hole as a function of time. If the ID is negative, the vent orifice coefficient is defined as a function of relative pressure, $P_{\text{air}}/P_{\text{bag}}$;

VARIABLE	DESCRIPTION
	see [Anagonye and Wang 1999]. In addition, LCC23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for C23 overrides LCC23.
A23	Vent orifice area: GT.0.0: Vent orifice area which applies to exit hole LT.0.0: The absolute value |A23| is a part ID, see [Anagonye and Wang, 1999]. With LCA23 = -1, |A23| is a part set representing venting holes. The venting leakage through each venting hole represented by a part in part set |A23| is output to abstat. The area of this part/set becomes the vent orifice area. With SIDTYP > 0, airbag pressure will not be applied to part/set |A23| representing venting holes if part/set |A23| is not included in SID, the part set representing the airbag. EQ.0.0: Set A23 to zero if LCA23 is ≠ 0.
LCA23	Load curve defining the vent orifice area which applies to exit hole as a function of *absolute* pressure, or LCA23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for A23 overrides LCA23.
CP23	Orifice coefficient for leakage (fabric porosity). Set to zero if LCCP23 is defined below.
LCCP23	Load curve defining the orifice coefficient for leakage (fabric porosity) as a function of time, or LCCP23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for CP23 overrides LCCP23.
AP23	Area for leakage (fabric porosity)
LCAP23	Load curve defining the area for leakage (fabric porosity) as a function of (absolute) pressure, or LCAP23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for AP23 overrides LCAP23.

Card 5	1	2	3	4	5	6	7	8
Variable	PE	RO	GC	LCEFR	POVER	PPOP	OPT	KNKDN
Type	F	F	F	I	F	F	F	I
Default	none	none	none	0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
PE	Ambient pressure
RO	Ambient density
GC	Gravitational conversion constant (mandatory - no default). If consistent units are being used for all parameters in the airbag definition, then unity should be input.
LCEFR	Optional curve for exit flow rate (mass/time) as a function of (gauge) pressure
POVER	Initial relative overpressure (gauge), P_{over} in control volume
PPOP	Pop Pressure: relative pressure (gauge) for initiating exit flow, P_{pop}
OPT	Fabric venting option. If nonzero, CP23, LCCP23, AP23, and LCAP23 are set to zero. Porosity leakage of each material is output to abstat. EQ.1: Wang-Nefske formulas for venting through an orifice are used. Blockage is not considered. EQ.2: Wang-Nefske formulas for venting through an orifice are used. Blockage of venting area due to contact is considered. EQ.3: Leakage formulas of Graefe, Krummheuer, and Siejak [1990] are used. Blockage is not considered. EQ.4: Leakage formulas of Graefe, Krummheuer, and Siejak [1990] are used. Blockage of venting area due to contact is considered. EQ.5: Leakage formulas based on flow through a porous media are used. Blockage is not considered. EQ.6: Leakage formulas based on flow through a porous media

VARIABLE	DESCRIPTION
	are used. Blockage of venting area due to contact is considered.
	EQ.7: Leakage is based on gas volume outflow as a function of pressure load curve. Blockage of flow area due to contact is not considered. Absolute pressure is used in the porous-velocity-versus-pressure load curve, given as FAC(p) in the *MAT_FABRIC card.
	EQ.8: Leakage is based on gas volume outflow as a function of pressure load curve. Blockage of flow area due to contact is considered. Absolute pressure is used in the porous-velocity-versus-pressure load curve, given as FAC(p) in the *MAT_FABRIC card.
KNKDN	*Optional* load curve ID defining the knock down pressure scale factor as a function of time. This option only applies to jetting. The scale factor defined by this load curve scales the pressure applied to airbag segments which do not have a clear line-of-sight to the jet. Typically, at very early times this scale factor will be less than unity and equal to unity at later times. The full pressure is always applied to segments which can see the jets.

Inflator Card. If the inflator is modeled (LCMT = 0), fill in the following card. *If not, include but leave blank.*

Card 6	1	2	3	4	5	6	7	8
Variable	IOC	IOA	IVOL	IRO	IT	LCBF		
Type	F	F	F	F	F	I		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
IOC	Inflator orifice coefficient
IOA	Inflator orifice area
IVOL	Inflator volume
IRO	Inflator density

VARIABLE	DESCRIPTION
IT	Inflator temperature
LCBF	Load curve defining burn fraction as a function of time

Temperature Dependent Heat Capacities Card. Include this card when CV = 0.

Card 7	1	2	3	4	5	6	7	8
Variable	TEXT	A	B	MW	GASC	HCONV		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
TEXT	Ambient temperature
A	First molar heat capacity coefficient of inflator gas (e.g., Joules/mole/K)
B	Second molar heat capacity coefficient of inflator gas, (e.g., Joules/mole/K^2)
MW	Molecular weight of inflator gas (e.g., Kg/mole).
GASC	Universal gas constant of inflator gas (e.g., 8.314 Joules/mole/K)
HCONV	Effective heat transfer coefficient between the gas in the air bag and the environment at temperature TEXT. If HCONV < 0, then HCONV defines a load curve of data pairs (time, hconv).

Criteria for Initiating Exit Flow Card. Additional card for the POP option to the *AIRBAG_WANG_NEFSKE card.

Card 8	1	2	3	4	5	6	7	8
Variable	TDP	AXP	AYP	AZP	AMAGP	TDURP	TDA	RBIDP
Type	F	F	F	F	F	F	F	I
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	none

VARIABLE	DESCRIPTION
TDP	Time delay before initiating exit flow after pop pressure is reached
AXP	Pop acceleration magnitude in the local x-direction. EQ.0.0: Inactive
AYP	Pop acceleration magnitude in the local y-direction. EQ.0.0: Inactive.
AZP	Pop acceleration magnitude in the local z-direction. EQ.0.0: Inactive
AMAGP	Pop acceleration magnitude. EQ.0.0: Inactive
TDURP	Time duration pop acceleration must be exceeded to initiate exit flow. This is a cumulative time from the beginning of the calculation, that is, it is not continuous.
TDA	Time delay before initiating exit flow after pop acceleration is exceeded for the prescribed time duration.
RBIDP	Part ID of the rigid body for checking accelerations against pop accelerations

Remarks:

The gamma law equation of state for the adiabatic expansion of an ideal gas is used to determine the pressure after preload:

$$p = (\gamma - 1)\rho e$$

where p is the pressure, ρ is the density, e is the specific internal energy of the gas, and γ is the ratio of the specific heats:

$$\gamma = \frac{c_p}{c_v} \ .$$

Here c_v is the specific heat at constant volume, and c_p is the specific heat at constant pressure. A pressure relation is defined as:

$$Q = \frac{p_e}{p} \ ,$$

where p_e is the external pressure and p is the internal pressure in the bag. A critical pressure relationship is defined as:

$$Q_{\text{crit}} = \left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma}{\gamma - 1}},$$

where γ is the ratio of specific heats as defined above and

$$Q \le Q_{\text{crit}} \Rightarrow Q = Q_{\text{crit}} \ .$$

Wang and Nefske define the mass flow through the vents and leakage by

$$\dot{m}_{23} = C_{23} A_{23} \frac{p}{R\sqrt{T_2}} Q^{\frac{1}{\gamma}} \sqrt{2 g_c \left(\frac{\gamma R}{\gamma - 1}\right)\left(1 - Q^{\frac{\gamma - 1}{\gamma}}\right)}$$

and

$$\dot{m}'_{23} = C'_{23} A'_{23} \frac{p}{R\sqrt{T_2}} Q^{\frac{1}{\gamma}} \sqrt{2 g_c \left(\frac{\gamma R}{\gamma - 1}\right)\left(1 - Q^{\frac{\gamma - 1}{\gamma}}\right)} \ .$$

It must be noted that the gravitational conversion constant must be given in consistent units. As an alternative to computing the mass flow out of the bag by the Wang-Nefske model, a curve for the exit flow rate depending on the internal pressure can be taken. Then, no definitions for C23, LCC23, A23, LCA23, CP23, LCCP23, AP23, and LCAP23 are necessary.

The airbag inflator assumes that the control volume of the inflator is constant and that the amount of propellant reacted can be defined by the user as a tabulated curve of fraction reacted versus time. A pressure relation is defined as:

$$Q_{\text{crit}} = \frac{p_c}{p_I} = \left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma}{\gamma - 1}} \ ,$$

where p_c is a critical pressure at which sonic flow occurs, and p_I is the inflator pressure. The exhaust pressure is given by

$$p_e = \begin{cases} p_a & \text{if } p_a \ge p_c \\ p_c & \text{if } p_a < p_c \end{cases}$$

where p_a is the pressure in the control volume. The mass flow into the control volume is governed by the equation:

$$\dot{m}_{\text{in}} = C_0 A_0 \sqrt{2 p_I \rho_I} \sqrt{\frac{g_c \gamma \left(Q^{\frac{2}{\gamma}} - Q^{\frac{\gamma+1}{\gamma}} \right)}{\gamma - 1}}$$

where C_0, A_0, and ρ_I are the inflator orifice coefficient, area, and gas density, respectively.

If OPT is defined, then for OPT set to 1 or 2 the mass flow rate out of the bag, $\dot{m}_{out}$ is given by:

$$\dot{m}_{out} = \sqrt{g_c} \left\{ \sum_{n=1}^{\text{nairmats}} [\text{FLC}(t)_n \times \text{FAC}(p)_n \times \text{Area}_n] \right\} \sqrt{2p\rho} \sqrt{\frac{\gamma \left(Q^{\frac{2}{\gamma}} - Q^{\frac{\gamma+1}{\gamma}} \right)}{\gamma - 1}}$$

where, ρ is the density of airbag gas, "nairmats" is the number of fabrics used in the airbag, and Area_n is the current unblocked area of fabric number n.

If OPT is set to 3 or 4 then:

$$\dot{m}_{\text{out}} = \left\{ \sum_{n=1}^{\text{nairmats}} [\text{FLC}(t)_n \times \text{FAC}(p)_n \times \text{Area}_n] \right\} \sqrt{2(p - p_{\text{ext}})\rho}$$

while for OPT set to 5 or 6:

$$\dot{m}_{\text{out}} = \left\{ \sum_{n=1}^{\text{nairmats}} [\text{FLC}(t)_n \times \text{FAC}(p)_n \times \text{Area}_n] \right\} (p - p_{\text{ext}}) \quad .$$

For OPT set to 7 or 8 (may be comparable to an equivalent model ALE model):

$$\dot{m}_{\text{out}} = \sum_{n=1}^{\text{nairmats}} \text{FLC}(t)_n \times \text{FAC}(p)_n \times \text{Area}_n \times \rho_n \quad .$$

Note that for different OPT settings, $\text{FAC}(p)_n$ has different meanings (all units shown just as demonstrations):

1. For OPT of 1, 2, 3 and 4, $\text{FAC}(p)$ is unit-less.
2. For OPT of 5 and 6, $\text{FAC}(p)$ has a unit of (s/m).
3. For OPT of 7 or 8, $\text{FAC}(p)$ is the gas volume outflow through a unit area per unit time thus has the unit of speed,

$$[\text{FAC}(p)] = \frac{[\text{volume}]}{[\text{area}][t]} = \frac{[\text{L}]^3}{[\text{L}]^2[t]} = \frac{[L]}{[t]} = [\text{velocity}].$$

Multiple airbags may share the same part ID since the area summation is over the airbag segments whose corresponding part IDs are known. Currently, we assume that no more

than ten materials are used per bag for purposes of the output. This constraint can be eliminated if necessary.

The total mass flow out will include the portion due to venting, that is, constants C23 and A23 or their load curves above.

If CV = 0. then the constant-pressure specific heat is given by:

$$c_p = \frac{(a + bT)}{MW}$$

and the constant-volume specific heat is then found from:

$$c_v = c_p - \frac{R}{MW}$$

Additional Cards required for JETTING models:

The following additional cards are defined for the WANG_NEFSKE_JETTING and WANG_NEFSKE_MULTIPLE_JETTING options, two further cards are defined for each option. The jet may be defined by specifying either the coordinates of the jet focal point, jet vector head and secondary jet focal point, or by specifying three nodes located at these positions. The nodal point option is recommended when the location of the airbag changes as a function of time.

WANG_NEFSKE_JETTING Card. Include this card if the WANG_NEFSKE_JETTING keyword option is used.

Card 9a	1	2	3	4	5	6	7	8
Variable	XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	CA	BETA
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	1.0
Remark	1	1	1	1	1	1		

WANG_NEFSKE_MULTIPLE_JETTING Card. Include this card if the WANG_-NEFSKE_MULTIPLE_JETTING keyword option is used.

Card 9b	1	2	3	4	5	6	7	8
Variable	XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	LCJRV	BETA
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	1.0
Remark	1	1	1	1	1	1		

VARIABLE	DESCRIPTION
XJFP	x-coordinate of jet focal point, meaning the virtual origin in Figures 3-1 and 3-2.
YJFP	y-coordinate of jet focal point, meaning the virtual origin in Figures 3-1 and 3-2.
ZJFP	z-coordinate of jet focal point, meaning the virtual origin in Figures 3-1 and 3-2.
XJVH	x-coordinate of jet vector head to defined code centerline
YJVH	y-coordinate of jet vector head to defined code centerline
ZJVH	z-coordinate of jet vector head to defined code centerline
CA	Cone angle, α, defined in radians. LT.0.0: $\lvert\alpha\rvert$ is the load curve ID defining cone angle as a function of time
LCJRV	Load curve ID giving the spatial jet relative velocity distribution; see Figures 3-1, 3-2, and 3-3. The jet velocity is determined from the inflow mass rate and scaled by the load curve function value corresponding to the value of the angle ψ. Typically, the values on the load curve vary between 0 and unity. See *DEFINE_CURVE.
BETA	Efficiency factor, β, which scales the final value of pressure obtained from Bernoulli's equation. LT.0.0: $\lvert\beta\rvert$ is the load curve ID defining the efficiency factor as a

VARIABLE	DESCRIPTION
	function of time

Jetting Card. This card is included if the either the WANG_NEFSKE_JETTING or WANG_NEFSKE_MULTIPLE_JETTING keyword option is used.

Card 10	1	2	3	4	5	6	7	8
Variable	XSJFP	YSJFP	ZSJFP	PSID	ANGLE	NODE1	NODE2	NODE3
Type	F	F	F	I	F	I	I	I
Default	none	none	none	none	none	0	0	0
Remark						1	1	1

VARIABLE	DESCRIPTION
XSJFP	x-coordinate of secondary jet focal point, passenger side bag. If the coordinates of the secondary point are (0,0,0) then a conical jet (driver's side airbag) is assumed.
YSJFP	y-coordinate of secondary jet focal point
ZSJFP	z-coordinate of secondary jet focal point
PSID	Optional part set ID; see *SET_PART. If zero, all elements are included in the airbag.
ANGLE	Cutoff angle in degrees. The relative jet velocity is set to zero for angles greater than the cutoff. See Figure 3-3. This option applies to the MULTIPLE jet only.
NODE1	Node ID located at the jet focal point, that is, the virtual origin in Figures 3-1 and 3-2
NODE2	Node ID for node along the axis of the jet
NODE3	Optional node ID located at secondary jet focal point

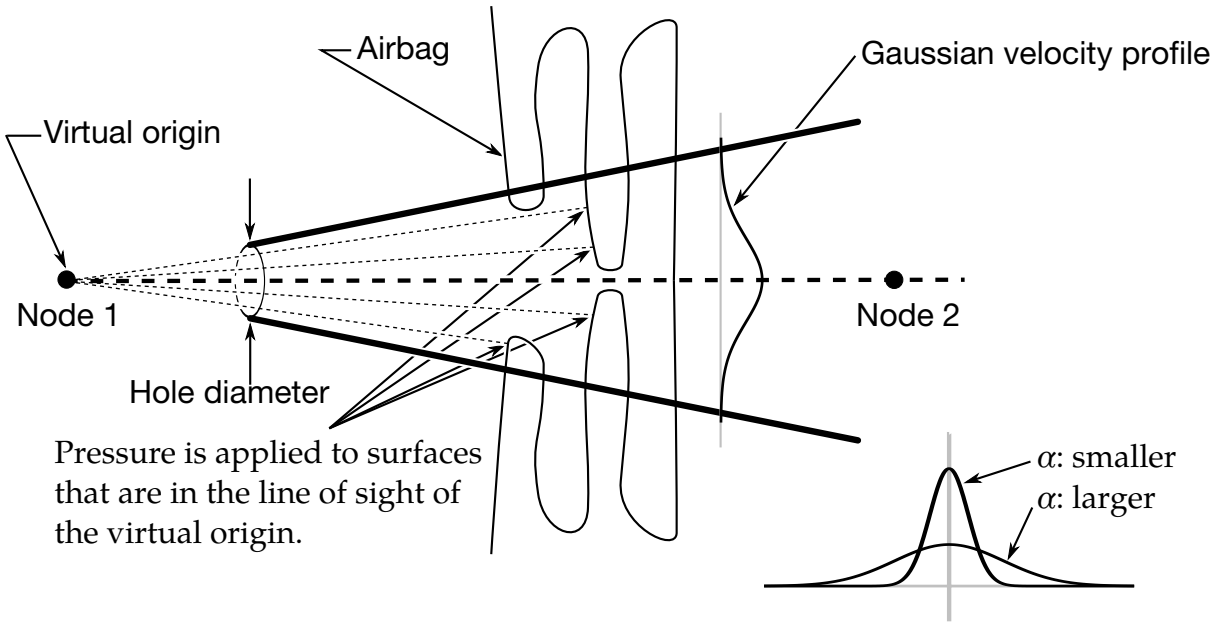


Figure 3-1. Jetting configuration for driver's side airbag (pressure applied only if centroid of surface is in line-of-sight)

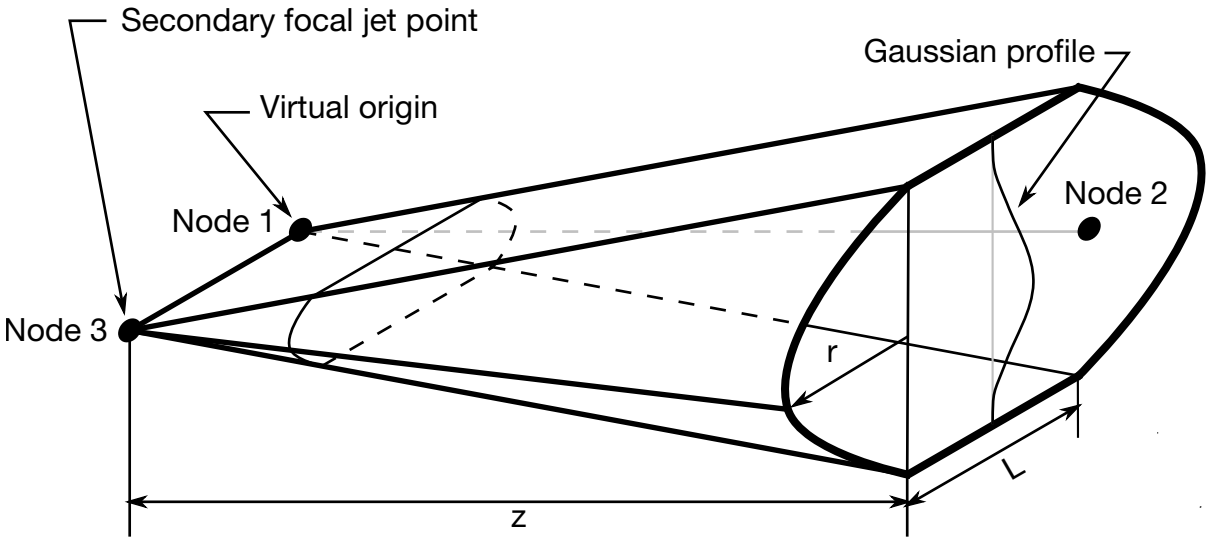


Figure 3-2. Jetting configuration for the passenger's side bag.

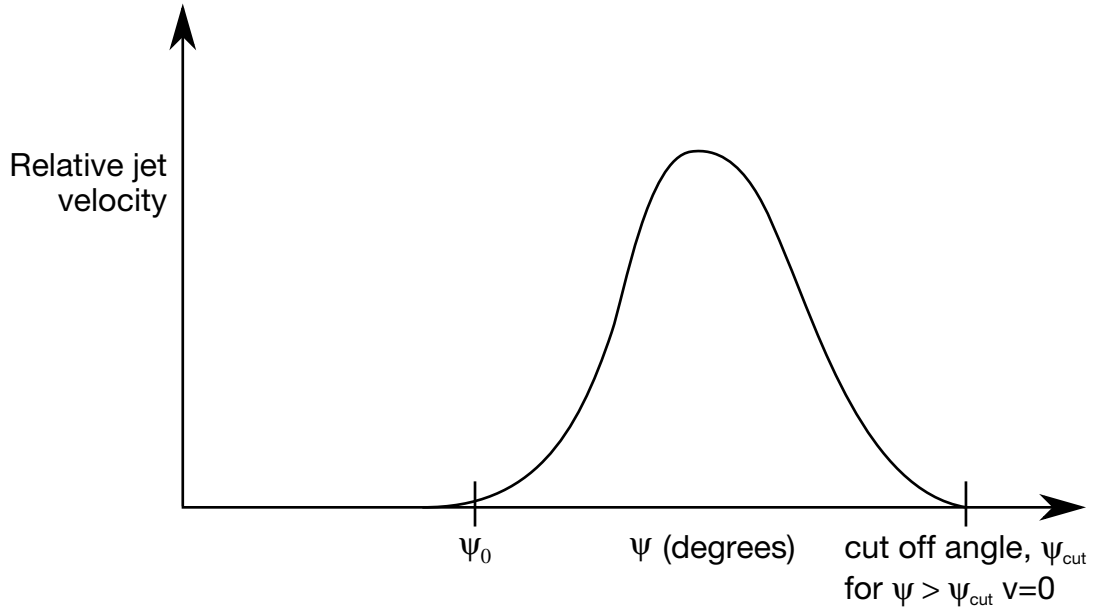


Figure 3-3. Normalized jet velocity versus angle for multiple jet driver's side airbag

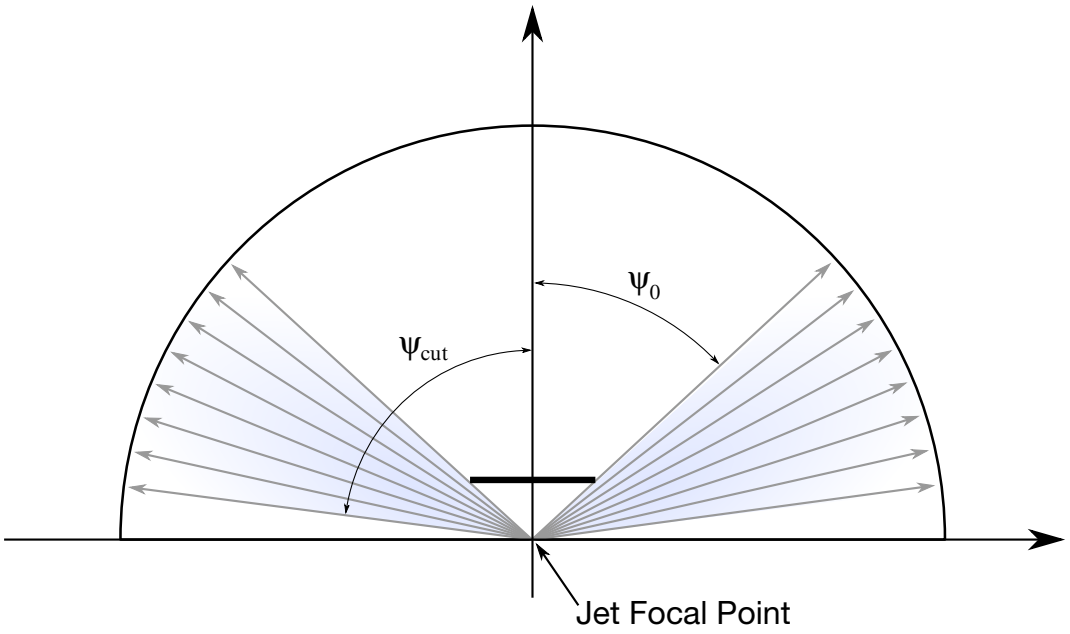


Figure 3-4. Multiple jet model for driver's side airbag. Typically, ψ_{cut} (see input ANGLE) is close to 90°. The angle ψ_0 is included to indicate that there is some angle below which the jet is negligible; see Figure 3-3.

Remarks:

1. **Jet Direction.** It is assumed that the jet direction is defined by the coordinate method (XJFP, YJFP, ZJFP) and (XJVH, YJVH, ZJVH) unless both NODE1 and NODE2 are defined. In which case the coordinates of the nodes give by NODE1, NODE2 and NODE3 will override (XJFP, YJFP, ZJFP) and (XJVH, YJVH, ZJVH). The use of nodes is recommended if the airbag system is undergoing rigid body motion. The nodes should be attached to the vehicle to allow for the coordinates of the jet to be continuously updated with the motion of the vehicle.

2. **Pressure Distribution.** The jetting option provides a simple model to simulate the real pressure distribution in the airbag during the breakout and early unfolding phase. Only the surfaces that are in the line of sight to the virtual origin have an increased pressure applied. With the optional load curve LCRJV, the pressure distribution with the code can be scaled according to the so-called relative jet velocity distribution.

3. **Jetting Configuration.** For passenger side airbags the cone is replaced by a wedge type shape. The first and secondary jet focal points define the corners of the wedge and the angle α then defines the wedge angle.

4. **Surfaces for Applying Pressure.** Instead of applying pressure to all surfaces in the line of sight of the virtual origin(s), a part set can be defined to which the pressure is applied.

5. **Jet Focal Point Placement.** Care must be used to place the jet focal point within the bag. If the focal point is outside the bag, inside surfaces will not be visible so jetting pressure will not be applied correctly.

Additional Card required for CM option:

The following additional card is defined for the WANG_NEFSKE_JETTING_CM and WANG_NEFSKE_MULTIPLE_JETTING_CM options.

CM Card. Additional card required for CM keyword option.

Card 11	1	2	3	4	5	6	7	8
Variable	NREACT							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
NREACT	Node for reacting jet force. If zero, the jet force will not be applied.

Remarks:

Compared with the standard LS-DYNA jetting formulation, the Constant Momentum option has several differences. Overall, the jetting usually has a more significant effect on airbag deployment than the standard LS-DYNA jetting: the total force is often greater and does not reduce with distance from the jet.

The velocity at the jet outlet is assumed to be a choked (sonic) adiabatic flow of a perfect gas. Therefore the velocity at the outlet is given by:

$$v_{\text{outlet}} = \sqrt{\gamma RT} = \sqrt{\frac{(c_p - c_v)Tc_p}{c_v}} \ .$$

The density in the nozzle is then calculated from conservation of mass flow.

$$\dot{m} = \rho_0 v_{\text{outlet}} A_{\text{outlet}}$$

This is different from the standard LS-DYNA jetting formulation, which assumes that the density of the gas in the jet is the same as atmospheric air, and then calculates the jet velocity from conservation of mass flow.

The velocity distribution at any radius, r, from the jet centerline and distance, z, from the focus, $v_{r,z}$, relates to the velocity of the jet centerline, $v_{r=0,z}$ in the same way as the standard LS-DYNA jetting options:

$$v_{r,z} = v_{r=0,z} e^{-\left(\frac{r}{\alpha z}\right)^2}$$

The velocity at the jet centerline, $v_r = 0$, at the distance, z, from the focus of the jet is calculated such that the momentum in the jet is conserved.

$$\begin{aligned} \text{momentum at nozzle} &= \text{momentum at z} \\ \rho_0 v_{\text{outlet}}^2 A_{\text{outlet}} &= \rho_0 \int v_{\text{jet}}^2 dA_{\text{jet}} \\ &= \rho_0 v_{r=0,Z}^2 \left\{ b + F\sqrt{b} \right\} \end{aligned}$$

where, $b = \frac{\pi(\alpha z)^2}{2}$, and F is the distance between the jet foci (for a passenger jet).

Finally, the pressure exerted on an airbag element in view of the jet is given as:

$$p_{r,z} = \beta \rho_0 v_{r,z}^2$$

By combining the equations above

$$p_{r,z} = \frac{\beta \dot{m} v_{\text{outlet}} \left[e^{-(r/\alpha z)^2} \right]^2}{\left\{ \frac{\pi(\alpha z)^2}{2} + F\sqrt{\frac{\pi(\alpha z)^2}{2}} \right\}}$$

The total force exerted by the jet is given by

$$F_{\text{jet}} = \dot{m} v_{\text{outlet}} \ ,$$

which is independent of the distance from the nozzle. Mass flow in the jet is not necessarily conserved, because gas is entrained into the jet from the surrounding volume. By contrast, the standard LS-DYNA jetting formulation conserves mass flow but not momentum. This has the effect of making the jet force reduce with distance from the nozzle.

The jetting forces can be reacted onto a node (NREACT), to allow the reaction force through the steering column or the support brackets to be modeled. The jetting force is written to the ASCII abstat file and the binary xtf file.

***AIRBAG_LOAD_CURVE**_{OPTION}_

Card Summary:

The additional card required for the LOAD_CURVE option is Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

STIME	LCID	RO	PE	PO	T	TO	

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	STIME	LCID	RO	PE	P0	T	T0	
Type	F	I	F	F	F	F	F	
Default	0.0	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
STIME	Time at which pressure is applied. LCID is offset by this amount.
LCID	Load curve ID defining pressure as a function of time; see *DEFINE_CURVE.
RO	Initial density of gas (ignored if LCID > 0)
PE	Ambient pressure (ignored if LCID > 0)
P0	Initial gauge pressure (ignored if LCID > 0)
T	Gas Temperature (ignored if LCID > 0)
T0	Absolute zero on temperature scale (ignored if LCID > 0)

Remarks:

Within this simple model the control volume is inflated with a pressure defined as a function of time or calculated using the following equation if LCID = 0.

$$P_{\text{total}} = C\rho(T - T_0)$$

$$P_{gauge} = P_{\text{total}} - P_{\text{ambient}}$$

The pressure is uniform throughout the control volume.

*AIRBAG_LINEAR_FLUID_{OPTION}

Card Summary:

The additional cards required for the LINEAR_FLUID option begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

BULK	RO	LCINT	LCOUTT	LCOUTP	LCFIT	LCBULK	LCID

Card 4. This card is optional.

P_LIMIT	P_LIMLC	NONULL					

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	BULK	RO	LCINT	LCOUTT	LCOUTP	LCFIT	LCBULK	LCID
Type	F	F	I	I	I	I	I	I
Default	none	none	none	optional	optional	optional	optional	none

VARIABLE	DESCRIPTION
BULK	K, bulk modulus of the fluid in the control volume. Constant as a function of time. Define if LCBULK = 0.
RO	ρ, density of the fluid: GT.0.0: Constant value LT.0.0: |RO| refers to a load curve ID for a curve giving density as a function of time.
LCINT	$F(t)$, input flow curve defining mass per unit time as a function of time; see *DEFINE_CURVE.
LCOUTT	$G(t)$, output flow curve defining mass per unit time as a function of time. This load curve is optional.
LCOUTP	$H(p)$, output flow curve defining mass per unit time as a function of pressure. This load curve is optional.
LFIT	$L(t)$, added pressure as a function of time. This load curve is optional.
LCBULK	Curve defining the bulk modulus as a function of time. This load curve is optional, but if defined, the constant, BULK, is not used.
LCID	Load curve ID defining pressure as a function of time. See Remark 1.

Card 4 is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	P_LIMIT	P_LIMLC	NONULL					
Type	F	I	I					
Default	optional	optional	optional					

VARIABLE	DESCRIPTION
P_LIMIT	Limiting value on total pressure (optional)
P_LIMLC	Curve defining the limiting pressure value as a function of time. If nonzero, P_LIMIT is ignored.
NONULL	A flag determining whether pressure is applied on null material parts (see Remark 2): EQ.0: Apply pressure everywhere inside the airbag. NE.0: Do not apply pressure on null material part of the airbag.

Remarks:

1. **Pressure and mass flow rate.** This model is intended for the simulation of hydroforming processes or similar problems. The pressure is controlled by the mass flowing into the volume and by the current volume. The pressure is uniformly applied to the control volume.

   If LCID = 0, then the pressure is determined from:

$$P(t) = K(t)\ln\left[\frac{V_0(t)}{V(t)}\right] + L(t) \ ,$$

   where

$$\begin{aligned} P(t) &= \text{Pressure,} \\ V(t) &= \text{Volume of fluid in compressed state,} \\ V_{0(t)} &= V_0(t) \\ &= \frac{M(t)}{\rho} \\ &= \text{Volume of fluid in uncompressed state,} \\ M(t) &= M(0) + \int F(t)dt - \int G(t)dt - \int H(p)dt \\ &= \text{Current fluid mass,} \end{aligned}$$

$$M(0) = V(0)\rho$$
$$= \text{Mass of fluid at time zero } P(0) = 0.$$

By setting LCID ≠ 0, a pressure time history may be specified for the control volume, and the mass of fluid within the volume is then calculated from the volume and density.

Note the signs used in the equation for $M(t)$. The mass flow should always be defined as positive since the output flow is subtracted.

2. **NONULL.** The NONULL flag is useful for hydroforming simulations. In this type of simulation, the part set that makes up the airbag will usually include a part ID of null shells, defined by *MAT_NULL. The null shells together with a deformable sheet blank will form an airbag, which upon pressurization, will push the blank into a die cavity. This feature is available in SMP as of Dev 136254.

***AIRBAG_HYBRID**_{OPTIONS}

***AIRBAG_HYBRID_JETTING**_{OPTIONS}

Card Summary:

The additional cards required for HYBRID and HYBRID_JETTING options begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if the ID keyword option is used. To use the *AIRBAG_-INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

ATMOST	ATMOSP	ATMOSD	GC	CC	HCONV		

Card 4. This card is required.

C23	LCC23	A23	LCA23	CP23	LCP23	AP23	LCAP23

Card 5. This card is required.

OPT	PVENT	NGAS	LCEFR	LCIDM0	VNTOPT		

Card 5.1. Include NGAS pairs of Cards 5.1 and 5.2.

LCIDM	LCIDT		MW	INITM	A	B	C

Card 5.2. Include NGAS pairs of Cards 5.1 and 5.2.

FMASS							

Card 6. This card is included for HYBRID_JETTING only.

XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	CA	BETA

Card 7. This card is included for HYBRID_JETTING only.

XSJFP	YSJFP	ZSJFP	PSID	IDUM	NODE1	NODE2	NODE3

Card 8. This card is included if the CM keyword option is used with HYBRID_JETTING.

NREACT							

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	ATMOST	ATMOSP	ATMOSD	GC	CC	HCONV		
Type	F	F	F	F	F	F		
Default	none	none	none	none	1.0	none		

VARIABLE	DESCRIPTION
ATMOST	Atmospheric temperature
ATMOSP	Atmospheric pressure
ATMOSD	Atmospheric density
GC	Universal molar gas constant

VARIABLE	DESCRIPTION
CC	Conversion constant EQ.0.0: Set to 1.0
HCONV	Effective heat transfer coefficient between the gas in the air bag and the environment at temperature at ATMOST. If HCONV < 0.0, then HCONV defines a load curve of data pairs (time, hconv).

Card 4	1	2	3	4	5	6	7	8
Variable	C23	LCC23	A23	LCA23	CP23	LCP23	AP23	LCAP23
Type	F	I	F	I	F	I	F	I
Default	none	0	none	0	none	0	none	0

VARIABLE	DESCRIPTION
C23	Vent orifice coefficient which applies to exit hole. Set to zero if LCC23 is defined below.
LCC23	The absolute value, |LCC23|, is a load curve ID. If the LCC23 is positive, the load curve defines the vent orifice coefficient which applies to the exit hole as a function of time. If the LCC23 is negative, the vent orifice coefficient is defined as a function of relative pressure, P_{air}/P_{bag}; see [Anagonye and Wang 1999]. In addition, LCC23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for C23 overrides LCC23.
A23	If defined as a positive number, A23 is the vent orifice area which applies to the exit hole. If defined as a negative number, with LCA23 ≠ -1, the absolute value |A23| is a part ID; see [Anagonye and Wang 1999]. With LCA23 = -1, a negative A23 represents venting holes by a part set |A23|. The venting leakage through each venting hole represented by a part in part set |A23| is output to abstat. The area of this part/set becomes the vent orifice area. With SIDTYP > 0, airbag pressure will not be applied to part/set |A23| representing venting holes if part/set |A23| is not included in SID, the part set representing the airbag. Set A23 to zero if a positive LCA23 is defined below.
LCA23	Load curve ID defining the vent orifice area which applies to exit

VARIABLE	DESCRIPTION
	hole as a function of *absolute* pressure, or LCA23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for A23 overrides LCA23.
CP23	Orifice coefficient for leakage (fabric porosity). Set to zero if LCCP23 is defined below.
LCCP23	Load curve ID defining the orifice coefficient for leakage (fabric porosity) as a function of time, or LCCP23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for CP23 overrides LCCP23.
AP23	Area for leakage (fabric porosity)
LCAP23	Load curve ID defining the area for leakage (fabric porosity) as a function of (absolute) pressure, or LCAP23 can be defined through *DEFINE_CURVE_FUNCTION. A nonzero value for AP23 overrides LCAP23.

Card 5	1	2	3	4	5	6	7	8
Variable	OPT	PVENT	NGAS	LCEFR	LCIDM0	VNTOPT		
Type	I	F	I	I	I	I		
Default	none	none	none	0	0	0		

VARIABLE	DESCRIPTION
OPT	Fabric venting option; if nonzero CP23, LCCP23, AP23, and LCAP23 are set to zero. Porosity leakage of each material is output to abstat. EQ.1: Wang-Nefske formulas for venting through an orifice are used. Blockage is not considered. EQ.2: Wang-Nefske formulas for venting through an orifice are used. Blockage of venting area due to contact is considered. EQ.3: Leakage formulas of Graefe, Krummheuer, and Siejak [1990] are used. Blockage is not considered. EQ.4: Leakage formulas of Graefe, Krummheuer, and Siejak

VARIABLE	DESCRIPTION
	[1990] are used. Blockage of venting area due to contact is considered. EQ.5: Leakage formulas based on flow through a porous media are used. Blockage due to contact is not considered. EQ.6: Leakage formulas based on flow through a porous media are used. Blockage due to contact is considered. EQ.7: Leakage is based on gas volume outflow versus pressure load curve. Blockage of flow area due to contact is not considered. Absolute pressure is used in the porous-velocity-versus-pressure load curve, given as FAC(P) in the *MAT_FABRIC card. EQ.8: Leakage is based on gas volume outflow versus pressure load curve. Blockage of flow area due to contact is considered.
PVENT	Gauge pressure when venting begins
NGAS	Number of gas inputs to be defined below (including initial air). The maximum number of gases is 17.
LCEFR	Optional curve for exit flow rate (mass/time) versus (gauge) pressure
LCIDM0	Optional curve representing inflator's total mass inflow rate. When defined, LCIDM in the following 2 × NGAS cards defines the molar fraction of each gas component as a function of time and INITM defines the initial molar ratio of each gas component.
VNTOPT	Additional options for venting area definition. For A23 ≥ 0 EQ.1: Vent area is equal to A23. EQ.2: Vent area is A23 plus the eroded surface area of the airbag parts. EQ.10: Same as VNTOPT = 2 For A23 < 0 EQ.1: Vent area is the increase in surface area of part |A23|. If there is no change in surface area of part |A23| over the initial surface area or if the surface area reduces from the initial area, there is no venting.

VARIABLE	DESCRIPTION
	EQ.2: Vent area is the total (not change in) surface area of part |A23| plus the eroded surface area of all other parts comprising the airbag.
	EQ.10: Vent area is the increase in surface area of part |A23| plus the eroded surface area of all other parts comprising the airbag.

Include NGAS pairs of Cards 5.1 and 5.2

Card 5.1	1	2	3	4	5	6	7	8
Variable	LCIDM	LCIDT		MW	INITM	A	B	C
Type	I	I		F	F	F	F	F
Default	none	none		none	none	none	none	none

Include NGAS pairs of Cards 5.1 and 5.2.

Card 5.2	1	2	3	4	5	6	7	8
Variable	FMASS							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
LCIDM	Load curve ID for inflator mass flow rate (equal to 0 for gas in the bag at time = 0)
	GT.0: Piecewise linear interpolation
	LT.0: Cubic spline interpolation
LCIDT	Load curve ID for inflator gas temperature (equal to 0 for gas in the bag at time = 0)
	GT.0: Piecewise linear interpolation

VARIABLE	DESCRIPTION
	LT.0: Cubic spline interpolation
MW	Molecular weight
INITM	Initial mass fraction of gas component present in the airbag, prior to injection of gas by the inflator. INITM is used to calculate the averaged gas properties and the initial mass of all gas components, assuming that the airbag is initially at ambient temperature and under ambient pressure. The sum of INITM of all gas components should be 1.0.
A	Coefficient for molar heat capacity of inflator gas at constant pressure, (e.g., Joules/mole/^oK)
B	Coefficient for molar heat capacity of inflator gas at constant pressure, (e.g., Joules/mole/$^oK^2$)
C	Coefficient for molar heat capacity of inflator gas at constant pressure, (e.g., Joules/mole/$^oK^3$)
FMASS	Fraction of additional aspirated mass

Additional Cards required for HYBRID_JETTING and HYBRID_JETTING_CM:

The following two additional cards are defined for the HYBRID_JETTING options. The jet may be defined by specifying either the coordinates of the jet focal point, jet vector head and secondary jet focal point, or by specifying three nodes located at these positions. The nodal point option is recommended when the location of the airbag changes as a function of time.

Card 6	1	2	3	4	5	6	7	8
Variable	XJFP	YJFP	ZJFP	XJVH	YJVH	ZJVH	CA	BETA
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none
Remark	1	1	1	1	1	1		

VARIABLE	DESCRIPTION
XJFP	x-coordinate of jet focal point, that is., the virtual origin in Figures 3-1 and 3-2. See Remark 1 below.
YJFP	y-coordinate of jet focal point, that is, the virtual origin in Figures 3-1 and 3-2.
ZJFP	z-coordinate of jet focal point, that is, the virtual origin in Figures 3-1 and 3-2.
XJVH	x-coordinate of jet vector head to defined cone centerline
YJVH	y-coordinate of jet vector head to defined cone centerline
ZJVH	z-coordinate of jet vector head to defined cone centerline
CA	Cone angle, α, defined in radians. LT.0.0: $\lvert\alpha\rvert$ is the load curve ID defining cone angle as a function of time
BETA	Efficiency factor, β, which scales the final value of pressure obtained from Bernoulli's equation. LT.0.0: $\lvert\beta\rvert$ is the load curve ID defining the efficiency factor as a function of time

Card 7	1	2	3	4	5	6	7	8
Variable	XSJFP	YSJFP	ZSJFP	PSID	IDUM	NODE1	NODE2	NODE3
Type	F	F	F	I	F	I	I	I
Default	none	none	none	none	none	0	0	0
Remark					2	1	1	1

VARIABLE	DESCRIPTION
XSJFP	x-coordinate of secondary jet focal point, passenger side bag. If the coordinate of the secondary point is (0,0,0), then a conical jet (driver's side airbag) is assumed.

VARIABLE	DESCRIPTION
YSJFP	y-coordinate of secondary jet focal point
ZSJFP	z-coordinate of secondary jet focal point
PSID	Optional part set ID; see *SET_PART. If zero, all elements are included in the airbag.
IDUM	Dummy field (variable not used)
NODE1	Node ID located at the jet focal point, that is, the virtual origin in Figures 3-1 and 3-2 . See Remark 1 below.
NODE2	Node ID for node along the axis of the jet. See Figures 3-1 and 3-2.
NODE3	Optional node ID located at secondary jet focal point. See Figure 3-2.

Additional card required for HYBRID_JETTING_CM option.

Card 8	1	2	3	4	5	6	7	8
Variable	NREACT							
Type	I							
Default	none							
Remark	4							

VARIABLE	DESCRIPTION
NREACT	Node for reacting jet force. If zero the jet force will not be applied.

Remarks:

1. **Jetting.** It is assumed that the jet direction is defined by the coordinate method (XJFP, YJFP, ZJFP) and (XJVH, YJVH, ZJVH) unless both NODE1 and NODE2 are defined. In which case the coordinates of the nodes given by NODE1, NODE2 and NODE3 will override (XJFP, YJFP, ZJFP) and (XJVH, YJVH, ZJVH). The use of nodes is recommended if the airbag system is undergoing rigid body motion. The nodes should be attached to the vehicle to

allow for the coordinates of the jet to be continuously updated with the motion of the vehicle.

The jetting option provides a simple model to simulate the real pressure distribution in the airbag during the breakout and early unfolding phase. Only the surfaces that are in the line of sight to the virtual origin have an increased pressure applied. With the optional load curve LCRJV, the pressure distribution with the code can be scaled according to the so-called relative jet velocity distribution.

For passenger side airbags the cone is replaced by a wedge type shape. The first and secondary jet focal points define the corners of the wedge and the angle α then defines the wedge angle.

Instead of applying pressure to all surfaces in the line of sight of the virtual origin(s), a part set can be defined to which the pressure is applied.

2. **IDUM.** This variable is not used and has been included to maintain the same format as the WANG_NEFSKE_JETTING options.

3. **Focal Point Placement.** Care must be used to place the jet focal point within the bag. If the focal point is outside the bag, inside surfaces will not be visible so jetting pressure will not be applied correctly.

4. **NREACT.** See the description related to the WANG_NEFSKE_JETTING_CM option. For the hybrid inflator model the heat capacities are compute from the combination of gases which inflate the bag.

*AIRBAG_HYBRID_CHEMKIN_{OPTION}

Card Summary:

The HYBRID_CHEMKIN model includes 3 control cards (Cards 3 – 5). For each gas species an additional set of cards must follow consisting of a control card (Card 6) and several thermodynamic property data cards. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

LCIDM	LCIDT	NGAS	DATA	ATMT	ATMP	RG	

Card 4. This card is required.

HCONV							

Card 5. This card is required.

C23	A23						

Card 6. For each gas species (see NGAS on Card 3), include one of this card followed by several thermo-dynamic property data cards whose format depends on the DATA field on Card 3. The next keyword ("*") card terminates this input.

CHNAME	MW	LCIDN	FMOLE	FMOLET			

Card 7a.1. This card is included if and only if DATA = 1.

TLOW	TMID	THIGH					

Card 7a.2. This card is included if and only if DATA = 1.

ALOW	BLOW	CLOW	DLOW	ELOW	FLOW	HLOW	

Card 7a.3. This card is included if and only if DATA = 1.

AHIGH	BHIGH	CHIGH	DHIGH	EHIGH	FHIGH	HHIGH	

Card 7b. This card is included if and only if DATA = 3.

A	B	C	D	E			

Data Card Definitions:

Card 3	1	2	3	4	5	6	7	8
Variable	LCIDM	LCIDT	NGAS	DATA	ATMT	ATMP	RG	
Type	I	I	I	I	F	F	F	

VARIABLE	DESCRIPTION
LCIDM	Load curve specifying input mass flow rate as a function of time: GT.0: piece wise linear interpolation LT.0: cubic spline interpolation

VARIABLE	DESCRIPTION
LCIDT	Load curve specifying input gas temperature as a function of time: GT.0: piece wise linear interpolation LT.0: cubic spline interpolation
NGAS	Number of gas inputs to be defined below (including initial air)
DATA	Thermodynamic database (see remarks): EQ.1: NIST database (3 additional property cards are required below) EQ.2: CHEMKIN database (no additional property cards are required) EQ.3: Polynomial data (1 additional property card is required below)
ATMT	Atmospheric temperature
ATMP	Atmospheric pressure
RG	Universal gas constant

Card 4	1	2	3	4	5	6	7	8
Variable	HCONV							
Type	F							
Default	0.							

VARIABLE	DESCRIPTION
HCONV	Effective heat transfer coefficient between the gas in the air bag and the environment at temperature ATMT. If HCONV < 0, then HCONV defines a load curve of data pairs (time, hconv).

Card 5	1	2	3	4	5	6	7	8
Variable	C23	A23						
Type	F	F						
Default	0.	0.						

VARIABLE	DESCRIPTION
C23	Vent orifice coefficient
A23	Vent orifice area

Gas Species Control Card. For each gas species include a set of cards consisting of this card followed by several thermo-dynamic property data cards whose format depends on the DATA field on Card 3. The next keyword ("*") card terminates this input.

Card 6	1	2	3	4	5	6	7	8
Variable	CHNAME	MW	LCIDN	FMOLE	FMOLET			
Type	A	F	I	F	F			
Default	none	none	0	none	0.			

VARIABLE	DESCRIPTION
CHNAME	Chemical symbol for this gas species (e.g., N2 for nitrogen, AR for argon). Required for DATA = 2 (CHEMKIN), optional for DATA = 1 or DATA = 3.
MW	Molecular weight of this gas species.
LCIDN	Load curve specifying the input mole fraction versus time for this gas species. If > 0, FMOLE is not used.
FMOLE	Mole fraction of this gas species in the inlet stream.
FMOLET	Initial mole fraction of this gas species in the tank.

NIST Data Cards. Include this card if DATA = 1. The required data can be found on the NIST web site at http://webbook.nist.gov/chemistry/.

Card 7a.1	1	2	3	4	5	6	7	8
Variable	TLOW	TMID	THIGH					
Type	F	F	F					
Default	none	none	none					

NIST Data Cards. Include this card if DATA = 1. The required data can be found on the NIST web site at http://webbook.nist.gov/chemistry/.

Card 7a.2	1	2	3	4	5	6	7	8
Variable	ALOW	BLOW	CLOW	DLOW	ELOW	FLOW	HLOW	
Type	F	F	F	F	F	F	F	
Default	none	none	none	none	none	none	none	

NIST Data Cards. Include this card if DATA = 1. The required data can be found on the NIST web site at http://webbook.nist.gov/chemistry/.

Card 7a.3	1	2	3	4	5	6	7	8
Variable	AHIGH	BHIGH	CHIGH	DHIGH	EHIGH	FHIGH	HHIGH	
Type	F	F	F	F	F	F	F	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
TLOW	Curve fit low temperature limit
TMID	Curve fit low-to-high transition temperature
THIGH	Curve fit high temperature limit

VARIABLE	DESCRIPTION
ALOW, …, HLOW	Low temperature range NIST polynomial curve fit coefficients (see remarks)
AHIGH, …, HIGH	High temperature range NIST polynomial curve fit coefficients (see remarks)

Polynomial Fit Card. This card is included if DATA = 3.

Card 7b	1	2	3	4	5	6	7	8
Variable	A	B	C	D	E			
Type	F	F	F	F	F			
Default	none	0.	0.	0.	0.			

VARIABLE	DESCRIPTION
A	Coefficient; see remarks.
B	Coefficient; see remarks.
C	Coefficient; see remarks.
D	Coefficient; see remarks.
E	Coefficient; see remarks.

Remarks:

Specific heat curve fits:

$$\text{NIST:} \quad c_p = \frac{1}{M}\left(a + bT + cT^2 + dT^3 + \frac{e}{T^2}\right)$$

$$\text{CHEMKIN:} \quad c_p = \frac{\overline{R}}{M}\left(a + bT + cT^2 + dT^3 + eT^4\right)$$

$$\text{POLYNOMIAL:} \quad c_p = \frac{1}{M}\left(a + bT + cT^2 + dT^3 + eT^4\right)$$

where,

$$\overline{R} = \text{universal gas constant } 8.314 \frac{\text{Nm}}{\text{mole} \times K}$$

$$M = \text{gas molecular weight}$$

*AIRBAG_FLUID_AND_GAS_{OPTION}

Card Summary:

The additional cards required for the *AIRBAG_FLUID_AND_GAS option begin with Card 3. See the "core cards" section of *AIRBAG for cards preceding Card 3.

Card ID. This card is included if and only if the ID keyword option is used. To use the *AIRBAG_INTERACTION keyword ID Cards are required.

ABID	HEADING

Card 1. This card is required.

SID	SIDTYP	RBID	VSCA	PSCA	VINI	MWD	SPSF

Card 2a. This card is read when RBID > 0.

N							

Card 2a.1. This card is read when RBID > 0. Define N constants for the user subroutine. Include only the number of cards necessary, that is, for nine constants use 2 cards.

C1	C2	C3	C4	C5			

Card 2b.1. This card is read when RBID < 0.

AX	AY	AZ	AMAG	TDUR			

Card 2b.2. This card is read when RBID < 0.

DVX	DVY	DVZ	DVMAG				

Card 2b.3. This card is read when RBID < 0.

UX	UY	UZ	UMAG				

Card 3. This card is required.

XWINI	XWADD	XW	P	TEND	RHO	LCXW	LCP

Card 4. This card is required.

GDIR	NPROJ	IDIR	IIDIR	KAPPA	KBM	GRAV	

Data Card Definitions:

Additional cards required for FLUID_AND_GAS option. (For card 1 see the "core cards" section of *AIRBAG.) Currently this option only works in SMP and explicit analysis.

Card 3	1	2	3	4	5	6	7	8
Variable	XWINI	XWADD	XW	P	TEND	RHO	LCXW	LCP
Type	F	F	F	F	F	F	I	I
Default	none	none	none	none	none	none	none	none

Card 4	1	2	3	4	5	6	7	8
Variable	GDIR	NPROJ	IDIR	IIDIR	KAPPA	KBM	GRAV	
Type	F	I	I	I	F	F	F	
Default	none	3	none	none	1.0	none	10.0	

VARIABLE	DESCRIPTION
XWINI	Fluid level at time $t = 0$ in |GDIR| direction
XWADD	Fluid level filling increment per time step
XW	Final fluid level in filling process
P	Gas pressure at time t = TEND
TEND	Time when gas pressure P is reached
RHO	Density of the fluid (for example, for water, RHO ≈ 1000.0 kg/m^3)
LCXW	Load curve ID for fluid level as a function of time. XW, XWADD, and XWINI are ignored with this option.
LCP	Load curve ID for gas pressure as a function of time. P and TEND are ignored with this option.

VARIABLE	DESCRIPTION
GDIR	Global direction of gravity. Use negative numbers for the negative direction, such as -3.0 for the negative global z-axis. EQ.1.0: Global x-direction EQ.2.0: Global y-direction EQ.3.0: Global z-direction
NPROJ	Number of projection directions (only global axis) for volume calculation
IDIR	First direction of projection (if |NPROJ| ≠ 3), only global axis.
IIDIR	Second direction of projection (if |NPROJ| = 2), only global axis.
KAPPA	Adiabatic exponent
KBM	Bulk modulus of the fluid (e.g. for water, BKM ≈ 2080 N/mm^2)
GRAV	Gravity. The default is 10.0, which is only correct when the units are m/s^2.

Remarks:

The *AIRBAG_FLUID_AND_GAS option models a quasi-static multi-chamber fluid/gas structure interaction in a simplified way, including three possible load cases: (i) only gas, (ii) only incompressible fluid, or (iii) the combination of incompressible fluid with additional gas "above." See Figure 3-5.

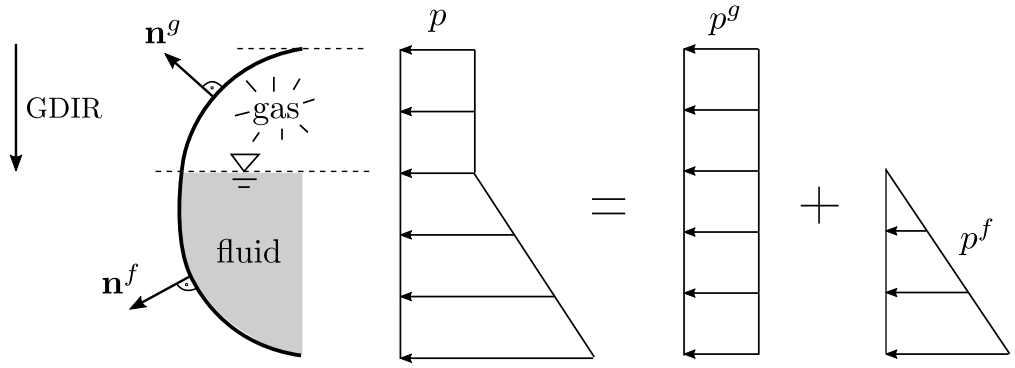


Figure 3-5. Hydrostatic pressure distribution in a chamber filled with gas and incompressible fluid

The theory is based on the description of gases and fluids as energetically equivalent pressure loads. The calculation of the fluid volume is carried out using the directions of projection and a non-normalized normal vector. This model, therefore, requires that the normal of the shell elements belonging to a filled structure must point outwards. Holes are not detected, but can be taken into account as described below.

In case of a pure gas (no fluid), the *AIRBAG_SIMPLE_PRESSURE_VOLUME and *AIRBAG_FLUID_AND_GAS cards give identical results as they are based on the same theory. The update of the gas pressure due to volume change is calculated with the following simple gas law:

$$p^g = \left(1 - \text{KAPPA} \times \frac{v^g - v^g_{\text{old}}}{v^g_{\text{old}}}\right) p^g_{\text{old}}$$

with adiabatic exponent KAPPA and gas volume v^g.

The theory of incompressible fluids is based on the variation of the potential energy and an update of the water level due to changes in the volume and the water surface, see Haßler and Schweizerhof [2007], Haßler and Schweizerhof [2008], Rumpel and Schweizerhof [2003], and Rumpel and Schweizerhof [2004].

In case of multiple fluid/gas filled chambers each chamber requires an additional *AIRBAG_FLUID_AND_GAS card. Some of the parameters, which are called local parameters, only belong to a single chamber (such as gas pressure). In contrast, global parameters belong to all chambers (such as the direction of gravitation).

Because the theory only applies to quasi-static fluid-structure interaction, the load must be applied slowly, so the kinetic energy is almost zero throughout the process.

All parameters input on Card 3 are local parameters describing filling the chamber. The water level and the gas pressure can be defined by curves using LCXW and LCP or by using the parameters XWINI, XW, XWADD, P, and TEND. When describing the fluid and gas filling using the parameters, the gas pressure at time $t = 0$ is set to 0, and the initial water level is set to XWINI in |GDIR|-direction. At each timestep, XWADD is added to the water level until XW is reached. The gas pressure will be raised until P is reached at time $t =$ TEND.

In general, global parameters belong to all chambers. To describe the global axis in GDIR, NPROJ, IDIR, and IIDIR, the following mapping applies: the x-axis is axis 1, the y-axis is axis 2, and the z-axis is axis 3.

The gas and fluid volume is calculated by using contour integrals in the global x-, y- and z-coordinates. If one of the boundaries is discontinuous in one or two global directions, these directions must be ignored in NPROJ, IDIR, and IIDIR. At least one direction of projection must be set (NPROJ = 1, IDIR = value), but it is recommended to use as many directions of projection as possible.

In case of a structure filled exclusively with fluid, IDIR and IIDIR should not be set to |GDIR|. In case of holes in a structure (for example, take advantage of symmetry planes), IDIR and IIDIR should not be set to the normal direction of the plane describing the hole or symmetry plane.

An example of a water-filled tube structure illustrating how to use NPROJ, IDIR, IIDIR, and GDIR is shown in Figure 3-6. In this example, gravity is acting opposite to the global z-axis. In this case, then, GDIR = -3. The structure is filled exclusively with water, so the projection direction cannot be set to |GDIR| = 3. To use the symmetry of the tube, only half of the structure has been modeled. The normal of the symmetry plane is in the y-direction, so the projection direction cannot be set to 2. Because the symmetry axes (2 and 3) are not allowed, the only direction of projection is 1; therefore, NPROJ = 1 and IDIR = 1.

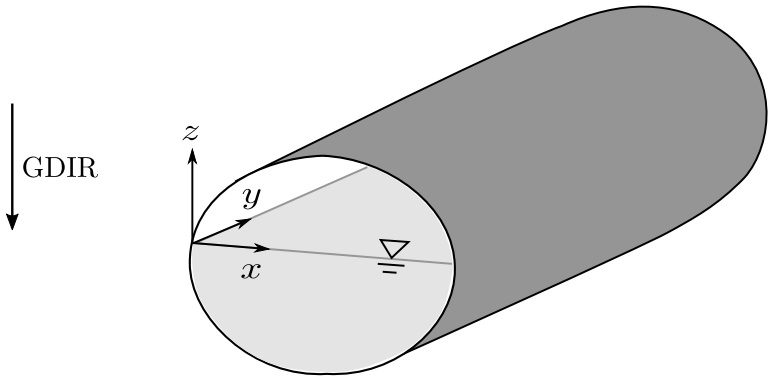


Figure 3-6. Example for water filled tube structure

For further explanations and examples, see Haßler and Schweizerhof [2007], Haßler and Schweizerhof [2008], and Maurer, Gebhardt, and Schweizerhof [2010].

The possible entries for NPROJ, IDIR, and IIDIR are:

NPROJ	IDIR	IIDIR
3		
2	1	2
2	1	3
2	2	3
1	1	
1	2	
1	3	

*AIRBAG_ALE

Purpose: Provide a simplified approach to defining the deployment of the airbag using the ALE capabilities with an option to switch from the initial ALE method to a control volume (CV) method similar to *AIRBAG_HYBRID at a chosen time. An enclosed airbag (and possibly the airbag canister/compartment and/or a simple representation of the inflator) shell structure interacts with the inflator gas(es). This definition provides a single fluid to structure coupling for the airbag-gas interaction during deployment in which the CV input data may be used directly.

Card Summary:

Card 1. This card is required.

SID	SIDTYP					MWD	SPSF

Card 2. This card is required.

ATMOST	ATMOSP		GC	CC	TNKVOL	TNKFINP	

Card 3. This card is required. See *CONSTRAINED_LAGRANGE_IN_SOLID.

NQUAD	CTYPE	PFAC	FRIC	FRCMIN	NORMTYP	ILEAK	PLEAK

Card 4. This card is required.

IVSETID	IVTYPE	IBLOCK	VNTCOF				

Card 5a. This card is for an ALE mesh defined with a part ID. NZ must not be defined (NZ = 0) to activate this card.

IDA	IDG	NZ	MOVERN	ZOOM			

Card 5b. This card is included to automatically define an ALE mesh. It is activated when NZ ≠ 0.

NX	NY	NZ	MOVERN	ZOOM			

Card 5b.1. This card is included when NZ > 0.

X0	Y0	Z0	X1	Y1	Z1		

Card 5b.2. This card is included when NZ > 0.

X2	Y2	Z2	Z3	Y3	Z3		

Card 6. This card is required.

SWTIME		HG	NAIR	NGAS	NORIF	LCVEL	LCT

Card 7. This card is required.

			MWAIR	INITM	AIRA	AIRB	AIRC

Card 8. Include NGAS of this card.

LCMF			MWGAS		GASA	GASB	GASC

Card 9. Include NORIF of this card.

NODEID	VECID	ORIFARE					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SIDTYP					MWD	SPSF
Type	I	I					F	F
Default	none	none					0	0

VARIABLE	DESCRIPTION
SID	Set ID. This set ID contains the Lagrangian elements (segments) which make up the airbag and possibly the airbag canister/compartment and/or a simple representation of the inflator. See Remark 1.
SIDTYP	Set type: EQ.0: Segment set EQ.1: Part set
MWD	Mass weighted damping factor, D, used during the CV phase
SPSF	Stagnation pressure scale factor, $0 \leq \gamma \leq 1$. SPSF is needed during the CV phase.

Ambient Environment Card.

Card 2	1	2	3	4	5	6	7	8
Variable	ATMOST	ATMOSP		GC	CC	TNKVOL	TNKFINP	
Type	F	F		F	F	F	F	
Default	0.	0.		none	1.0	0.0	0.0	

VARIABLE	DESCRIPTION
ATMOST	Atmospheric ambient temperature. See Remark 2.
ATMOSP	Atmospheric ambient pressure. See Remark 2.
GC	Universal molar gas constant.
CC	Conversion constant. EQ.0: Set to 1.0.
TNKVOL	Tank volume from the inflator tank test or inflator canister volume. See Remark 9 and Card 6. **LCVEL = 0 and TNKFINP is defined:** TNKVOL is the defined tank. Inlet gas velocity is estimated by LS-DYNA method (testing). **LCVEL = 0 and TNKFINP is not defined:** TNKVOL is the estimated inflator canister volume inlet gas velocity is estimated automatically by the Lian-Bhalsod-Olovsson method. **LCVEL ≠ 0:** This must be left blank.
TNKFINP	Tank final pressure from the inflator tank test data. Only define this parameter for option 1 of TNKVOL definition above. See Remark 9.

Coupling Card. See keyword *CONSTRAINED_LAGRANGE_IN_SOLID.

Card 3	1	2	3	4	5	6	7	8
Variable	NQUAD	CTYPE	PFAC	FRIC	FRCMIN	NORMTYP	ILEAK	PLEAK
Type	I	I	F	F	F	I	I	F
Default	4	4	0.1	0.0	0.3	0	2	0.1

VARIABLE	DESCRIPTION
NQUAD	Number of (quadrature) coupling points for coupling Lagrangian parts to ALE solid parts. If NQUAD = n, then n × n coupling points will be parametrically distributed over the surface of each Lagrangian segment. See Remark 12.
CTYPE	Coupling type (see Remark 12): EQ.4: Penalty coupling with coupling in the normal direction under compression only (default). EQ.6: Penalty coupling in which coupling is under both tension and compression in the normal direction for the unfolded region and under only compression in the normal direction for folded region.
PFAC	Penalty factor. PFAC is a scale factor for scaling the estimated stiffness of the interacting (coupling) system. It is used to compute the coupling forces to be distributed on the Lagrangian and ALE parts. See Remark 13. GT.0: Fraction of estimated critical stiffness. LT.0: -PFAC is a load curve ID. The curve defines the relative coupling pressure (y-axis) as a function of the tolerable fluid penetration distance (x-axis).
FRIC	Coupling coefficient of friction
FRCMIN	Minimum fluid volume fraction in an ALE element to activate coupling.
NORMTYP	Penalty coupling spring direction: EQ.0: Normal vectors are interpolated from nodal normals.

VARIABLE	DESCRIPTION
	EQ.1: Normal vectors are interpolated from segment normals.
ILEAK	Leakage control flag. Default = 2 (with energy compensation).
PLEAK	Leakage control penalty factor (default = 0.1)

Venting Hole Card.

Card 4	1	2	3	4	5	6	7	8
Variable	IVSETID	IVTYPE	IBLOCK	VNTCOF				
Type	I	I	I	F				
Default	0	0	0	0.0				

VARIABLE	DESCRIPTION
IVSETID	Set ID defining the venting hole surface(s). See Remark 4.
IVTYPE	Set type of IVSETID: EQ.0: Part Set (default). EQ.1: Part ID. EQ.2: Segment Set.
IBLOCK	Flag for considering blockage effects for porosity and vents (see Remark 5): EQ.0: no (blockage is NOT considered, default). EQ.1: yes (blockage is considered).
VNTCOF	Vent Coefficient for scaling the flow. See Remark 6.

ALE Mesh from Part ID Card. Parameters for transformation of the ALE mesh specified with part IDs.

Card 5a	1	2	3	4	5	6	7	8
Variable	IDA	IDG	NZ	MOVERN	ZOOM			
Type	I	I	I	I	I			
Default	none	none	↓	0	0			

VARIABLE	DESCRIPTION
IDA	Part ID of the initial air mesh
IDG	Part ID of the initial gas mesh
NZ	Leave blank to activate.
MOVERN	ALE mesh automatic motion option. EQ.0: ALE mesh is fixed in space. GT.0: Node group ID. See *ALE_REFERENCE_SYSTEM_-NODE ALE mesh can be moved with PRTYP = 5, mesh motion follows a coordinate system defined by 3 reference nodes. See Remark 7.
ZOOM	ALE mesh automatic expansion option: EQ.0: Do not expand ALE mesh EQ.1: Expand/contract ALE mesh by keeping all airbag parts contained within the ALE mesh (equivalent to PRTYP = 9). See Remark 8.

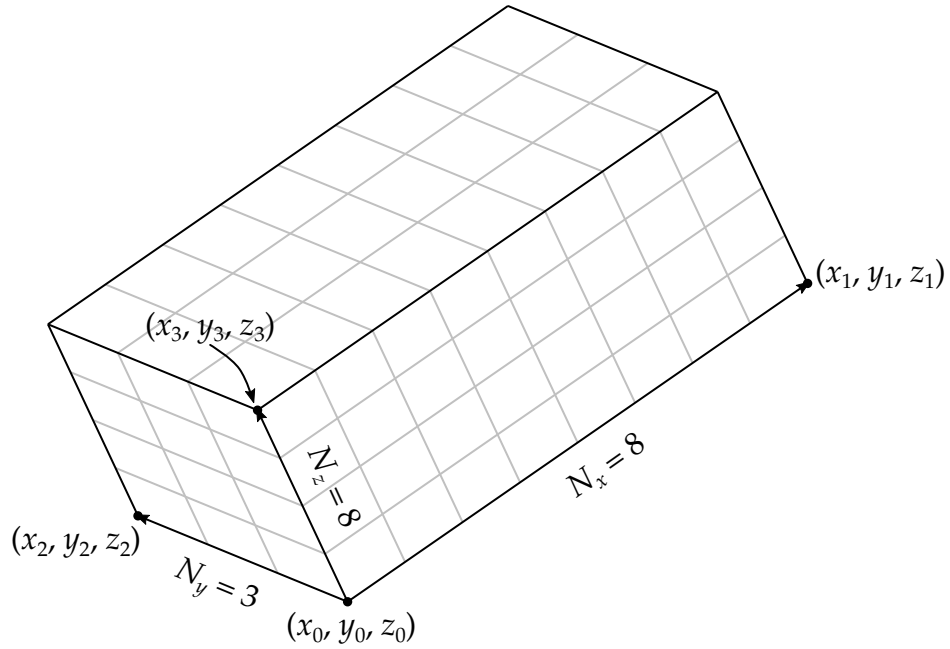


Figure 3-7. Illustration of automatic mesh generation for the ALE mesh in a hexahederal region

Automatic ALE Mesh Card. Parameters for ALE mesh automatic definition and its transformation. See Figure 3-7.

Card 5b	1	2	3	4	5	6	7	8
Variable	NX	NY	NZ	MOVERN	ZOOM			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

VARIABLE	DESCRIPTION
NX	NX is the number of ALE elements to be generated in the x-direction.
NY	NY is the number of ALE elements to be generated in the y-direction.
NZ	NZ is the number of ALE elements to be generated in the z-direction.
MOVERN	ALE mesh automatic motion option.

VARIABLE	DESCRIPTION
	EQ.0: ALE mesh is fixed in space. GT.0: Node group ID. See *ALE_REFERENCE_SYSTEM_-NODE ALE mesh can be moved with PRTYP = 5, mesh motion follows a coordinate system defined by 3 reference nodes. See Remark 7.
ZOOM	ALE mesh automatic expansion option: EQ.0: Do not expand ALE mesh. EQ.1: Expand/contract ALE mesh by keeping all airbag parts contained within the ALE mesh (equivalent to PRTYP = 9). See Remark 8.

Origin for ALE Mesh Card. Include Cards 5b.1 and 5b.2 when NZ > 0.

Card 5b.1	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	X1	Y1	Z1		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

Card 5b.2	1	2	3	4	5	6	7	8
Variable	X2	Y2	Z2	Z3	Y3	Z3		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
X0, Y0, Z0	Coordinates of origin for ALE mesh generation (node0).
X1, Y1, Z1	Coordinates of point 1 for ALE mesh generation (node1). x-extent = node1 − node0

VARIABLE	DESCRIPTION
X2, Y2, Z2	Coordinates of point 2 for ALE mesh generation (node2). y-extent = node2 − node0
X3, Y3, Z3	Coordinates of point 3 for ALE mesh generation (node3). z-extent = node3 − node0

Miscellaneous Parameters Card.

Card 6	1	2	3	4	5	6	7	8
Variable	SWTIME		HG	NAIR	NGAS	NORIF	LCVEL	LCT
Type	F		F	I	I	I	I	I
Default	10^{16}		0.	0	0	0	0	0
Remarks	3						9	10

VARIABLE	DESCRIPTION
SWTIME	Time to switch from ALE method to control volume (CV) method. Once switched, a method similar to that used by the *AIRBAG_-HYBRID card is used.
HG	Hourglass control for ALE fluid mesh(es).
NAIR	Number of air components. For example, this equals 2 when air contains 80% of N_2 and 20% of O_2. If air is defined as a single gas, then NAIR = 1.
NGAS	Number of inflator gas components.
NORIF	Number of point sources or orifices. This determines the number of point source cards to be read.
LCVEL	Load curve ID for inlet velocity (see also TNKVOL & TNKFINP of Card 2 above). This is the same estimated velocity curve used in *SECTION_POINT_SOURCE_MIXTURE card.
LCT	Load curve ID for inlet gas temperature (see *AIRBAG_HYBRID).

Air Component Card. Include NAIR cards, one for each air component.

Card 7	1	2	3	4	5	6	7	8
Variable				MWAIR	INITM	AIRA	AIRB	AIRC
Type				F	F	F	F	F
Default				0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
MWAIR	Molecular weight of air component
INITA	Initial Mass Fraction of air component(s)
AIRA	First coefficient of molar heat capacity at constant pressure (e.g., J/mole/K). See Remark 11.
AIRB	Second coefficient of molar heat capacity at constant pressure (e.g., J/mole/K^2). See Remark 11.
AIRC	Third coefficient of molar heat capacity at constant pressure (e.g., J/mole/K^3). See Remark 11.

Gas Component Card. Include NGAS cards, one for each gas component.

Card 8	1	2	3	4	5	6	7	8
Variable	LCMF			MWGAS		GASA	GASB	GASC
Type	I			F		F	F	F
Default	none			0		0	0.	0.

VARIABLE	DESCRIPTION
LCMF	Load curve ID for mass flow rate (see *AIRBAG_HYBRID, e.g., kg/s). See Remark 11.
MWGAS	Molecular weight of inflator gas components.

VARIABLE	DESCRIPTION
GASA	First coefficient of molar heat capacity at constant pressure (e.g., J/mole/K). See Remark 11.
GASB	Second coefficient of molar heat capacity at constant pressure (e.g., J/mole/K^2). See Remark 11.
GASC	Third coefficient of molar heat capacity at constant pressure (e.g., J/mole/K^3). See Remark 11.

Point Source Cards. Include NORIF cards, one for each point source.

Card 9	1	2	3	4	5	6	7	8
Variable	NODEID	VECID	ORIFARE					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
NODEID	The node ID defining the point source.
VECID	The vector ID defining the direction of flow at the point source.
ORIFARE	The orifice area at the point source.

Remarks:

1. **Lagrangian Elements for Airbag Components.** This set ID typically contains the Lagrangian segments of the 3 parts that are coupled to the inflator gas: airbag, airbag canister (compartment), and inflator. As in all control-volume, orientation of elements representing bag and canister should point outward. During the ALE phase the segment normal will be reversed automatically for fluid-structure coupling. *However, the orientation of inflator element normal vectors should point to its center.* See Figure 3-8.

2. **Atmospheric Density.** Atmospheric density for the ambient gas (air) can be computed from

$$\rho_{\text{amb}} = \frac{P_{\text{amb}}}{RT_{\text{amb}}}$$

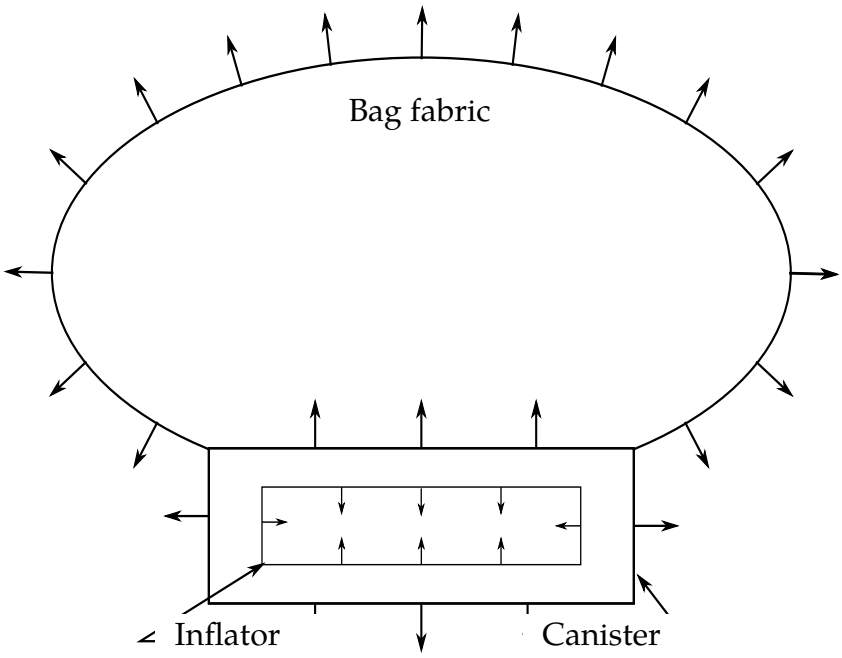


Figure 3-8. Arrows indicate "outward" normal

3. **ALE to Control Volume Switch.** Since *all* ALE related activities will be turned off after the switch from ALE method to control volume method, no other ALE coupling will exist beyond t = SWTIME.

4. **Venting.** Vent definition will be used for ALE venting. Upon switching, area of the segments will be used for venting as a23 in *AIRBAG_HYBRID.

5. **Fabric Porosity.** Fabric porosity for ALE and *AIRBAG_HYBRID can be defined on *MAT_FABRIC. Define FLC and FAC on *MAT_FABRIC. FVOPT 7 and 8 will be used for both ALE and *AIRBAG_HYBRID. IBLOCK = 0 will use FVOPT = 7 and IBLOCK = 1 will use FVOPT=8.

6. **Vent Coefficient.** VCOF will be used to scale the vent area for ALE venting and this coefficient will be used as vent coefficient c23 for *AIRBAG_HYBRID upon switching.

7. **ALE Mesh Motion.** If the airbag moves with the vehicle, set MOVERN = GROUPID; this GROUPID is defined using *ALE_REFERENCE_SYSTEM_NODE. The 3 nodes defined in *ALE_REFERENCE_SYSTEM_NODE will be used to transform the ALE mesh. The point sources will also follow this motion. This simulates PRTYP = 5 in the *ALE_REFERENCE_SYSTEM_GROUP card.

8. **Mesh Expansion.** Automatic expansion/contraction of the ALE mesh to follow the airbag expansion can be turned on by setting zoom = 1. This feature is particularly useful for fully folded airbags requiring very fine ale mesh initially. As

the airbag inflates the ale mesh will be automatically scaled such that the airbag will be contained within the ALE mesh. This simulates PRTYP = 9 in the *ALE_-REFERENCE_SYSTEM_GROUP card.

9. **Inlet Gas Velocity.** There are 3 methods for defining the inlet gas velocity:

   a) Inlet gas velocity is estimated by LS-DYNA method (testing), if

   $$\text{LCVEL} = 0 \Rightarrow \text{TNKVOL} = \text{Tank volume}$$

   and

   $$\text{TNKFINP} = \text{Tank final pressure from tank test data.}$$

   b) Inlet gas velocity is estimated automatically by Lian-Bhalsod-Olovsson method if,

   $$\text{LCVEL} = 0 \Rightarrow \text{TNKVOL} = \text{Tank volume.}$$

   and

   $$\text{TNKFINP} = \text{blank.}$$

   c) Inlet gas velocity is defined by user via a load curve LCVEL if,

   $$\begin{aligned} \text{LCVEL} &= \text{n}, \\ \text{TNKVOL} &= 0, \end{aligned}$$

   and

   $$\text{TNKFINP} = 0$$

10. **Sampling Points.** LCT and LCIDM should have the same number of sampling points.

11. **Heat Capacity at Constant Pressure.** The per-mass-unit, temperature-dependent, constant-pressure heat capacity is

$$C_p(T) = \frac{\left[A + BT + CT^2\right]}{MW}.$$

where,

$$A = \tilde{C}_{p_0}$$

these quantities often have units of,

$C_p(T)$	A	B	C
$\frac{\text{J}}{\text{kg} \times \text{K}}$	$\frac{\text{J}}{\text{mole} \times \text{K}}$	$\frac{\text{J}}{\text{mole} \times \text{K}^2}$	$\frac{\text{J}}{\text{mole} \times \text{K}^3}$

12. **Coupling.** Sometimes CTYPE = 6 may be used for a complex folded airbag. NQUAD = 2 may be used as a starting value and increased as necessary depending on the relative mesh resolutions of the Lagrangian and ALE meshes.

13. **PFAC.** Use a load curve for PFAC whenever possible. It tends to be more robust.

Related Cards:

AIR ⟶ *PART (AMMG2)
*SECTION_SOLID
*MAT_GAS_MIXTURE

GAS ⟶ *PART (AMMG1)
*SECTION_POINT_SOURCE_MIXTURE
*MAT_GAS_MIXTURE

Couplings ⟶ *CONSTRAINED_LAGRANGE_IN_SOLID

ALE Mesh Motion ⟶ *ALE_REFERENCE_SYSTEM_GROUP

Control Volume ⟶ *AIRBAG_HYBRID

Vent ⟶ *AIRBAG_ALE/Card4

Example 1:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*AIRBAG_ALE
$#1    SID   SIDTYPE      NONE      NONE      NONE      NONE       MWD      SPSF
       123         1         0         0         0         0       0.0       0.0
$#2 ATMOST    ATMOSP      NONE        GC        CC    TNKVOL     TNKFP
    298.15 1.0132E-4         0     8.314       0.0       0.0       0.0
$#3  NQUAD     CTYPE      PFAC      FRIC    FRCMIN  NRMTYPE     ILEAK     PLEAK
         4         4     -1000       0.0       0.3         0         2       0.1
$#4 VSETID  IVSETTYP    IBLOCK  VENTCOEF
         1         2         0      1.00
$#5NXIDAIR   NYIDGAS        NZ    MOVERN      ZOOM
     50000     50003         0         0         0
$#6 SWTIME      NONE        HG      NAIR      NGAS     NORIF     LCVEL       LCT
   1000.00     0.000     1.e-4         1         1         8      2002      2001
$#7 AIR          NONE      NONE     MWAIR     INITM      AIRA      AIRB      AIRC
         0         0         0   0.02897      1.00    29.100   0.00000   0.00000
$#8 GASLCM      NONE      NONE     MWGAS      NONE      GASA      GASB      GASC
      2003         0         0    0.0235         0    28.000   0.00000   0.00000
$#9 NODEID    VECTID  ORIFAREA
    100019         1 13.500000
    100020         2 13.500000
    100021         3 13.500000
    100022         4 13.500000
    100023         5 13.500000
    100024         6 13.500000
    100017         7 13.500000
    100018         8 13.500000
$ PFAC CURVE = penalty factor curve.
*DEFINE_CURVE
$     lcid      sidr       sfa       sfo      offa      offo    dattyp
```

```
      1000         0       0.0       2.0       0.0       0.0
$                a1                 o1
                0.0          0.00000000
           1.0000000        4.013000e-04
*SET_SEGMENT_TITLE
vent segments (defined in IVSETID)
         1       0.0       0.0       0.0       0.0
      1735      1736       661      1697       0.0       0.0       0.0       0.0
      1735      2337      1993      1736       0.0       0.0       0.0       0.0
      1735      1969      1988      2337       0.0       0.0       0.0       0.0
      1735      1697       656      1969       0.0       0.0       0.0       0.0
*DEFINE_VECTOR
$#     vid        xt        yt        zt        xh        yh        zh
         1       0.0       0.0-16.250000 21.213200 21.213200-16.250000
         2       0.0       0.0-16.250000 30.000000-1.000e-06-16.250000
         3       0.0       0.0-16.250000 21.213200-21.213200-16.250000
         4       0.0       0.0-16.250000-1.000e-06-30.000000-16.250000
         5       0.0       0.0-16.250000-21.213200-21.213200-16.250000
         6       0.0       0.0-16.250000-30.0000001.0000e-06-16.250000
         7       0.0       0.0-16.250000-21.213200 21.213200-16.250000
         8       0.0       0.0-16.2500001.0000e-06 30.000000-16.250000
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

In this example, pre-existing background air mesh with part ID 50000 and gas mesh with part ID 50003 are used. Thus NZ = 0. There is no mesh motion nor expansion allowed. An inlet gas velocity curve is provided.

Example 2:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ SIDTYP: 0=SGSID; 1=PSID
*AIRBAG_ALE
$#1    SID   SIDTYPE      NONE      NONE      NONE      NONE       MWD      SPSF
         1         1         0        0.        0.        0.        0.        0.
$#2 ATMOST    ATMOSP      NONE        GC        CC    TNKVOL     TNKFP
       298.   101325.       0.0     8.314        1.    6.0E-5         0
$#3  NQUAD     CTYPE      PFAC      FRIC    FRCMIN  NRMTYPE     ILEAK     PLEAK
         2         6      -321       0.0       0.3         1         2       0.1
$#4 VSETID  IVSETTYP    IBLOCK  VENTCOEF
         0         0         0         0
$#5NXIDAIR   NYIDGAS        NZ    MOVERN      ZOOM
        11        11         9
$5b     x0        y0        z0        x1        y1        z1  NOT-USED  NOT-USED
      -0.3      -0.3    -0.135       0.3      -0.3    -0.135
$5c     x2        y2        z2        x3        y3        z3  NOT-USED  NOT-USED
      -0.3       0.3    -0.135      -0.3      -0.3      0.39
$#6 SWTIME      NONE        HG      NAIR      NGAS     NORIF     LCVEL       LCT
   0.04000     0.005     1.e-4         2         1         1         0         2
$#7 AIR         NONE      NONE     MWAIR     INITM      AIRA      AIRB      AIRC
                                   0.028      0.80    27.296   0.00523
                                   0.032      0.20    25.723   0.01298
$#8 GASLCM      NONE      NONE     MWGAS      NONE      GASA      GASB      GASC
         1                        0.0249                29.680   0.00880
$#9 NODEID    VECTID  ORIFAREA
      9272         1   1.00e-4
$ Lagrangian shell structure to be coupled to the inflator gas
*SET_PART_LIST
         1       0.0       0.0       0.0       0.0
         1         2         3
*DEFINE_VECTOR
$0.100000E+01,  10.000000000
$      vid        xt        yt        zt        xh        yh        zh
```

```
         1       0.0       0.0       0.0       0.0       0.0  0.100000
$ bag penetration ~ 1 mm  <====>  P_coup ~ 500000 pascal ==> ~ 5 atm
*DEFINE_CURVE
$     lcid      sidr       sfa       sfo      offa      offo    dattyp
       321         0       0.0       0.0       0.0       0.0
$                a1                  o1
                0.0                 0.0
          0.00100000       5.0000000e+05
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

In this example, LS-DYNA automatically creates the background ALE mesh with:

NX = 11 ⇒ 11 elements in the *x*-direction
NY = 11 ⇒ 11 Elements in the *y*-direction
NZ = 9 ⇒ 9 Elements in the *z*-direction

*AIRBAG_CPG_{OPTION}

Available options include:

<BLANK>

ID

TITLE

Use either ID or TITLE to specify an airbag ID and a heading for the airbag.

Purpose: Define an airbag using the Continuum-based Particle Gas (CPG) method. CPG is a particle-based compressible gas solver.

> **NOTE:** This model requires orienting surface normal vectors *inward*, unlike the traditional Control Volume or CV method (also known as Uniform Pressure or UP method) for which they must point *outward*.

Card Summary:

Card ID. Include this card if using the ID or TITLE keyword options.

BAGID	HEADING	

Card 1. This card is required.

SID1	STYPE1	SID2	STYPE2	BLOCK	NPDATA		SPSID

Card 2. This card is required.

HLEN	UNIT	ADAID	TATM	PATM	NVENT	BIRTH	TSW

Card 3. This card is required.

	NGAS	NORIF	NID1	NID2	NID3		

Card 4. Additional cards for NPDATA > 0 (see Card 1). Define NPDATA cards, one for each part or part set, providing additional data.

SIDH	STYPEH	HCONV	PFRIC				

Card 5. Include NVENT instantiations of this card (see Card 2).

SID3	STYPE3	HLENV	LCTC	LCPC	DVID	PPOP	

Card 6. This card is required.

PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR		

Card 7. Include NGAS of this card. See Card 3.

LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*	INFG*i*	

Card 8. Include NORIF of this card. See Card 3.

SSID*i*	HLENO	ROLOC		INFO*i*			

Data Card Definitions:

Title Card. Additional card for the ID or TITLE keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	BAGID	HEADING						
Type	I	A60						

VARIABLE	DESCRIPTION
BAGID	Airbag ID. This must be a unique number.
HEADING	Airbag descriptor. We suggest using unique descriptions.

Card 1	1	2	3	4	5	6	7	8
Variable	SID1	STYPE1	SID2	STYPE2		NPDATA		SPSID
Type	I	I	I	I		I		I
Default	none	0	0	0		0		0

VARIABLE	DESCRIPTION
SID1	Part or part set ID defining the complete airbag. See Remarks 1 and 2.

VARIABLE	DESCRIPTION
STYPE1	Set type: EQ.0: Part EQ.1: Part set
SID2	Part or part set ID defining the embedded parts of the airbag. See Remark 2.
STYPE2	Set type: EQ.0: Part EQ.1: Part set
BLOCK	Blocking options (see Remark 9): EQ.0: Off. Blocking is not considered. EQ.1: Elements with nonzero contact forces are fully blocked, meaning porosity and venting leakage are set to zero. EQ.4: Venting leakage is not affected by contact forces, meaning blocking is not considered at vents. Porosity leakage is set to zero for elements with nonzero contact forces. EQ.11: Venting leakage is set to zero for vent elements with nonzero contact forces. Porosity leakage is scaled by a factor that depends on the ratio of contact pressure to gas pressure. EQ.14: Venting leakage is not affected by contact forces, meaning blocking is not considered at vents. Porosity leakage is scaled by a factor that depends on the ratio of contact pressure to gas pressure.
NPDATA	Number of parts or part sets with additional data
SPSID	Optional sampling part set ID for geodesic redistribution of CPG particles on surface shell elements (see Remark 10).

Card 2	1	2	3	4	5	6	7	8
Variable	HLEN	UNIT	ADAID	TATM	PATM	NVENT	BIRTH	TSW
Type	F	I	I	F	F	I	F	F
Default	0.	0	0	293 K	1 atm	0	0.	0.

VARIABLE	DESCRIPTION
HLEN	Average interparticle distance. See Remark 3.
UNIT	Unit system: EQ.0: kg-mm-ms-K EQ.1: SI EQ.2: tonne-mm-s-K
ADAID	Optional ID referencing *DEFINE_CPG_ADAPTIVE
TATM	Atmospheric temperature
PATM	Atmospheric pressure
NVENT	Number of vent hole parts or part sets
BIRTH	Birth time for the CPG solver. Note that there should be no deformation occurring in any part of SID1 prior to birth time.
TSW	If greater than 0.0, time at which the algorithm switches to the control volume method with uniform pressure.

Card 3	1	2	3	4	5	6	7	8
Variable		NGAS	NORIF	NID1	NID2	NID3		
Type		I	I	I	I	I		
Default		none	none	0	0	0		.

VARIABLE	DESCRIPTION
NGAS	Number of gas components
NORIF	Number of orifices
NID1 - NID3	Three nodes that define a moving coordinate system. By default, the coordinate system is fixed. See Remark 8.

NPDATA Card. Additional cards for NPDATA > 0. Define NPDATA cards, one for each part or part set with additional data.

Card 4	1	2	3	4	5	6	7	8
Variable	SIDH	STYPEH	HCONV	PFRIC				
Type	I	I	F	F				
Default	none	none	none	0.0				

VARIABLE	DESCRIPTION
SIDH	Part or part set ID defining additional part data
STYPEH	Set type: EQ.0: Part EQ.1: Part set
HCONV	Convective heat transfer coefficient used to calculate heat loss from the airbag's external surface to the ambient. See Remark 4.
PFRIC	Friction factor F_r. See Remark 5.

Vent Hole Card. Additional cards for NVENT > 0. Define NVENT cards, one for vent hole.

Card 5	1	2	3	4	5	6	7	8
Variable	SID3	STYPE3	HLENV	LCT	LCP	DVID	PPOP	
Type	I	I	F	I	I	I	F	
Default	0	none	optional	0	0	0	0.0	

VARIABLE	DESCRIPTION
SID3	Part or part set ID defining vent holes
STYPE3	Set type: EQ.0: Part EQ.1: Part set
HLENV	Optional local average interparticle distance centered on the vent. See Remark 7.
LCT	Optional load curve to impose temperature as a function of time in cases where backflow is detected. Otherwise, TATM is used. See Remark 6.
LCP	Optional load curve to impose pressure as a function of time. Otherwise, PATM is used. See Remark 6.
DVID	Optional vent ID. See *DEFINE_CPG_VENT. When active, LCT, LCP and PPOP on this line are ignored and replaced with *DEFINE_CPG_VENT input.
PPOP	Pressure difference between the interior and ambient pressure (PATM) for opening the vent holes. Once the vents are open, they stay open. EQ.0.0: The vent holes are open at the beginning of the simulation.

Air Card. Mandatory card.

Card 6	1	2	3	4	5	6	7	8
Variable	PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR		
Type	F	F	F	F	F	F		
Default	PATM	TATM	none	none	0.0	0.0		

VARIABLE	DESCRIPTION
PAIR	Initial pressure inside bag
TAIR	Initial temperature inside bag
XMAIR	Molar mass of gas initially inside bag: LT.0.0: -XMAIR references the ID of a *DEFINE_CPG_GAS_PROPERTIES keyword that defines the gas thermodynamic properties. Note that AAIR, BAIR, and CAIR are ignored.
AAIR – CAIR	Constant, linear, and quadratic heat capacity parameters

Gas Cards. NGAS additional cards, one for each gas (card format for i^{th} gas).

Card 7	1	2	3	4	5	6	7	8
Variable	LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*	INFG*i*	
Type	I	I	F	F	F	F	I	
Default	none	none	none	none	0.0	0.0	1	

VARIABLE	DESCRIPTION
LCM*i*	Mass flow rate curve for gas component *i*,
LCT*i*	Temperature load curve for gas component *i*
XM*i*	Molar mass of gas component *i*. LT.0.0: The absolute value of XM*i* references the ID of a *DE-

VARIABLE	DESCRIPTION
	FINE_CPG_GAS_PROPERTIES keyword that defines the gas thermodynamic properties. Note that A*i*, B*i*, and C*i* are ignored.
A*i* - C*i*	Constant, linear, and quadratic heat capacity parameters for gas component *i*
INFG*i*	Inflator ID for this gas component (defaults to 1)

Orifice Cards. NORIF additional cards, one for each orifice (card format for i^{th} orifice).

Card 8	1	2	3	4	5	6	7	8
Variable	SSID*i*	HLENO	ROLOC		INFO*i*			
Type	I	F	F		I			
Default	none	optional	none		1			

VARIABLE	DESCRIPTION
SSID*i*	Shell ID defining the location of nozzle *i*. If a negative value is entered, the absolute value refers to a shell set ID. See Remark 7.
HLENO	Optional local average interparticle distance centered on the orifice
ROLOC	Optional radius definition that overwrites the automatic refinement size centered around the orifice when HLENO is defined. See Remark 7.
INFO*i*	Inflator ID for this orifice (default = 1)

Remarks:

1. **Closed geometry.** The parts that define the external volume of the airbag must consist of shells that form a closed geometry. If an inconsistent list of parts is provided, the airbag volume will fail to properly generate. See Figure 3-9 for an example of a closed geometry contour.

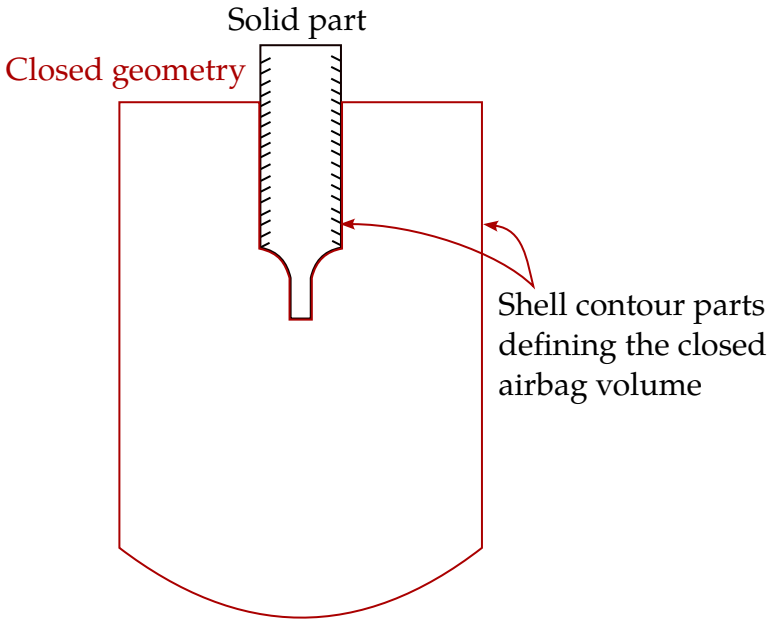


Figure 3-9. Example of a closed contour. Here, the inflator has originally been modelled as a solid. Thus, a layer of shell elements must be defined to correctly define a closed airbag.

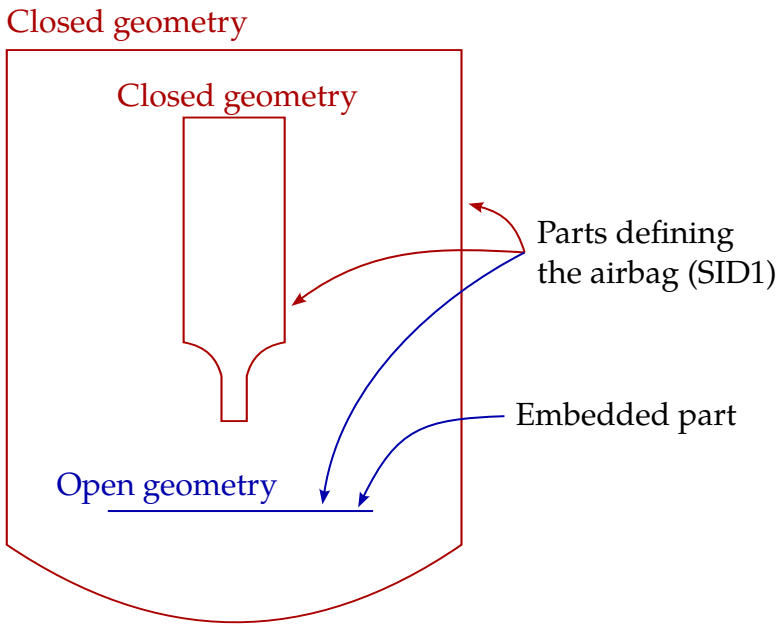


Figure 3-10. SID1 contains all parts that participate in the airbag analysis. SID2 is restricted to the embedded parts. Embedded parts are open geometries within the airbag volume. The parts listed in SID1 but not listed in SID2 must collectively represent a closed geometry.

2. **Embedded parts.** The embedded parts in the airbag consist of all the open geometries within the airbag volume. Figure 3-10 illustrates a confusing scenario when determining the embedded parts. In this example, the inflator-shaped

part within the airbag volume participates in the definition of airbag volume but is not an open geometry. It must therefore be included in SID1 but not SID2. Conversely, the thin plate participates in the airbag analysis and represents an open geometry. Thus, it is included in both SID1 and SID2.

3. **HLEN.** We recommend keeping this value at around the airbag's surface mesh size or slightly larger.

4. **Thermal coupling.** Coupling between CPG and the implicit thermal solver is available for shell surfaces. Heat is exchanged between CPG particles and the corresponding shell structures according to:

$$\dot{Q} = hA(T_{\text{struct}} - T_{\text{CPG}}) \ ,$$

where h is the heat transfer coefficient input with HCONV and A the shell area. This heat flux is then incorporated in the energy equation of the CPG particles and imposed as a Neuman boundary condition on the shell elements. Proper time interpolation is performed to ensure consistency between the two solvers (the implicit thermal solver time step is generally larger).

5. **Friction.** The friction factor F_r acts as a scaling value on the friction force that is added to the fluid particles close to the wall. The friction force is based on the fluid viscosity (see *DEFINE_CPG_GAS_PROPERTIES) and velocity gradient. For typical airbag applications, effects are expected to be small unless very large values are chosen for F_r.

6. **Backflow.** The velocity direction at a given time automatically determines the state of the vent, meaning whether it is in backflow mode or in outflow mode. If in subsonic outflow mode, static pressure is imposed. If in subsonic backflow mode, total pressure and total temperature are imposed. In the case of supersonic outflow, user-defined pressure values are ignored and replaced by the upstream pressure. In the case of supersonic backflow, static pressure and total temperature are imposed. When in backflow mode, the gas fraction of air is set to 1.0 (0.0 for all other gases).

7. **Selecting orifice and vent areas.** Airbag vent and orifice areas are typically very small. Conversely, traditional CFD recommendations require at least ten elements or points to define any inlet or outlet area properly. Following those guidelines may result in very large models. In practice, a compromise can be found by increasing the numerical surface area of the orifice and keeping enough particles to represent the inlet properly. HLENO and HLENV facilitate locally refining the point cloud around the orifices and vents. When used, the solver progressively transitions the particle distance from HLENO to HLEN for orifices (or HLENV to HLEN for vents) based on an automatic criterion. In the cases of orifices, ROLOC can overwrite the default automatically estimated radius of the transition zone. To manually define a local particle distance, use *MESH_SIZE_SHAPE.

8. **Moving coordinate system.** Defining a moving coordinate system can be used when the entire airbag is attached to a moving part (such as a moving vehicle). The gas velocity is initialized according to the initial velocities of the coordinate system, and the point cloud follows the motion of the coordinate system (translation and rotation).

9. **Blockage treatment.** Contact forces are used to determine whether an element that belongs to a vent or a porous fabric part should be identified as blocked. This blockage prevents or restricts gas flow from exiting at this location. For example, if BLOCK = 11, vent elements in contact are fully blocked, and porosity leakages depend on the ratio of contact pressure to gas pressure:

$$\mathbf{V}_{\text{blocked}} = \min\left(1.,\max\left(0.,1.-\frac{P_{\text{contact}}}{P_{\text{gas}}}\right)\right)\mathbf{V}_{\text{unblocked}}$$

10. **Particle resampling on surfaces.** By default, surface elements are sampled independently on an element-by-element basis, with each shell element receiving at least one CPG particle and potentially more depending on its surface area and the selected HLEN value. When HLEN is significantly larger than the underlying mesh size, this strategy can create a substantial discrepancy between the particle spacing on the airbag surface and that within the airbag volume, potentially impacting computational performance. For such cases, a part set can be specified through SPSID. Setting to SPSID to a nonzero value activates an additional filtering procedure that removes surface particles located too close to one another based on their geodesic distance along the surface. As a consequence, the number of surface particles associated with the specified part set is fixed for the duration of the simulation. Therefore, this option should be used with caution and is generally not recommended for components expected to undergo large deformations, such as null shells, vents, or similar highly deformable parts.

*AIRBAG_INTERACTION

Purpose: To define two connected airbags which vent into each other.

Card 1	1	2	3	4	5	6	7	8
Variable	AB1	AB2	AREA	SF	PID	LCID	IFLOW	EXCP
Type	I	I	F	F	I	I	I	I
Default	none	none	none	none	0	0	0	0

VARIABLE	DESCRIPTION
AB1	First airbag ID, as defined on *AIRBAG card.
AB2	Second airbag ID, as defined on *AIRBAG card.
AREA	Orifice area between connected bags. LT.0.0: |AREA| is the load curve ID defining the orifice area as a function of absolute pressure. EQ.0.0: AREA is taken as the surface area of the part ID defined below.
SF	Shape factor. LT.0.0: |SF| is the load curve ID defining vent orifice coefficient as a function of relative time.
PID	Optional part ID of the partition between the interacting control volumes. AREA is based on this part ID. If PID is negative, the blockage of the orifice part due to contact is considered,
LCID	Load curve ID defining mass flow rate as a function of pressure difference, see *DEFINE_CURVE. If LCID is defined, AREA, SF and PID are ignored.
IFLOW	Flow direction LT.0: One way flow from AB1 to AB2 only. EQ.0: Two way flow between AB1 and AB2. GT.0: One way flow from AB2 to AB1 only.

VARIABLE	DESCRIPTION
EXCP	Excluding partition part, PID, from bag pressure application. This option applies to *AIRBAG_WANG_NEFSKE and *AIRBAG_HYBRID only. EQ.0: bag pressure will be applied to PID. EQ.1: PID is excluded from bag pressure application.

Remarks:

Mass flow rate and temperature load curves for the secondary chambers must be defined as null curves, for example, in the DEFINE_CURVE definitions give two points (0.0,0.0) and (10000.,0.0).

All input options are valid for the following airbag types:

*AIRBAG_SIMPLE_AIRBAG_MODEL

*AIRBAG_WANG_NEFSKE

*AIRBAG_WANG_NEFSKE_JETTING

*AIRBAG_WANG_NEFSKE_MULTIPLE_JETTING

*AIRBAG_HYBRID

*AIRBAG_HYBRID_JETTING

The LCID defining mass flow rate as a function of pressure difference may additionally be used with:

*AIRBAG_LOAD_CURVE

*AIRBAG_LINEAR_FLUID

If the AREA, SF, and PID are used to define the interaction, then the airbags must contain the same gas, that is, C_p, C_V and g must be the same. The flow between bags is governed by formulae which are similar to those of Wang-Nefske.

***AIRBAG_PARTICLE**_{OPTION1}_..._{OPTION6}

Available options include:

OPTION1 applies to the MPP implementation only.

MPP

OPTION2 also applies to the MPP implementation only. When the DECOMPOSITON option is present, *CONTROL_MPP_DECOMPOSITION_BAGREF and *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS are automatically inserted into the input if they are not already present.

DECOMPOSITION

With *OPTION3* you can specify an airbag ID and a heading for the airbag.

ID

TITLE

OPTION4:

MOLEFRACTION (see Remark 10)

INFLATION (see Remark 17)

JET (see Remark 18)

Include *OPTION5* to specify the airbag volume with a segment set:

SEGMENT

OPTION6 allows you to give the birth and death times for the current CPM airbag.

TIME

Purpose: To define an airbag using the particle method. This method is also sometimes referred to as CPM (Corpuscular Particle Method)

> **NOTE:** This model requires that surface normal vectors be oriented *inward*, unlike the traditional Control Volume or CV method (also known as Uniform Pressure or UP method) for which they must point *outward*. To check bag and chamber integrity, see the CPMERR option on the *CONTROL_CPM card.

Card Summary:

Card MPP. Include this card if using the MPP keyword option.

SX	SY	SZ					

Card ID. Include this card if the using the ID or TITLE keyword options.

BAGID	HEADING	

Card T. Include this card if using the TIME keyword option.

BIRTH	DEATH						

Card 1. This card is required.

SID1	STYPE1	SID2	STYPE2	BLOCK	NPDATA	FRIC	IRPD

Card 2. Include this card if using the SEGMENT keyword option.

SEGSID							

Card 3. This card is required.

NP	UNIT	VISFLG	TATM	PATM	NVENT	TEND	TSW

Card 4. Include this card if using the JET keyword option.

JNODE							

Card 5. When the card after Card 3 or 4 depending on if the JET keyword option is used begins with a "+" character, the input reader processes it as this card; otherwise this card is skipped.

TSTOP	TSMTH	OCCUP	REBL	SIDSV	PSID1	TSPLIT	SFFDC

Card 5.1. When Card 5 is included and the card after Card 5 begins with a "+" character, the input reader processes it as this card; otherwise, this card is skipped.

SFIAIR4	IDFRIC	IAIRSTAT	DPWR	VNDAMP	JETLEN	JETEND	SHLOFF

Card 6. Include this card if UNIT = 3. See Card 3.

	MASS		TIME		LENGTH		

Card 7. This card is required.

IAIR	NGAS	NORIF	NID1	NID2	NID3	CHM	CD_EXT

Card 8. Include this card when STYPE2 = 2 (see Card 1). Define SID2 cards, one for each internal part or part set.

SIDUP	STYUP	PFRAC	LINKING				

Card 9. Additional cards for NPDATA > 0 (see Card 1). Define NPDATA cards, one for each heat convection part or part set.

SIDH	STYPEH	HCONV	PFRIC	SDFBLK	KP	INIP	CP

Card 10. Define NVENT of this card (see Card 3).

SID3	STYPE3	C23	LCTC23	LCPC23	ENH_V	PPOP	

Card 11. Include this card when IAIR ≠ 0. See Card 7.

PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR	NPAIR	NPRLX

Card 12. Include this card if using the MOLEFRACTION keyword option.

LCMASS							

Card 13. Include NGAS of this card. See Card 7.

LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*	INFG*i*	

Card 14. Include NORIF of this card. See Card 7. Card 14.1 may be required to directly follow an instantion of Card 14 if VD*i* = -3 or -4

NID*i*	AN*i*	VD*i*	CA*i*	INFO*i*	IMOM	IANG	CHM_ID

Card 14.1. Include this card in a set with Card 14 if VD*i* = -3 or -4.

XOFF							

Data Card Definitions:

MPP Card. Additional card for MPP keyword option.

Card MPP	1	2	3	4	5	6	7	8
Variable	SX	SY	SZ					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
SX, SY, SZ	Scale factor for each direction used during the MPP decomposition. For instance, increasing SX from 1 to 10 increases the probability that the model is divided along the *x*-direction.

Title Card. Additional card for ID or TITLE keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	BAGID	HEADING						
Type	I	A60						

VARIABLE	DESCRIPTION
BAGID	Airbag ID. This must be a unique number. The BAGID is referenced in, for example, *INITIAL_AIRBAG_PARTICLE_POSITION.
HEADING	Airbag descriptor. It is suggested that unique descriptions be used.

TIME Card. Additional card for TIME keyword option.

Card T	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH						
Type	F	F						
Default	none	10^{20}						

VARIABLE	DESCRIPTION
BIRTH, DEATH	Time to activate/deactivate CPM algorithm. CPM particles will be generated after the birth time.

Card 1	1	2	3	4	5	6	7	8
Variable	SID1	STYPE1	SID2	STYPE2	BLOCK	NPDATA	FRIC	IRPD
Type	I	I	I	I	I	I	F	I
Default	none	0	0	0	0	0.0	0.0	0

VARIABLE	DESCRIPTION
SID1	Part or part set ID defining the complete airbag.
STYPE1	Set type: EQ.0: Part EQ.1: Part set
SID2	Part or part set ID defining the internal parts of the airbag
STYPE2	Set type: EQ.0: Part EQ.1: Part set EQ.2: Number of parts to read (not recommended for general use)

VARIABLE	DESCRIPTION
BLOCK	Blocking. Block must be set to a two-digit number: $\text{BLOCK} = \text{M} \times 10 + \text{N},$ The 10's digit controls the treatment of particles that escape due to deleted elements (particles are always tracked and marked). M.EQ.0: Active particle method which causes particles to be put back into the bag. M.EQ.1: Particles are leaked through unmeshed vents. See Remark 3. M.EQ.2: Blockage logic is turned on when the contact pressure is greater than the bag/chamber pressure for the external vents. The 1's digit controls the treatment of leakage. N.EQ.0: Always consider porosity leakage without considering blockage due to contact. N.EQ.1: Check if airbag node is in contact or not. If yes, 1/4 (quad) or 1/3 (tri) of the segment surface will not have porosity leakage due to contact. N.EQ.2: Same as 1 but no blockage for external vents N.EQ.3: Same as 1 but no blockage for internal vents N.EQ.4: Same as 1 but no blockage for all vents.
NPDATA	Number of parts or part sets data
FRIC	Friction factor F_r if -1.0 < FRIC ≤ 1.0. Otherwise, LE.-1.0: |FRIC| is the curve ID which defines F_r as a function of the part pressure. GT.1.0: FRIC is the *DEFINE_FUNCTION ID that defines F_r. See Remark 2.
IRPD	Dynamic scaling of particle radius flag: EQ.0: Off EQ.1: On

SEGMENT Card. This card is included if the SEGMENT keyword option is used.

Card 2	1	2	3	4	5	6	7	8
Variable	SEGSID							
Type	I							

VARIABLE	DESCRIPTION
SEGSID	ID for a segment set. The segments define the volume and should belong to the parts from SID1.

Card 3	1	2	3	4	5	6	7	8
Variable	NP	UNIT	VISFLG	TATM	PATM	NVENT	TEND	TSW
Type	I	I	I	F	F	I	F	F
Default	200,000	0	0	293 K	1 atm	0	10^{10}	10^{10}

VARIABLE	DESCRIPTION
NP	Number of particles
UNIT	Unit system: EQ.0: kg-mm-ms-K EQ.1: SI EQ.2: tonne-mm-s-K EQ.3: User defined units (see Remark 11)
VISFLG	Visible particles. This field affects only the CPM database. See Remark 5. EQ.0: Default to 1 EQ.1: Output particle's coordinates, velocities, mass, radius, spin energy, translational energy EQ.2: Output reduced data set with coordinates only EQ.3: Suppress CPM database

VARIABLE	DESCRIPTION
TATM	Atmospheric temperature
PATM	Atmospheric pressure
NVENT	Number of vent hole parts or part sets
TEND	Time when all (NP) particles have entered bag. See Remark 14.
TSW	Time at which algorithm switches to control volumes.

Jet Card. This card is included if the JET keyword option is used.

Card 4	1	2	3	4	5	6	7	8
Variable	JNODE							
Type	I							

VARIABLE	DESCRIPTION
JNODE	Node ID on which to apply the reaction force from the thrust (see Remark 18)

Optional Control Cards. When the card after Card 3 or 4 depending on if the JET keyword option is used begins with a "+" character, the input reader processes it as this card; otherwise this card is skipped.

Card 5	1	2	3	4	5	6	7	8
Variable	TSTOP	TSMTH	OCCUP	REBL	SIDSV	PSID1	TSPLIT	SFFDC
Type	F	F	F	I	I	I	F	F
Default	10^{10}	1.0 ms	0.1	0	none	none	none	1.0

VARIABLE	DESCRIPTION
TSTOP	Time at which front tracking switches from IAIR = 4 to IAIR = 2. See Card 7.

VARIABLE	DESCRIPTION
TSMTH	To avoid sudden jumps in the pressure signal during switching, the front tracking is tapered during a transition period. If set to zero, the default time of 1.0 millisecond will be applied.
OCCUP	Particles occupy OCCUP percent of the airbag's volume (default = 10%). This field can be used to balance computational cost and signal quality. OCCUP ranges from 0.001 to 0.1.
REBL	If the option is ON, all energy stored from damping will be evenly distributed as vibrational energy to all particles. This improves the pressure calculation in certain applications. EQ.0: Off (Default) EQ.1: On
SIDSV	Part set ID for internal shell part. The volume occupied by this part is excluded from the bag volume. These internal parts must be consistently oriented for the excluded volume to be correctly calculated.
PSID1	Part set ID for external parts that have normals pointed outward. This option is usually used with an airbag integrity check when two CPM bags are connected with bag interaction. Therefore, one of the bags can have the correct shell orientation, but the shared parts for the second bag will have the wrong orientation. This option will automatically flip the parts defined in this set in the second bag during integrity checking.
TSPLIT	Start time to activate particle splitting algorithm. See Remark 15.
SFFDC	Scale factor for the force decay constant. SFFDC has a range of [0.01,100.0]. The default value is 1.0. The value given here will replaced the values from *CONTROL_CPM.

Optional Control Cards. When Card 5 is included and the card after Card 5 begins with a "+" character, the input reader processes it as this card; otherwise this card is skipped.

Card 5.1	1	2	3	4	5	6	7	8
Variable	SFIAIR4	IDFRIC	IAIRSTAT	DPWR	VNDAMP	JETLEN	JETEND	SHLOFF
Type	F	I	I	F	F	F	F	I
Default	1.0	0	0	1.0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
SFIAIR4	Scale factor for the ratio of initial air particles to inflator gas particles for IAIR = 4. Smaller values weaken the effect of gas front tracking.
IDFRIC	Direction of particle-to-fabric impact force: EQ.0: No change (default) EQ.1: The force is applied in the segment's normal direction.
IAIRSTAT	By default, initial air particles cannot pass through pores and vents when IAIR = 4. This flag can change the treatment: EQ.0: No change (default) EQ.1: Allow passing through pores and vents
DPWR	Exponent for decaying the partial pressure of ambient air inside the airbag for IAIR = 4. See Remark 19.
VNDAMP	Damping coefficient against the motion in the segment's normal direction. 0. ≤ VNDAMP ≤ 1.0.
JETLEN	Estimated particle jet distance. Based on this distance, the particle-to-particle contact evaluation frequency is reduced to preserve the jetting effect from the inflator orifice.
JETEND	End time for the JETLEN option
SHLOFF	Flag for specifying where the exterior surface of the airbag is when the exterior is made of shell elements. This exterior surface is used to determine the volume of the airbag. EQ.0: The exterior surface is taken as the mid-plane of the shell

VARIABLE	DESCRIPTION
	(default).
	EQ.1: The exterior surface is offset from the mid-plane of the shell by half the shell element's thickness.

Optional unit card. Additional card when UNIT = 3.

Card 6	1	2	3	4	5	6	7	8
Variable		MASS		TIME		LENGTH		
Type		F		F		F		
Default		none		none		none		

VARIABLE	DESCRIPTION
MASS, TIME, LENGTH	Conversion factor from current unit to MKS unit. For example, if the current unit is using kg-mm-ms, the input should be 1.0, 0.001, 0.001.

Card 7	1	2	3	4	5	6	7	8
Variable	IAIR	NGAS	NORIF	NID1	NID2	NID3	CHM	CD_EXT
Type	I	I	I	I	I	I	I	F
Default	0	none	none	0	0	0	None	0.

VARIABLE	DESCRIPTION
IAIR	Initial gas inside bag considered: EQ.0: No EQ.1: Yes, using control volume method. EQ.-1: Yes, using control volume method. In this case ambient air enters the bag when PATM is greater than bag pressure. EQ.2: Yes, using the particle method.

VARIABLE	DESCRIPTION
	EQ.4: Yes, using the particle method. Initial air particles are used for the gas front tracking algorithm, but they do not apply forces when they collide with a segment. Instead, a uniform pressure is applied to the airbag based on the ratio of air and inflator particles. In this case NPRLX must be negative so that forces are not applied by the initial air.
	EQ.-4: Yes, using the particle method. Similar to 4, except the PV energy created to go against the ambient pressure is removed from the airbag system based on the following equation: $$PV = \sum_{\text{segments}} \frac{\text{Collision}_{\text{air}}}{\text{Collision}_{\text{particle}}} V_{\text{seg}} P_{\text{atm}} A_{\text{seg}} dt$$
NGAS	Number of gas components
NORIF	Number of orifices
NID1 - NID3	Three nodes defining a moving coordinate system for the direction of flow through the gas inlet nozzles (Default = fixed system)
CHM	Chamber ID used in *DEFINE_CPM_CHAMBER. See Remark 7.
CD_EXT	Drag coefficient of the external air. If the value is not zero, the inertial effect from external air will be considered and forces will be applied in the normal direction on the exterior airbag surface.
	GT.0: Drag coefficient
	LT.0: Curve ID for time dependent drag coefficient

NOTE: To model drag, you must define ambient air properties with IAR.

Internal Part Set Cards. Additional Cards for STYPE2 = 2. Define SID2 cards, one for each internal part or part set.

Card 8	1	2	3	4	5	6	7	8
Variable	SIDUP	STYUP	PFRAC	LINKING				
Type	I	I	F	I				
Default	none	none	0.	none				

VARIABLE	DESCRIPTION
SIDUP	Part or part set ID defining the internal parts that pressure will be applied to. This internal structure acts as a valve to control the external vent hole area. Pressure will be applied only after switch to UP (uniform pressure) using TSW.
STYUP	Set type: EQ.0: Part EQ.1: Part set
PFRAC	Fraction of pressure to be applied to the set (0.0 to 1.0). If PFRAC = 0.0, no pressure is applied to internal parts.
LINKING	Part ID of an internal part that is coupled to the external vent definition. The minimum area of this part or the vent hole will be used for actual venting area.

Heat Convection Part Set Cards. Additional cards for NPDATA > 0. Define NPDATA cards, one for each heat convection part or part set.

Card 9	1	2	3	4	5	6	7	8
Variable	SIDH	STYPEH	HCONV	PFRIC	SDFBLK	KP	INIP	CP
Type	I	I	F	F	F	F	I	F
Default	none	none	none	FRIC	1.0	0.	0	none

VARIABLE	DESCRIPTION
SIDH	Part or part set ID defining part data.
STYPEH	Set type: EQ.0: Part EQ.1: Part set EQ.2: Part. HCONV is the *DEFINE_CPM_NPDATA ID. EQ.3: Part set. HCONV is the *DEFINE_CPM_NPDATA ID.
HCONV	Convective heat transfer coefficient used to calculate heat loss from the airbag external surface to the ambient. See *AIRBAG_HYBRID developments (Resp. P.O. Marklund). LT.0: |HCONV| is a load curve ID that defines the heat convection coefficient as a function of time. *DEFINE_CURVE_FUNCTION is supported for specifying this curve. When STYPEH is greater than 1, HCONV is a *DEFINE_CPM_NPDATA ID.
PFRIC	Friction factor F_r if -1.0 < PFRIC ≤ 1.0. Defaults to FRIC from Card 1 if undefined. Otherwise, LE.-1.0: |PFRIC| is the curve ID that defines F_r as a function of the part pressure. GT.1.0: PFRIC is the *DEFINE_FUNCTION ID that defines F_r. See Remark 2.
SDFBLK	Scaling down factor for blockage factor (Default = 1.0, no scaling down). The valid factor will be (0.0,1.0]. If 0.0, it will set to 1.0.
KP	Thermal conductivity of the part. See Remark 9.
INIP	Place initial air particles on surface. EQ.0: Yes (default) EQ.1: No This feature excludes surfaces from initial particle placement. This option is useful for preventing particles from being trapped between adjacent fabric layers.
CP	Specific heat (see Remark 16).

Vent Hole Card. Additional cards for NVENT > 0. Define NVENT cards, one for vent hole.

Card 10	1	2	3	4	5	6	7	8
Variable	SID3	STYPE3	C23	LCTC23	LCPC23	ENH_V	PPOP	
Type	I	I	F	I	I	I	F	
Default	0	none	1.0	0	0	0	0.0	

VARIABLE	DESCRIPTION
SID3	Part or part set ID defining vent holes.
STYPE3	Set type: EQ.0: Part EQ.1: Part set which each part being treated separately EQ.2: Part set and all parts are treated as one vent. See Remark 13.
C23	GE.0: Vent hole coefficient, a parameter of Wang-Nefske leakage. A value between 0.0 and 1.0 can be input. See Remark 1. LT.0: ID for *DEFINE_CPM_VENT. Some extended options can only be defined through *DEFINE_CPM_VENT. Please check this keyword for more information.
LCTC23	Load curve defining the vent hole coefficient as a function of time. LCTC23 can be defined through *DEFINE_CURVE_FUNCTION. If omitted, a curve equal to 1.0 used. See Remark 1.
LCPC23	Load curve or function defining the vent hole coefficient as a function of pressure. If omitted a curve equal to 1.0 is used. See Remark 1. GT.0: Load curve ID for *DEFINE_CURVE_FUNCTION or *DEFINE_CURVE LT.0: *DEFINE_FUNCTION ID
ENH_V	Enhanced venting option. See Remark 8.

VARIABLE	DESCRIPTION
	EQ.0: Off (default) EQ.1: On EQ.2: Two-way flow for internal vent; treated as hole for external vent (see Remark 8)
PPOP	Pressure difference between interior and ambient pressure (PATM) to open the vent holes. Once the vents are open, they will stay open.

Air Card. Additional card for IAIR ≠ 0.

Card 11	1	2	3	4	5	6	7	8
Variable	PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR	NPAIR	NPRLX
Type	F	F	F	F	F	F	I	I/F
Default	PATM	TATM	none	none	0.0	0.0	0	0

VARIABLE	DESCRIPTION
PAIR	Initial pressure inside bag
TAIR	Initial temperature inside bag
XMAIR	Molar mass of gas initially inside bag: LT.0: -XMAIR references the ID of a *DEFINE_CPM_GAS_PROPERTIES keyword that defines the gas thermodynamic properties. Note that AAIR, BAIR, and CAIR are ignored.
AAIR - CAIR	Constant, linear, and quadratic heat capacity parameters
NPAIR	Number of air particles. See Remark 6.
NPRLX	Number of cycles to reach thermal equilibrium. See Remark 6. LT.0: If more than 50% of the collision to fabric is from initial air particles, the contact force will not apply to the fabric segment in order to keep its original shape. If the number contains ".", "e" or "E", NPRLX will treated as an

VARIABLE	DESCRIPTION
	end *time* rather than as a cycle count. When the keyword option INFLATION is used, NPRLX is also the number of cycles for maintaining the initial pressure given by PAIR.

MOLEFRACTION Card. Additional card for the MOLEFRACTION option.

Card 12	1	2	3	4	5	6	7	8
Variable	LCMASS							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
LCMASS	Total mass flow rate curve for the MOLEFRACTION option

Gas Cards. NGAS additional cards, one for each gas (card format for i^{th} gas).

Card 13	1	2	3	4	5	6	7	8
Variable	LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*	INFG*i*	
Type	I	I	F	F	F	F	I	
Default	none	none	none	none	0.0	0.0	1	

VARIABLE	DESCRIPTION
LCM*i*	Mass flow rate curve for gas component *i*, unless the MOLEFRACTION option is used. If the MOLEFRACTION option is used, then it is the time dependent molar fraction of the total flow for gas component *i*.
LCT*i*	Temperature load curve/DEFINE_FUNCTION for gas component *i*

VARIABLE	DESCRIPTION
XM*i*	Molar mass of gas component *i*. LT.0: The absolute value of XM*i* references the ID of a *DEFINE_CPM_GAS_PROPERTIES keyword that defines the gas thermodynamic properties. Note that A*i*, B*i*, and C*i* are ignored.
A*i* - C*i*	Constant, linear, and quadratic heat capacity parameters for gas component *i*
INFG*i*	Inflator ID for this gas component (defaults to 1). The user inflator routine can only be defined using *DEFINE_CPM_GAS_PROPERTIES. Please check this extra card for the interface I/O.

Orifice Cards. NORIF additional cards, one for each orifice (card format for i^{th} orifice).

Card 14	1	2	3	4	5	6	7	8
Variable	NID*i*	AN*i*	VD*i*	CA*i*	INFO*i*	IMOM	IANG	CHM_ID
Type	I	F	I	F	I	I	I	I
Default	none	↓	none	30°	1	0	0	0

VARIABLE	DESCRIPTION
NID*i*	Node ID/Shell ID defining the location of nozzle *i*. See Remark 12.
AN*i*	GT.0: Area of nozzle *i* (default: all nozzles are assigned the same area). LT.0: Load curve ID. Time dependent area of nozzle *i*.
VD*i*	GT.0: Vector ID. Initial direction of gas inflow at nozzle *i*. LT.0: Values in the NID*i* fields are interpreted as shell IDs. See Remark 12. EQ.-1: Direction of the gas inflow is the shell's normal direction EQ.-2: Direction of the gas inflow is the reverse direction from the shell's normal direction EQ.-3: Direction of the gas inflow is the shell's normal direction. This option requires including an additional card (Card

VARIABLE	DESCRIPTION
	14.1) for setting the offset from the shell's center point. EQ.-4: Direction of gas inflow is the reverse direction from the shell's normal direction. This option requires including an additional card (Card 14.1) for setting the offset from the shell's center point.
CA*i*	Cone angle in degrees (defaults to 30°). This option is used only when IANG is equal to 1.
INFO*i*	Inflator ID for this orifice. (default = 1)
IMOM	Inflator reaction force (R5.1.1 release and later): EQ.0: Off EQ.1: On
IANG	Activation for cone angle to use for friction calibration (not normally used; eliminates thermal energy of particles from inflator). EQ.0: Off (default) EQ.1: On
CHM_ID	Chamber ID where the inflator node resides. Chambers are defined using *DEFINE_CPM_CHAMBER.

Additional Orifice Cards. Include this card in a set with Card 14 if VD*i* = -3 or -4.

Card 14.1	1	2	3	4	5	6	7	8
Variable	XOFF							
Type	F							
Default	0.0							

VARIABLE	DESCRIPTION
XOFF	Offset distance from the shell's center to the inflator position

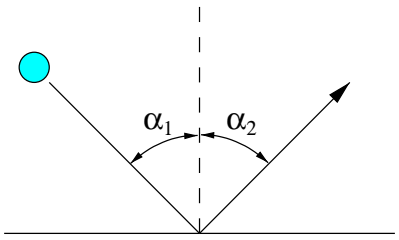


Figure 3-11. Particle Deflection

Remarks:

1. **Formula for total vent hole coefficient.** The value must be between 0.0 and 1.0.

   Total vent hole coefficient = min(max(C23 × LCTC23 × LCPC23, 0.0), 1.0)

2. **Surface roughness.** Friction factor to simulate the surface roughness. If the surface is frictionless the particle incoming angle α_1 is equal to the deflection angle α_2 (see Figure 3-11).

   The surface roughness F_r and the total angle α will have the following relationships:

   $$0.0 \le F_r \le 1.0$$
   $$\alpha = \alpha_1 + \alpha_2$$

   For the special case when

   $$F_r = 1.0 \ ,$$

   the incoming particle will bounce back from its incoming direction, so

   $$\alpha = 0.0 \ .$$

   For $-1.0 < F_r < 0.0$, particles will bounce towards the surface by the following relationship

   $$\alpha = 2\left[\alpha_1 - F_r\left(\frac{\pi}{4} - \frac{\alpha_1}{2}\right)\right] \ .$$

   If $F_r \le -1.0$, the absolute value is the curve ID which defines the F_r as a function of the part pressure.

   If $F_r > 1.0$, the value is the *DEFINE_FUNCTION ID. Currently, the code will pass the following 3 values in this function: airbag pressure, airbag volume, and current time. The function will return the value of F_r. A simple example of user provide function with ID 900 is shown below.

```
*DEFINE_FUNCTION
       900
float pfric100(float pressure_bag, float volume_bag, float current_t)
/* function using bag volume to set Fr  */
{
```

```
float x=0., vol1=0.5, vol2=1.1, vol3=1.2;
  if(volume_bag>=0 && volume_bag<vol1 ) {
    x=1.2;
  } elseif (volume_bag>=vol1 && volume_bag<vol2 ) {
    x=1;
  } elseif (volume_bag>=vol2 && volume_bag<vol3) {
    x=1-(volume_bag-vol2)/vol3;
  } elseif (volume_bag>=vol3 ) {
    x=0.;
  }
  return -0.06*x;
}
```

3. **Blocking and BLOCK field.** Setting the 10's digit to 1 allows for physical holes in an airbag. In this case, particles that are far away from the airbag are disabled. In most cases, these are particles that have escaped through unclosed surfaces due to physical holes, failed elements, etc. This reduces the bucket sort search distance.

4. **Convection energy balance.** The change in energy due to convection is given by

$$\frac{\mathrm{d}E}{\mathrm{d}t} = A \times \mathrm{HCONV} \times \left(T_{\mathrm{bag}} - T_{\mathrm{atm}}\right) \ ,$$

where

$$\begin{aligned} A &= \text{part area} \\ \mathrm{HCONV} &= \text{user-defined heat convection coefficient} \\ T_{\mathrm{bag}} &= \text{the weighted average temperature impacting particles} \\ T_{\mathrm{atm}} &= \text{ambient temperature} \end{aligned}$$

5. **Output files.** Particle time history data is output to the d3plot database. LS-PrePost can display and fringe this data. To reduce runtime memory requirements, VISFLG controls the data output. Note that VISFLG only affects version 4 of the CPM output. Version 3 is the default, so you need to change the value of CPMOUT on *CONTROL_CPM for VISFLG to have any effect.

6. **Spatial distribution equilibration for airbag particles.** Total number of particles initialized is NP + NPAIR. Since the initial air particles are placed at the surface of the airbag segments with correct velocity distribution initially, particles are not randomly distributed in space. It requires a finite number of relaxation cycles, NPRLX, to allow particles to move and produce better spatial distribution.

   Since the momentum and energy transfer between particles are based on perfect elastic collisions, the CPM solver would like to keep a similar mole per particle between the inflator and initial air particles. The CPM solver will check the following factor:

$$\mathrm{factor} = \left| 1 - \frac{\text{mole per particle of initial air}}{\text{mole per particle of inflator gas}} \right| \ .$$

If the factor is more than 10% apart, the code will issue the warning message with the tag, (SOL+1232), and provide the suggested NPAIR value. The NPAIR value should be adjusted based on the application. For example, this message should be ignored if for certain impact analyses the simulation is setup with only initial air, that is, no inflator gas.

7. **Remark concerning *DEFINE_CPM_CHAMBER.** By default, initial air particles are evenly placed on airbag segments, which cannot sense the local volume. This creates an incorrect pressure field if the bag has several distinct pockets. *DEFINE_CPM_CHAMBER allows you to initialize air particles using the volume ratios of regions of the airbag. The particles are distributed proportional to the defined chamber volume to achieve better pressure distribution.

   When looking at the airbag statistics in abstat_cpm, plotting the output for the chambers is more meaningful than for the whole bag. The lumped bag data could possibly give wrong pressure and temperature outputs, which can be misleading.

8. **Enhanced venting.** When enhanced venting is on, the vent hole's equivalent radius R_{eq} will be calculated. Particles within R_{eq} on the high pressure side of the vent hole geometry center will be moved toward the hole. This will increase the collision frequency near the vent for particles to detect small structural features and produce better flow through the vent hole.

   When ENH_V equals 1, particles flow from high to low pressure only. When EHN_V equals 2, particles can flow in both directions for internal vent.

   Particles encountering external vents are released. The ambient pressure is not taken into account, and the particle will be released regardless of the value of the pressure in the bag/chamber. Therefore, the vent rate will be sensitive to the vent location.

9. **Effective convection heat transfer coefficient.** If the thermal conductivity, KP, is given, then the effective convection heat transfer coefficient is given by

$$H_{\text{eff}} = \left(\frac{1.0}{\text{HCONV}} + \frac{\text{shell thickness}}{\text{KP}} \right)^{-1},$$

   where the part thickness comes from the SECTION database. If KP is not given, H_{eff} defaults to HCONV.

10. **MOLEFRACTION option.** Without the MOLEFRACTION option, a flow rate is specified for each species on the LCM*i* fields of Card 13. With the MOLEFRACTION option the total mass flow rate is specified in the LCMASS field on Card 12 and the molar fractions are specified in the LCM*i* fields of Card 13.

11. **User-defined units.** If UNIT = 3 is used, there is no default value for TATM and PATM, and proper values must be provided. Unit conversion factors must also be provided so that the code sets the correct universal gas constant as well as some other variables.

12. **Shell-based nozzle.** Node ID and shell ID based nozzles should not be used in the same airbag definition. The nozzle location is taken to be at the center of the shell, and the initial nozzle direction can be defined by the shell's normal vector or by its reversed normal vector. The airbag definition must include the shell element. This vector transforms with the moving coordinate system defined by NID1 - NID3. The nozzle area is set on the AN*i* fields, and shell area is not taken into account; therefore, the mass flow rate distribution with shells is determined in the same way as it is with nozzles defined by nodes.

13. **Merge part set for vent.** When STYPE3 = 2, the first part in the set is designated as the master. All remaining parts are merged into the master so that enhanced venting is treated correctly. ABSTAT_CPM output will be associated with the master part. This option has the same effect as manually merging elements into the master part.

14. **TEND.** If TEND is not defined, particle mass and release rate will be calculated using the whole of the mass-flow curves LCM*i*, regardless of termination time. TEND can be used to limit the curve data but must be equal to or greater than the analysis termination time.

15. **TSPLIT.** Particles that exit by porosity leakage or a vent are removed from the system. If TSPLIT is set, the code keeps track of the number of removed particles (A) and active particles (B) every 200 cycles after time TSPLIT. Once A is greater than B, each active particle will be split into A/B + 1 particles for a better particle density in the volume.

16. **CP.** If the specific heat of the structure part is given, the initial temperature of this part has the initial air temperature, and the part mass (M) is automatically calculated. Heat transfer between gas and structure is based on the convection equation. The time history of the part temperature is stored in the abstat_cpm database under the field of *part_temp*

$$E = \text{AREA} \times H_{\text{eff}} \times \left(T_{\text{gas}} - T_{\text{part}}(t-1)\right)$$

$$T_{\text{part}}(t) = T_{\text{part}}(t-1) + E/(M \times CP)$$

17. **INFLATION keyword option.** The pressure in the closed volume may gradually deviate from the initial value due to volume changes caused by the pressure difference, such as during the process of tire inflation. INFLATION is designed to maintain this initial pressure by adding mass to the initial gas in the closed volume during the NPRLX time steps.

18. **JET keyword option.** For CPM vent, the flow equation is used to construct the probability density function to release or reflect the particle. The algorithm does not create the thrusting effect. To get correct thrust force created from the discharge, the thrust force is evaluated from the following equation and the reaction force is applied to the JNODE input from user.

$$F_{\text{thrust}} = \dot{m}(v_{\text{sound}} - v_{\text{exit}}) + A_{\text{vent}}(P_{\text{bag}} - P_{\text{ambient}})$$
$$F_{\text{JNODE}} = -F_{\text{thrust}}$$

19. **DPWR.** For IAIR = 4, a fraction of the ambient air pressure is applied to the segment from inside the airbag against ambient pressure external to the airbag based on the following equation:

$$\mu = \left(\frac{N_{\text{air}}}{N_{\text{air}} + N_{\text{infl}}}\right)^{\text{DPWR}} .$$

Here,

μ = fraction of ambient pressure applied inside the airbag
N_{air} = number of collisions from initial air particles
N_{infl} = number of collisions from inflator gas particles

This internal air pressure prevents the fabric layers from pressing against each other prior to inflation. This equation models the partial pressure from the air becoming negligible as the inflator gas builds up inside the airbag. Since the ratio is less than 1, the larger the DPWR, the faster this internal air pressure is removed. If this air pressure decreases too quickly, the gap between fabric layers will close due to ambient pressure because the inflator gas does not increase quickly enough.

***AIRBAG_REFERENCE_GEOMETRY**_{OPTION}_{OPTION}_{OPTION}

Available options include:

<BLANK>

BIRTH

RDT

ID

Purpose: If the reference configuration of the airbag is taken as the folded configuration, the geometrical accuracy of the deployed bag will be affected by both the stretching and the compression of elements during the folding process. Such element distortions are very difficult to avoid in a folded bag. By reading in a reference configuration such as the final unstretched configuration of a deployed bag, any distortions in the initial geometry of the folded bag will have no effect on the final geometry of the inflated bag. This is because the stresses depend only on the deformation gradient matrix:

$$F_{ij} = \frac{\partial x_i}{\partial X_j}$$

where the choice of X_j may coincide with the folded or unfold configurations. It is this unfolded configuration which may be specified here.

When the BIRTH option is specified an additional card setting the BIRTH parameter is activated. The BIRTH parameter specifies a critical time value before which the reference geometry is *not used*. Until the BIRTH time is reached the input geometry is used for (1) inflating the airbag and for (2) determining the time step size, even when the RDT option is set.

NOTE: This card does not support multiple birth times. The last BIRTH value read will be used for *all* preceding *AIRBAG_REFERENCE_GEOMETRY_BIRTH definitions. RGBRTH in *MAT_FABRIC supports a material dependent birth time.

When the RDT option is active the time step size will be based on the reference geometry once the solution time exceeds the birth time. This option is useful for shrunken bags where the bag does not carry compressive loads and the elements can freely expand before stresses develop. If this option is not specified, the time step size will be based on the current configuration and will increase as the area of the elements increase. The default may be much more expensive but possibly more stable.

ID card. Additional card for keyword option ID.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	SX	SY	SZ	NIDO	IOUT		
Type	I	F	F	F	I	I		
Default	none	1.0	1.0	1.0	1st NID	0		

Birth Card. Additional card for keyword option BIRTH.

Card 1	1	2	3	4	5	6	7	8
Variable	BIRTH							
Type	F							
Default	0.0							

Node Cards. For each node ID having an associated reference position include an additional card in format 2. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z				
Type	I	F		F		F				
Default	none	0.		0.		0.				

VARIABLE	DESCRIPTION
ID	Card ID
SX, SY, SZ	Scale factor in each direction
NIDO	Node ID for origin. Default is the first node ID defined in this keyword.

VARIABLE	DESCRIPTION
IOUT	Flag for outputting the current reference node coordinates EQ.0: Do not output the current reference node coordinates. EQ.1: Store the current reference node coordinates for output to a keyword file. See Remark 4.
BIRTH	Time at which the reference geometry activates
NID	Node ID for which a reference configuration is defined. Nodes defined in this section must also appear under the *NODE input. It is only necessary to define the reference coordinates of nodal points, if their coordinates are different than those defined in the *NODE section.
X	x coordinate
Y	y coordinate
Z	z coordinate

Remarks:

1. **Smaller reference geometry.** Note that a reference geometry which is smaller than the initial airbag geometry will not induce initial tensile stresses.

2. **Liner.** If a liner is included and the parameter LNRC is set to 1 in *MAT_FABRIC, compression is disabled in the liner until the reference geometry is reached, i.e., the fabric element becomes tensile.

3. **Nodes shared by foam and fabric.** Some shell and solid elements modeling the fabric and foam, respectively, share nodes. The reference geometry specified by either *INITIAL_FOAM_REFERENCE_GEOMETRY or *AIRBAG_REFERENCE_GEOMETRY will be applied to both the shell and solid elements that share these nodes. However, having different reference geometry for shell and solid elements sharing common nodes can be achieved by using *INITIAL_FOAM_REFERENCE_GEOMETRY for solid elements and *AIRBAG_SHELL_REFERENCE_GEOMETRY for shell elements.

4. **IOUT file.** When IOUT = 1, the current reference coordinates are output to keyword file [JOBID].AIRBAG_REFERENCE_GEOMETRY.TRANSFORMED. The [JOBID] indicates that this is replaced with the job ID.

*AIRBAG_SALE_{OPTION1}_{OPTION2}

Available settings of *OPTION1* are:

<BLANK>

EXTRA

With *OPTION1* set to EXTRA, an extra line containing advanced options is included in the input. These advanced options enable changing certain default S-ALE airbag parameters.

Available settings of *OPTION2* are:

<BLANK>

ID

TITLE

To specify an airbag ID and a heading for the airbag, set *OPTION2* to either ID or TITLE.

Purpose: Define an airbag using the S-ALE method. This keyword serves as a wrapper that is translated into several S-ALE keywords during the keyword read-in phase. See Remark 1.

> **NOTE:** This model assumes that surface normal vectors be oriented *inward*, unlike the traditional Control Volume or CV method (also known as Uniform Pressure or UP method) for which they must point *outward*. If the assumption of *inward* orientation is *not* valid, IFLIP = 1 needs to be set. See Remark 2 for a detailed explanation.

Card Summary:

Card ID. Include this card when *OPTION2* is either ID or TITLE.

ID	HEADING	

Card 1. This card is required.

SID1	STYPE1	SID2	STYPE2				

Card 2. This card is required.

	UNIT		TATM	PATM	NVENT		TSW

Card 3. This card is required.

	NGAS	NORIF					

Card 4. Define NVENT of this card. See Card 2.

SID3	STYPE3	C23					

Card 5. This card is required. Define air material properties.

PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR		

Card 6. Include NGAS of this card. See Card 3.

LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*		

Card 7. Include NORIF of this card. See Card 3.

NID*i*	AN*i*	VD*i*					

Card 8. Include this card if *OPTION1* is set to EXTRA.

IFLIP	LCVEL	PMAX			ILOCK	ILVL	DECAY

Data Card Definitions:

Title Card. Additional card for ID or TITLE keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A60						

VARIABLE	DESCRIPTION
ID	Airbag ID
HEADING	Airbag heading

Card 1	1	2	3	4	5	6	7	8
Variable	SID1	STYPE1	SID2	STYPE2				
Type	I	I	I	I				
Default	none	0	0	0				

VARIABLE	DESCRIPTION
SID1	Part or part set ID defining the complete airbag
STYPE1	Set type: EQ.0: Part EQ.1: Part set
SID2	Part or part set ID defining the internal parts of the airbag
STYPE2	Set type: EQ.0: Part EQ.1: Part set

Card 2	1	2	3	4	5	6	7	8
Variable		UNIT		TATM	PATM	NVENT		TSW
Type		I		F	F	I		F
Default		0		293 K	1 atm	0		10^{10}

VARIABLE	DESCRIPTION
UNIT	Unit system: EQ.0: kg-mm-ms-K EQ.1: SI EQ.2: tonne-mm-s-K

VARIABLE	DESCRIPTION
TATM	Atmospheric temperature
PATM	Atmospheric pressure
NVENT	Number of vent hole parts or part sets. See Remark 5.
TSW	Time at which the algorithm switches to control volumes

Card 3	1	2	3	4	5	6	7	8
Variable		NGAS	NORIF					
Type		I	I					
Default		none	none					

VARIABLE	DESCRIPTION
NGAS	Number of gas components
NORIF	Number of orifices

Vent Hole Card. Additional cards for NVENT > 0. Define NVENT cards, one for each vent hole.

Card 4	1	2	3	4	5	6	7	8
Variable	SID3	STYPE3	C23					
Type	I	I	F					
Default	0	none	1.0					

VARIABLE	DESCRIPTION
SID3	Part or part set ID defining vent holes
STYPE3	Set type: EQ.0: Part

VARIABLE	DESCRIPTION
	EQ.1: Part set
	EQ.2: Segment set
C23	Vent hole coefficient, a parameter of Wang-Nefske leakage

Air Card. Mandatory card to define air properties.

Card 5	1	2	3	4	5	6	7	8
Variable	PAIR	TAIR	XMAIR	AAIR	BAIR	CAIR		
Type	F	F	F	F	F	F		
Default	PATM	TATM	none	none	0.0	0.0		

VARIABLE	DESCRIPTION
PAIR	Initial pressure inside the bag (optional if PATM is set)
TAIR	Initial temperature inside the bag (optional if TATM is set)
XMAIR	Molar mass of the gas initially inside the bag
AAIR - CAIR	Constant, linear, and quadratic heat capacity parameters

Gas Cards. NGAS additional cards, one for each gas (card format for i^{th} gas).

Card 6	1	2	3	4	5	6	7	8
Variable	LCM*i*	LCT*i*	XM*i*	A*i*	B*i*	C*i*		
Type	I	I	F	F	F	F		
Default	none	none	none	none	0.0	0.0		

VARIABLE	DESCRIPTION
LCM*i*	Mass flow rate curve for gas component *i*
LCT*i*	Temperature load curve for gas component *i*

VARIABLE	DESCRIPTION
XM*i*	Molar mass of gas component *i*.
A*i* – C*i*	Constant, linear, and quadratic heat capacity parameters for gas component *i*

Orifice Cards. NORIF additional cards, one for each orifice (card format for i^{th} orifice).

Card 7	1	2	3	4	5	6	7	8
Variable	NID*i*	AN*i*	VD*i*					
Type	I	F	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
NID*i*	Node ID defining the location of nozzle *i*
AN*i*	Area of nozzle *i*. See Remark 6.
VD*i*	Vector ID defining of gas inflow direction at nozzle *i*

EXTRA Card. Include this card is *OPTION1* is set to EXTRA. This card facilitates changing certain pre-determined parameters in S-ALE simulations.

Card 8	1	2	3	4	5	6	7	8
Variable	IFLIP	LCVEL	PMAX			ILOCK	ILVL	DECAY
Type	I	I	F			I	I	F
Default	0	0	PATM			0	0	0.0

VARIABLE	DESCRIPTION
IFLIP	Flag to flip the normal vectors for airbag shell parts. *ALE_STRUCTURED_FSI expects shell segment normal vectors to point inward, toward the inflator gas to which it couples. See Remark 2.

VARIABLE	DESCRIPTION
LCVEL	Load curve ID for a curve giving the gas inlet velocity as a function of time. See Remark 3.
PMAX	Maximum impact pressure. This is used to set penalty stiffness. See Remark 4.
ILOCK	Load curve ID for specifying offset distance. This offset distance is used to avoid "contact locking" between airbag fabric from different chambers. See Remark 7.
ILVL	Number of airbag segments to look downstream from the gas front. ILVL and DECAY are the two flags needed to activate a mechanism to pre-open the closed airbag tube in front of the gas front to allow more gas flow. See Remark 8.
DECAY	Decaying factor. See Remark 8.

Remarks:

1. **Conversion of this keyword.** During the keyword reader phase, this keyword translates to a combination of several different keywords, namely:

   - *ALE_STRUCTURED_MULTI-MATERIAL_GROUP
   - *MAT_ALE_GAS_MIXTURE_ADV
   - *INITIAL_GAS_MIXTURE
   - *ALE_STRUCTURED_POINT_SOURCE
   - *ALE_STRUCTURED_FSI
   - *SET_MULTI-MATERIAL_GROUP_LIST
   - *DEFINE_SALEAB_PARAMETERS

   The converted keywords are output in saleabcvrt.inc.

2. **IFLIP.** *ALE_STRUCTURED_FSI requires the structure's segment normal vectors to point toward the ALE fluids to which the segments are coupled. If the airbag shell parts' normal vectors point outward, use this flag to reverse the segment normal vectors in the FSI coupling segment set without changing the shell elements' connectivity. This flag provides an easy conversion from CV (control volume) airbags to S-ALE airbags because CV bag definitions require airbag segment normal vectors to point outwards.

3. **LCVEL.** The inlet velocity load curve is optional. If zero, a virtual tank test is performed at the initialization phase using the gas inflow data.

4. **PMAX.** PMAX is an estimated value of the maximum impact pressure applied by the gas onto the airbags. This value is used to set up the stiffness in the FSI between the airbag and inflator gas. The bulk modulus of a gas depends largely on its current pressure. For example, γp for isentropic ideal gas.

   Setting up the "right" penalty stiffness is the most important factor in constructing FSI successfully. From prior experience, we apply the maximum impact pressure at 1/10 of the ALE element size. Because it is impossible to obtain this maximum impact pressure, PMAX gives this estimate. It does not need to be exact. One-half to two times different is good enough. If left blank, the maximum impact pressure is taken as 1 atm.

5. **VENT.** The Wang-Nefske leakage algorithm is used. It is the same approach used by OPT = 2 in *AIRBAG_WANG_NEFSKE. Blockage is always on.

6. **AN*i*.** The orifice area is a mandatory input as its value is used in the following calculations:

   a) Making an estimate of the inflator volume. A virtual tank test is performed before the simulation to obtain an estimated gas velocity curve if LCVEL is not prescribed.

   b) Calculating the injected gas volume in a given time cycle. The gas injected in a cycle pushes the gases away and creates a volume for itself. That volume is calculated as the inlet velocity times orifice area times time step.

   c) Distributing the inflow gas between different nozzles. AN*i* divided by the sum of all nozzle areas scales the inflow gas flow rate to calculate the amount of gas flowing through nozzle *i*.

7. **ILOCK.** This option helps solve so-called "contact locking". When gases from two different chambers enter the same S-ALE cell, airbag fabric segments from different chambers can interlock. With ILOCK enabled, a separate FSI card is added to keep gas from other chambers at an offset distance away. ILOCK specifies how this offset distance varies with time. A constant offset equal to 1.5 to 3 times the S-ALE cell size is a good starting point.

8. **Gas front tracking and pre-opening.** A narrow opening in airbag structures is a challenge for continuum methods like S-ALE. If the distance between the two airbag layers is smaller than the S-ALE cell size, the gas flow chokes simply because the gas sees no way to go through. Refining the S-ALE mesh is not a solution here as the opening could be prohibitively small.

   Setting ILVL and DECAY invokes a method that prevents this issue. The method tracks the gas front and forces open a few layers of airbag segments downstream to create a space inlet where the gas can flow. Then, it applies an

artificial force on those fabric segments to pop them up slightly This artificial force is obtained from the gas front FSI pressure. The first flag, ILVL, specifies the number of segments the method looks downstream from the gas front for pre-opening. The second flag, DECAY, serves two purposes:

a) It scales the FSI pressure at the gas front.

b) It fades away this artificial force by capping the new artificial pressure each cycle with DECAY times the last cycle's artificial pressure.

*AIRBAG_SHELL_REFERENCE_GEOMETRY_{OPTION}_{OPTION}

Available options include:

<BLANK>

RDT

ID

Purpose: Usually, the input in this section is not needed; however, sometimes it is convenient to use disjoint pre-cut airbag parts to define the reference geometries. If the reference geometry is based only on nodal input, this is not possible since in the assembled airbag the boundary nodes are merged between parts. By including the shell connectivity with the reference geometry, the reference geometry can be based on the pre-cut airbag parts instead of the assembled airbag. The elements, that are defined in this section, must have identical element IDs as those defined in the *ELEMENT_SHELL input, but the nodal IDs, which may be unique, are only used for the reference geometry. These nodes are defined in the *NODE section but can be additionally defined in *AIRBAG_REFERENCE_GEOMETRY. The element orientation and n1-n4 ordering must be identical to the *ELEMENT_SHELL input.

When the RDT option is active the time step size will be based on the reference geometry once the solution time exceeds the birth time which can be defined by RGBRTH of *MAT_FABRIC.

ID card. Additional card for keyword option ID.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	SX	SY	SZ	NID	IOUT		
Type	I	F	F	F	I	I		
Default	none	1.0	1.0	1.0	↓	0		

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	N3	N4				
Type	I	I	I	I	I	I				
Default	none	none	none	none	none	none				

VARIABLE	DESCRIPTION
ID	Card ID
SX, SY, SZ	Scale factor in each direction
NID	Node ID for origin. The default is the first node ID defined in this keyword.
IOUT	Flag for outputting the current reference node coordinates EQ.0: Do not output the current reference node coordinates. EQ.1: Store the current reference node coordinates for output to a keyword file. See Remark 1.
EID	Element ID
PID	Optional part ID; see *PART. The part ID is not used in this section.
N1	Nodal point 1
N2	Nodal point 2
N3	Nodal point 3
N4	Nodal point 4

Remarks:

1. **IOUT file.** When IOUT = 1, the current reference coordinates are output to keyword file [JOBID].AIRBAG_REFERENCE_GEOMETRY.TRANSFORMED. The [JOBID] indicates that this is replaced with the job ID.

*ALE

The Arbitrary-Lagrangian-Eulerian, or ALE, method is an explicit Finite Element hydrocode. It handles multiple materials (multi-materials) in a single ALE mesh. ALE meshes are neither fixed in space nor attached to a material. The ALE mesh(es) may transform (translate, rotate, and expand) as needed. These transformations can reduce the number of elements required, potentially improving computational efficiency.

A model can contain more than one ALE mesh. Within each ALE mesh multiple ALE materials may move around, deform, and impact each other. If some Lagrangian structures (meshes) overlap with the ALE domain, the ALE materials can interact with these structures. These structures may be constructed as any combination of Lagrangian, SPH, SPG and DEM parts. This fluid-structure interaction (FSI) capability enables analyzing very complex FSI problems. The ALE solver does not have turbulence or boundary layer features, so it is not intended for modeling the traditional CFD problems which require those features.

Generally, the ALE mesh is unstructured to accommodate complex geometries. For simple rectilinear geometries, a structured, logically regular, mesh can be used with the Structure ALE (S-ALE) solver. This solver has algorithmic simplifications, memory reductions, and performance enhancements due to the type of mesh.

ALE *does not* support implicit time integration, nor does it support dynamic relaxation. Furthermore, except for ALE formulation 5, which *does* support contact, ALE *does not*, in general, support contact.

In three dimensions, ALE supports *only* one-point solid elements. These solid elements can either be hexahedral, pentahedral, or tetrahedral. Pentahedrons and tetrahedrons are treated as degenerate hexahedron elements. For each ALE multi-material, strain and stress is evaluated in each solid element at a single integration point. In this sense, the ALE element formulation is equivalent to ELEFORM 1 solid formulation.

Input required for the ALE capability is specified using *ALE cards. The keyword cards in this section, excluding those specifically for the S-ALE solver, are listed in alphabetical order:

*ALE_AMBIENT_HYDROSTATIC

*ALE_BURN_SWITCH_MMG

*ALE_COUPLING_NODAL_CONSTRAINT

*ALE_COUPLING_NODAL_DRAG

*ALE_COUPLING_NODAL_PENALTY

*ALE_COUPLING_RIGID_BODY

*ALE_ESSENTIAL_BOUNDARY

*ALE_FAIL_SWITCH_MMG

*ALE_FRAGMENTATION

*ALE_FSI_PROJECTION

ALE_FSI_SWITCH_MMG_*{OPTION}

*ALE_FSI_TO_LOAD_NODE

*ALE_INJECTION

*ALE_MAPPING

*ALE_MAPPING_FROM_LAGRANGIAN

*ALE_MESH_INTERFACE

*ALE_MULTI-MATERIAL_GROUP

*ALE_PRESCRIBED_MOTION

*ALE_REFERENCE_SYSTEM_CURVE

*ALE_REFERENCE_SYSTEM_GROUP

*ALE_REFERENCE_SYSTEM_NODE

*ALE_REFERENCE_SYSTEM_SWITCH

*ALE_REFINE

*ALE_SMOOTHING

*ALE_SWITCH_MMG

*ALE_TANK_TEST

*ALE_UP_SWITCH

Keywords specifically for the S-ALE solver are:

*ALE_STRUCTURED_FSI

*ALE_STRUCTURED_MESH

*ALE_STRUCTURED_MESH_CONTROL_POINTS

*ALE_STRUCTURED_MESH_JOIN

*ALE_STRUCTURED_MESH_MOTION

*ALE_STRUCTURED_MESH_REFINE

*ALE_STRUCTURED_MESH_TRIM

*ALE_STRUCTURED_MESH_VOLUME_FILLING

*ALE_STRUCTURED_MULTI_DOMAIN_CONTACT

*ALE_STRUCTURED_MULTI-MATERIAL_GROUP

*ALE_STRUCTURED_POINT_SOURCE

For simplifying applying boundary conditions to S-ALE meshes, see:

*BOUNDARY_SALE_MESH_FACE

For other input information related to the ALE capability, see keywords:

*BOUNDARY_AMBIENT_EOS

*CONSTRAINED_EULER_IN_EULER

*CONSTRAINED_LAGRANGE_IN_SOLID

*CONTROL_ALE

*DATABASE_FSI

*INITIAL_VOID

*INITIAL_VOLUME_FRACTION

*INITIAL_VOLUME_FRACTION_GEOMETRY

*SECTION_SOLID

*SECTION_POINT_SOURCE_FOR_GAS_ONLY

*SECTION_POINT_SOURCE_MIXTURE

*SET_MULTIMATERIAL_GROUP_LIST

For a single gaseous material:

*EOS_LINEAR_POLYNOMIAL

*EOS_IDEAL_GAS

*MAT_NULL

For multiple gaseous materials:

*MAT_GAS_MIXTURE

*INTIAL_GAS_MIXTURE

*ALE_AMBIENT_HYDROSTATIC

Purpose: When an ALE model contains one or more ambient (or reservoir-type) ALE parts (ELFORM = 11 and AET = 4), this command may be used to initialize the hydrostatic pressure field in the ambient ALE domain due to gravity. A *LOAD_BODY_*OPTION* keyword must be defined when using this keyword. The associated *INITIAL_HYDROSTATIC_ALE keyword may be used to define a similar initial hydrostatic pressure field for the regular ALE domain (not reservoir-type region).

Card 1	1	2	3	4	5	6	7	8
Variable	ALESID	STYPE	VECID	GRAV	PBASE	RAMPTLC		
Type	I	I	I	F	F	I		
Default	none	0	none	0.0	0.0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	NID	MMGBL						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ALESID	ALESID defines the reservoir-type. ALE domain/mesh whose hydrostatic pressure field due to gravity is being initialized by this keyword. See Remark 4.
STYPE	ALESID set type (see Remark 4): EQ.0: Part set ID EQ.1: Part ID EQ.2: Solid set ID for 3D case EQ.3: Shell set ID for 2D case
VECID	Vector ID of a vector defining the direction of gravity

VARIABLE	DESCRIPTION
GRAV	Magnitude of the gravitational acceleration. For example, in metric units the value is usually set to 9.80665 m/s^2.
PBASE	Nominal or reference pressure at the top surface of all fluid layers. By convention, the gravity direction points from the top layer to the bottom layer. Each fluid layer must be represented by an ALE multi-material group ID (AMMGID or MMG). See Remark 1.
RAMPTLC	A ramping time function load curve ID. This curve (via *DEFINE_-CURVE) defines how gravity is ramped up as a function of time. Given GRAV value above, the curve's ordinate varies from 0.0 to 1.0, and its abscissa is the (ramping) time. See Remark 2.
NID	Node ID defining the top of an ALE fluid (AMMG) layer.
MMGBL	AMMG ID of the fluid layer immediately below this NID. Each node is defined in association with one AMMG layer below it. See Remark 3. In case of S-ALE, AMMG name (AMMGNM) could be also used in place of AMMGID. See Remark 5.

Remarks:

1. **Pressure in multi-layer fluids.** For models using multi-layer ALE Fluids the pressure at the top surface of the top fluid layer is set to PBASE and the hydrostatic pressure is computed as following

$$P = P_{\text{base}} + \sum_{i=1}^{N_{\text{layers}}} \rho_i g h_i \, .$$

2. **Hydrostatic pressure ramp up.** If RAMPTLC is activated (i.e. not equal to "0"), then the hydrostatic pressure is effectively ramped up over a user-defined duration and kept steady. When this load curve is defined, do not define the associated *INITIAL_HYDROSTATIC_ALE card to initialize the hydrostatic pressure for the non-reservoir ALE domain. The hydrostatic pressure in the regular ALE region will be initialized indirectly because of the hydrostatic pressure generated in the reservoir-type ALE domain. The same load curve should be used to ramp up gravity in a corresponding *LOAD_BODY card. With this approach, any submerged Lagrangian structure coupled to the ALE fluids will have time to equilibrate to the proper hydrostatic condition.

3. **Limitation on EOS model.** This keyword only supports *EOS_GRUNEISEN and *EOS_LINEAR_POLYNOMIAL, but only in the following two cases:

$$c_3 = c_4 = c_5 = c_6 = 0, \qquad E_0 = 0$$
$$c_4 = c_5 > 0, \qquad c_1 = c_2 = c_3 = c_6 = 0, \qquad V_0 = 0.$$

4. **S-ALE usage.** When this feature is used with S-ALE, the part and part set options might not make much sense because all elements inside an S-ALE mesh are assigned to one single part ID. Thus, for S-ALE, ALESID should refer to a solid set ID (STYPE = 2) for 3D or shell set ID (STYPE = 3) for 2D, where the set contains the ALE boundary elements on which you want to prescribe hydrostatic pressures. To generate the set, use *SET_SOLID_GENERAL or *SET_SHELL_GENERAL with OPTION = SALECPT.

5. **AMMG name for S-ALE.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, each AMMG can only be referred to by their AMMGID. The AMMGID for each AMMG is based on the order of appearance of the AMMG in the input deck. For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_MULTI-MATERIAL_GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the field AMMGNM. Each AMMG defined with that keyword can then be referred with either its name or its AMMGID (which is again based on order of appearance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, then the AMMGIDs may change. Then, you must find all those references and change them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

Example:

Model Summary: Consider a model consisting of two ALE parts, air on top of water.

H3 = AMMG1 = Air part above

H4 = AMMG2 = Water part below

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ ALE materials (fluids) listed from top to bottom:
$
$ NID AT TOP OF A LAYER SURFACE         ALE MATERIAL LAYER BELOW THIS NODE
$ TOP OF 1st LAYER -------> 1681        ----------------------------------------
$                                       Air above   = PID 3 = H3 = AMMG1 (AET=4)
$ TOP OF 2nd LAYER -------> 1671        ----------------------------------------
$                                       Water below = PID 4 = H4 = AMMG2 (AET=4)
$ BOTTOM ---------------------->        ----------------------------------------
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_AMBIENT_HYDROSTATIC
$   ALESID     STYPE     VECID      GRAV     PBASE   RAMPTLC
        34         0        11   9.80665  101325.0         9
$      NID    MMGBL
      1681         1
      1671         2
*SET_PART_LIST
```

```
        34
         3         4
*ALE_MULTI-MATERIAL_GROUP
         3         1
         4         1
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH       CID
        11       0.0       1.0       0.0       0.0       0.0       0.0
*DEFINE_CURVE
         9
               0.000             0.000
               0.001             1.000
              10.000             1.000
*LOAD_BODY_Y
$     LCID        SF    LCIDDR        XC        YC        ZC
         9   9.80665         0       0.0       0.0       0.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_BURN_SWITCH_MMG

Purpose: Change a volume fraction of an explosive modeled by an ALE multi-material group (AMMGID) into the explosion product (another ALE group). The reaction starts if some ignition conditions are met. These conditions and the reaction rate computing the explosive fraction to be switched are specified by *DEFINE_FUNCTION functions. The functions take as many arguments as there are fields specified on Card 4. This keyword is supported for ALE and S-ALE.

Card Summary:

Card 1. This card is required.

MMGFR	MMGTO	NVARLINE					

Card 2. This card is required.

REACT							

Card 3. This card is required.

IGNI	IGNIV	IGNIVF					

Card 4. Include NVARLINES of this card.

VAR	VAR	VAR	VAR	VAR	VAR	VAR	VAR

Card 5. Include as many of this card as desired. The next keyword ("*") card terminates this input.

PAR	PAR	PAR	PAR	PAR	PAR	PAR	PAR

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MMGFR	MMGTO	NVARLINE					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
MMGFR	ALE multi-material-group (explosive) before the switch
MMGTO	ALE multi-material-group (explosion product) after the switch
NVARLINE	Number of lines with arguments in the functions REACT, IGNI and IGNIV

Card 2	1	2	3	4	5	6	7	8
Variable	REACT							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
REACT	ID of the *DEFINE_FUNCTION function controlling the reaction rate. This function determines the explosive volume fraction to be switched.

Card 3	1	2	3	4	5	6	7	8
Variable	IGNI	IGNIV	IGNIVF					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
IGNI	ID of the *DEFINE_FUNCTION function controlling the conditions of ignition
IGNIV	ID of the *DEFINE_FUNCTION function computing the ignition front speed. See Remark 1.

VARIABLE	DESCRIPTION
IGNIVF	Flag that activates computing the ignition front as a material interface between MMGFR and MMGTO. This flag will automatically turn on if *both* IGNI and IGNIVF are undefined (see Remark 2). EQ.0: Not activated EQ.1: Activated

Variable Cards. Cards defining the function arguments. Include NVARLINES of this card.

Card 4	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
VAR*i*	Variables that provide the arguments for REACT, IGNI, and IGNIV (see Remark 3): EQ.1: *xx*-stress for MMGFR EQ.2: *yy*-stress for MMGFR EQ.3: *zz*-stress for MMGFR EQ.4: *xy*-stress for MMGFR EQ.5: *yz*-stress for MMGFR EQ.6: *zx*-stress for MMGFR EQ.7: Plastic strain for MMGFR EQ.8: Internal energy for MMGFR EQ.9: Bulk viscosity for MMGFR EQ.10: Volume from the previous cycle for MMGFR GE.11 and LE.20: Other auxiliary variables for MMGFR GE.21 and LE.40: Auxiliary variables for MMGTO (*xx*-stress, …)

VARIABLE	DESCRIPTION
EQ.41:	Mass for MMGFR
EQ.42:	Mass for MMGTO
EQ.43:	Volume fraction for MMGFR
EQ.44:	Volume fraction for MMGTO
EQ.45:	Material volume for MMGFR
EQ.46:	Material volume for MMGTO
EQ.47:	Time step
EQ.48:	Time
EQ.49:	Cycle
GE.50 and LE.57:	x-positions of the ALE nodes
GE.58 and LE.65:	y-positions of the ALE nodes
GE.66 and LE.73:	z-positions of the ALE nodes
GE.74 and LE.81:	x-velocities of the ALE nodes
GE.82 and LE.89:	y-velocities of the ALE nodes
GE.90 and LE.97:	z-velocities of the ALE nodes
GE.98 and LE.105:	x-accelerations of the ALE nodes
GE.106 and LE.113:	y-accelerations of the ALE nodes
GE.114 and LE.121:	z-accelerations of the ALE nodes
GE.122 and LE.129:	Masses of the ALE nodes

Parameter Cards. Cards defining parameters to control user-defined routines in the user-subroutine Fortran files:

- `alebrnswmg_ignitioncondition`
- `alebrnswmg_ignitionspeed`
- `alebrnswmg_reactionrate`

They can be used along with or instead of the functions IGNI, IGNIV, and REACT. The usermat version of LS-DYNA is required to access these routines. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 5	1	2	3	4	5	6	7	8
Variable	PAR1	PAR2	PAR3	PAR4	PAR5	PAR6	PAR7	PAR8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PAR*i*	User-defined routines parameters

Remarks:

1. **Nodal Lighting Times.** Each ALE node has a lighting time. When reached for a given node *i*, the reaction in the elements connected to this node can begin. The lighting time of another node (*j*) of an element connected to node *i* is computed by dividing the distance from *j* to *i* by the ignition speed provided by IGNIV. The first nodes to be ignited can be selected with *INITIAL_DETONATION. If the detonation point (located by (X, Y, Z) in *INITIAL_DETONATION) is close to a node, this node becomes the first node to be ignited. If the detonation point is close to an element center, the nodes of this element are first ignited.

2. **Ignition Front.** If IGNIVF = 1, the ignition front is a material interface between MMGFR and MMGTO. Every node behind the front is ignited; if a node in an element with MMGFR and MMGTO is on the MMGTO side, its lighting time is set to the current time to activate the reaction in the elements connected to this node.

3. **Variable Specification.** The variables are presented to the function IDFUNC as floating point data. The order of the arguments appearing in the *DEFINE_-FUNCTION should match the order of VAR*i* specified on Card 4 (for this

keyword). For example, when there is one card in format 4 containing "48, 49, 41, 42", then the time (48), the cycle (49), and the masses (41 & 42) should be the first, second, third, and fourth arguments to the function defined on the *DEFINE_FUNCTION keyword.

If there is a blank column between 2 VAR*i* fields, the list between the values in these two fields is specified. For example, if the card contains "1, ,6", then the six stresses (1 through 6) are selected as arguments.

*ALE_COUPLING_NODAL_CONSTRAINT_{OPTION}

Available options include:

<BLANK>

ID

TITLE

Purpose: This keyword activates constraint coupling between ALE materials and non-ALE (structure) nodes. The structure nodes may belong to Lagrangian solid, shell, beam, thick shell, or discrete sphere (see *ELEMENT_DISCRETE_SPHERE) elements. In contrast to *ALE_COUPLING_NODAL_PENALTY, caution should be exercised when using EFG, SPG or SPH nodes, as meshless methods generally do not satisfy essential boundary conditions, leading to energy dissipation.

> This keyword requires a 3D ALE formulation. It is, therefore, incompatible with parts defined using *SECTION_ALE2D or *SECTION_ALE1D.

If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP	CTYPE	MCOUP		
Type	I	I	I	I	I	I		
Default	none	none	0	0	1	0		

Card 2	1	2	3	4	5	6	7	8
Variable	START	END				FRCMIN		
Type	F	F				F		
Default	0	10^{10}				0.5		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LSDYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70)
STRSID	ID for a part, part set, segment set, or node set giving the structure. The structure may include Lagrangian solid, shell, beam, thick shell, or discrete sphere elements. EFG, SPH, or EFG nodes may be used, but the boundary conditions may not be satisfied.
ALESID	ID for a part or part set specifying the ALE solid elements
STRSTYP	Set type for STRSID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	Set type for ALESID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
CTYPE	Coupling type: EQ.1: Constrained velocity only EQ.2: Constrained acceleration and velocity
MCOUP	Multi-material option (see Remarks). EQ.0: Couple with all multi-material groups

VARIABLE	DESCRIPTION
	LT.0: MCOUP must be an integer. –MCOUP refers to a set ID of an ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP.
START	Start time for coupling
END	End time for coupling
FRCMIN	Only to be used with nonzero MCOUP. Minimum volume fraction of the fluid materials included in the list of AMMGs to activate coupling. Default value is 0.5.

Remarks:

When MCOUP is a negative integer, say for example MCOUP = -123, then an ALE multi-material set-ID (AMMSID) of 123 must exist. This is an ID defined by a *SET_MULTI-MATERIAL_GROUP_LIST card.

*ALE_COUPLING_NODAL_DRAG

Available options include:

<BLANK>

ID

TITLE

Purpose: This command provides a coupling mechanism to model the drag interaction between ALE fluids and discrete element forms (particles). The particles are assumed to be of spherical shape being either SPH elements or discrete elements (*ELEMENT_DISCRETE_SPHERE). The coupling forces are proportional to the relative speed between the fluid and particles plus the buoyancy force due to gravitational loading.

> This keyword requires a 3D ALE formulation. It is, therefore, incompatible with parts defined using *SECTION_ALE2D or *SECTION_ALE1D.

If the title is not defined, LS-DYNA will automatically generate an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP				
Type	I	I	I	I				
Default	none	none	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	START	END		FCOEF			DIRECG	GRAV
Type	F	F		F			I	F
Default	0	10^{10}		1.0			none	0.0

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70).
STRSID	ID for a part, part set, segment set, or node set specifying the particles. The particles can be SPH or discrete elements.
ALESID	ID for a part or part set ID giving the ALE solid elements. See Remark 1.
STRSTYP	Particle set type: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	ALE set type: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
START	Start time for coupling
END	End time for coupling
FCOEF	Drag coefficient scale factor or function ID to calculate drag coefficient GT.0: Drag coefficient scale factor.

VARIABLE	DESCRIPTION
	LT.0: The absolute value of FCOEF is the Function ID of the user provided function to calculate drag coefficient; See Remark 1.
DIRECG	Gravity force direction. EQ.1: Global x direction EQ.2: Global y direction EQ.3: Global z direction
GRAV	Gravity value. This value is used to calculate buoyance force.

Remarks:

1. **Drag Coupling Force.** The drag coupling force between the particles and ALE fluids takes the following form:

$$F_{\text{drag}} = c_{\text{drag}} \times \frac{1}{2}\rho v^2 \times \frac{1}{4}\pi d^2 \ ,$$

where c_{drag} is the drag coefficient, ρ is the density of the fluid in which the particle is submerged, v is the relative velocity between the particle and the fluid, and d is the diameter of the particle.

The default drag coefficient is a function of Reynolds's number and calculated by using the following formula:

$$c_{\text{drag}} = \left(0.63 + \frac{4.8}{\sqrt{\text{Re}}}\right)^2.$$

You can define your own function of drag coefficient. To do that, you need to define a function using *DEFINE_FUNCTION and assign the negative function ID to FCOEF flag.

An example is provided below to illustrate the setup. It is equivalent to the default drag coefficient calculation.

```
*ALE_COUPLING_NODAL_DRAG
     10001         1         3         1
                                    -10                                 3      9.81
*DEFINE_FUNCTION
        10
float cd(float re)
{
  float cd;
  cd=(0.63+4.8/sqrt(re))*(0.63+4.8/sqrt(re));
  if (cd > 2.0) cd = 2.0;
  return cd;
}
```

*ALE_COUPLING_NODAL_PENALTY

Available options include:

<BLANK>

ID

TITLE

Purpose: This command provides a penalty coupling mechanism between ALE materials and non-ALE (structure) nodes. The structure nodes may belong to Lagrangian solid, shell, beam, thick shell, or discrete (*ELEMENT_DISCRETE_SPHERE) elements. In contrast to *ALE_COUPLING_NODAL_CONSTRAINT, EFG, SPG, and SPH nodes *are* supported.

> This keyword is incompatible with parts that use *SECTION_ALE2D or *SECTION_ALE1D, meaning it requires a 3D ALE formulation.

If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP		MCOUP		
Type	I	I	I	I		I		
Default	none	none	0	0		0		

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	PFORM	PFAC		FRCMIN		
Type	F	F	I	F		F		
Default	0	10^{10}	0	0.1		0.5		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LSDYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70)
STRSID	ID of a part, part set, segment set, or node set giving the structure. The structure may include Lagrangian elements, EFG, SPG, or SPH.
ALESID	ID of a part or part set specifying the ALE solid elements
STRSTYP	Set type for STRSID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	Set type of ALESID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
MCOUP	Multi-material option (see Remarks): EQ.0: Couple with all multi-material groups LT.0: MCOUP must be an integer. –MCOUP refers to a set ID of an ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP.
START	Start time for coupling

VARIABLE	DESCRIPTION
END	End time for coupling
PFORM	Penalty stiffness formulations: EQ.0: Mass based penalty stiffness EQ.1: Bulk modulus based penalty stiffness EQ.2: Penalty stiffness is determined by the user-provided load curve between penetration and penalty pressure.
PFAC	Penalty stiffness factor (PFORM = 0 or 1) for scaling the estimated stiffness of the interacting (coupling) system or load curve ID (PFORM = 2).
FRCMIN	Only to be used with nonzero MCOUP. Minimum volume fraction of the fluid materials included in the list of AMMGs to activate coupling. Default value is 0.5.

Remarks:

When MCOUP is a negative integer, say for example MCOUP = -123, then an ALE multi-material set-ID (AMMSID) of 123 must exist. This is an ID defined by a *SET_MULTI-MATERIAL_GROUP_LIST card.

*ALE_COUPLING_RIGID_BODY

Purpose: This command serves as a simplified constraint type coupling method between ALE fluids and a Lagrange rigid body.

In certain FSI simulations structure deformation is either small or not of the interest. Often these structures are modeled as rigid bodies to shorten the simulation time and reduce the complexity. For such kind of problems, a full scale ALE/FSI simulation is costly in both simulation time and memory. This keyword provides a light-weight alternative FSI method for systems with minimal structural response.

It has a similar input format to *ALE_ESSENTIAL_BOUNDARY and maybe regarded as being an extension of the essential boundary feature. The documentation for *ALE_ESSENTIAL_BOUNDARY_BODY applies, in large part, to this card also.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NSID						
Type	I	I						
Default	none	none						

ALE Coupling Interfaces Cards. Include one card for each part, part set or segment to define ALE coupling interface. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	ICTYPE	IEXCL				
Type	I	I	I	I				
Default	none	none	1	none				

VARIABLE	DESCRIPTION
PID	Rigid body part ID
NSID	Node set ID defining ALE boundary nodes to follow rigid body motion

VARIABLE	DESCRIPTION
ID	Set ID defining a part, part set or segment set ID of the ALE coupling interface
IDTYPE	Type of set ID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID) EQ.2: Segment set ID (SGSID)
ICTYPE	Constraint type: EQ.1: No flow through all directions EQ.2: No flow through normal direction (slip condition)
IEXCL	Segment Set ID to be excluded from applying ALE essential boundary condition. For example, inlet/outlet segments.

Remarks:

For ICTYPE = 2, the constrained direction(s) at each surface node comes in part from knowing whether the node is a surface node, an edge node, or a corner node. If the ALE mesh boundary is identified by part(s) (IDTYPE = 0/1), edge and corner nodes are automatically detected during the segment generation process. However, this automatic detection is not foolproof for complicated geometries. Identifying the ALE mesh boundary using segment sets (IDTYPE = 2) is generally preferred for complicated geometries in order to avoid misidentification of edge and corner nodes. When segment sets are used, the edge and corner nodes are identified by their presence in multiple segment sets where each segment set describes a more or less smooth, continuous surface. The intersections of these surfaces are used to identify edge/corner nodes.

*ALE_ESSENTIAL_BOUNDARY

Purpose: This command applies and updates essential boundary conditions on ALE or S-ALE boundary surface nodes. Updating the boundary conditions is important if the ALE mesh moves according to *ALE_REFERENCE_SYSTEM_GROUP. If the mesh does not move, it's more correct to call it an Eulerian mesh rather than an ALE mesh, but *ALE_ESSENTIAL_BOUNDARY can be applied nonetheless.

Certain engineering problems need to constrain the flow along the ALE mesh boundary. A simple example would be water flowing in a curved tube. Using the *ALE_ESSENTIAL_BOUNDARY approach, the tube material is not modeled and there is no force coupling between the fluid and the tube, rather the interior volume of the tube is represented by the location of the ALE mesh. Defining SPC boundary conditions with a local coordinate system at each ALE boundary node would be extremely inconvenient in such a situation. The *ALE_ESSENTIAL_BOUNDARY command applies the desired constraints along the ALE surface mesh automatically. You only need to specify the part(s) or segment set(s) corresponding to the ALE boundary surfaces and the type of constraint desired.

Boundary Condition Cards. Include one card for each part, part set or segment on which essential boundary conditions are applied. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	ICTYPE	IEXCL				
Type	I	I	I	I				
Default	none	none	1	none				

VARIABLE	DESCRIPTION
ID	Set ID defining a part, part set or segment set ID of the ALE or S-ALE mesh boundary.
IDTYPE	Type of set ID: EQ.0: Part set ID (PSID). EQ.1: Part ID (PID). EQ.2: Segment set ID (SGSID).

VARIABLE	DESCRIPTION
ICTYPE	Constraint type: EQ.1: No flow through all directions. EQ.2: No flow through normal direction. (slip condition)
IEXCL	Segment set ID to be excluded from applying ALE or S-ALE essential boundary condition. For example, inlet/outlet segments.

Remarks:

For ICTYPE = 2, the constrained direction(s) at each surface node comes in part from knowing whether the node is a surface node, an edge node, or a corner node. If the ALE mesh boundary is identified by part(s) (IDTYPE = 0 or 1), edge/corner nodes are automatically detected during the segment generation process. However, this automatic detection is not foolproof for complicated geometries. Identifying the ALE mesh boundary using segment sets (IDTYPE = 2) is generally preferred for complicated geometries in order to avoid misidentification of edge/corner nodes. When segment sets are used, the edge/corner nodes are identified by their presence in multiple segment sets where each segment set describes a more or less smooth, continuous surface. In short, the junctures or intersections of these surfaces identify edge/corner nodes.

***ALE_FAIL_SWITCH_MMG**_*{OPTION}*

Purpose: This card is used to allow the switching of an ALE multi-material-group ID (AMMGID) if a failure criteria is reached. If this card is not used and *MAT_VACUUM has a multi-material group in the input deck, failed ALE groups are replaced by the group for *MAT_VACUUM.

Available options include:

<BLANK>

ID

TITLE

A title for the card may be input between the 11th and 80th character on the title-ID line. The optional title line precedes all other cards for this command.

The user can explicitly define a title for this coupling.

Title Card. Additional card for the ID or TITLE options to keyword.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMG-SID before the switch. The AMMG-SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID points to one or more AMMGs. See Remark 1.

VARIABLE	DESCRIPTION
TO_MMG	This is the AMMG-SID after the switch. The AMMG-SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST card. This SID points to one or more AMMGs. See Remark 1.

Remarks:

1. **AMMG Switch.** There is a correspondence between the FR_MMG and TO_MMG. Consider an example where:

   a) The FR_MMG SID points to a SID = 12 (the SID of its SMMGL card is 12, and this SID contains AMMG 1 and AMMG 2)

   b) The TO_MMG points to a SID = 34 (the SID of the SMMGL card is 34, and this SID contains AMMG 3 and AMMG 4)

   Then, AMMG 1, if switched, will become AMMG 3, and AMMG 2, if switched, will become AMMG 4.

2. **History Variables.** In the switch example above, if AMMG 1 in FR_MMG has n_1 history variables and its corresponding ALE group AMMG 3 in TO_MMG has n_3 history variables, the number of history variables copied from AMMG 1 to AMMG 3 is $\min(n_1, n_3)$.

*ALE_FRAGMENTATION

Purpose: When a material fails in tension, this card is used to switch the failed material to vacuum. When used with FRAGTYP = 2, the solver monitors the volume increment that causes material failure amd converts only the volume increment into vacuum. This approach can be used to model material fragmentation.

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	FRAGTYP					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMGID of the failed material.
TO_MMG	This is the AMMGID of the vacuum to which the failed material is being switched.
FRAGTYP	Flag defining whether the failed material is completely or partially switched to vacuum. EQ.1: Fully switch; all failed material is switched to vacuum (see Remark 1). EQ.2: Partially switch; only the volume expansion from the last time step is switched to vacuum (see Remark 2).

Remarks:

The Lagrange element contains only one material. Once the failure criterion is met in a Lagrange element, the whole element is marked as "failed" and either deleted or kept from further element force calculation.

However, for multi-material ALE elements, such an approach is not practical as these elements are occupied by multiple materials. Failure, therefore, cannot be adequately modeled at the element level. Instead we convert the failed material inside an ALE element to vacuum. The effect is similar to element deletion in Lagrange simulations. The failed material, once switched to vacuum, is excluded from any future element force calculation.

1. **Switch to vacuum (FRAGTYPE = 1).** By default, multi-material elements switch failed materials to vacuum. This switch involves assigning the full volume fraction of the failed material, say AMMG 1, in an element to vacuum, say AMMG 2.

   FRAGTYP = 1 is equivalent to the default treatment. However, with this card the vacuum AMMG can be explicitly specified. In the case that more than one vacuum AMMG exist, it is strongly recommended to use the FRAGTYP = 1 approach to eliminate ambiguity. It is also helpful during post-processing since it is possible to see the material interface of the switched material by assigning a dedicated vacuum AMMG to the switched material.

2. **Fragmentation (FRAGTYPE = 2).** FRAGYP = 2 models material fragmentation. Note that the FRAGTYP = 1 approach leads to loss of mass and, consequently, dissipates both momentum and energy. With FRAGTYP = 2, instead of converting the full volume of the failed material to vacuum, LS-DYNA only converts the material expansion to vacuum. This approach conserves mass and, therefore, momentum and energy.

   To illustrate how this fragmentation model works, consider a tension failure example. At the time step when the material fails, LS-DYNA calculates the material expansion in the current step and converts this volume to vacuum. The stresses and other history variables are left unchanged, so that in the next time step it will again fail. The expansion in the next time step will be also converted to vacuum. This process continues until maybe at a later time the gap stops growing or even starts to close due to compression.

Example:

Consider a simple bar extension example:

FR_MMG:	H5	=	AMMG1	= Metal bar
	H6	=	AMMG2	= Ambient air
TO_MMG:	H7	=	AMMG3	= Dummy vacuum part

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_FRAGMENTATION
$    FR_MMG    TO_MMG   FRAGTYP
         1         3         2
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_FSI_PROJECTION

Purpose: This card provides a coupling method for simulating the interaction between a Lagrangian material set (structure) and ALE material set (fluid). The nearest ALE nodes are projected onto the Lagrangian structure surface at each time step. This method does not conserve energy, as mass and momentum are transferred using the constraint based approach.

Card 1	1	2	3	4	5	6	7	8
Variable	LAGSID	ALESID	LSIDTYP	ASIDTYP	SMMGID	ICORREC	INORM	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH						
Type	F	F						
Default	0.0	10^{10}						

VARIABLE	DESCRIPTION
LAGSID	A set ID defining the Lagrangian part(s) for this coupling (structures).
ALESID	A set ID defining the ALE part(s) for this coupling (fluids).
LSIDTYP	Lagrangian set ID type: EQ.0: Part set ID (PSID), EQ.1: Part ID (PID).
ASIDTYP	ALE set ID type: EQ.0: Part set ID (PSID), EQ.1: Part ID (PID).

VARIABLE	DESCRIPTION
SMMGID	A set ID referring to a group of one or more ALE-Multi-Material-Group (AMMG) IDs which represents the ALE materials interacting with the Lagrangian structure. This SMMGID is a set ID defined by *SET_MULTI-MATERIAL_GROUP_LIST.
ICORREC	Advection error correction method (see Remark 1): EQ.1: ALE mass is conserved. Leaked mass is moved, EQ.2: ALE mass is almost conserved, EQ.3: No correction performed (default). ALE mass is conserved. Some leakage may occur. This may be the best solution.
INORM	Type of coupling: EQ.0: Couple in all directions, EQ.1: Couple in compression and tension (free sliding), EQ.2: Couple in compression only (free sliding). This choice requires ICORREC = 3.
BIRTH	Start time for coupling.
DEATH	End time for coupling.

Remarks:

1. **Advection Errors.** As the ALE nodes are projected onto the closest Lagrangian surface, there may be some advection errors introduced. These errors may result in a small element mass fraction being present on the "wrong" side of the coupled Lagrangian surface. There are 3 possible scenarios:

   a) Mass on the wrong side of the Lagrangian structure may be moved to the right side. This may cause P oscillations. No leakage will occur.

   b) Mass on the wrong side is deleted. Mass on the right side is scaled up to compensate for the lost mass. No leakage will occur.

   c) Mass on the wrong side is allowed (no correction performed). Some leakage may occur. This may be the most robust and simplest approach.

Example:

Model Summary:

H1 = AMMG1 = background air mesh
H2 = AMMG1 = background air mesh
S3 = cylinder containing AMMG2
S4 = dummy target cylinder for impact

The gas inside S3 is AMMG2. S3 is given an initial velocity and it will impact S4.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
         1         1
         2         1
*SET_MULTI-MATERIAL_GROUP_LIST
        22
         2
*ALE_FSI_PROJECTION
$   LAGSID    ALESID   LSIDTYP   ASIDTYP    SMMGID   ICORREC     INORM
         3         1         1         1        22         3         2
$    BIRTH     DEATH
       0.0      20.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_FSI_SWITCH_MMG_{OPTION}

Purpose: This card is used to allow the switching of an ALE multi-material-group ID (AMMGID) of a fluid as that fluid passes across a monitoring surface. This monitoring surface may be a Lagrangian shell structure or a segment set. It does not have to be included in the Lagrangian structure set of the coupling card: *CONSTRAINED_LAGRANGE_IN_SOLID. However, at least one coupling card must be present in the model.

Available options include:

<BLANK>

ID

TITLE

An ID number (up to 8 digits) may be defined for this switch command in the first 10-character space. A title for the card may be input between the 11th and 80th character on the title-ID line. The optional title line precedes all other cards for this command.

The user can explicitly define a title for this coupling.

ID Card. Additional card for the Title or ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	NQUAD	XOFF	BTIME	DTIME	NFREQ	NFOLD
Type	I	I	I	F	F	F	I	I
Default	none	0	1	0.0	0.0	1.0E20	1	0

Card 2	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	XLEN					
Type	I	I	F					
Default	none	none	0.0					

VARIABLE	DESCRIPTION
SID	A set ID defining a monitoring surface over which an ALE fluid flows across, and its ALE multi-material-group-ID (AMMGID) is switched. The monitoring surface may be a Lagrangian shell structure or a segment set. This surface, if Lagrangian, does not have to be included in the coupling definition (see Remark 4).
STYPE	Set ID type of the above SID: EQ.0: Part set ID (PSID) (default). EQ.1: Part ID (PID). EQ.2: Segment set ID (SGSID).
NQUAD	The number of flow-sensor points to be distributed over each monitoring surface/segment. There should be enough sensor points distributed to monitor the flow in each ALE element intersected by this monitoring surface (default = 1; see Remark 3).
XOFF	An offset distance away from the monitoring surface, beyond which the AMMGID switching occurs. The direction of XOFF is defined by the normal vector of the monitoring segment. This offset distance, in general, should be at least 2 ALE element widths away from, and beyond, the monitoring interface (default = 0.0).
BTIME	Start time for the AMMGID switch to be activated (default = 0.0).
DTIME	Ending time for the AMMGID switch (default = 1.0E20).
NFREQ	Number of computational cycles between ALE switch check (default = 1).

VARIABLE	DESCRIPTION
NFOLD	Flag for checking folding logic: EQ.0: Off EQ.1: On. LS-DYNA will check if the monitoring segment is in the fold which is applicable to airbags. If the monitoring segment is still located within a folded (shell) region, then no switching is allowed yet until it has unfolded.
FR_MMG	This is the AMMG SID before the switch. The AMMG SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID points to one or more AMMGs. See Remark 1.
TO_MMG	This is the AMMG SID after the switch. The AMMG SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST card. This SID points to one or more AMMGs. See Remark 1.
XLEN	This is an absolute distance for distributing the flow sensor points over the ALE elements. To make sure that at least 1 sensor point, defined on each Lagrangian segment, is present in each ALE element to track the flow of an AMMG, XLEN may be estimated as roughly half the length of the smallest ALE element in the mesh. See Remark 3.

Remarks:

1. **AMMG SIDs.** There is a correspondence between the FR_MMG and TO_MMG. Consider an example where:

   a) The FR_MMG SID points to a SID = 12 (the SID of its SMMGL card is 12, and this SID contains AMMG 1 and AMMG 2)

   b) The TO_MMG points to a SID = 34 (the SID of the SMMGL card is 34, and this SID contains AMMG 3 and AMMG 4)

   Then, AMMG 1, if switched, will become AMMG 3, and AMMG 2, if switched, will become AMMG 4.

2. **Database Files.** The ID option must be activated if the parameter SWID is used in the *DATABAS_FSI card. Then the accumulated mass of an AMMG that goes through a tracking surface, and being switched, will be reported via the

parameter "mout" in the dbfsi ASCII output file (or equivalently the "POROSITY" parameter inside LS-PrePost ASCII plotting option).

3. **Sensor Point Interval.** When both NQUAD and XLEN are defined, whichever gives smaller sensor-point interval distance will be used. XLEN may give better control as in the case of a null shell acting as the monitoring surface. As this null shell is stretched, NQUAD distribution of sensor-points may not be adequate, but XLEN would be.

4. **Coupling Cards.** The monitoring surface does not have to be included in the Lagrangian structure set of the coupling card. However, at least one coupling card must be present in the model. The monitoring segment set can be made up of Lagrangian or ALE nodes.

Example:

Consider a simple airbag model with 3 part IDs:

H25 (AMMG1) = Inflator gas injected into the airbag.
H24 (AMMG2) = Air outside the airbag (background mesh).
H26 (AMMG3) = Dummy AMMG of inflator gas after it passes through a vent hole.
S9 = A Lagrangian shell part representing a vent hole.
S1 = A Lagrangian shell part representing the top half of an airbag.
S2 = A Lagrangian shell part representing the bottom half of an airbag.

The inflator gas inside the airbag is distinguished from the inflator gas that has passed through the monitoring surface (vent hole) to the outside of the airbag by assigning different ALE multi-material group set ID to each. The dummy fluid part (H26) should have the same material and EOS model IDs as the before-switched fluid (H25).

TO_MMG = 125
⇒ AMMGID (before switch) = *SET_MULTI-MATERIAL_GROUP_LIST(125) = 1
⇒ PART = PART(AMMGID1) = H25

FR_MMG = 126
⇒ AMMGID (before switch) = *SET_MULTI-MATERIAL_GROUP_LIST(126) = 3
⇒ PART = PART(AMMGID3) = H26

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
        25         1
        24         1
        26         1
*DATABASE_FSI
$     TOUT                                  [STYPE: 0=PSID ; 1=PID ; 2=SGSID]
    0.1000
$ DBFSI_ID       SID     STYPE  AMMGSWID  LDCONVID
```

```
         1         1         1
         2         2         1
         3         9         1     90000
*SET_MULTI-MATERIAL_GROUP_LIST
       125
         1
*SET_MULTI-MATERIAL_GROUP_LIST
       126
         3
*ALE_FSI_SWITCH_MMG_ID
     90000
$       SID   SIDTYPE     NQUAD      XOFF     BTIME     DTIME     NFREQ      FOLD
         9         1         3     -20.0       5.0       0.0         1         1
$   Fr_MMG    To_MMG     XCLEN
       125       126        5.
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

Note:

1. The *DATABASE_FSI card tracks 3 surface entities: (a) top half of an airbag, (b) bottom half of an airbag, and (c) the vent hole monitoring surface where the AMMGID of the inflator gas is switched.

2. The amount of mass passing through the vent hole during the switch is output to a parameter called “pleak” in a dbfsi ASCII file. See *DATABASE_FSI.

3. The *ALE_FSI_SWITCH_MMG_ID card track any flow across S9 and switch the AMMGSID from 125 (AMMG 1) to 126 (AMMG 3).

*ALE_FSI_TO_LOAD_NODE

Purpose: Output the ALE coupling forces from *CONSTRAINED_LAGRANGE_IN_SOLID for CTYPE = 4 in keyword format, so they may be applied directly in another run.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	NSID	IOPT					
Type	I	I	I					
Default	none	none	0					

Remaining card is optional.†

Path Card. Optional card for the directory path (see Remark 4).

Card 2	1	2	3	4	5	6	7	8
Variable	PATH							
Type	A80							
Default	optional							

VARIABLE	DESCRIPTION
DT	Output intervals
NSID	Node Set ID. See *SET_NODE.
IOPT	Options to map the coupling data between 2 runs: EQ.0: The keyword alefsiloadnode.k (see Remark 1) is created at the end of the run by LS-DYNA. EQ.1: A database of coupling forces is dumped without the conversion to a keyword file at the end of the run (see Remark 2). The database can be treated by a program (alefsiloadnode.exe) to write alefsiloadnode.k. EQ.2: The database of coupling forces created by IOPT = 1 (see Remark 2) is read back. The structure meshes should be

VARIABLE	DESCRIPTION
	identical. The forces are directly applied to the nodes without using *LOAD_NODE. The parameter DT is not read, but NSID is read.
	EQ.3: A database of coupling accelerations is dumped at the end of the run (see Remark 3).
	EQ.4: The database of coupling accelerations created by IOPT = 3 (see Remark 3) is read back. The structure meshes can be different. The accelerations are interpolated at the nodes provided by NSID. The parameters DT and NSID are read.
PATH	Path to the directory where the databases are created.

Remarks:

1. **Keyword Output File.** The name of the output keyword file is alefsiloadnode.k. For each node, this file contains three *LOAD_NODE for each global direction and three *DEFINE_CURVE for the coupling force histories.

2. **Database for IOPT = 1 and 2.** The name of the database is alefsi2ldnd.tmp (or alefsi2ldnd.tmp00… in MPP). It should be in the directory of the second run for IOPT = 2. The database lists the coupling forces by node. The structure meshes (and their MPP decomposition) for the IOPT = 1 and IOPT = 2 runs should be the same.

3. **Databases for IOPT = 3 and 4.** The names of the databases are alefsi2ldnd.tmp (or alefsi2ldnd.tmp00… in MPP) and alefsi2lndndx.tmp. They should be in the directory of the 2nd run for IOPT = 4. The file alefsi2ldnd.tmp lists the coupling accelerations by node (coupling acceleration = coupling force / nodal mass). The file alefsi2lndndx.tmp lists the initial nodal coordinates and coupling segment connectivities (see *CONSTRAINED_LAGRANGE_IN_SOLID for the coupling segment). The structure meshes for the IOPT = 3 and IOPT = 4 runs can be different. The IOPT = 3 initial geometry stored in alefsi2lndndx.tmp is used to interpolate the coupling accelerations (saved in alefsi2ldnd.tmp) at the nodes provided by NSID. For the interpolation to work, these nodes should be within the IOPT = 3 coupling segment initial locations.

4. **Directory path.** The filenames of the databases can be prefixed by the variable PATH to control in which directory the databases should be created. Since the directory path and file name are concatenated, PATH should end with "/" (or "\" in Windows).

*ALE_INJECTION

Purpose: Select nodes and elements in a user defined volume to prescribe the velocities of the selected nodes and set the materials in the selected elements with material(s) from a multi-material group set.

Card 1	1	2	3	4	5	6	7	8
Variable	MMGSET	SEGSET	GLOBAL	LCE	LCRVL			
Type	I	I	I	I	I			
Default	none	none	0	0	0			

Remaining cards are optional.†

Velocity Card. Optional card for the nodal motion (see Remark 1).

Card 2	1	2	3	4	5	6	7	8
Variable	LCVT	VECT	LCVR	VECR	BOXV	XG	YG	ZG
Type	I	I	I	I	I	F	F	F
Default	0	0	0	0	0	0.0	0.0	0.0

Surface Card. Optional card for defining the geometry to be filled by MMGSET (see Remark 1)

Card 3	1	2	3	4	5	6	7	8
Variable	SURFCT	NDIV	XL	YL	ZD	ZU	XC	YC
Type	I	I	F	F	F	F	F	F
Default	0	3	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
MMGSET	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_-LIST).
SEGSET	Segment set ID (see *SET_SEGMENT). A local coordinate system is created for each segment. See Remark 2.
GLOBAL	Three digit flag to control how to select the elements, how to prescribe the velocities, and how to define the geometrical parameters of Cards 2 and 3 (including BOXV): EQ.__0: Geometrical parameters are local to the segments of SEGSET EQ.__1: Geometrical parameters are natural to SEGSET segments (see Remark 3 and Figure 4-1) EQ._0_: Velocities are applied in local coordinate systems attached to each segment of SEGSET EQ._1_: Velocities are applied in the global coordinate system EQ.0__: Select the elements and nodes in the local volume around each segment of SEGSET EQ.1__: Select the elements in the global volume formed by all the segments of SEGSET. The nodes are selected in the local volume around each segment of SEGSET. EQ.2__: Select the elements and nodes in the global volume formed by all the segments of SEGSET. Velocities are applied in the global coordinate system.
LCE	Curve ID for the internal energy (see Remark 6): GT.0: Load curve ID; see *DEFINE_CURVE. See Remark 2. LT.0: -LCE is the function ID for the internal energy which depends on 26 arguments: time, number of cycles, and nodal coordinates of the 8 nodes for the ALE element. See *DEFINE_FUNCTION. See Remark 5.
LCRVL	Curve ID for the relative volume (see Remark 6): GT.0: Load curve ID; see *DEFINE_CURVE. See Remark 2. LT.0: -LCRVL is the function ID for the relative volume which depends on 26 arguments: time, number of cycles, and nodal coordinates of the 8 nodes for the ALE element. See *DEFINE_FUNCTION. See Remark 5.

VARIABLE	DESCRIPTION
LCVT	Curve ID for the translational velocity: GT.0: Load curve ID; see *DEFINE_CURVE. LT.0: -LCVT is the function ID for the translational velocity which depends on 5 arguments: time, number of cycles, and nodal coordinates. See *DEFINE_FUNCTION. See Remark 5.
VECT	Vector to orient the translation. See *DEFINE_VECTOR.
LCVR	Curve ID for the rotational velocity: GT.0: Load curve ID; see *DEFINE_CURVE. LT.0: -LCVR is the function ID for the rotational velocity which depends on 5 arguments: time, number of cycles, and nodal coordinates. See *DEFINE_FUNCTION. See Remark 5.
VECR	Vector to orient the rotational axis (see *DEFINE_VECTOR).
BOXV	Box (see *DEFINE_BOX) defining the region where the velocities are applied (see Remark 7).
XG, YG, ZG	Position of the rotation center (see Remark 8).
SURFCT	Flag to define the surface, inside which the nodes and elements are selected: LT.0: -SURFCT is the function ID (see *DEFINE_FUNCTION) for the rotational velocity with 17 arguments: time, number of cycles, ALE element center coordinates, segment nodal coordinates. EQ.0: Ellipsoid: $\frac{x^2}{\mathrm{XL}^2}+\frac{y^2}{\mathrm{YL}^2}+\frac{z^2}{\mathrm{ZL}^2}=1$ where ZL = −ZD if $z<0$ and ZL = ZU if $z>0$ (see Remark 9 and Figure 4-3) EQ.1: Ellipse-based cylinder: $\frac{x^2}{\mathrm{XL}^2}+\frac{y^2}{\mathrm{YL}^2}=1$ where −ZD < z < ZU (see Remark 10 and Figure 4-4)

VARIABLE	DESCRIPTION
	EQ.2: Truncated ellipse-based cone: $\frac{x^2}{X^2}+\frac{y^2}{Y^2}=1$ where $0<z<\text{ZD}$, $X=(1-z/\text{ZU})\text{XL}$, and $Y=(1-z/\text{ZU})\text{YL}$. Also, ZD < ZU, so ZU is the apex z-coordinate of the cone (see Figure 4-5).
	EQ.3: Drop geometry meaning a cone for $-\text{ZD}<z<0$ and half an ellipsoid for $0<z<\text{ZU}$ (see Remark 11 and Figure 4-6)
	EQ.4: Box with side lengths $-\text{XL}<x<\text{XL}$, $-\text{YL}<y<\text{YL}$, and $-\text{ZD}<z<\text{ZU}$ (see Figure 4-7)
	EQ.5: Segment based cylinder (see Remark 12 and Figure 4-8)
NDIV	Number of divisions of an element cut by the surface SURFCT to compute the volume fractions (see Remark 13 and Figure 4-2).
XL	Length of the geometry SURFCT in the local x-direction
YL	Length of the geometry SURFCT in the local y-direction
ZD	Length for the geometry SURFCT in the local z-direction for $z<0$, except for SURFCT = 2 where $z>0$. ZD can be input as a negative or positive value.
ZU	Length for the geometry SURFCT in the local z-direction for $z>0$
XC	x-coordinate in the segment of the local coordinate center (see Remark 14).
YC	y-coordinate in the segment of the local coordinate center (see Remark 14).

Remarks:

1. **Card 1 Definition Only.** If only Card 1 is defined, the velocities are not prescribed. The surface to select the elements to apply the MMGSET is a sphere inscribed in the segment.

2. **Segment Local Coordinate System.** A local coordinate system is created at the center of each segment (see Figure 4-1). The normal is the local z-axis. The local x-axis is parallel to the vector that is the average of 2 vectors: the vector from Node 1 to Node 2 and the vector from Node 4 to Node 3 for a quadrangle. For a triangle, the local x-axis is parallel to the vector from Node 1 to Node 2.

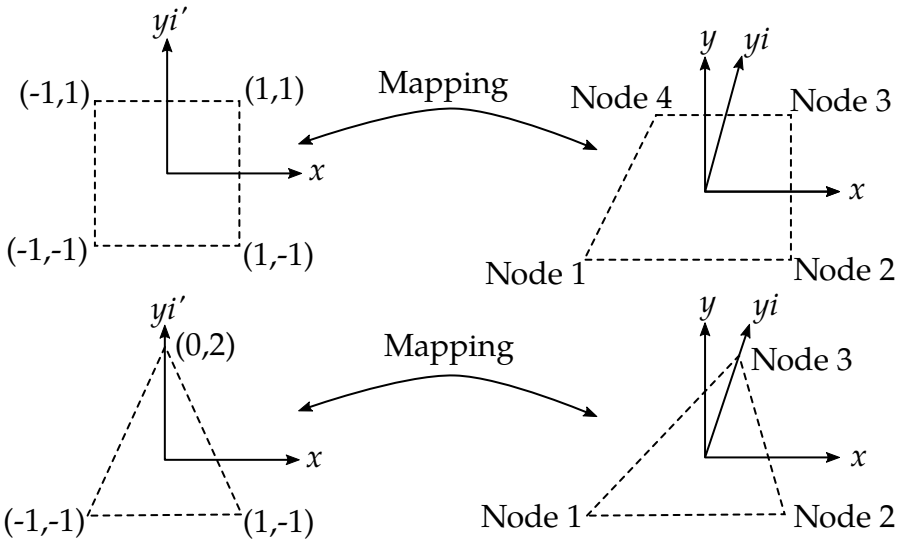


Figure 4-1. Isoparametric mapping if GLOBAL = 1

The local y-axis is the cross product of the local z- and x-axis. A 4th local axis called yi-axis can be considered for the isoparametric mapping (see Remark 3) if the segment is asymmetrical with the y-axis. For a quadrangle, the yi-axis is parallel to the vector that is an average of the 2 other sides: the vector from Node 1 to Node 4 and the vector from Node 2 to Node 3. For a triangle, the local yi-axis is parallel to the vector that points from the midpoint of the edge between Nodes 1 and 2 to Node 3.

3. **Isoparametric Mapping.** If GLOBAL = 1, the geometrical parameters, namely BOXV, XG, YG, ZG, XL, YL, ZD, ZU, XC, and YC, are natural to each segment in SEGSET. Therefore, inside the quadrangle segment, the coordinates XG, YG, ZG, XC, and YC are isoparametric, so they vary from -1 to 1, with 0 at the center. For the triangle, Node 1 is at $(-1,-1)$, Node 2 is at $(1,-1)$, and Node 3 is at $(0,2)$. The matching coordinates in the segment are in the segment-centered coordinate system (x,yi,z) (see Remarks 2 and 14). The coordinates in BOXV and lengths XL, YL, ZD, ZU are multiples of the characteristic lengths in each direction of the segment-centered orthonormal coordinate system (x,y,z) (see Remarks 2 and 14). For quadrangles, the characteristic length in the x-direction is an average of the lengths of the side from Node 1 to Node 2 and the side from Node 3 to Node 4. Similarly, the characteristic length in the y-direction is an average of the of the lengths of the side from Node 1 to Node 4 and the side from Node 2 to Node 3. For triangles, the characteristic length in the x-direction is the length of the side between Nodes 1 and 2; the characteristic length in the y-direction is the length of the median that joins the side length of Nodes 1 and 2 to the vertex of Node 3. For the z-direction, the characteristic length is an average of the characteristic x- and y-lengths.

4. **Load Curve Birth and Death Times.** The smallest and largest times in the curves are the birth and death times. If these time frames are equal to (or slightly

larger than) the time step, the related velocities can be initialized during the first cycle. If the first abscissa is a negative time with no ordinate, the curve is considered periodic and the negative time is the period.

5. **Function Birth and Death Times.** Birth and death times for functions can be considered. The functions are called twice:

   a) the first time with a negative time to determine if the function will be applied. The returned value is positive if the function is to be applied. If the returned value is negative, then the function is not applied.

   b) the second time to evaluate the function if the value returned from the first function call is positive.

6. **Internal Energy and Relative Volume Curves.** The internal energy and relative volume curves are equivalent to the ambient curves in *BOUNDARY_AMBIENT_EOS. So the materials in MMGSET should have *EOS to use these curves. If these curves are not defined, the initial values for the internal energies and relative volumes are applied (like for the ambients).

7. **Prescribed Velocity.** If BOXV is undefined, the velocities are applied to all the nodes inside the surface defined by SURFCT and the nodes of elements cut by this surface. If BOXV is defined, all the nodes inside the box have their velocities being prescribed including nodes outside the surface SURFCT.

8. **Rotation Center.** If the rotation center, namely XG, YG, and ZG, is undefined, the rotation is applied around an axis defined by VECR passing through the center of mass of the groups defined by MMGSET. In an ALE 2D model, this center is shifted to the y-axis.

9. **Ellipsoid Geometry.** The ellipsoid can have different lengths above ($z > 0$) and below ($z < 0$) the segment (see Figure 4-3). If only XL is defined, YL = ZD = ZU = XL, so the surface geometry is the surface of a sphere.

10. **Axisymmetric Cylinder Geometry.** If only XL is defined, YL = XL, so the surface is defined as an axisymmetric cylinder (see Figure 4-4).

11. **Drop Geometry.** For SURFCT = 3, the geometry is an ellipsoid-based cone is below the segment ($z < 0$) and half of an ellipsoid above the segment ($z > 0$). The geometry looks like a drop (or an ice cream cone. See Figure 4-6).

12. **Segment Projection Geometry.** For SURFCT = 5, a node or element center is inside this geometry if $-\text{ZD} < z < \text{ZU}$ and if the node or element center projection on the segment plane is inside this segment (see Figure 4-8).

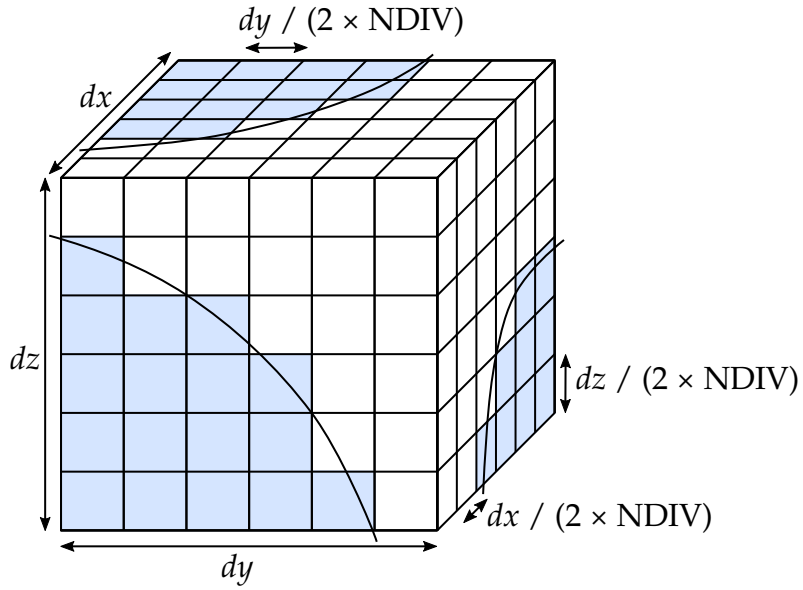


Figure 4-2. Division of an ALE element into $8 \times \text{NDIV}^3$ sub-cells to compute the volume fractions of elements cut by the surface defined by SURFCT

13. **Sub-Volumes.** Like NTRACE in *INITIAL_VOLUME_FRACTION_GEOMETRY, the division of an element into sub-volumes allows the computation of volume fractions of an element cut by the surface SURFCT. The definition of division differs from that for NTRACE in *INITIAL_VOLUME_FRACTION_GEOMETRY. For the initialization of *s*the volume fractions, the number of sub-volumes along an element edge is 2 × NTRACE + 1 for *INITIAL_VOLUME_FRACTION_GEOMETRY while it's 2 × NDIV for this keyword (see Figure 4-2).

14. **Segment Center.** The segment center is the average position of the segment nodes. By default, the local coordinate system is centered at this point. XC and YC can shift this center in the segment plane.

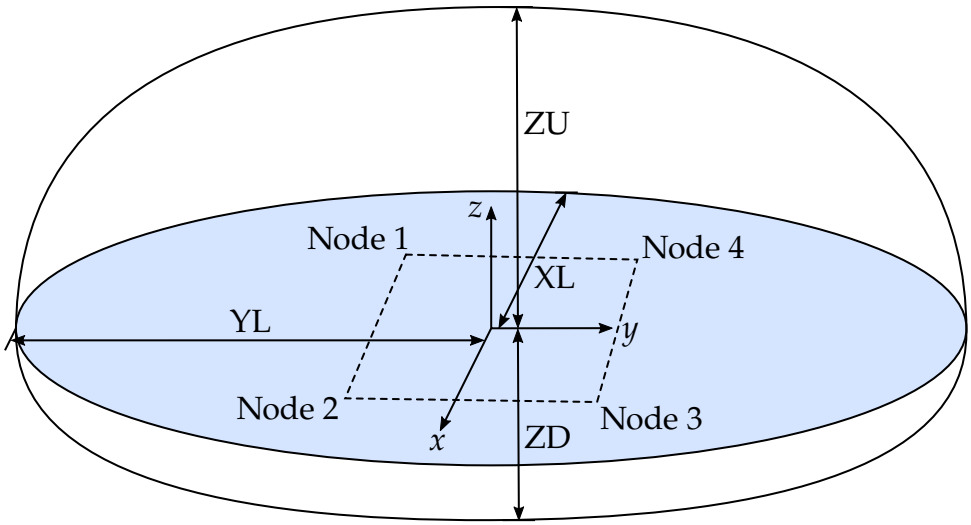


Figure 4-3. Surface for SURFCT = 0

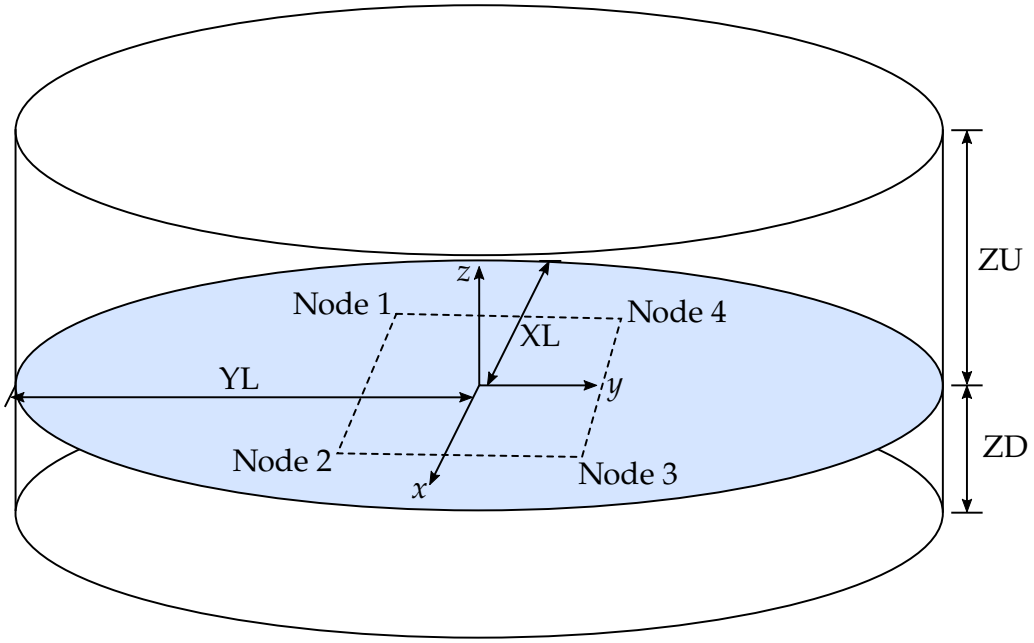


Figure 4-4. Surface for SURFCT = 1

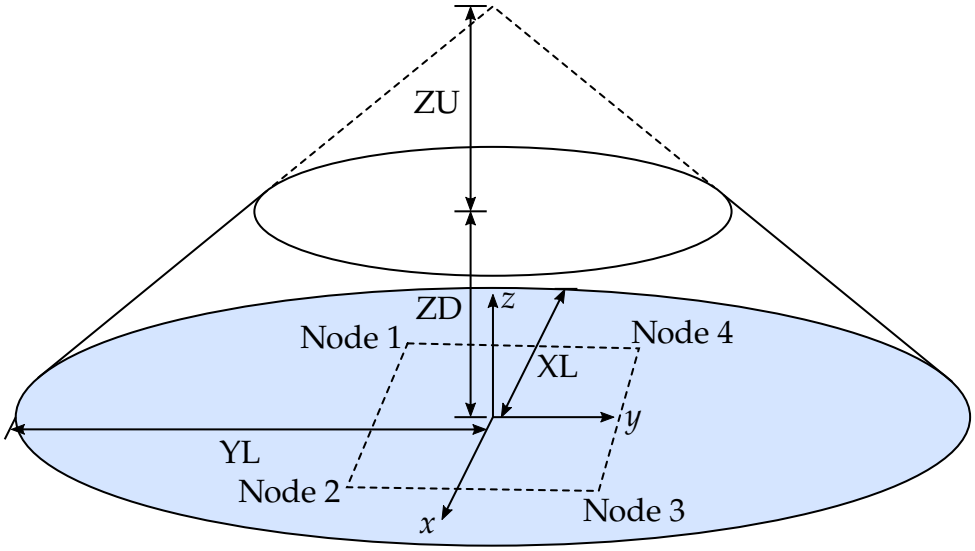


Figure 4-5. Surface for SURFCT = 2

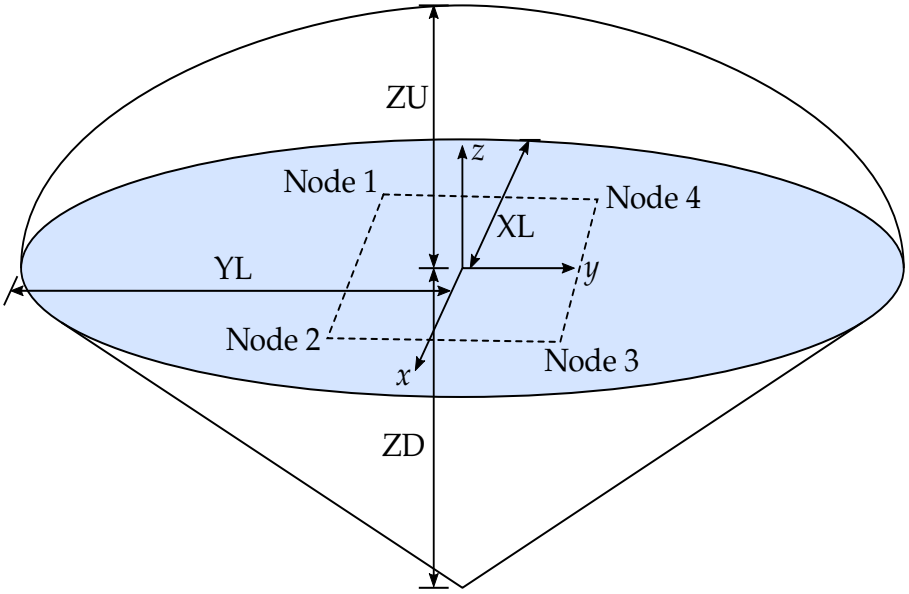


Figure 4-6. Surface for SURFCT = 3

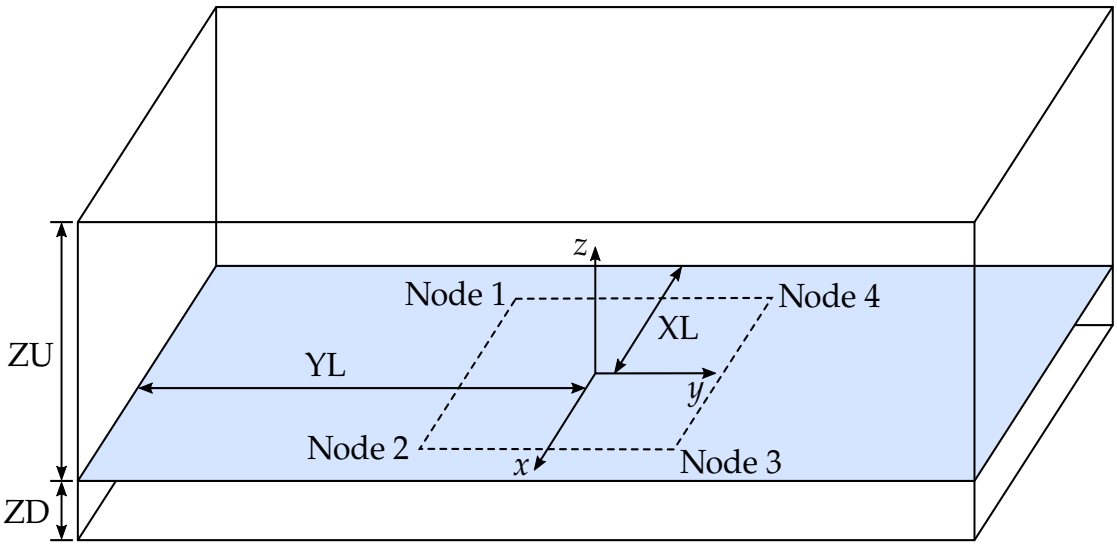


Figure 4-7. Surface for SURFCT = 4

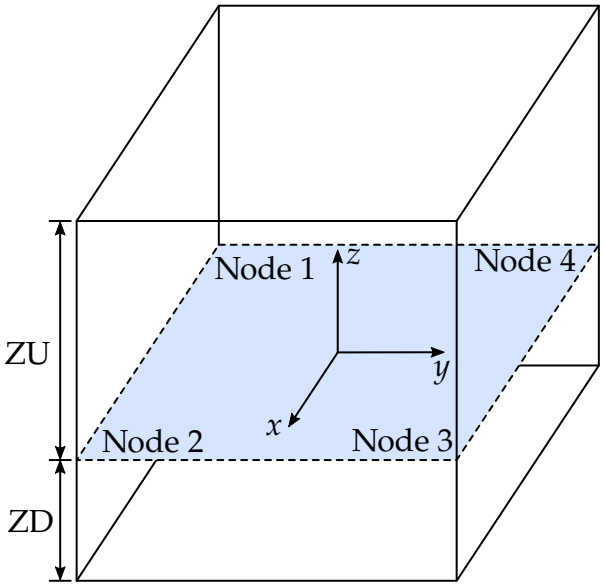


Figure 4-8. Surface for SURFCT = 5

*ALE_MAPPING

Purpose: This card maps ALE data from a previous run to a region of elements at a given time. Data are read from or written to a mapping file specified with "map=" on the command line (see Remark 1). This keyword is supported for S-ALE and ALE. To map data at the initial time to the whole ALE domain (not just a region of elements) see *INITIAL_ALE_MAPPING.

The following transitions are allowed:

1D → 2D	2D → 2D	3D → 3D
1D → 3D	2D → 3D	

Card Summary:

Card 1. This card is required.

AMMSID	RW						

Card 2. This card and all subsequent cards are optional.

NTIM	TBEG	TEND	ND				

Card 3. This card is optional. It defines the location of the mapping for RW > 0. For RW = -1, if the subsequent cards are included, this card must be included as a blank line (all entries will be ignored).

VECID	ANGLE	XP	YP	ZP			

Card 4. This card is optional.

ID	TYPE	NVOL					

Card 4.1. Include NVOL of this card.

VOLTYP	VECID1	DW1	XL	YL	ZL	DW2	DV2

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	AMMSID	RW						
Type	I	I						
Default	none	-1						

VARIABLE	DESCRIPTION
AMMSID	Set ID of ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP. See Remark 2.
RW	Flag defining if the keyword reads or writes in the mapping (see Remarks 3 and 4): EQ.-1: Write to the mapping file. See Remark 4. GT.0: Read from the mapping file. The magnitude of RW defines which part of the output to the mapping file is to be read in if more than one *ALE_MAPPING keywords was used to write the mapping file in the previous run. For example, if there was only one keyword used for writing the mapping file (most of the cases), set RW = 1 during the read.

"When" Card. Optional card to define the times of the mapping (see Remark 4).

Card 2	1	2	3	4	5	6	7	8
Variable	NTIM	TBEG	TEND	ND				
Type	I	F	F	I				
Default	1	↓	ENDTIM	0				

VARIABLE	DESCRIPTION
NTIM	Number of times to write to the mapping file or data entry for the time selected to be read from the mapping file (see Remark 4):

VARIABLE	DESCRIPTION
	RW.EQ.-1: Number of times to write to the mapping file between the times TBEG and TEND. RW.GT.0: Data entry to be read if, during the previous run, a keyword *ALE_MAPPING with RW = -1 wrote several times to the mapping file. In most cases, only one time was output in which case NTIM = 1.
TBEG	RW.EQ.-1: Time to start writing to the mapping (TBEG = ENDTIM by default). See Remark 4. RW.GT.0: Time to which the read data entry indicated by NTIM will be mapped (TBEG = 0.0 by default).
TEND	For RW = -1 only, time to stop writing to the mapping file. See Remark 4.
ND	For RW = -1 only, ID of a node for which ALE data is written in the mapping file as soon as the velocity at the specified node is nonzero

"Where" Card. Optional card to define the location of the mapping for RW > 0 (see Remarks 5 and 6). For RW = -1, if the subsequent cards are included, this card must be included as a blank line (all entries will be ignored).

Card 3	1	2	3	4	5	6	7	8
Variable	VECID	ANGLE	XP	YP	ZP			
Type	I	F	F	F	F			
Default	0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
VECID	ID of the symmetric axis defined using *DEFINE_VECTOR. The fields XT, YT, and ZT in *DEFINE_VECTOR define the location of the origin in the previous run. See Remarks 5 and 6.
ANGLE	Angle of rotation in degrees around an axis defined by *DEFINE_VECTOR for the 3D to 3D mapping. See Remark 6.
XP	x-position of a point on a plane used by specific mappings (only for 2D plane strain to 3D mappings). See Remark 6.

VARIABLE	DESCRIPTION
YP	*y*-position of a point on a plane used by specific mappings (only for 2D plane strain to 3D mappings). See Remark 6.
ZP	*z*-position of a point on a plane used by specific mappings (only for 2D plane strain to 3D mappings). See Remark 6.

Region Card. Optional card to define which elements are targeted by the mapping (see Remark 7).

Card 4	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NVOL					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
ID	Part ID, part set ID, or element set ID. See Remark 7.
TYPE	Type of "ID" (see Remark 7): EQ.0: part set ID EQ.1: part ID EQ.2: element set ID
NVOL	Number of volumes in which the elements are selected for the mapping. See Remark 7.

Volume Card. Include NVOL of this card to define the volumes selecting the elements targeted by the mapping. See Remarks 7, 8, and 9.

Card 4.1	1	2	3	4	5	6	7	8
Variable	VOLTYP	VECID1	DW1	XL	YL	ZL	DW2	DV2
Type	I	I	F	F	F	F	F	F
Default	none	0	0.0	0.0	0.0	0.0	DW1	DV1

VARIABLE	DESCRIPTION
VOLTYP	Type of volume containing the selected elements for the mapping. The magnitude of VOLTYP indicates the type of volume. VOLTYP > 0 flags that the elements *inside* the volume are selected while VOLTYP < 0 indicates that the elements *outside* the volume are selected. The volume depends on geometrical lengths in a local coordinate system defined by orthonormal axis called **u**, **v** and **w** (see Remarks 7, 8, and 9). |VOLTYP|.EQ.1: Trapezoidal prism (see Figure 4-10) |VOLTYP|.EQ.2: Elliptic truncated cone (see Figure 4-11) |VOLTYP|.EQ.3: Ellipsoid (see Figure 4-12)
VECID1	ID of the local **u**-axis defined using *DEFINE_VECTOR. See Remark 8. LE.0: Axis used to define special case volumes. See Remark 9.
DW1	Length in the local **w**-axis direction. See Remarks 8 and 9.
XL	Global *x*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
YL	Global *y*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
ZL	Global *z*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
DW2	Length in the local **w**-axis direction (DW2 = DW1 by default). See Remark 8.

VARIABLE	DESCRIPTION
DV2	Length in the local **v**-axis direction (DV2 = DV1 by default). See Remark 8.

Remarks:

1. **Mapping file.** To make one mapping, only the command-line argument "map=" is necessary. If RW = -1, the mapping file will be created and ALE data histories will be written to this file. If RW > 0, the mapping file will be read and ALE data histories will be used to interpolate the ALE variables of the selected elements.

   To include several successive mappings in the file, the prompt "map1=" is necessary. If RW = -1 and the prompt "map1=" is in the command line, the ALE data are written to the mapping file given by "map1=." However, reading the map file is the same procedure as the single mapping case using "mapping=" on the command line.

   The mapping file contains the following nodal and element data:

   - nodal coordinates
   - nodal velocities
   - part IDs
   - element connectivities
   - element centers
   - densities
   - volume fractions
   - stresses
   - plastic strains
   - internal energies
   - bulk viscosities
   - relative volumes

2. **Mapping of multi-material groups.** The routines initiated by this keyword require knowing which mesh will be initialized with the mapping data and more specifically which multi-material groups will be mapped (if RW > 0) or output (for a next run if RW = -1). The field AMMSID refers to the *SET_MULTI-MATERIAL_GROUP_LIST card. That keyword defines a list of material groups in the current run. The number of entries in this list should match the number of entries from the previous run (as a reminder the data entries of multi-material

groups are defined in *ALE_MULTI-MATERIAL_GROUP). For instance, if the previous model has 3 groups and the current one has 5 groups, the following is a possible mapping:

The 1st group (previous) ⇒ the 3rd group (current),
The 2nd group (previous) ⇒ the 5th group (current) and,
The 3rd group (previous) ⇒ the 4th group (current).

In this case, the *SET_MULTI-MATERIAL_GROUP_LIST card should be as follows:

```
*SET_MULTI-MATERIAL_GROUP_LIST
300
3,5,4
```

3. **Several keywords writing to the mapping file.** Several instantiations of *ALE_MAPPING in a single run with RW = -1 can write to the mapping file. The order with which the instantiations write to the file is important for retrieving the correct data. The first definition of *ALE_MAPPING with RW = -1 in the input deck will have the first position, the second *ALE_MAPPING with RW = -1 will have the second position, and so on. For instance, if 3 *ALE_MAPPING keywords output data to the mapping file during a previous run, then setting RW = 2 in the current run will cause LS-DYNA to read the data written in the previous run from the second instantiation.

4. **Writing several times to the mapping file with a single keyword.** If NTIM > 1, *ALE_MAPPING with RW = -1 can write several times in the mapping file. When the time reaches TBEG, the data for TBEG are written to the mapping file. Then, data are output to the file NTIM-1 at (TBEG − TEND)/(NTIM − 1) time intervals. If NTIM < 2, the data are only output for TBEG. The order of writing is in time order which is important for reading the data later. For instance, if the first run outputs data 4 times (NTIM = 4), then using RW = 1 and NTIM = 3 in a later run causes LS_DYNA to read the third time that data were written by the first run.

5. **Previous origin.** Several *ALE_MAPPING with RW > 0 can map the same set of data to different parts of the current mesh by changing the origin of the previous run. The location of the origin is set by the fields XT, YT, and ZT in the *DEFINE_VECTOR referenced with VECID.

6. **Orientation vector.** For a mapping file created by a previous asymmetric model, the symmetric axis orientation in the current model is specified by VECID. For a mapping file created by a 3D or 1D spherical model, the orientation of the vector VECID is computed but ignored. When 2D plane strain data are mapped in a 3D calculation, the point (XP,YP,ZP) together with the vector, VECID, define the plane in the 3D mesh where the mapping occurs.

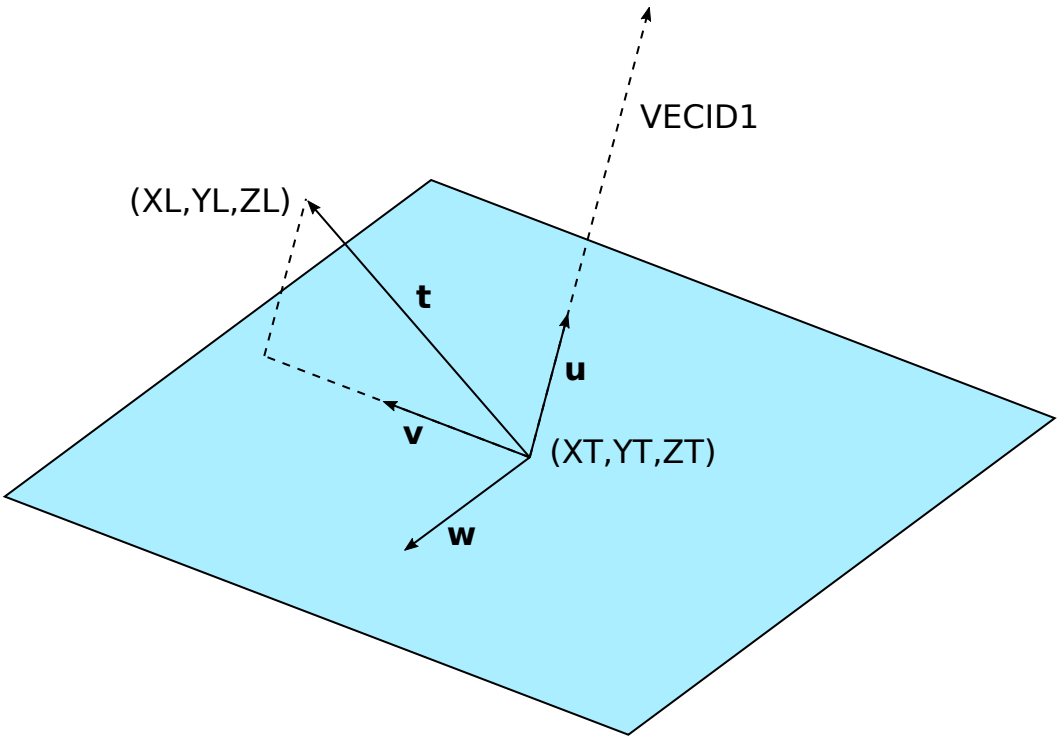


Figure 4-9. Local Coordinate System for the Defined Volumes

7. **Elements for mapping.** It's possible to select which elements should be the target of a mapping (if RW > 0) or have their data output in a mapping file (if RW = -1). The elements can be selected by part or element sets. Along with this first selection, volumes can defined (if NVOL > 0) to adjust the selection. The intersection of these volumes and sets provides the final selection.

   If NVOL > 0, NVOL extra lines at the end of the keyword should be added to define volumes for the element selection. Each line defines a volume. If VOLTYP > 0, the elements inside the intersection of these volumes are selected. If VOLTYP < 0, the elements outside the intersection of these volumes are selected.

8. **Defining volumes.** Defining the three different kinds of volumes from the input fields requires a set of very similar steps. Defining VOLTYP = 1, the trapezoidal prism, will be discussed in detail to illustrate these steps. Then the differences in how the fields are used for the other two volumes are highlighted following this discussion.

   The first step required to construct the trapezoidal prism from the input fields is to define a local coordinate (**u**, **v**, **w**).

   a) *Defining the Local Coordinate System.* The local coordinate system is defined using a vector and a point, namely VECID1 and (XL,YL,ZL). The origin of

the coordinate system is the tail of VECID1 which is the point (XT,YT,ZT) set in the *DEFINE_VECTOR input. See Figure 4-9.

i) **u**-*axis.* The vector **u** is the unit direction of VECID1.

ii) **v**-*axis.* From here the other two vectors can be constructed on the plane perpendicular to **u** using the point (XL,YL,ZL). Note that for this construction to work, (XL,YL,ZL) *cannot* be along the **u** vector. Let **t** = (XL,YL,ZL) − (XT,YT,ZT). Then, **v** is the unit vector along the projection of **t** onto the plane perpendicular to **u**.

iii) **w**-*axis.* **w** is then the unit vector that completes the right-hand coordinate system.

The trapezoidal prism is constructed by connecting two faces defined with the local coordinate system. The two faces are both perpendicular to **u**. See Figure 4-10.

b) *First Face of the Trapezoidal Prism.* The first face is a rectangle on a plane that includes the point (XT,YT,ZT) and is perpendicular to **u**. The rectangle is centered at the point (XT,YT,ZT), and it has its sides oriented such that they align with the **v** and **w** vectors. The length of the side oriented along the **v**-axis is 2 × DV1. DV1 is the length of the projection of **t** onto the plane. The length of the side oriented along the **w**-axis is 2 × DW1, where DW1 is an input field.

c) *Second Face of the Trapezoidal Prism*. The second face of the trapezoidal is defined very similarly to the first. It is also a rectangle on a plane perpendicular to **u**, but the rectangle is centered on the point (XH,YH,ZH). This point is set in *DEFINE_VECTOR for VECID1. The sides are oriented the same way as in the first face definition. In this case the length of the side oriented along the **v**-axis is 2 × DV2 which is an input field. The length of the side oriented along the **w**-axis is 2 × DW2, where DW1 is an input field.

VOLTYP = 2, the elliptic truncated cone, is basically the same as the trapezoidal prism, except that it has two elliptic faces instead of two rectangular faces centered at the same locations as the rectangular faces. The ellipse semi-axes are aligned along **v** and **w** with lengths of DV1 and DW1 on the first face and lengths of DV2 and DW2 on the second face. DV1 is defined the same way as for the trapezoidal prism. See Figure 4-11.

VOLTYP = 3, the ellipsoid, is defined using an ellipse that is the same as the first face of the VOLTYP = 2. That ellipse is one of the symmetry planes of the ellipsoid. The ellipsoid is centered at the point (XT,YT,ZT). It has semi-axes that are the length of VECID1, DV1, and DW1 along **u**, **v**, and **w**, respectively. Both DV2 and DW2 are not used in this case. See Figure 4-12.

9. **Shortcuts for defining common types of volumes.** The keyword reader allows for certain shortcuts to define cylinders, ellipsoids with circular cross-sections, and spheres. All of these volumes can be constructed through the process described in Remark 8.

   a) *Shortcut for Cylinder.* With VOLTYP = 2, a cylinder can be defined more simply with a vector and radius. To do this, let VECID1 be the axis of the cylinder, but to indicate to LS-DYNA that a shortcut is being used input the ID as a negative number. Set DW1 as the radius of the cylinder. No other input is required. The length of the cylinder is the length of VECID1. Its faces will be centered on the (XT,YT,ZT) and (XH,YH,ZH), just like an arbitrary truncated cone with ellipsoid faces.

   b) *Shortcut for Ellipsoid with Circular Cross-Section.* To obtain an ellipsoid with a circular cross-section, set VOLTYP = 3 and pick a radius and the axis perpendicular to the cross-section. Set VECID1 to a the negative of the *DEFINE_VECTOR ID for the axis. Set the field DW1 as the radius. This input will create an ellipsoid with a circular cross-section centered at the point (XT,YT,ZT). Like the normal ellipsoid definition, the remaining semi-axis will have the same length as VECID1.

   c) *Shortcut for Sphere.* To obtain a sphere, set VOLTYP = 3 and VECID = 0. The only other required input are a radius which will be set in the field DW1 and a center point which will be set using (XL,YL,ZL).

10. **Comparison to *BOUNDARY_ALE_MAPPING.** Both *ALE_MAPPING and *BOUNDARY_ALE_MAPPING, can write multiple sets of data to a file at a given frequency between certain times. The big difference between these two keywords is reading the data. With *ALE_MAPPING, you select only one set of data to map from the previous run to the current run which is mapped at a time you select. With *BOUNDARY_ALE_MAPPING, all data output between t = BIRTH and t = DEATH from the previous run is mapped to the elements in the current run. The discrete histories from the previous run are interpolated.

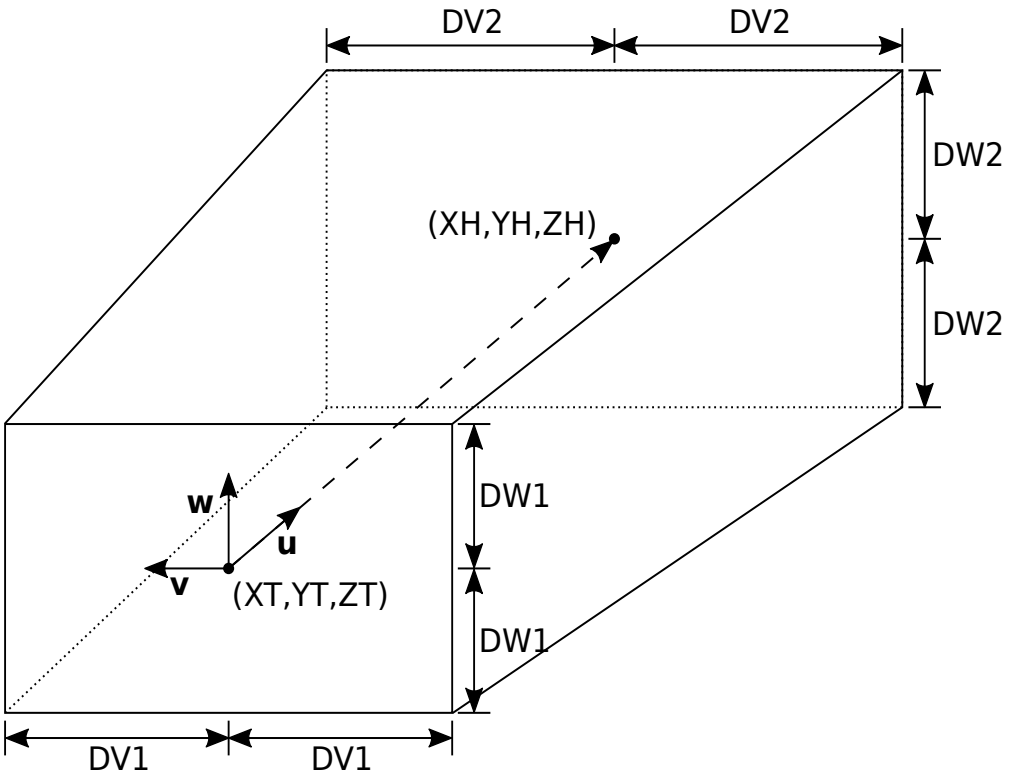


Figure 4-10. Trapezoidal Prism (VOLTYP = 1)

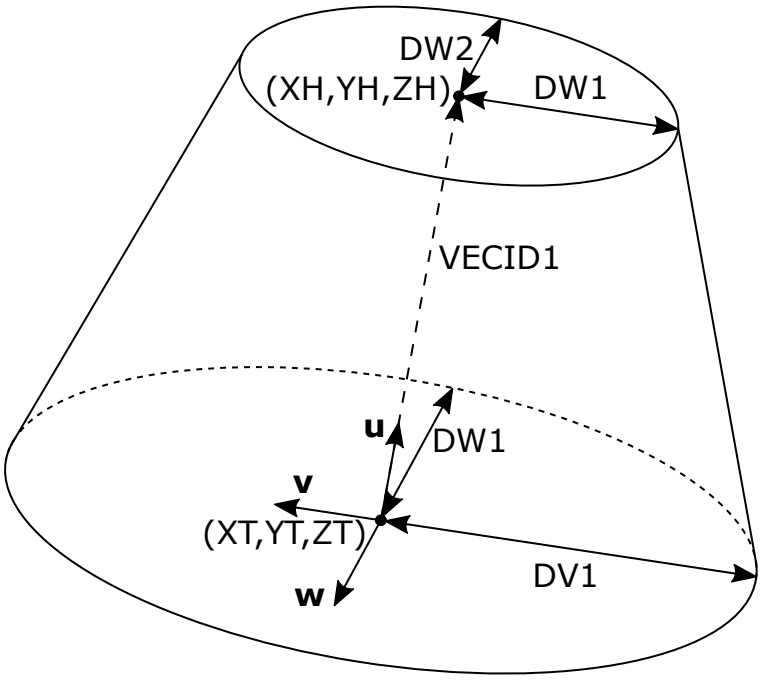


Figure 4-11. Elliptic Truncated Cone (VOLTYP = 2)

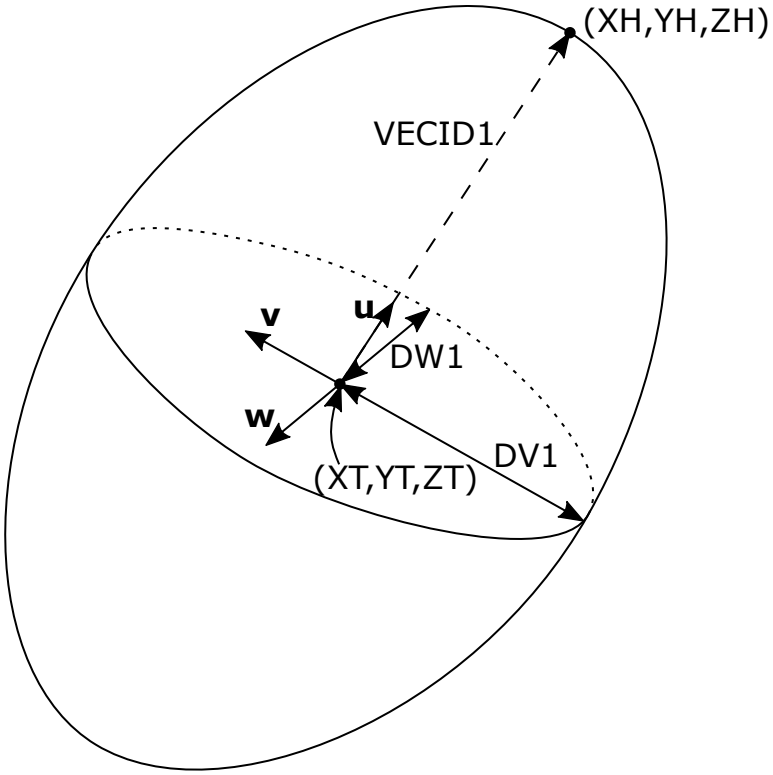


Figure 4-12. Ellipsoid (VOLTYP = 3)

*ALE_MAPPING_FROM_LAGRANGIAN

Purpose: Map data from a Lagrangian run (that is, non-ALE solid model) to ALE solids. This keyword supports both 2D and 3D meshes. It maps data from 2D Lagrangian meshes to 2D ALE meshes, and from 3D Lagrangian meshes to 3D ALE meshes. Data are written to a mapping file called lag2ale_map (see Remark 1) that can be read back to initialize a 2D or 3D ALE mesh (see *INITIAL_ALE_MAPPING). To generate the mapping file, include this keyword in the input deck for the Lagrangian simulation. The produced mapping file is supported for both ALE and S-ALE.

Card Summary:

Card 1. This card is required to select which Lagrangian parts are mapped.

LAGPID	LAGPTYP						

Card 2. This card is required to generate the ALE mesh to which Lagrangian data are mapped.

NX	NY	NZ	NPX	NPY	NPZ	ALEPID	

Card 3. This card is optional. It controls how the volumes shared by the Lagrangian and ALE elements are computed.

METHOD	DIV						

Data Card Definitions:

Lagrangian Card. Card to select which Lagrangian parts are mapped.

Card 1	1	2	3	4	5	6	7	8
Variable	LAGPID	LAGPTYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
LAGPID	Part or part set ID for Lagrangian parts involved in the mapping
LAGPTYP	Type of LAGPID:

VARIABLE	DESCRIPTION
	EQ.0: ID is a part set ID (see *SET_PART)
	EQ.1: ID is a part ID (see *PART)

ALE Mesh Card. Card to create the ALE mesh to which Lagrangian data are mapped.

Card 2	1	2	3	4	5	6	7	8
Variable	NX	NY	NZ	NPX	NPY	NPZ	ALEPID	
Type	I	I	I	I	I	I	I	
Default	none	none	none	0	0	0	none	

VARIABLE	DESCRIPTION
NX, NY, NZ	Number of ALE elements in each direction of the global coordinate system. These parameters create a structured box mesh. See Remark 2. For 2D, NZ is ignored.
NPX, NPY, NPZ	Number of extra elements to pad the box mesh beyond its lower and upper limits in each direction of the global coordinate system. See Remark 2. For 2D, NPZ is ignored.
ALEPID	Part ID of the ALE mesh

Intersection Computation Card. Optional card to control how the volumes shared by the Lagrangian and ALE elements are computed (see Remark 3).

Card 3	1	2	3	4	5	6	7	8
Variable	METHOD	DIV						
Type	I	I						
Default	0	10						

VARIABLE	DESCRIPTION
METHOD	Method to compute volumes (areas in 2D) at the intersection of

VARIABLE	DESCRIPTION
	Lagrangian and ALE elements: EQ.0: Both METHOD = 1 and METHOD = 2 are applied by default. See Remark 3. EQ.1: The intersection volumes are exactly computed. EQ.2: The intersection volumes are evaluated with DIV. See Remark 3.
DIV	Division of ALE element edges to create subcells, in which volumes (areas in 2D) inside Lagrangian elements are added up by METHOD = 2 to approximate the intersection volumes at the intersection between ALE and Lagrangian elements. See Remark 3.

Remarks:

1. **Output files.** When the Lagrangian run normally terminates with this keyword in the input deck, the following files are generated:

File Name	Description
lag2ale_lagmesh.k	Lagrangian mesh selected by this keyword
lag2ale_alemesh.k	ALE mesh generated by this keyword
lag2ale_volfrac.k	Fractions of the ALE element volumes (areas in 2D) occupied by Lagrangian elements. These volume fractions are listed in this file using *INITIAL_VOLUME_FRACTION.
lag2ale_mass	Lagrangian element masses distributed across the ALE mesh
lag2ale_stress.k	Lagrangian element stresses distributed across the ALE mesh. These stresses are listed in this file with *INITIAL_STRESS_SOLID.
lag2ale_velo.k	Lagrangian velocities interpolated at ALE nodes. These velocities are listed in this file with *INITIAL_VELOCITY_NODE.
lag2ale_map	Mapping file that can be read back with the command-line argument "map = " and *INITIAL_ALE_MAPPING to initialize the run of an ALE model with a mesh similar to the one in lag2ale_alemesh.k (that is, you can opt for a different ALE mesh, but its size, position and element size should be close to lag2ale_alemesh.k for the mapping to work properly)

2. **ALE mesh.** The ALE mesh, to which Lagrangian data are mapped, is made of identical parallelepiped hexahedral (quadrilateral in 2D) solids. The nodal

connectivities and nodal coordinates are listed under *ELEMENT_SOLID (*ELEMENT_SHELL for 2D) and *NODE, respectively, in a file called lag2ale_alemesh.k. The structured mesh has 2 zones:

a) A core mesh is a box of NX × NY × NZ elements. The box exactly embeds the Lagrangian parts selected by LAGPID. In other words, the dimensions of all these Lagrangian parts in each direction give the box size.

b) The core mesh can be padded with extra layers of elements. NPX, NPY, and NPZ are the number of extra elements in each global direction.

3. **Mesh intersection.** If a Lagrangian element is superimposed with an ALE element from the mesh created by this keyword, the volume that they share can be evaluated by 2 methods:

a) *METHOD = 1.* The computation of the intersections between edges and faces yields a volume that can be calculated in most cases.

b) *METHOD = 2.* The intersection volume is approximated by summing the volumes of smaller DIV × DIV × DIV subcells that:

i) divide the ALE element volumes and

ii) are inside the Lagrangian element

For the default (METHOD = 0), both methods are applied. In this case, if METHOD = 1 fails, METHOD = 2 will provide the intersection volume.

*ALE_MESH_INTERFACE

Purpose: Mesh with triangular shells the material interfaces (isosurfaces of the volume fraction 0.5) of ALE groups selected by a multi-material group set. The keyword also meshes the volume of these materials with tetrahedra. The meshes are output in keyword format when the run reaches the termination time or when the user applies a sense switch (Ctrl-C, then enter a SW code when prompted; see Sense Switch Controls in Getting Started). This keyword is supported for ALE and S-ALE.

Card 1	1	2	3	4	5	6	7	8
Variable	MMGSET	NOWRT	VOLRAT	INTERP				
Type	I	I	F	I				
Default	none	0	0.0	0				

Remeshing Card. Optional card controlling remeshing of the interface (see Remark 2).

Card 2	1	2	3	4	5	6	7	8
Variable	EDGMIN	EDGMAX						
Type	F	F						
Default	0.0	0.0						

VARIABLE	DESCRIPTION
MMGSET	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_LIST). The materials (or ALE groups) in this set are selected to be meshed.
NOWRT	File output flag. NOWRT is interpreted digit-wise, NOWRT = [*PNML*]: $$\text{NOWRT} = L + M \times 10 + N \times 100 + P \times 1000$$ The 1s digit controls the output of the initial triangular mesh on the material interfaces:
	L.EQ.0: Write the initial triangular meshes for the material interfaces (see Remark 1). The mesh is output to a

VARIABLE	DESCRIPTION
	keyword file called alemeshmatint.k. L.EQ.1: Do not output alemeshmatint.k. The 10s digit controls the output of the remesh for the triangular mesh on the material interface: M.EQ.0: Write a triangular mesh on the material interfaces after remeshing (see Remark 2) to a keyword file called aleremeshmatint.k. M.EQ.1: Do not output aleremeshmatint.k. The 100s digit controls the output of the mesh in the material volumes: N.EQ.0: Write a tetrahedral mesh of the material volumes to a keyword file called alemeshmatvol.k. N.EQ.1: Do not output alemeshmatvol.k. N.EQ.2: Write a hexahedral mesh in 3D or a quadrilateral mesh in 2D for the materials selected by MMGSET in alemeshmatvol.k. The 1000s digit controls the output of a mapping file that can be read back with *INITIAL_LAG_MAPPING to initialize the tetrahedral mesh in alemeshmatvol.k: P.EQ.0: Write the Lagrangian mapping file called alemeshmap. P.EQ.1: Do not output alemeshmap.
VOLRAT	Volume ratio: LT.2.0: Mesh volume ratio beyond which the mesh is output (see Remark 3) EQ.2.0: Volume fraction threshold in an ALE hexahedra beyond which this element is output in aleremeshmatvol.k.
INTERP	Interpolation method: EQ.0: The ALE hexahedron data are interpolated at the Lagrangian tetrahedron centers. EQ.1: The intersection volumes between ALE hexahedra and Lagrangian tetrahedra are computed, and the ALE data are mapped to the Lagrangian elements with a volume-averaged method.
EDGMIN	Minimum triangle edge applied during remeshing (see Remark 2)

VARIABLE	DESCRIPTION
EDGMAX	Maximum triangle edge applied during remeshing (see Remark 2)

Remarks:

1. **Initial material interface mesh.** The isosurfaces of the volume fraction 0.5 (material interfaces) for the selected ALE groups (materials) are first meshed with triangles, whose areas depend on the ALE element face sizes and how the material interface cuts these elements. For instance, if a material interface in an ALE element is parallel to one of its faces, the area of the 2 triangles meshing the interface will be half the face area. If the material interface only cuts one corner of an ALE element, the resulting triangular surface will be one shell. In this case, if the material volume is small, the triangle meshing the material interface will be small as well. Thus, the interface mesh can have triangles of very different sizes. If the material volume needs to be meshed, a remeshing of its surface is required. If a material is in contact with the ALE mesh boundary, this contact surface will be meshed to ensure a closed volume.

2. **Remeshing of the material interface.** The material interface is remeshed based on the coordinates of the first try to mesh it. The remeshing code is the same as the one controlled by *CONTROL_REMESHING. Similar to RMIN and RMAX for that keyword, each triangle edge of the new mesh should be between a minimum EDGMIN and maximum EDGMAX. If these parameters are not provided, EDGMIN and EDGMAX are the minimum and maximum edge size of the initial mesh, respectively.

3. **Mesh volume ratio threshold.** A material can be divided into several volumes. Some of them could be so small that they would be useless or difficult to mesh. If one of these fragments has a ratio of its volume over the total material volume smaller than VOLRAT, the fragment will not be meshed.

4. **Part IDs in alemeshmatint.k.** If a material is fragmented into N volumes, each of these fragments will have a specific part ID in alemeshmatint.k. The ID for fragment i with $1 \leq i \leq N$ of the j^{th} ALE group is $\times 1000 + j$.

5. **File names.** If there are several *ALE_MESH_INTERFACE, files are output for each of these keywords. The base names for a given instance of this keyword are appended with a number that refers to the order in which that instance is included, meaning the output for the second instance, for example, would have a 2 appended.

*ALE_MULTI-MATERIAL_GROUP

Purpose: This command defines the appropriate ALE material groupings for interface reconstruction when two or more ALE Multi-Material Groups (AMMG) are present in a model. This card is required when ELFORM = 11 in the *SECTION_SOLID card or when ALEFORM = 11 in *SECTION_ALE1D or *SECTION_ALE2D. Each data line represents one ALE multi-material group (AMMG), with the first line referring to AMMGID 1, second line AMMGID 2, etc. Each AMMG represents one unique "fluid" which may undergo interaction with any Lagrangian structure in the model.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	IDTYPE						
Type	I	I						
Default	none	0						
Remarks	1							

VARIABLE	DESCRIPTION
SID	Set ID
IDTYPE	Set type: EQ.0: Part set, EQ.1: Part.

Remarks:

1. **ELFORM.** When ELFORM = 12 in the *SECTION_SOLID card (single material and void), this card should not be used. In one model, ELFORM = 12 cannot be used together with ELFORM = 11. If possible, it is recommended that ELFORM = 11 be used as it is the most robust and versatile formulation for treating multi-material ALE parts.

2. **AMMGID Assignment.** Each AMMG is automatically assigned an ID (AMMGID) and consists of one or more PART IDs. The interface of each AMMGID is reconstructed as it evolves dynamically. Each AMMGID is represented by one material contour color in LS-PrePost.

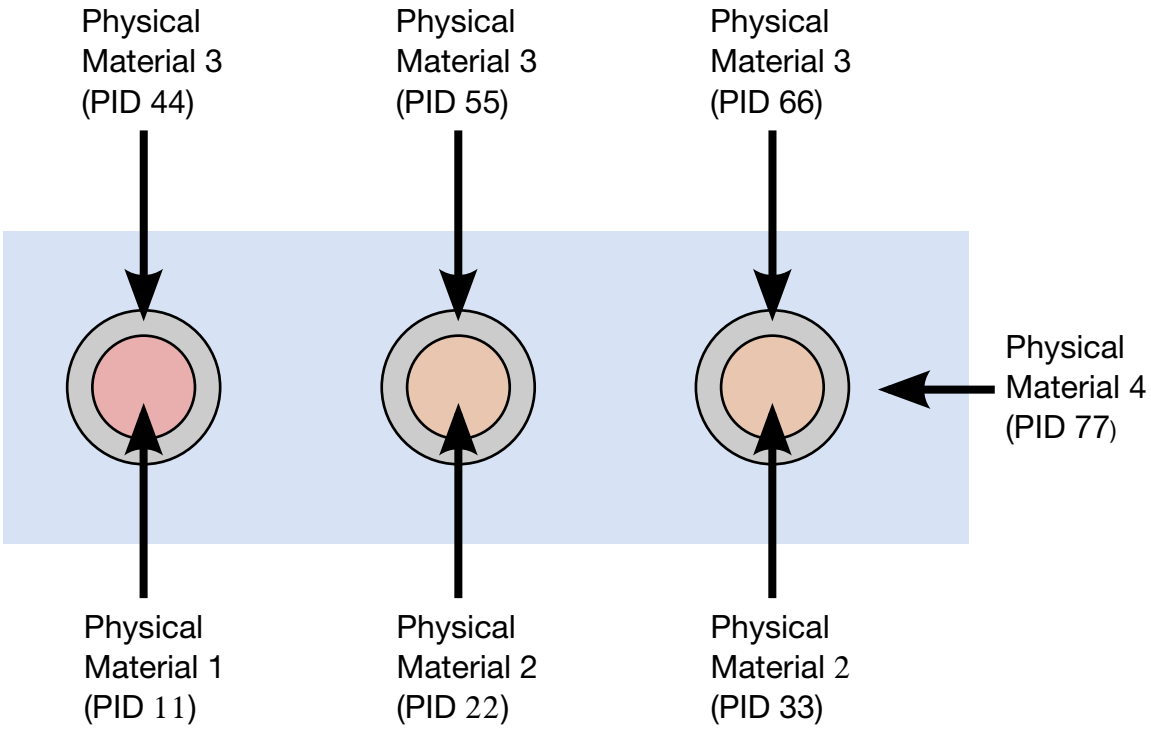


Figure 4-13. Schematic illustration of Example 1.

3. **ALE Elements and AMMGs.** The maximum number of AMMGIDs allowed has been increased to 20. However, there may be 2, at most 3, AMMGs inside an ALE element at any time. If there are more than 3 AMMGs inside any 1 ALE element, the ALE mesh needs refinement. Better accuracy is obtained with 2 AMMGs in mixed elements.

4. **Plotting AMMGIDs.** To plot these AMMGIDs in LS-PrePost:

   [FCOMP] ⇒ [MISC] ⇒ [VOLUME FRACTION OF AMMGID #] ⇒ [APPLY]

   (Note: Contour definitions maybe different for gas mixture application)

5. **Model Description.** It is very important to distinguish among the

   a) Physical materials,

   b) PART IDs, and

   c) AMMGIDs.

   A *PART may be any mesh component. In ALE formulation, it is simply a geometric entity and a time = 0.0 concept. This means a *PART may be a mesh region that can be filled with one or more AMMGIDs at time zero, via a volume filling command (*INITIAL_VOLUME_FRACTION_GEOMETRY). An AM-

MGID represents a physical material group which is treated as one material entity (represented by 1 material color contour in LS-PrePost plotting). AMMGID is used in dealing with multiple ALE or Eulerian materials. For example, it can be used to specify an ALE group in a coupling card.

Example 1:

Consider a purely Eulerian model containing 3 containers containing 2 different physical materials (fluids 1 and 2). All surrounded by the background material (maybe air). The containers are made of the same material, say, metal. Assume that these containers explode and spill the fluids. We want to track the flow and possibly mixing of the various materials. Note that all 7 parts have ELFORM = 11 in their *SECTION_SOLID cards. So we have total of 7 PIDs, but only 4 different physical materials. See Figure 4-13.

Approach 1: If we want to track only the interfaces of the **physical** materials.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*SET_PART
         1
        11
*SET_PART
         2
        22        33
*SET_PART
         3
        44        55        66
*SET_PART
         4
        77
*ALE_MULTI-MATERIAL_GROUP
         1         0   ← 1st line = 1st AMMG ⇒ AMMGID = 1
         2         0   ← 2nd line = 2nd AMMG ⇒ AMMGID = 2
         3         0   ← 3rd line = 3rd AMMG ⇒ AMMGID = 3
         4         0   ← 4th line = 4th AMMG ⇒ AMMGID = 4
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

With this approach, we define only 4 AMMGs (NALEGP = 4). So in LS-PrePost, when plotting the material-group (history variable) contours, we will see 4 colors, one for each material group. One implication is that when the fluids from part 22 and part 33 flow into the same element, they will coalesce and no boundary distinction between them is maintained subsequently. While this may be acceptable for fluids at similar thermodynamic states, this may not be intuitive for solids. For example, if the solid container materials from parts 44, 55 and 66 flow into one element, they will coalesce "like a single fluid", and no interfaces among them are tracked. If this is undesirable, an alternate approach may be taken. It is presented next.

Approach 2: If we want to reconstruct as many interfaces as necessary, in this case, we follow the interface of each part.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
         1         1   ←  1st line  =  1st AMMG  ⇒ AMMGID = 1
         2         1   ←  2nd line  =  2nd AMMG  ⇒ AMMGID = 2
         3         1   ←  3rd line  =  3rd AMMG  ⇒ AMMGID = 3
         4         1   ←  4th line  =  4th AMMG  ⇒ AMMGID = 4
         5         1   ←  4th line  =  5th AMMG  ⇒ AMMGID = 5
         6         1   ←  4th line  =  6th AMMG  ⇒ AMMGID = 6
         7         1   ←  4th line  =  7th AMMG  ⇒ AMMGID = 7
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

There are 7 AMMGs in this case (NALEGP = 7). This will involve more computational cost for the additional tracking. Realistically, accuracy will be significantly reduced if there are more than 3 or 4 materials in any one element. In that case, higher mesh resolution may be required.

Example 2:

Oil	Water	Air
Group 1	Group 2	Group 3
Part IDs 1 and 2	Part ID 3	Part IDs 5, 6, and 7

The above example defines a mixture of three groups of materials (or "fluids"), oil, water and air, that is, the number of ALE multi-material groups (AMMGs) NALEGP = 3.

The first group contains two parts (materials), part IDs 1 and 2.

The second group contains one part (material), part ID 3.

The third group contains three parts (materials), part IDs 5, 6 and 7.

*ALE_PRESCRIBED_MOTION

Purpose: Define an imposed nodal motion on a set of multi-material ALE groups for a given time span (if this time frame is short enough, velocity initialization by group can be considered). This keyword is supported for both ALE and S-ALE.

Card 1	1	2	3	4	5	6	7	8
Variable	MMSID	INSIDE	SIDR					
Type	I	I	I					
Default	none	0	0					

Remaining cards are optional.†

Translational Velocity Card. Optional card for the translational velocity (see Remark 1).

Card 2	1	2	3	4	5	6	7	8
Variable	LCVTX	LCVTY	LCVTZ					
Type	I	I	I					
Default	Rem 1	Rem 1	Rem 1					

Rotational Velocity Card. Optional card for the rotational velocity (see Remark 1).

Card 3	1	2	3	4	5	6	7	8
Variable	LCVRX	LCVRY	LCVRZ					
Type	I	I	I					
Default	Rem 1	Rem 1	Rem 1					

Center of Rotation Card. Optional card for defining a center of rotation.

Card 4	1	2	3	4	5	6	7	8
Variable	XG	YG	ZG					
Type	F	F	F					
Default	Rem 4	Rem 4	Rem 4					

VARIABLE	DESCRIPTION
MMSID	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_LIST).
INSIDE	Flag to define which nodes the motion is prescribed for (see Remark 2): EQ.0: Nodes connected to at least one ALE element that is at minimum partially filled by a group of MMSID EQ.1: Nodes connected to at least one ALE element that is fully filled by a group of MMSID EQ.2: Nodes only connected to ALE elements that are fully filled by a group of MMSID
SIDR	Flag controlling the use of this keyword during dynamic relaxation: EQ.0: Applies during the normal analysis phase only EQ.1: Applies during the dynamic relaxation phase but not the normal analysis phase EQ.2: Applies during both the dynamic relaxation and the normal analysis phases
LCVTX LCVTY LCVTZ	Curve IDs for the translation in each global direction; see *DEFINE_CURVE. See Remark 3.
LCVRX LCVRY LCVRZ	Curve IDs for the rotation around each global direction; see *DEFINE_CURVE. See Remark 3.
XG, YG, ZG	Position of the rotation center. See Remark 4.

Remarks:

1. **Zero velocity.** If only the 1st card is defined, the velocities of the nodes inside ALE groups defined by MMSID are set to zero.

2. **INSIDE.** Depending on the flag INSIDE, the motion is applied to nodes connected to elements partially or fully filled by ALE groups of the set MMSID. For INSIDE = 0, when an element is partially filled by one of these groups, the motion of all its nodes will be prescribed. INSIDE = 1 is more restrictive and only selects nodes connected to at least one element fully filled by a group in MMSID. INSIDE = 2 is even more selective: nodes connected to only fully filled elements are considered. So INSIDE = 1 can apply velocities to nodes at the boundary between fully filled and "empty" elements, while INSIDE = 2 only selects nodes inside the ALE group surrounded by fully filled elements.

3. **Birth and death times.** The smallest and largest times in the curves are the birth and death times of the related motions. If these time frames are equal to (or slightly larger than) the time step, the related velocities can be initialized during the first cycle.

4. **Rotation center.** If the rotation center (XG, YG, ZG) is undefined (no 4th card), the rotation is applied around an axis (defined by LCVRX, LCVRY, LCVRZ) passing through the center of mass of the groups defined by MMSID. In an ALE 2D model, this center is shifted to the Y-axis. If the card is included, but has blank fields, then the blank field values default 0.0.

*ALE_REFERENCE_SYSTEM_CURVE

Purpose: This command defines a motion and/or a deformation prescribed for a geometric entity, where a geometric entity may be any part, part set, node set, or segment set. The motion or deformation is completely defined by the 12 parameters shown in the equation below. These 12 parameters are defined in terms of 12 load curves. This command is required only when PRTYPE = 3 in the *ALE_REFERENCE_SYSTEM_GROUP command.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	LCID1	LCID2	LCID3	LCID4	LCID5	LCID6	LCID7	LCID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Card 3	1	2	3	4	5	6	7	8
Variable	LCID9	LCID10	LCID11	LCID12				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
ID	Curve group ID.
LCID1, …, LCID12	Load curve IDs.

Remarks:

1. The velocity of a node at coordinate (x, y, z) is defined as:

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_5 \\ f_9 \end{Bmatrix} + \begin{bmatrix} f_2 & f_3 & f_4 \\ f_6 & f_7 & f_8 \\ f_{10} & f_{11} & f_{12} \end{bmatrix} \begin{Bmatrix} x - \mathrm{XC} \\ y - \mathrm{YC} \\ z - \mathrm{ZC} \end{Bmatrix}$$

where $f_1(t)$ is the value of load curve LCID1 at time t, $f_2(t)$ is the value of load curve LCID2 at time t and so on. The functions $f_1(t), f_5(t)$, and $f_9(t)$ respectively correspond to the translation components in global x, y, and z direction, while the functions $f_2(t), f_7(t)$, and $f_{12}(t)$ correspond to and expansion or contraction along the x, y, and z axes.

The parameters XC, YC and ZC from the second data card of *ALE_REFERENCE_SYSTEM_GROUP define the center of rotation and expansion of the mesh. If the mesh translates, the center position is updated with $f_1(t), f_5(t)$, and $f_9(t)$.

If LCID8, LCID10, LCID3 are equal to -1, their corresponding values $f_8(t), f_{10}(t)$, and $f_3(t)$ will be equal to $-f_{11}(t)$, $-f_4(t)$, and $-f_6(t)$ so as to make a skew symmetric matrix thereby inducing a rigid rotation of the mesh about the axis **v** defined by the triple,

$$\mathbf{v} = \left(f_{11}(t), f_4(t), f_6(t)\right)$$

Example:

Consider a motion that consists of translation in the x and y direction only. Thus only $f_1(t)$ and $f_5(t)$ are required. Hence only 2 load curve ID's need be defined:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
$       SID     STYPE      PRTYP       PRID     BCTRAN      BCEXP      BCROT     ICOORD
          1         0          3         11          0          7          0
$        XC        YC         ZC     EXPLIM
          0         0          0          0
*ALE_REFERENCE_SYSTEM_CURVE
$ CURVESID
         11
$     LCID1      LCID2      LCID3      LCID4      LCID5      LCID6      LCID7     LCID8
        111          0          0          0        222          0          0         0
$     LCID9     LCID10     LCID11     LCID12
          0          0          0          0
*DEFINE_CURVE
$      lcid       sidr        sfa        sfo       offa       offo     dattyp
        111
$                   a1                    o1
                  0.00                   5.0
                  0.15                   4.0
*DEFINE_CURVE
$      lcid       sidr        sfa        sfo       offa       offo     dattyp
        222
$                   a1                    o1
                  0.00                  -1.0
```

```
                0.15                -5.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_REFERENCE_SYSTEM_GROUP

Purpose: Associate a geometric entity to a reference system type. A geometric entity may be any part, part set, node set, or segment set of a model (or a collection of meshes). A reference system type refers to the possible transformation allowed for a geometric entity (or mesh). This command defines the type of reference system or transformation that a geometric entity undergoes. In other words, it prescribes how a specified mesh can translate, rotate, expand, contract, be fixed in space, etc.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	PRTYPE	PRID	BCTRAN	BCEXP	BCROT	ICR/NID
Type	I	I	I	I	I	I	I	I
Default	none	0	0	0	0	0	0	0

Card 2	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	EXPLIM	EFAC		FRCPAD	IEXPND
Type	F	F	F	F	F		F	I
Default	0.0	0.0	0.0	∞	0.0		0.1	0

Remaining cards are optional.[†]

Card 3	1	2	3	4	5	6	7	8
Variable	IPIDXCL	IPIDTYP						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
SID	Set ID
STYPE	Set type: EQ.0: part set EQ.1: part EQ.2: node set EQ.3: segment set
PRTYPE	Reference system type (see Remark 1): EQ.0: Eulerian EQ.1: Lagrangian EQ.2: Normal ALE mesh smoothing EQ.3: Prescribed motion following load curves; see *ALE_REFERENCE_SYSTEM_CURVE. EQ.4: Automatic mesh motion following mass weighted average velocity in ALE mesh EQ.5: Automatic mesh motion following a local coordinate system defined by three user defined nodes; see *ALE_REFERENCE_SYSTEM_NODE. EQ.6: Switching in time between different reference system types; see *ALE_REFERENCE_SYSTEM_SWITCH. EQ.7: Automatic mesh expansion in order to enclose up to twelve user defined nodes; see *ALE_REFERENCE_SYSTEM_NODE. EQ.8: Mesh smoothing option for shock waves, where the element grid contracts in the vicinity of the shock front: this may be referred to as the Delayed-ALE option. It controls how much the mesh is to be moved during the remap step. This option requires the definition of the 5th parameter Card 2, EFAC; see below for definition. EQ.9: Allowing the ALE mesh(es) to: 1. Translate and/or rotate to follow a local Lagrangian reference coordinate system (whose *ALE_REFERENCE_SYSTEM_NODE card ID is defined by the BCTRAN parameter)

VARIABLE	DESCRIPTION
	2. Expand or contract to enclose a Lagrangian part set ID defined by the PRID parameter. 3. Has a Lagrangian node ID be defined by the ICR/NID parameter to be the center of the ALE mesh expansion.
PRID	A parameter giving additional information depending on the reference system (PRTYPE) choice: PRTYPE.EQ.3: PRID defines a load curve group ID specifying an *ALE_REFERENCE_SYSTEM_CURVE card for mesh translation. This defines up to 12 curves which prescribe the motion of the system. PRTYPE.EQ.4: PRID defines a node set ID (*SET_NODE), for which a mass average velocity is computed. This velocity controls the mesh motion. PRTYPE.EQ.5: PRID defines a node group ID specifying an *ALE_REFERENCE_SYSTEM_NODE card, via which, three nodes forming a local coordinate system are defined. PRTYPE.EQ.6: PRID defines a switch list ID specifying an *ALE_REFERENCE_SYSTEM_SWITCH card. This defines the switch times and the reference system choices for each time interval between the switches. PRTYPE.EQ.7: PRID defines a node group ID specifying an *ALE_REFERENCE_SYSTEM_NODE card. Up to 12 nodes in space forming a region to be enveloped by the ALE mesh are defined. PRTYPE.EQ.9: PRID defines a Lagrangian part set ID (PSID) defining the Lagrangian part(s) whose range of motion is to be enveloped by the ALE mesh(es). This is useful for airbag modeling.
If PRTYPE = 4 or PRTYPE = 5, then	
BCTRAN	BCTRAN is a translational constraint (Remark 3). EQ.0: no constraints EQ.1: constrained x translation

VARIABLE	DESCRIPTION
	EQ.2: constrained y translation
	EQ.3: constrained z translation
	EQ.4: constrained x and y translation
	EQ.5: constrained y and z translation
	EQ.6: constrained z and x translation
	EQ.7: constrained x, y, and z translation
Else If PRTYPE = 9, then	
BCTRAN	BCTRAN is a node group ID from a *ALE_REFERENCE_SYSTEM_NODE card prescribing a local coordinate system (3 node IDs) whose motion is to be followed by the ALE mesh(es).
Else	
BCTRAN	Ignored
End if	
BCEXP	For PRTYPE = 4 and 7, BCEXP is an expansion constraint; otherwise it is ignored (Remark 3).
	EQ.0: no constraints
	EQ.1: constrained x expansion
	EQ.2: constrained y expansion
	EQ.3: constrained z expansion
	EQ.4: constrained x and y expansion
	EQ.5: constrained y and z expansion
	EQ.6: constrained z and x expansion
	EQ.7: constrained x, y, and z expansion
BCROT	BCROT is a rotational constraint (Remark 3). It is ignored unless PRTYPE = 4.
	EQ.0: no constraints
	EQ.1: constrained x rotation
	EQ.2: constrained y rotation

VARIABLE	DESCRIPTION
	EQ.3: constrained z rotation
	EQ.4: constrained x and y rotation
	EQ.5: constrained y and z rotation
	EQ.6: constrained z and x rotation
	EQ.7: constrained x, y, and z rotation
If PRTYPE = 4	
ICR/(NID)	ICR is a flag the specifies the method LS-DYNA uses for determining the center point for expansion and rotation (Remark 3). EQ.0: The center is at center of gravity of the ALE mesh. EQ.1: The center is at (XC,YC,ZC), just a point in space (it does not have to be a defined node)
Else if PRTYPE = 9	
(ICR)/NID	NID sets the Lagrangian node ID for the node that anchors the center of ALE mesh expansion (Remark 2).
End if	
XC, YC, ZC	Center of mesh expansion and rotation for PRTYPE = 4; otherwise ignored. See ICR above.
EXPLIM	Limit ratio for mesh expansion and contraction. Each cartesian direction is treated separately. The distance between the nodes is not allowed to increase by more than a factor EXPLIM or decrease to less than a factor 1/EXPLIM. This flag applies only for PRTYPE = 4, otherwise it is ignored.
EFAC	Mesh remapping factor for PRTYPE = 8 only, otherwise it is ignored. EFAC ranges between 0.0 and 1.0. When EFAC approaches 1.0, the remapping approaches the Eulerian behavior. The smaller the value of EFAC, the closer the mesh will follow the material flow in the vicinity of a shock front, that is, approaching Lagrangian behavior.

VARIABLE	DESCRIPTION
	Note that excessively small values for EFAC can result in severe mesh distortions as the mesh follows the material flow. As time evolves, the mesh smoothing behavior will approach that of an Eulerian system.
FRCPAD	For PRTYPE = 9 this is an ALE mesh padding fraction, otherwise it is ignored. FRCPAD ranges from 0.01 to 0.2. If the characteristic Lagrange mesh dimension, dL_1, exceeds $$(1 - 2 \times \text{FRCPAD}) \times dL_A,$$ where dL_A is the characteristic length of the ALE mesh, then the ALE mesh is expanded so that $$dL_A = \frac{dL_1}{1 - 2 \times \text{FRCPAD}}.$$ This provides an extra few layers of ALE elements beyond the maximum Lagrangian range of motion. EQ.0.01: $dL_A = {dL_L}/{0.98} = dL_L \times 1.020408$ EQ.0.20: $dL_A = {dL_L}/{0.60} = dL_L \times 1.666667$
IEXPND	For PRTYPE = 9, this is an ALE mesh expansion control flag, otherwise it is ignored. EQ.0: Both mesh expansion and contraction are allowed. EQ.1: Only mesh expansion is allowed.
IPIDXCL	An ALE set ID to be excluded from the expansion and/or contraction only. Translation and rotation are allowed. For example, this may be used to prevent the ALE mesh (or part) at the inflator gas inlet region from expanding too much. High ALE mesh resolution is usually required to resolve the high speed flow of the gas into the airbag via point sources. See Remark 2.
IPIDTYPE	Set ID type of IPIDXCL: EQ.0: PSID EQ.1: PID

Remarks:

1. **Required Associated Cards.** Some PRTYP values may require a supplemental definition defined using the corresponding PRID. For example, PRTYP = 3 requires a *ALE_REFERENCE_SYSTEM_CURVE card. If PRID = *n*, then in the corresponding *ALE_REFERENCE_SYSTEM_CURVE card, ID = *n*. Similar association applies for any PRTYP (i.e. 3, 5, 6, or 7) which requires a definition for its corresponding PRID parameter.

2. **Mesh Centering.** For PRTYPE = 9, ICR/NID can be used to keep a high density ALE mesh centered on the region of greatest interest, such as the inflator orifices region in an airbag model. For example, in the case of nonsymmetrical airbag deployment, assuming that the ALE mesh is initially finer near the inlet orifices and gradually coarsened away from it, defining an "anchor node" at the center of the orifice location will keep the fine ALE mesh region centered on the orifice region. Then this fine ALE mesh region will not be shifted away (from the point sources) during expansion and translation. The ALE mesh can move and expand outward to envelop the Lagrangian airbag in such a way that the inlet is well resolved throughout the deployment.

3. **Additional Constraints.** The table below shows the applicability of the various choices of PRTYPE. Simple deductions from the functional definitions of the PRTYPE choices will clarify the applications of the various constraints. For example, when PRTYP = 3, nodal motion of the ALE mesh is completely controlled by the 12 curves. Therefore, no constraints are needed.

PRTYPE	ICR/NID	BCTRAN	BCROT	BCEXP
3	NO	NO	NO	NO
4	YES (ICR)	YES	YES	YES
5	NO	YES	NO	NO
6	NO	NO	NO	NO
7	NO	NO	NO	YES
8	NO	NO	NO	NO
9	YES (NID)	YES (NGID)	NO	NO

Example 1:

Consider a bird-strike model containing 2 ALE parts: a bird surrounded by air (or void). Part set ID 1 is defined containing both parts. To allow for the meshes of these 2 parts to

move with their combined mass-weighted-average velocity, PRTYPE = 4 is used. Note that BCEXP = 7 indicating mesh expansion is constrained in all global directions.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
$      SID     STYPE     PRTYP      PRID    BCTRAN     BCEXP     BCROT    ICOORD
         1         0         4         0         0         7         0
$       XC        YC        ZC    EXPLIM
         0         0         0         0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

Example 2:

Consider a bouncing ball model containing 2 ALE parts: a solid ball (PID 1) surrounded by air or void (PID 2). Part set ID 1 is defined containing both parts. To allow for the meshes of these 2 parts to move with 2 reference system types: (a) first, they move with their combined mass-weighted-average velocity between 0.0 and 0.01 second; and subsequently (between 0.01 and 10.0 seconds) their reference system is switched to (b) an Eulerian system (thus the mesh is fixed in space), a reference system "SWITCH" is required. This is done by setting PRTYPE = 6. This PRTYPE requires a corresponding *ALE_REFERENCE_SYSTEM_SWITCH card. Note that PRID = 11 in the *ALE_REFERENCE_SYSTEM_GROUP card corresponds to the SWITCHID = 11 in *ALE_REFERENCE_SYSTEM_SWITCH card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_REFERENCE_SYSTEM_GROUP
$      SID     STYPE     PRTYP      PRID    BCTRAN     BCEXP     BCROT    ICOORD
         1         0         6        11         0         7         7
$       XC        YC        ZC    EXPLIM   EULFACT SMOOTHVMX
         0         0         0         0       0.0
*ALE_REFERENCE_SYSTEM_SWITCH
$ SWITCHID
        11
$       t1        t2        t3        t4        t5        t6        t7
      0.01      10.0
$    TYPE1     TYPE2     TYPE3     TYPE4     TYPE5     TYPE6     TYPE7     TYPE8
         4         0
$      ID1       ID2       ID3       ID4       ID5       ID6       ID7       ID8
         0         0         0         0         0         0         0         0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_REFERENCE_SYSTEM_NODE

Purpose: Define a group of nodes that control the motion of an ALE mesh. This keyword is used *only* when PRTYPE = 5 or 7 in a corresponding *ALE_REFERENCE_SYSTEM_-GROUP card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8
Type	I	I	I	I	I	I	I	I

Enclosing Nodes Card. This card is only included if PRTYPE = 7.

Card 3	1	2	3	4	5	6	7	8
Variable	NID9	NID10	NID11	NID12				
Type	I	I	I	I				

VARIABLE	DESCRIPTION
ID	Node group ID for PRTYPE 5 or 7; see PRID on *ALE_REFERENCE_SYSTEM_GROUP.
NID1, …, NID12	User specified nodes. See Remarks 1 and 2.

Remarks:

1. **ALE Motion from Local Coordinate System.** For PRTYPE = 5, the ALE mesh is forced to follow the motion of a coordinate system, which is defined by three

nodes (NID1,NID2,NID3). These nodes are located at x_1, x_2, and x_3, respectively. The axes of the coordinate system, x', y', and z', are defined as:

$$x' = \frac{x_2 - x_1}{|x_2 - x_1|}$$
$$y' = z' \times x'$$
$$z' = x' \times \frac{x_3 - x_1}{|x' \times (x_3 - x_1)|}$$

Note that $x_1 \rightarrow x_2$ is the local x'axis, $x_1 \rightarrow x_3$ is the local y' axis and x' crosses y' gives the local z' axis. These 3 nodes are used to locate the reference system at any time. Therefore, their positions relative to each other should be as close to an orthogonal system as possible for better transformation accuracy of the ALE mesh.

2. **ALE Motion from Enclosing Nodes.** For PRTYPE = 7, the ALE mesh is forced to move and expand, so as to enclose up to twelve user defined nodes (NID1, …, NID12). This is a rarely used option.

Example 1:

Consider modeling the sloshing of water inside a rigid tank. Assuming there are 2 ALE parts, the water (PID 1) and air or void (PID 2) contained inside a rigid (Lagrangian) tank (PID 3). The outer boundary nodes of both ALE parts are merged with the inner tank nodes. Part set ID 1 is defined containing both ALE parts (PIDs 1 and 2). To allow for the meshes of the 2 ALE parts to move with the rigid Lagrangian tank, PRTYPE = 5 is used. The motion of the ALE parts then follows 3 reference nodes on the rigid tank. These 3 reference nodes must be defined by a corresponding *ALE_REFERENCE_SYSTEM_NODE card. In this case the reference nodes have the nodal IDs of 5, 6 and 7. Note that PRID = 12 in the *ALE_REFERENCE_SYSTEM_GROUP card corresponds to the SID = 12 in the *ALE_REFERENCE_SYSTEM_NODE card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
$       SID     STYPE      PRTYP       PRID    BCTRAN     BCEXP     BCROT    ICOORD
          1         0          5         12
$        XC        YC         ZC    EXPLIM
           0         0          0          0
*ALE_REFERENCE_SYSTEM_NODE
$     NSID
          12
$        N1        N2         N3         N4         N5         N6         N7         N8
           5         6          7
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*ALE_REFERENCE_SYSTEM_SWITCH

Purpose: Time-dependent switching between the reference system types for ALE geometric entities. Multiple switches can occur during the simulation. See PRTYPE in *ALE_REFERENCE_SYSTEM_GROUP. This command is required only when PRTYPE = 6 in *ALE_REFERENCE_SYSTEM_GROUP. See Example 2 in the *ALE_REFERENCE_SYSTEM_GROUP.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	T1	T2	T3	T4	T5	T6	T7	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 3	1	2	3	4	5	6	7	8
Variable	TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6	TYPE7	TYPE8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Card 4	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ID	Switch list ID; see *ALE_REFERENCE_SYSTEM_GROUP.
T1, …, T7	Times for switching reference system type. By default, the reference system TYPE1 occurs between time = 0.0 and time = T1, and TYPE2 occurs between time = T1 and time = T2, etc.
TYPE1, …, TYPE8	Reference system types (also see PRTYPE under *ALE_REFERENCE_SYSTEM_GROUP): EQ.0: Eulerian EQ.1: Lagrangian EQ.2: Normal ALE mesh smoothing EQ.3: Prescribed motion following load curves; see *ALE_REFERENCE_SYSTEM_CURVE. EQ.4: Automatic mesh motion following mass weighted average velocity in ALE mesh EQ.5: Automatic mesh motion following a local coordinate system defined by three user defined nodes; see *ALE_REFERENCE_SYSEM_NODE.
ID1, …, ID8	The corresponding PRID parameters supporting each PRTYPE used during the simulation.

Remarks:

1. **Switching Time.** The beginning time is assumed to be t = 0.0, and the starting PRTYPE is TYPE1. So at T1, the 1st switching time, PRTYPE is switched from TYPE1 to TYPE2, and so forth. This option can be complex in nature, so it is seldom applied.

*ALE_REFINE

See *CONTROL_REFINE_ALE.

*ALE_SMOOTHING

Purpose: This smoothing constraint either keeps a dependent ALE node at its initial parametric location along a line between two other ALE nodes (see Figure 4-14) or causes a dependent ALE node to follow the motion of two non-ALE nodes (see Remark 1). This constraint is active during each mesh smoothing operation. This keyword can be used with ALE solids, ALE shells, and ALE beams.

Card 1	1	2	3	4	5	6	7	8
Variable	DNID	NID1	NID2	IPRE	XCO	YCO	ZCO	
Type	I	I	I	I	F	F	F	
Default	none	none	none	0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
DNID	Dependent node or node set ID: GT.0: DNID is an ALE node. EQ.0: The dependent nodes are the nodes of an ALE mesh connected to the nodes in INID1. See Remark 2. LT.0: -DNID is the ID of an ALE node set. See Remark 2.
NID1	ID of first node or set for constraining the dependent nodes: GT.0: NID1 is a node. LT.0: -NID1 is a segment set ID if XCO = YCO = ZCO = 0.0. Otherwise, -NID1 is a node set ID. See Remark 2.
NID2	ID of second node or node set for constraining the dependent nodes: GT.0: NID2 is a node. EQ.0: The dependent node motion is solely controlled by NID1. See Remarks 2 and 3. LT.0: -NID2 is a node set ID. See Remark 2.
IPRE	EQ.0: Smoothing constraints are performed after mesh relaxation.

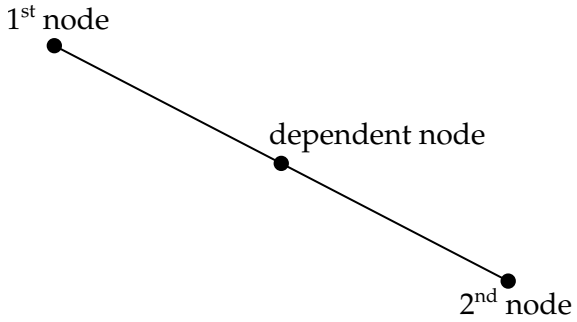


Figure 4-14. This simple constraint, which ensures that a constrained node remains on a straight line between two controlling nodes, is sometimes necessary during ALE smoothing.

VARIABLE	DESCRIPTION
	EQ.1: Smoothing constraints are performed before mesh relaxation.
XCO	x-coordinate of constraint vector
YCO	y-coordinate of constraint vector
ZCO	z-coordinate of constraint vector

Remarks:

1. **When NID1 and NID2 Nodes are not ALE Nodes.** If DNID, NID1 and NID2 are ALE nodes, the positions of NID1 and NID2 are interpolated to position DNID. If NID1 and NID2 are *not* ALE nodes, the motions of NID1 and NID2 are interpolated to move DNID.

2. **Constraint Generation.** If NID1 is a set ID, DNID and NID2 should be node set IDs or zeros. In this case, the constraints are created during initialization and printed out to a file called alesmoothingenerated.k.

   If NID1 is a node set, the constraints associated with a given node in NID1 are generated by finding the closest dependent nodes to an axis passing through the NID1 node and oriented by the constraint vector (XCO,YCO,ZCO). If NID1 is a segment set, each node in the given segments can constrain dependent nodes. The axis direction for a node is found by averaging all the normals of the segments connected to the node.

   If DNID = 0, NID1 should be a set of nodes or segments along the boundary of an ALE mesh. For a given node in NID1 or in a segment of NID1, constraints are created for all the nodes of the mesh found the closest to the axis described previously. The search for dependent nodes starts with nodes of elements

connected to the node in NID1 and stops when a boundary node with an element connectivity similar to that of the node in NID1 is reached or when a node in the set NID2 (if defined) is found.

3. **NID2 = 0 and DNID ≠ 0.** If NID2 = 0 and DNID is nonzero, NID1 should *not* be an ALE node or set. Otherwise, the positions of DNID and INID1 would match and the element volumes between them could be zero or negative. Only DNID and INID1 motion should match in such a case.

*ALE_STRUCTURED_FSI_{OPTION1}_{OPTION2}

Purpose: Perform Fluid-Structure Interaction (FSI) between Lagrangian structures modeled by shell/solids and ALE multi-material fluids using a structured ALE mesh (*ALE_STRUCTURED_MESH).

Available options for *OPTION1* include:

<BLANK>

ABEXT

ABINT

ABEXT invokes special logic for the exterior parts of airbags, while ABINT invokes special logic for interior parts.

Available options for *OPTION2* include:

<BLANK>

TITLE

With the keyword option TITLE, you can set a coupling (card) ID number and title for each coupling card. If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Comparison to *CONSTRAINED_LAGRANGE_IN_SOLID:

*ALE_STRUCTURED_FSI, as its name suggests, only works with the structured ALE solver. It mostly follows the format of *CONSTRAINED_LAGRANGE_IN_SOLID, but with a few exceptions:

- **Coupling Type.** Unlike *CONSTRAINED_LAGRANGE_IN_SOLID, *ALE_STRUCTURED_FSI only has one penalty formulation coupling method which is similar to a combination of CTYPE = 4 and 5 in *CONSTRAINED_LAGRANGE_IN_SOLID.
- **Number of Coupling Points.** For each Lagrangian segment, there are a certain number of coupling points, evenly distributed at the segment surface. Penalty springs are attached to those coupling points. Using *CONSTRAINED_LAGRANGE_IN_SOLID requires specifying the number while with this keyword, LS-DYNA automatically determines the number of coupling points during the initialization phase.
- **Leakage Control.** Leakage control is automated when using *ALE_STRUCTURED_FSI. Fluid leakage is detected and cured automatically with no user intervention needed.

- **Normal Type.** Normal type selection is automated, based on the local geometry. Users do not need to choose between nodal/segment normal.
- **Edge Coupling.** Edge coupling is automatic. Shell segments are picked out and the exposed edges are coupled. *CONSTRAINED_LAGRANGE_IN_SOLID_-EDGE is not needed.
- **Erosion Coupling.** Eroded solid elements will change the coupling segments. Segments belonging to eroded elements need to be deleted, and newly exposed segments need to be added. With *CONSTRAINED_LAGRANGE_IN_SOLID, CTYPE = 5 needs to be specified in order to activate this segment's modification algorithm. In the new *ALE_STRUCTURED_FSI, this process is always on and no flag is needed.

Title Card. Additional card for the TITLE keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	LSTRSID	ALESID	LSTRSTYP	ALESTYP				MCOUP
Type	I	I	I	I				I
Default	none	none	0	0				0

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	PFAC	FRIC	ILVL	FLIP	DECAY	OFFSET
Type	F	F	F	F	I	I	F	F
Default	0.0	10^{10}	0.1	0.0	0	0	0.0	0.0

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID. If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	Description of this coupling definition (A70)
LSTRSID	Set ID defining a part, part set, or segment set ID of the Lagrangian structure (see *PART, *SET_PART, or *SET_SEGMENT)
ALESID	Set ID defining a part or part set ID of the structured ALE mesh (see *PART)
LSTRSTYP	Set type of LSTRSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID) EQ.2: Segment set ID (SGSID)
ALESTYP	Set type of ALESID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
MCOUP	Multi-material(s) to be coupled (see Remark 1): EQ.0: Couple with all multi-material groups. EQ.-N: -N is the ID of *SET_MULTI-MATERIAL_GROUP_LIST.
START	Start time for coupling
END	End time for coupling
PFAC	Penalty factor. PFAC is a scale factor for scaling the estimated stiffness of the interacting (coupling) system. It is used to compute the coupling forces to be distributed on the Lagrangian and ALE parts. GT.0: Fraction of estimated critical stiffness LT.0: PFAC must be an integer, and -PFAC is a load curve ID. The curve defines the coupling pressure on the y-axis as a function of the penetration along the x-axis.
FRIC	Friction coefficient. The friction force is evaluated as the normal force multiplied by the friction coefficient. GT.0: Constant friction coefficient

VARIABLE	DESCRIPTION
	EQ.-N: Variable friction coefficient defined by a table with ID N. The friction coefficient is a function of coupling pressure and relative velocity. The table uses the coupling pressure as the parameter values. See Examples.
ILVL	Number of airbag segments to look downstream from the gas front. It is only used for the ABEXT keyword option. Together with DECAY, it activates pre-opening logic for gas-front segments to alleviate constrained flow typically observed in gas flow through long tubes. Please refer to *AIRBAG_SALE (specifically Remark 8) for a detailed explanation.
FLIP	A Lagrangian segment will couple to fluid on only one side of the segment. The segment normal is assumed to point toward the fluids to which it is coupled. If that is not the case, set FLIP to 1. EQ.0: No action EQ.1: Flip the segment normal, so it points toward the fluids to which it is to be coupled.
DECAY	Decaying factor. It is only used for the ABEXT keyword option. Together with ILVL, it activates pre-opening logic for gas-front segments to alleviate constrained flow typically observed in gas flow through long tubes. Please refer to *AIRBAG_SALE (specifically Remark 8) for a detailed explanation.
OFFSET	Specify how much to offset Lagrange segments away from the segment surface: EQ.0.0: No offset. For Lagrange shell elements, this setting means coupling occurs at the mid-plane of the shell element. This option is the default. GT.0.0: Fraction of Lagrange segment thickness. This option only applies to segments from Lagrange shell elements. For example, setting OFFSET to 0.5 offsets the coupling surface by half the shell thickness. LT.0.0: OFFSET must be an integer, and -OFFSET is a load curve ID. The curve defines the offset distance in absolute value on the y-axis as a function of the time along the x-axis. This option is supported for both Lagrange shell and solid elements.

Remarks:

1. **Multi-material coupling option.** MCOUP is used to specify which ALE multi-materials are to be coupled. In a typical simulation, we want to prevent some fluid(s) on one side of the structure from penetrating through to the other side. In this case, we would pick the AMMGs on one side and list them under *SET_MULTI-MATERIAL_GROUP_LIST.

   Coupling to all materials as activated by MCOUP = 0 is generally not recommended. LS-DYNA calculates the fluid coupling interface as the surface where the sum of coupled ALE materials occupies a volume fraction equal to 50%. Since MCOUP = 0 couples to all materials, the sum of all coupled ALE materials is, in this case, trivially 100%. Consequently, when MCOUP = 0, there will not be a fluid interface with which to track leakage.

2. **Output.** To obtain the sliding energy output for each FSI, specify a *DATABASE_SLEOUT in the input deck. Including this keyword causes a file called slefsi to be output. The format of slefsi follows sleout exactly.

Examples:

The following is a partial input deck that illustrates defining a variable friction table. Table 2 gives three coupling pressures that correspond to the three *DEFINE_CURVEs specified under the table definition (see *DEFINE_TABLE for details). Each curve specifies the relationship between relative velocity and friction coefficient for a given coupling pressure. For example, if the coupling pressure is 100, curve 4 is used. If coupling pressure = 50, both curves 3 and 4 are used. The friction coefficient is interpolated from the values given by the curves.

```
*ALE_STRUCTURED_FSI
$# lstrsid    alesid   lstrtyp   alestyp         -         -         -     mcoup
         1         1         0         1         0         0         0         -
1
$#   start       end      pfac      fric         -      flip
                          0.1        -2         0         0
*DEFINE_TABLE
$     tbid       sfa      offa
           2
$                   p
                  0.0
                100.0
               1000.0
$---------------------------------------------------------------------------
--
*DEFINE_CURVE
$ curve for p = 0.0
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         3         0       1.0       1.0       0.0       0.0         0         0
$               vel1                 mu1
                 0.0                 0.6
```

```
    1.0000000000e+02                0.5
*DEFINE_CURVE
$ curve for p = 100.0
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         4         0       1.0       1.0       0.0       0.0         0         0
$               vel2                 mu2
                 0.0                 0.5
    1.0000000000e+02                 0.4
*DEFINE_CURVE
$ curve for p = 1000.0
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         5         0       1.0       1.0       0.0       0.0         0         0
$               vel3                 mu3
                 0.0                 0.3
    1.0000000000e+02                 0.2
```

*ALE_STRUCTURED_MESH

Purpose: Generate a structured 2D or 3D mesh and invoke the Structured ALE (S-ALE) solver. Spacing parameters are input through one or more of the *ALE_STRUCTURED_MESH_CONTROL_POINTS cards. The local coordinate system is defined using the *ALE_STRUCTURED_MESH card.

In certain contexts, it is advantageous to use structured meshes. With structured meshes, the element and node connectivity are straightforward, and the searching algorithm used for ALE coupling is greatly simplified. Also, numerous checks are avoided because these meshes include only hex elements.

The S-ALE solver supports SMP, MPP, and MPP hybrid configurations. All three implementations require less simulation time and memory usage than the regular ALE solver. We, therefore, encourage using the S-ALE solver when the ALE mesh is structured.

Once an ALE mesh is generated using an *ALE_STRUCTURED_MESH card, this card invokes the S-ALE solver and performs the ALE advection time step. For fluid-structure interaction, *ALE_STRUCTURED_FSI is recommended over *CONSTRAINED_LAGRANGE_IN_SOLID. *ALE_STRUCTURED_FSI has better, automated leakage detection and control, has a much cleaner and easier input, and is designed for the S-ALE solver.

This keyword can be used multiple times. In each occurrence, an independent, separate mesh is constructed. These meshes can occupy different spatial domains or the same spatial domain. Simulations are executed in those meshes independently.

There are two related keywords: *ALE_STRUCTURED_MESH_TRIM to trim/untrim the generated mesh and *ALE_STRUCTURED_MESH_MOTION to control the mesh motion during the simulation.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	DPID	NBID	EBID				TDEATH
Type	I	I	I	I				F
Default	none	none	Rem 6	Rem 6				10^{16}

Card 2	1	2	3	4	5	6	7	8
Variable	CPIDX	CPIDY	CPIDZ	NID0	LCSID			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. A unique number must be specified.
DPID	Default Part ID. The elements generated are assigned to DPID. This part definition is automatically generated during the input phase and contains neither material nor element formulation information. Please see Remark 1. This ID also serves as an indicator for mesh merging, that is, if the same part ID is used for several adjacent generated meshes, they will be merged together to form a single mesh consisting of these sub-meshes. Please see Remark 5.
NBID	Nodes are generated and assigned with node IDs starting from NBID. If setting to a nonzero value (recommended), use an ID larger than the maximum existing node ID. If unset or 0 (not recommended), NBID is internally set to the maximum node ID + an offset. See Remark 6.
EBID	Elements are generated and assigned with element IDs starting from EBID. If setting to a nonzero value (recommended), use an ID larger than the maximum existing element ID. If unset or 0 (not recommended), EBID is internally set to the maximum element ID + an offset. See Remark 6.
TDEATH	Death time for this mesh. Please see Remark 3.
CPIDX, CPIDY, CPIDZ	Control point IDs defining node ID/value pairs along each local axis. See *ALE_STRUCTURED_MESH_CONTROL_POINTS. Setting CPIDX to 0 or -1 invokes the ALE to S-ALE converter. Please see Remark 4. Note that for 2D problems CPIDZ is ignored.

VARIABLE	DESCRIPTION
NID0	NID0 sets the mesh's origin node. During the simulation, prescribed motion applied to this node applies to the entire structure S-ALE mesh.
LCSID	Local coordinate system ID. Please see Remark 2.

Remarks:

1. **DPID.** DPID only consists of elements and nodes. It does not include material properties or integration rules. The requirement that a part ID be specified for these automatically generated S-ALE solid elements exists only to satisfy the legacy rule that every element must be associated with a part. You do not need to set up the *PART card for DPID. All part definitions used in this card only refer to the mesh, not the material.

2. **LCSID.** The data cards associated with a *DEFINE_COORDINATE_... keyword define the local coordinate system. This local coordinate system specifies the three cardinal directions used for generating the structured ALE mesh. The structured mesh can be made to rotate by specifying a rotating local coordinate system. To define a rotating local coordinate system, use *DEFINE_COORDINATE_NODES with FLAG = 1 and then apply the prescribed motion to the three coordinate nodes.

3. **TDEATH.** This option provides a way to terminate an ALE calculation while Lagrangian parts still evolve. Once this death time is reached, all elements belonging to this mesh are deleted. In addition to that, ALE FSI cards, including *ALE_STRUCTURED_FSI, *CONSTRAINED_LAGRANGE_IN_SOLID, and *ALE_COUPLING_NODAL_*OPTION*, are checked to see if related ALE elements were deleted. If no ALE elements are left on the fluid side for certain coupling cards, this coupling card is disabled from further calculation as well.

   For example, consider a 2-stage simulation. In the first stage, Lagrange structures are under certain loadings from ALE fluids through ALE/FSI. Later in the second stage, we continue simulating the Lagrange structure behavior while stopping all ALE-related calculations by setting TDEATH to the start time of the 2nd stage.

4. **ALE to S-ALE converter.** For existing ALE models with rectilinear meshes, *ALE_STRUCTURED_MESH can invoke the conversion of the ALE mesh into an S-ALE mesh. To use this feature, add a *ALE_STRUCTURED_MESH card in the model input with CPIDX = -1 or 0 leaving all other fields blank. LS-DYNA will then convert all ALE keywords to the S-ALE format and write the modified input to a file named saleconvrt.inc. The solver used to perform the analysis

depends on the value of CPIDX. If -1, the S-ALE solver is used; if 0, the ALE solver is used. Please note that this converter is only available in 3D.

5. **Merging meshes.** Two or more separated, but adjacent sub-meshes can be merged to form a mesh of complex shape. To do that, first set up two or more sub-meshes; make sure their mesh spacing is the same at their interfaces so that these interface nodes match each other (a tolerance of 1/100 element size is allowed); and finally use the SAME DPID in those *ALE_STRUCTURED_MESH cards. The code automatically checks and merges sub-meshes with the same DPID.

6. **Default behavior of NBID/EBID.** Generally, we recommend not leaving NBID and EBID blank or zero. When left blank or set to 0, the keyword reader scans the input deck to find the maximum user ID and then adds an offset to that maximum ID to generate NBID/EBID. The initial design had the offset set to 1. However, ID collisions began occurring due to other features autogenerating nodes/elements in the same manner. With R16.0, we changed this offset to 1,000,000 to help avoid this issue.

Example:

The following example generates a regular evenly distributed 0.2 by 0.2 by 0.2 box mesh that has 22 nodes along each direction. The generated mesh is aligned to the local coordinate system specified by nodes 2, 3, and 4 with an origin at node 1.

All the elements inside the mesh are assigned to part 1. Note that part 1 is not explicitly defined in the input. The necessary part definition is automatically generated and contains neither material definitions nor integration rules.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid                                    tdeath
         1         1    200001    200001                                     0.010
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1001      1001         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3      flag
       234         2         3         4         1
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2
                   1                  .0
                  22                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

*ALE_STRUCTURED_MESH_CONTROL_POINTS

Purpose: Provide spacing information used by the *ALE_STRUCTURED_MESH keyword to generate a 2D or 3D structured ALE mesh.

Each instance of the *ALE_STRUCTURED_MESH_CONTROL_POINTS card defines a one-dimensional mesh using control points. Each control point consists of a node number (see N in Card 2a/b) and a coordinate (see X in Card 2a/b). The first control point *must* be node 1, and the node number of the last point defines the total number of nodes. Between any two control points, the mesh is uniform or progressive, depending on the input. The *ALE_STRUCTURED_MESH card, in turn, defines a simple three-dimensional mesh from the triple product of three *ALE_STRUCTURED_MESH_CONTROL_POINTS one-dimensional meshes.

Card Summary:

Card 1. This card is required.

CPID		ICASE	SFO		OFFO		

Card 2a. For ICASE = 0, include at least two instantiations of this card to give the control points. A minimum of two cards is required to give the end points and the number of nodes. Input is terminated with the next keyword ("*") card.

N	X	RATIO	

Card 2b. For ICASE = 1 or 2, include at least two instantiations of this card to give the control points. A minimum of two cards is required to give the end points and the number of nodes. Input is terminated with the next keyword ("*") card.

N	X	XL	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	CPID		ICASE	SFO		OFFO		
Type	I		I	F		F		
Default	none		0	1.		0.		

VARIABLE	DESCRIPTION
CPID	Control Points ID. A unique number must be specified. This ID is to be referred to in the three fields labeled CPIDX, CPIDY, and CPIDZ in *ALE_STRUCTURED_MESH.
ICASE	A flag to trigger special algorithms for a more user-friendly input format for creating progressive mesh spacing: EQ.0: Ratio-based progressive mesh spacing (see Card 2a) EQ.1: Base element size progressive mesh spacing (see Card 2b and Remark 4) EQ.2: Base node progressive mesh spacing (see Card 2b and Remark 5)
SFO	Scale factor for ordinate value. This is useful for simple modifications. EQ.0.0: default set to 1.0.
OFFO	Offset for ordinate values. See Remark 1.

Point Cards for ICASE = 0. Include this card for ICASE = 0. Put one control point per card. Input is terminated at the next keyword ("*") card. *At least two cards are required with one having N = 1.*

Card 2a	1	2	3	4	5	6	7	8
Variable	N		X		RATIO			
Type	I		F		F			
Default	none		none		0.0			

VARIABLE	DESCRIPTION
N	Control point node number
X	Control point position
RATIO	Ratio for progressive mesh spacing. A progressively larger or smaller mesh will be generated between the control point that has a nonzero ratio specified and the control point following it. If a

VARIABLE	DESCRIPTION
	control point and the one following it both have RATIO as zero, the mesh is uniform between them. See Remarks 2 and 3.
	GT.0.0: Mesh size increases
	$dl_{n+1} = dl_n \times (1 + \text{RATIO})$
	LT.0.0: Mesh size decreases
	$dl_{n+1} = dl_n / (1 - \text{RATIO})$

Point Cards for ICASE = 1 or 2. Include this card for ICASE = 1 or 2. Put one control point per card. Input is terminated at the next keyword ("*") card. *At least two cards are required with one having N = 1.* See Remarks 4 and 5 for details about how to use these cards to obtain uniform and progressive meshes. See Remark 3.

Card 2b	1	2	3	4	5	6	7	8
Variable	N		X		XL			
Type	I		F		F			
Default	none		none/optional		0.0/none			

VARIABLE	DESCRIPTION
N	Control point node number
X	Control point position. This is required for ICASE = 1. For ICASE = 2, see Remark 5 for details about when to include this input.
XL	Length of element at the control. For ICASE = 1, see Remark 4 for details about when to enter this value. This input is required for ICASE = 2.

Remarks:

1. **Coordinates scaling.** The ordinate values are scaled after the offsets are applied, that is:

$$\text{Ordinate value} = \text{SFO} \times (\text{Defined value} + \text{OFFO}) .$$

2. **Progressive mesh spacing.** The formula used to calculate element length is as follows:

$$\mathrm{dl}_{\mathrm{base}} = |x_{\mathrm{end}} - x_{\mathrm{start}}| \times (\mathrm{factor} - 1)/(\mathrm{factor}^n - 1)$$

in which $\mathrm{dl}_{\mathrm{base}}$ is the smallest base length; x_{start} and x_{end} are the coordinates of the start and end points, respectively; factor = 1 + RATIO if RATIO > 0 or 1/(1 − RATIO) if RATIO < 0; and n is number of elements. Please note that element size either increases by RATIO if RATIO > 0 or decreases by − RATIO/(1 − RATIO) if RATIO < 0 each time. The overall effect is the same; that is, starting from the smallest element, each time the element size is increased by |RATIO|.

3. **Transition meshes.** If the mesh has different mesh sizes and transition zones between them (where the mesh size progressively increases or decreases), then the ratios and number of elements for these zones are reported in the messag file (look for messages KEY+1174). Updated lists of control points are output in the messag file for each *ALE_STRUCTURED_MESH_CONTROL_POINTS with more than 3 control points.

4. **ICASE = 1.** Assume we want to use progressive mesh spacing for a certain region. We know the number of elements and the overall length. And we have a "base" element size in mind. However, the generic setup requires a "ratio", which needs an iterative solution of a n^{th} order polynomial equation based on the "base" element size. ICASE = 1 aims to simplify the setup. When it is activated, we can input the "base" element size. The value of the "ratio" is internally calculated and used to generate the S-ALE mesh. We will use examples to illustrate this option.

   For this discussion, a region is defined by 2 point cards (2 instantiations of Card 2b). In the example below, *ALE_STRUCTURED_MESH_CONTROL_-POINTS with CPID 1001 has three regions. The first region spans [0, 0.0755]; the second one [0.0755, 0.1245]; and the third one [0.1245, 0.2].

   The RATIO field in the generic setup is now marked as XL. In this case, it specifies the element size at that location. The rule is simple: a region can have a total of 0, 1, or 2 nonzero XL values specified at its two ends. 1 occurrence causes progressive mesh spacing. 0 or 2 occurrences cause an evenly spaced mesh. Please note that when 2 values of XL are given, they must have the same value since the mesh is evenly spaced. Otherwise, the input is not logically correct.

   In the example below:

   a) Region 1 has progressive spacing, and the preferred element size at the 8^{th} node is 0.0070.

   b) In region 2, the mesh is evenly distributed.

c) Region 3 has progressive spacing, and the preferred element size at the 15th node is 0.0070.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
$     CPID                ICASE
      1001                    1
$                x1                  x2                  xl
                  1                  .0
                  8              0.0755              0.0070
                 15              0.1245              0.0070
                 22                 0.2
```

The above setup will produce a progressive mesh for regions 1 and 3 (indicated by 1 XL value for each region's two end points) and a uniform mesh for region 2 (indicated by 2 XL values for the region's two end points). The element size is 0.0155 at the 1st node; gradually reduces to 0.007 at the 8th node; remains unchanged until the 15th node; then gradually increased to 0.0155 at the 22nd node.

We include two other examples to illustrate the versatility of the setup. For CPID 1002, regions 1 and 3 have progressive meshes and region 2 has a uniform mesh. For CPID 1003, regions 1 and 2 both have progressive meshes.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002                    1
$                x1                  x2                  xl
                  1                  .0              0.0155
                  8              0.0755
                 15              0.1245
                 22                 0.2              0.0155
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003                    1
$                x1                  x2                  xl
                  1                  .0
                 11                 0.1              0.0070
                 21                 0.2
```

5. **ICASE = 2.** For ICASE = 2, the idea is to build the progressive spaced mesh starting from a point and extend the mesh to the left and the right. To use this option, we need to first pick a "base node" and put it at a location by specifying its coordinate. The "base node" in the example below is the 8th node with a coordinate of 0.0755. Next, we specify the element size at all the nodes listed. In the example below, we want the element size to be 0.0155 at the 1st node, 0.007 at the 8th node, 0.007 at the 15th node and 0.0155 at the 22nd node.

Internally, the mesh is divided into 3 regions as discussed for ICASE = 1 in Remark 4. Progressive meshes are used between two control points with different element lengths while uniform meshes are used between two control points with the same element length. In the example below, regions 1 and 3 have progressive mesh spacing while region 2 has an evenly distributed mesh.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
$     CPID                ICASE
      1001                    1
$                x1                  x2                  xl
```

```
                  1                                0.0155
                  8             0.0755             0.0070
                 15                                0.0070
                 22                                0.0155
```

We include two other examples to illustrate the versatility of the setup. For CPID 1002, regions 1 and 3 have progressive meshes and region 2 has a uniform mesh. For CPID 1003, regions 1 and 2 both have progressive meshes.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002                   1
$                 x1                   x2                   xl
                  1                                       0.0155
                  8                                       0.0070
                 15                                       0.0070
                 22                  0.2                  0.0155

*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003                   1
$                 x1                   x2                   xl
                  1                                       0.0155
                 12                  0.1                  0.0070
                 22                                       0.0155
```

Examples:

1. The example below generates a regular box mesh. Its size is 0.2 × 0.2 × 0.2. It is generated in a local coordinate system defined by nodes 2, 3, and 4 that originates from node 1.

   The local *x*-axis mesh spacing is defined by control points ID 1001. It has 21 nodes evenly distributed from 0.0 to 0.2. The local *y*-axis is defined by ID 1002 and has twice the number of elements of 1001. It has 41 nodes evenly distributed from 0.0 to 0.2. The local *z*-axis is defined by ID 1003. It has 31 nodes and covers from 0.0 to 0.2. The mesh is two times finer in the region between node 6 and node 26.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid
         1         1    200001    200001
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1002      1003         1       234
*DEFINE_COORDINATE_NODES
$       cid      nid1      nid2      nid3
        234         2         3         4
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                  x1                   x2
                    1                   .0
                   21                   .2
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002
$                  x1                   x2
                    1                   .0
                   41                   .2
```

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003
$                 x1                  x2
                   1                  .0
                   6                 .05
                  26                 .15
                  31                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

2. This example shows how to generate a progressively larger/smaller mesh spacing. The mesh geometry is the same as the example above. In the x-direction the mesh becomes progressively smaller between nodes 1 and 8. For these 7 elements, each element is 0.1/1.1 = 9.09% smaller than its left neighbor. Between nodes 15 and node 22, the elements become progressively larger by 10% in comparison to each element's left neighbor. The 7 elements in the middle between node 8 and 15 are of equal length.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid
         1         1    200001    200001
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1002      1003         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3
       234         2         3         4
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2              ratio
                   1                  .0              -0.1
                   8          0.06666667
                  15          0.13333333               0.1
                  22                  .2
```

*ALE_STRUCTURED_MESH_JOIN

Purpose: Join structured ALE meshes generated by the *ALE_STRUCTURED_MESH keyword card. This keyword merges nodes along a common boundary between the specified meshes.

As many cards as needed can be input with each card representing one mesh generated with *ALE_STRUCTURED_MESH to be joined to the greater mesh. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
MSHID	S-ALE mesh ID for the mesh generated with *ALE_STRUCTURED_MESH

*ALE_STRUCTURED_MESH_MOTION

Purpose: Control the motion of a structured 3D ALE mesh generated by *ALE_STRUCTURED_MESH. Two motions are implemented. With the first, the mesh follows the center of mass of certain fluids and expands/contracts at the same rate as the specified fluids. With the second, the mesh follows the motion of a Lagrangian structure and expands/contracts so that the structure is covered by the mesh.

Card Summary:

Include as many of the following card types as needed. For example, you can include 1b, 1a, 1a, 1a, and 1b in a single input. This input ends at the next keyword ("*") card.

Each card represents a mesh motion operation. The OPTION field specifies the kind of S-ALE mesh motion:

- FOLLOW_GC makes the mesh follow the mass center of certain fluid(s) in the mesh and expand/contract at the same rate as those fluid(s). This mesh motion is useful, for instance, when modeling a moving projectile hitting a target.
- COVER_LAG makes the S-ALE structured mesh expand or contract so that the Lagrangian structure remains fully covered by the S-ALE mesh. It does not account for rigid body motion of the structured mesh. To apply rigid body motion, specify NID0 in *ALE_STRUCTURED_MESH. The node used for NID0 and NODCEN may be the same. This mesh motion is particularly useful for modeling airbag deployment.

Card 1a. Include this card when OPTION = FOLLOW_GC.

MSHID	OPTION	AMMGSID	EXPLIM				ISYM

Card 1b. Include this card when OPTION = COVER_LAG.

MSHID	OPTION	SID	STYPE	NODCEN		FRCPAD	

Data Card Definitions:

Fluid-Controlled Motion Card. Use this card when OPTION = FOLLOW_GC.

Card 1a	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	AMMGSID	EXPLIM				ISYM
Type	I	A	I	F				I
Default	none	none	0	1.0				0

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID defined in *ALE_STRUCTURED_MESH
OPTION	FOLLOW_GC
AMMGSID	Set of ALE multi-material group list IDs which the mesh follows. Please refer to *SET_MULTI-MATERIAL_-GROUP_LIST card for details.
EXPLIM	Limit ratio for mesh expansion and contraction. The distance between the nodes is not allowed to increase by more than a factor EXPLIM or decrease to less than a factor 1/EXPLIM. Default value of 1.0 means no expansion/contraction.
ISYM	A three digit number, [XYZ], to define symmetry: $$\text{ISYM} = 100 \times X + 10 \times Y + Z$$ Each digit specifies one direction, X for the local x, Y for local y, and Z for local z (local x, y, and z are defined in *ALE_STRUCTURED_MESH). Each digit can have the following values: EQ.0: No symmetry EQ.1: Symmetry plane along the face with a normal vector in the negative direction EQ.2: Symmetry plane along the face with a normal vector in the positive direction For example, ISYM = 201 means quarter symmetry with symmetry planes at the faces with normal vectors $+x$ and $-z$. 111 means 1/8 symmetry with symmetry planes at the

VARIABLE	DESCRIPTION
	faces with normal vectors *-x*, *-y* and *-z*.

Structure-Controlled Motion Card. Include this card if OPTION = COVER_LAG.

Card 1b	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	SID	STYPE	NODCEN		FRCPAD	
Type	I	A	I	I	I		F	
Default	none	none	none	0	none		0.1	

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. Defined in *ALE_STRUCTURED_MESH.
OPTION	COVER_LAG
SID	Set ID to identify the Lagrange structure
STYPE	Set type for SID: EQ.0: Part set EQ.1: Part EQ.2: Segment set EQ.3: Node set
NODCEN	Optional node ID used as the center of mesh expansion
FRCPAD	Multiplier that causes the S-ALE mesh to extend beyond the Lagrangian structure in each of the local directions of the S-ALE mesh (see *ALE_STRUCTURED_MESH). In each direction, the length of the S-ALE is calculated as: $$l_i^S = (1 + 2 \times \text{FRCPAD}) \times l_i^L$$ where l_i^S is the length of the S-ALE mesh in the i^{th} direction ($i = 1, 2, 3$) and l_i^L is the longest length of the Lagrangian structure in the i^{th} direction. The padding beyond the structure is applied evenly to each side. FRCPAD prevents the fluid-structure interaction (FSI) from occurring on the edge of the S-ALE mesh.

*ALE_STRUCTURED_MESH_REFINE_{OPTION}

For *OPTION* the available options are:

<BLANK>

REGION

Purpose: Refine existing structured ALE (S-ALE) meshes generated by the *ALE_STRUCTURED_MESH keyword. This keyword automatically updates sets defined using the values SALECPT and SALEFAC in the OPTION field for the keywords *SET_SOLID_GENERAL, *SET_SEGMENT_GENERAL, and *SET_NODE_GENERAL. As a result, this card is the only modification needed in the input deck for refining an S-ALE mesh.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	IFX	IFY	IFZ				
Type	I	I	I	I				
Default	none	1	1	1				

Local Refinement. Include this card for the REGION keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	IMIN	IMAX	JMIN	JMAX	KMIN	KMAX		
Type	I	I	I	I	I	I		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
MSHID	S-ALE mesh ID for the S-ALE mesh to be refined
IFX, IFY, IFZ	Refinement factor for each local direction. The number of elements in each direction of the new mesh is the refinement factor for that direction multiplied by the current number of elements in that direction. They must be integers.

VARIABLE	DESCRIPTION
	IFX.LT.0: |IFX| is the level of refinement in all directions.
IMIN,IMAX, JMIN,JMAX, KMIN,KMAX	Minimum and maximum nodal indices (or control points) along each direction of the S-ALE mesh that defines a region to refine. These integers should be between the smallest and largest control points of *ALE_STRUCTURED_MESH_CONTROL_POINTS keywords called by *ALE_STRUCTURED_MESH to create the mesh with the specified MSHID. Note that for 2D S-ALE, KMIN and KMAX are ignored.

Remarks:

1. **Multi-material problems.** This keyword provides a modeling technique to handle multi-material ALE problems. In comparison to pure Lagrange problems, models containing multi-material ALE fluids are often time consuming and memory demanding. Therefore, a recommended approach to this problem is constructing a concept model with a coarse mesh to estimate the computational resources required and then refining the concept model mesh gradually until convergence is achieved. This keyword minimizes the user's effort to follow such a procedure.

Example:

The example below generates a regular evenly distributed box mesh with mesh ID 1. The S-ALE mesh has 22 nodes along each direction, and the overall size is 0.2 by 0.2 by 0.2. S-ALE mesh 1 is generated using the local coordinate system defined by nodes 2, 3, and 4 with the origin located at node 1.

```
*ALE_STRUCTURED_MESH
$     mshid      dpid      nbid      ebid
          1         1    200001    200001
$     cpidx     cpidy     cpidz      nid0     lcsid
       1001      1001      1001         1       234
*DEFINE_COORDINATE_NODES
$       cid      nid1      nid2      nid3
        234         2         3         4
*SET_SOLID_GENERAL
$       SID
        100
$    OPTION     MSHID       XMN       XMX       YMN       YMX       ZMN       ZMX
    SALECPT         1         1        22         1        22        11        22
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2
```

```
                  1                  .0
                 22                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
       5   0.0000000e+00   0.0000000e+00   0.0000000e+00
*END
```

If at a later time we decided to make the mesh finer, we can simply add the following card to the input deck:

```
*ALE_STRUCTURED_MESH_REFINE
$    mshid       ifx       ify       ifz
         1         3         3         3
```

The mesh now contains 63 elements (64 nodes) in each direction. The solid element set 100 now contains elements ranging between nodes (1,1,31) and (64,64,64), instead of the original (1,1,11) and (22,22,22).

*ALE_STRUCTURED_MESH_TRIM

Purpose: Perform trim/un-trim operations on a structured ALE mesh generated by *ALE_STRUCTURED_MESH. This keyword is supported for 2D and 3D S-ALE meshes.

As many cards as needed can be input, with each card representing one mesh trim/un-trim operation. This input ends at the next keyword ("*") card. These operations are completed one by one, in the order of their cards' appearance. Please see the examples below.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	OPER	IOUTIN	E1	E2	E3	E4
Type	I	A	I	I	I or F	I or F	I or F	I or F
Default	0	none	0	0	none	none	none	None

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID
OPTION	There are several available trim options: PARTSET, SEGSET, SLDSET, SHLSET, BEAMSET, PLANE, CYLINDER, BOXCOR, BOXCPT, and SPHERE. See the table below for more details.
OPER	To trim or un-trim, that is, to delete the picked elements or keep them. EQ.0: Trim (default) EQ.1: Keep
IOUTIN	Flag to select which elements to trim, that is, "outside" or "inside" the specified object defined with the OPTION and E*n*. For PARTSET, SEGSET, SLDSET, SHLSET, and BEAMSET options, "outside" is defined as the region to which the segment normal points. EQ.0: Outside (default) EQ.1: Inside

VARIABLE	DESCRIPTION
E1, E2, E3, E4	These values have different definitions for different options. See the table below for details.

The "OPTION" column in the table below enumerates the allowed values for the OPTION variable as well as describing E1, …, E4 for each OPTION. Each of the following operations accepts up to four arguments but may take fewer. Values of E*n* left unspecified are ignored.

OPTION	DESCRIPTION
PARTSET	Trim using a Lagrangian part set. E1 is the shell or solid part set ID. E2 is the distance. Elements farther away than the distance in the direction of the shell or solid outer face normal vectors are deleted/kept (depending on the value of IOUTIN). Note that only elements on one side will be deleted. To delete the elements on both sides, repeat the card with the IOUTIN value reversed.
SEGSET	Trim using a Lagrangian segment set. E1 is the segment set ID. E2 is the distance. Elements farther away than the distance in the direction of the segment normal vectors are deleted/kept (depending on the value of IOUTIN). Note that only elements on one side will be deleted. To delete both sides, repeat the card with the IOUTIN value reversed.
SLDSET	Trim using a Lagrangian solid set. E1 is the solid set ID. E2 is the distance. Elements farther away than the distance in the direction of the solid outer face normal vectors (depending on the value of IOUTIN) are deleted/kept. Note that only elements on one side will be deleted. To delete both sides, repeat the card with the IOUTIN value reversed.
SHLSET	Trim using a Lagrangian shell set. E1 is the shell set ID. E2 is the distance. Elements farther away than the distance in the direction of the shell face normal vectors in 3D (or shell outer side normal vectors in 2D) are deleted/kept (depending on the value of IOUTIN). Note that only elements on one side will be deleted. To delete both sides, repeat the card with the IOUTIN value reversed.
BEAMSET	Trim using a Lagrangian beam set for a 2D mesh. E1 is the beam set ID. E2 is the distance. Elements farther away than the distance in the direction of the beam normal vectors are

OPTION	DESCRIPTION
	deleted/kept (depending on the value of IOUTIN). Note that only elements on one side will be deleted. To delete both sides, repeat the card with the IOUTIN value reversed.
PLANE	Trim by plane. E1 is the node ID of a node on the plane. E2 is another node ID off the plane. And vector E2 – E1 is normal to the plane. Note that "outside" the plane is the portion of the mesh to which the plane's normal vector points, and "inside" the plane is the portion of the mesh on the other side of the plane.
CYLINDER	Trim by CYLINDER. E1 and E2 are node IDs of the center nodes at two ends. E3 and E4 are the radius at those two ends.
BOXCOR	Trim by box. E1 is the box ID. Please refer to *DEFINE_BOX for details on setting up a box in the global coordinate system or *DEFINE_BOX_LOCAL in the local coordinate system.
BOXCPT	Trim by box. The box is defined using S-ALE control points (CPT). E1 is box ID. Please refer to *DEFINE_BOX for details on setting up a box.
SPHERE	Trim by sphere. E1 is the node ID of the sphere center node. E2 is the radius of the sphere.

Examples:

This example shows how to trim a mesh generated by *ALE_STRUCTURED_MESH. We use the same mesh as the example in the *ALE_STRUCTURED_MESH manual page, but now we trim the mesh so any elements outside of a sphere centered at (0.0,0.0,0.0) with a radius of 0.1 are deleted.

```
*ALE_STRUCTURED_MESH
$     mshid       pid      nbid      ebid
          1         1    200001    200001
$      nptx      npty      nptz      nid0     lcsid
       1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$     mshid    option      oper    ioutin       nid    radius
          1    SPHERE                            1      0.10
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
```

```
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

If instead we want to trim the elements inside the sphere, we simply change ioutin to 1 as follows.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                   1         1      0.10
*END
```

If we want to trim the elements between two spheres both centered at (0.0,0.0,0.0) with radii of 0.05 and 0.1, respectively, multiple methods requiring two cards can achieve this. In one method, first delete all the elements inside the large sphere (0.1 radius). Then un-delete, the elements inside the small sphere (0.05 radius).

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                   1         1      0.10
         1    SPHERE         1         1         1      0.00
*END
```

Another method is to first delete all the elements outside the small sphere (0.05 radius) and then un-delete the elements outside the large sphere (0.1 radius).

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                   0         1      0.05
         1    SPHERE         1         0         1      0.10
*END
```

To use BOXCPT, we define a box using S-ALE control point numbers. The example below deletes all elements outside of a box with two endpoints at (8.0,8.0,8.0) and (15.0,15.0,15.0) in S-ALE control points.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
```

```
         1         1    200001    200001
$      nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$     mshid    option      oper    ioutin     boxid
         1    BOXCPT                 0         1
*DEFINE_BOX
$     boxid       xmn       xmx       ymn       ymx       zmn       zmx
         1         8        15         8        15         8        15
*END
```

*ALE_STRUCTURED_MESH_VOLUME_FILLING

Purpose: Perform volume filling operations on a structured ALE mesh generated by the *ALE_STRUCTURED_MESH keyword card.

In a typical Structured ALE (S-ALE) simulation, *ALE_STRUCTURED_MESH constructs the S-ALE mesh. The mesh includes a set of hex solid elements and their nodes. *ALE_-STRUCTURED_MULTI-MATERIAL_GROUP is used to specify the multi-materials flowing in that mesh and their material properties.

With the above defined, we have the S-ALE mesh which defines our domain for the ALE simulation, and the materials flowing inside, but we have no idea of how those materials occupy the domain. We have two methods for identifying how the materials occupy the domain:

1. The first is to use *INITIAL_VOLUME_FRACTION. With this keyword, we list elements one by one and then for each AMMG in each element, its volume fraction. This approach is tedious and impractical, so it is rarely used.

2. The second way is to fill the volume automatically based on instructions given with *ALE_STRUCTURED_MESH_VOLUME_FILLING. Each instance of *ALE_STRUCTURED_MESH_VOLUME_FILLING with its data cards represents an instruction. Those instructions are executed consecutively, based on the order of appearance in the input file. Each instruction fills the volume either inside or outside of a geometry type (field GEOM in Card 2) with a certain AMMG.

Each *ALE_STRUCTURED_MESH_VOLUME_FILLING keyword has two data cards. Multiple *ALE_STRUCTURED_MESH_VOLUME_FILLING keywords can be defined and are executed in the order of appearance. Please see the examples below.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID		AMMGTO		NSAMPLE			VID
Type	I		A		I			I
Default	0		0		3			none

Card 2	1	2	3	4	5	6	7	8
Variable	GEOM	IN/OUT	E1	E2	E3	E4	E5	
Type	A	I	I or F	I or F	I or F	I or F	I or F	
Default	none	0	none	none	none	none	none	

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. A unique number must be specified.
AMMGTO	AMMG name (AMMGNM on *ALE_STRUCTURED_MULTI-MATERIAL_GROUP) of the ALE multi-material group filling the geometry. See *ALE_STRUCTURED_MULTI-MATERIAL_GROUP for reference.
NSAMPLE	Number of sampling points. In case an element is partially filled, in each direction, 2 × NSAMPLE + 1 points are generated. These $(2 \times \text{NSAMPLE} + 1)^3$ points, each representing a volume, are used to determine if its volume is in or out.
VID	ID of *DEFINE_VECTOR card. This flag is used to assign an initial velocity to the material filling the domain. Fields 2 to 5 (XT, YT, ZT) of the *DEFINE_VECTOR card are used to define the initial translational velocities. Please refer to Example 1 below for an example of usage.
GEOM	Geometry types. They are: ALL, PARTSET, PART, SEGSET, SLDSET, SHLSET, BEAMSET, PLANE, CYLINDER, BOXCOR, BOXCPT, and ELLIPSOID. Refer to the table below for more information.
IN/OUT	To fill inside or outside of the geometry. For the PARTSET, PART, SEGSET, SLDSET, SHLSET, and BEAMSET options, inside is taken as in the normal direction of the container's segments (see Remark 1). EQ.0: Inside (default) EQ.1: Outside
E*i*	These values have different definitions for different options. See the table below for details.

The "GEOM" column in the table below enumerates the allowed values for the geometry variable, as well as describing E1, …, E5 for each geometry type. Each of the following operations accepts up to 5 arguments but may take fewer. Values of E*n* left unspecified are ignored.

GEOM	DESCRIPTION
ALL	Fill all the volume inside the mesh. No additional variables are needed.
PARTSET	Geometry defined by a part set. E1 is a shell, thick shell, or solid element part set ID. E2 is the offset from the geometry. See Remarks 2, 2, 5, 6, and 7.
PART	Geometry defined by a part. E1 is a shell, thick shell, or solid element part ID. E2 is the offset from the geometry. See Remark 2, 2, 5, 6, and 7.
SEGSET	Geometry defined by a segment set. E1 is a segment set ID. E2 is the offset from the geometry. See Remarks 2 and 4.
SLDSET	Geometry defined by a solid element set. E1 is a solid set ID. E2 is the offset from the geometry. See Remarks 1 and 2.
SHLSET	Geometry defined by a shell set. E1 is a shell set ID. E2 is the offset from the geometry. See Remarks 2, 5, and 6.
BEAMSET	Geometry defined by a beam set. E1 is a beam set ID. E2 is the offset from the geometry. See Remarks 2 and 7.
PLANE	Geometry defined by a plane. E1 is the node ID of a node on the plane. E2 is another node ID off the plane. The vector going from node E1 to node E2 is the normal vector of the plane.
CYLINDER	Geometry defined by a cylinder. E1 and E2 are node IDs of the center nodes at the two ends. E3 and E4 are the radii at those two ends.
BOXCOR	Geometry defined by a box. E1 is the BOX ID. See *DEFINE_BOX for details on setting up a box in the global coordinate system or *DEFINE_BOX_LOCAL in the local coordinate system.
BOXCPT	Geometry defined by a box. The box is defined using S-ALE control points (CPT). E1 is BOX ID. See *DEFINE_BOX.

GEOM	DESCRIPTION
ELLIPSOID	Geometry defined by an ellipsoid. E1 is the node ID of the ellipsoid center node. E2, E3, and E4 are the radii along x, y, and z directions, respectively. E5 is the local coordinate system ID if a local coordinate system is used; see *DEFINE_COORDINATE_SYSTEM.

Remarks:

1. **Geometry specified with solid elements.** When solid elements specify the geometry for GEOM = PARTSET, PART, or SLDSET, then their boundaries define the container geometry. The inside/outside direction depends on the normal vectors of the segments on the boundaries of the container.

2. **Offset direction for PART, PARTSET, SEGSET, SLDSET, SHLSET, and BEAMSET.** For GEOM = PART, PARTSET, SEGSET, SLDSET, SHLSET, or BEAMSET, you can specify an offset distance for the fluid interface from the container's mesh. The direction of the offset depends on the normal vectors associated with the mesh of the container (see Remark 1). If the offset specified with E2 is greater than 0, then the offset is along the normal vectors from the boundary of the container. If it is less than 0, then it is offset in the negative normal direction from the boundary of the container.

3. **Files output for subsequent analyses.** This keyword causes file alevfrc.inc and salevfrc.lsda to be output at the end of the simulation. salevfrc.lsda is an lsda version of alevfrc.inc. alevfrc.inc can be included with *INCLUDE for subsequent analyses while salevfrc.lsda can be included with *INITIAL_VOLUME_FRACTION_LSDA. These files pre-fill the volume fractions and store the trimmed mesh which can save time for subsequent analyses.

4. **Geometry specified with segments.** When segments specify the geometry for GEOM = SEGSET, then they define the container geometry. The inside/outside direction and normal vectors depend on the segment node order in *SET_SEGMENT.

5. **Geometry specified with shell elements in 3D.** When shell elements specify the geometry for GEOM = PARTSET, PART, and SHLSET, then they define the container geometry. The inside/outside direction and normal vectors depend on the shell node order in *ELEMENT_SHELLqw (nodal order similar to the ones for *SET_SEGMENT).

6. **Geometry specified with shell elements in 2D.** When shell elements specify the geometry for GEOM = PARTSET, PART, and SHLSET, then their boundaries define the container geometry. The inside/outside direction depends on the normal vectors of the segments on the boundaries of the container.

7. **Geometry specified with beam elements in 2D.** When beam elements specify the geometry for GEOM = PARTSET, PART, and BEAMSET, then they define the container geometry. The inside/outside direction and normal vectors depend on the beam node order in *ELEMENT_BEAM (a beam normal points to the right of a vector from the 1st to 2nd node).

Examples:

1. This example uses two *ALE_STRUCTURED_MESH_VOLUME_FILLING cards. The first fills all volume in a mesh with AMMG 1, outside air, with an AMMGNM "airout". The second fills AMMG 2, inside air, with an AMMGNM "airin" in a spherical domain. We use the same mesh as the example in the *ALE_STRUCTURED_MESH card manual page.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz
      1001      1001      1001
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             airout
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             airin
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID                          0.03                           234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3      flag
       234         2         3         4         1
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
       5   0.1000000e+00   0.1000000e+00   0.1000000e+00
*END
```

Or we could fill every element with AMMG 2 and then switch the domain outside of the sphere to AMMG 1 as follows:

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             airin
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             airout
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID         1                0.03                           234
```

If we want to give the sphere filled in by AMMG 2 an initial velocity, simply define a *DEFINE_VECTOR card as below and assign its ID to field VID.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1              airin
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID                          0.03                           234
*DEFINE_VECTOR
$      vid        xt        yt        zt
         1      100.      -20.       0.0
```

2. In this model, we fill the whole mesh with AMMG 4 (“vacuum”) first. And then fill AMMG 1 (“water1”), AMMG 2 (“water2”), and AMMG 3 (“air”) into 3 containers, each defined by a Lagrangian shell part. Four cards are required to do that.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             vacuum
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water1
$     geom    in/out       pid    offset
      PART                2001
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water2
$     geom    in/out       pid    offset
      PART                2002
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1               air
$     geom    in/out       pid    offset
      PART                2003
```

The above filling by PART cards assumes the shell normal vectors point inward to the container. If the shell normal vectors point outwards, we need to assign the in/out value to 1. Assuming shell part 2003’s normal is pointing outwards, the card will need to be set as:

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1               air
$     geom    in/out       pid    offset
      PART         1      2003
```

3. To use BOXCPT, we define a box using S-ALE control point indices.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water1
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water2
```

```
$      geom     in/out      boxid
    BOXCPT                     1
*DEFINE_BOX
$     boxid        xmn        xmx        ymn        ymx        zmn        zmx
          1          8         15          8         15          8         15
```

*ALE_STRUCTURED_MULTI_DOMAIN_CONTACT

Purpose: Define the coupling between ALE materials in two overlapping, multi-material S-ALE meshes. This command enables "Eulerian contact" between different multi-materials residing in two separate meshes, where the multi-materials in each mesh maintain their own distinct velocity fields.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID1	MSHID2	PFAC	FRIC				
Type	I	I	F	F				
Default	0	0	0.1	0.0				

VARIABLE	DESCRIPTION
MSHID1	Mesh ID of the first S-ALE mesh
MSHID2	Mesh ID of the second S-ALE mesh
PFAC	Penalty factor. PFAC is a scale factor for scaling the estimated stiffness of the coupling system. It is used to compute the coupling forces. GT.0.0: Fraction of estimated critical stiffness LT.0.0: |PFAC| refers to a load curve ID. The curve gives the coupling pressure as a function of the penetration.
FRIC	Friction coefficient. The friction force is evaluated as the normal force multiplied by the friction coefficient. GT.0.0: Constant friction coefficient LT.0.0: Variable friction coefficient defined by a table with ID |FRIC|. The friction coefficient is a function of coupling pressure and relative velocity. The table uses the coupling pressure as the parameter values. See Examples in *ALE_STRUCTURED_FSI.

Remarks:

1. **Model description.** Consider two overlapping S-ALE meshes. Each S-ALE mesh contains two physical materials, such as **a** vacuum and a metal. This

keyword provides **an** "Eulerian contact" **interaction** between the two metals, **as** each resides in a separate S-ALE mesh, **enabling** each metal **to have** its own deformation **history/field**.

2. **Mesh alignment and trimming.** Although the two S-ALE meshes can feature different overall geometries, they must be perfectly coincident within the overlapping domain. A straightforward way to ensure this alignment is to use the same set of *ALE_STRUCTURED_MESH_CONTROL_POINTS across all S-ALE meshes. Each individual mesh can then be trimmed using *ALE_STRUCTURED_MESH_TRIM to reduce its computational domain.

3. **Contact pressure.** Contact pressure is built up in two overlapping S-ALE elements if their combined material fill fraction exceeds 1.0 (penalty formulation).

4. **Materials.** One of the multi-material groups in each mesh must be *MAT_VACUUM. The vacuum multi-material in each mesh occupies the space taken up by the non-vacuum multi-material in the opposite mesh.

5. **Multiple card support.** This keyword supports multiple occurrences. For example, if there are three multi-materials, they can be modeled separately across three different meshes using two contact cards: one to couple AMMG 1 in Mesh 1 to AMMG 2 in Mesh 2 and another to couple AMMG 2 in Mesh 2 to AMMG 3 in Mesh 3.

Example:

Consider an S-ALE multi-material model, where:

- Mesh 1 contains Metal A (AMMG 1, *MAT_NULL) with the remainder of the space filled with vacuum (AMMG 2, *MAT_VACUUM), and
- Mesh 2 contains Metal B (AMMG 3, *MAT_NULL) with the remainder of the space filled with vacuum (AMMG 2, *MAT_VACUUM).

Note that AMMG 2 can be reused here, as it resides in a separate mesh from Mesh 1.

The two meshes align perfectly with each other. When Metal A is given an initial velocity in the positive x-direction, it causes AMMG 1 to come into contact with AMMG 3. Unlike the single-mesh approach, where both metals are forced to deform identically within a mixed cell, this keyword allows each metal to maintain its own independent deformation and velocity fields.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_STRUCTURED_MULTI-MATERIAL_GROUP
$    AMMGNM       MID     EOSID
     metalA         1         1
     vacuum         2
     metalB         3         3
*ALE_STRUCTURED_MULTI_DOMAIN_CONTACT
$    MSHID1    MSHID2      PFAC      FRIC
```

```
        11        12       -11       0.1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*ALE_STRUCTURED_MULTI-MATERIAL_GROUP_{OPTION}

Purpose: Specify the material properties of each ALE multi-material group (AMMG) used by the S-ALE solver. Each AMMG represents one unique "fluid" which flows in the S-ALE mesh.

To set up a typical Structured ALE (S-ALE) simulation, use *ALE_STRUCTURED_MESH, *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, and *ALE_STRUCTURED_-MESH_VOLUME_FILLING. *ALE_STRUCTURED_MESH constructs the S-ALE mesh. The mesh includes a set of hex solid (quad for 2D) elements and their nodes. *ALE_-STRUCTURED_MULTI-MATERIAL_GROUP provides the ALE material properties. *ALE_STRUCTURED_MESH_VOLUME_FILLING fills the domain with these AMMGs.

> **NOTE:** You can use *ALE_MULTI-MATERIAL_GROUP to specify the AMMGs. However, this keyword provides a simpler interface because you specify the material and EOS here instead of needing a separate part ID. Because of this, you do not need a section definition. This keyword uses ELFORM = 11 on *SECTION_SOLID for 3D, ELFORM = 13 on *SECTION_ALE2D for 2D plane strain, ELFORM = 14 or 15 on *SECTION_ALE2D for 2D axisymmetric, ELFORM = 7 on *SECTION_ALE1D for 1D plane strain, ELFORM = 8 on *SECTION_ALE1D for 1D axisymmetric, and ELFORM = -8 on *SECTION_ALE1D for 1D spherical. This keyword also allows you to give each AMMG a name and reference pressure (see Remarks 2 and 3).

Available options include:

<BLANK>

AXISYM

AXIVOL

PLNEPS

1DAXISYM

1DUNIAXE

1DSPHERE

No keyword option (<BLANK>) is for 3D S-ALE simulations. The element formulation is equivalent to ELFORM = 11 on *SECTION_SOLID. The PLNEPS, AXISYM, and AX-IVOL options are for 2D S-ALE simulations to set up plane strain and axisymmetric element formulations. They correspond to ELFORM = 13, 14, and 15 in *SECTION_ALE2D,

respectively. The 1DUNIAXE, 1DAXISYM, and 1DSPHERE options are for 1D S-ALE simulations to set up 1D plane strain, 1D axisymmetric and 1D spheric element formulations. They correspond to ELFORM = 7, 8, and -8 in *SECTION_ALE1D, respectively.

Card 1	1	2	3	4	5	6	7	8
Variable	AMMGNM	MID	EOSID					PREF
Type	A	I	I					F
Default	none	none	none					0.

VARIABLE	DESCRIPTION
AMMGNM	AMMG name. This name is required to identify the AMMG (S-ALE fluid). It is not case-sensitive, but it needs to be unique. See Remark 2.
MID	Material ID
EOSID	Equation-of-state ID
PREF	Reference pressure of this AMMG. See Remark 3.

Remarks:

1. **AMMGID.** Each AMMG is automatically assigned an ID (AMMGID) according to its order of appearance in the input deck. The AMMG defined in the first line has an AMMGID = 1, the second line AMMGID = 2, and so on and so forth. This ID later could be used in other keyword cards, such as *SET_MULTI-MATERIAL_GROUP. The general ALE solver uses this convention which is inherited by the S-ALE solver.

2. **AMMG name.** Traditionally keywords refer to ALE multi-materials with their AMMGIDs. This ID is automatically assigned based on the order of appearance of each AMMG as discussed in Remark 1. This arrangement works in most cases but has its problems. For example, if you want to switch the order of the AMMGs or want to add or remove certain AMMGs, all references to those AMMGs are affected and need to be modified/corrected. Fixing those references is often forgotten, leading to errors.

   The name specified with AMMGNM on this keyword provides an alternative way to refer to AMMGs. Other ALE keyword cards, such as *SET_MULTI-MA-

TERIAL_GROUP_LIST, accept it in place of AMMGID for S-ALE simulations. For example, suppose the third AMMG defined with *ALE_STRUCTURED_-MULTI-MATERIAL_GROUP has AMMGNM = air. In the other keywords, we could either input 3 or air where it asks for AMMGID.

We recommend using AMMGNM over AMMGID as it is more user-friendly and less error-prone. We also suggest using AMMGNM in a consistent manner, meaning using AMMGNM across all ALE keywords in the input deck. Mixed usage of AMMGNM and AMMGID does not affect the run, but LS-PrePost does not support it.

3. **PREF.** PREF allows you to provide different reference pressures for each AMMG. With this, we can handle mismatched reference pressures among different AMMGs.

   Each ALE material definition (*MAT +*EOS) has an implicit reference pressure. Different AMMGs in the same model are generally required to have the same reference pressure. For example, assume we have two AMMGs, air and water. For the *EOS, we need to carefully choose E0 and V0 values so that their initial pressure matches this reference pressure, most likely atmospheric pressure.

   Sometimes mismatch could happen. For example, air and high explosives may have different reference pressures. Air may have a reference pressure of 1 bar, but the high explosive defined with *EOS_JWL may have a reference pressure of 0 bar. Before PREF, this discrepancy was considered small and ignored. PREF resolves this discrepancy by allowing us to input 101325.0 as PREF for air and 0.0 as PREF for high explosive. Then, when force is evaluated for an element, this PREF pressure will be subtracted off from that of the corresponding material.

*ALE_STRUCTURED_POINT_SOURCE

Purpose: Specify an inlet gas injection boundary condition for a multiple-gas mixture inflow using a set of point source(s) in a structured ALE (S-ALE) mesh.

For airbag simulations using the S-ALE method, please refer to *AIRBAG_SALE, a wrapper card internally translated into several S-ALE keywords including *ALE_STRUCTURED_POINT_SOURCE during the keyword read-in phase.

Card 1	1	2	3	4	5	6	7	8
Variable	AMMGNM	LCVEL	LCT	NORIF				
Type	A	I	I	I				
Default	none	0	none	none				

Gas Component Mass Flow Rate Card. Include as many instantiations of this card as needed to define the mass flow rates of all the gas components.

Card 2	1	2	3	4	5	6	7	8
Variable	LCM1	LCM2	LCM3	LCM4	LCM5	LCM6	LCM7	LCM8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Orifice Cards. Include NORIF orifice cards; one for each orifice.

Card 3	1	2	3	4	5	6	7	8
Variable	NID*i*	VD*i*	AN*i*					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
AMMGNM	ALE multi-material group (AMMG) name. This specifies the AMMG of the inlet gas. See *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, specifically Remark 2 of that keyword.
LCVEL	Load curve ID for a curve giving the gas inlet velocity as a function of time. See Remark 1.
LCT	Load curve ID for a curve giving the gas inlet temperature as a function of time.
NORIF	Number of orifices
LCM*i*	Load curve ID for the mass flow rate of each gas component. See Remark 2.
NID*i*	Node ID defining the location of nozzle *i*
VD*i*	Vector ID defining of gas inflow direction at nozzle *i*
AN*i*	Area of nozzle *i*. See Remark 3.

Remarks:

1. **LCVEL.** The inlet velocity load curve is optional. If zero, a virtual tank test using the gas inflow data is performed during the initialization phase.

2. **Mass flow rate.** The gas mixture is assumed to have a uniform temperature ($\overline{T} \approx T_i$) and inlet velocity. However, the species in the mixture may each have a different inlet mass flow rate.

3. **AN*i*.** The orifice area is a mandatory input as its value is used in the following calculations:

   a) Making an estimate of the inflator volume. A virtual tank test is performed before the simulation to obtain an estimated gas velocity curve if LCVEL is not prescribed.

   b) Calculating the injected gas volume in a given time cycle. The gas injected in a cycle pushes the gases away and creates a volume for itself. That volume is calculated as the inlet velocity times orifice area times time step.

   c) Distributing the inflow gas between different nozzles. AN*i* divided by the sum of all nozzle areas scales the inflow gas flow rate to calculate the amount of gas flowing through nozzle *i*.

*ALE_SWITCH_MMG

Purpose: This card changes a fraction of an ALE multi-material-group (AMMGID) into another group. The fraction is to be specified by a *DEFINE_FUNCTION function. The function take as many arguments as there are fields specified on the cards in format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	IDFUNC	IDSEGSET	IDSLDSET	NCYCSEG	NCYCSLD	
Type	I	I	I	I	I	I	I	
Default	none	none	none	0	0	50	50	

Variable Cards. Cards defining the function arguments. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMG-SID before the switch. The AMMG-SID corresponds to the SID defined on a *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID refers to one or more AMMGs. See Remark 1.
TO_MMG	This is the AMMG-SID after the switch. The AMMG-SID corresponds to the SID defined on a *SET_MULTI-MATERIAL_GROUP_LIST card. This SID refers to one or more AMMGs. See Remark 1.
IDFUNC	ID of a *DEFINE_FUNCTION function. This function determines the material fraction to be switched. See Example 1.

VARIABLE	DESCRIPTION
IDSEGSET	ID of *SEGMENT_SET that is used to pass geometric properties to the function specified by IDFUNC. *This field is optional*. LS-DYNA computes the segment center positions and normal vectors. For each ALE element, this data can be passed to the function IDFUNC for the segment that is closest to the element center. See Example 2.
IDSLDSET	The ID of a *SOLID_SET specifying which elements are affected by this particular instance of the *ALE_SWITCH_MMG keyword. *This field is optional*. If undefined, *ALE_SWITCH_MMG affects all ALE elements. The element centers are computed and can be used as variables in the function IDFUNC.
NCYCSEG	Number of cycles between each update of the segment centers and normal vectors (if a segment set is defined). For each update, a bucket sort is applied to find the closest segment to each ALE element. If the segment nodes are fully constrained, the segment centers and normal vectors are computed only one time.
NCYCSLD	Number of cycles between each update of the ALE element centers. For each update, a bucket sort is applied to find the closest segment to each ALE element. If the element nodes do not move (as with AFAC = -1 in *CONTROL_ALE), the element centers are computed exactly once.
VAR*i*	Variables used by IDFUNC which must be in the argument order (See Remark 2): EQ.0: See Remark 3. EQ.1: *xx*-stress for FR_MMG EQ.2: *yy*-stress for FR_MMG EQ.3: *zz*-stress for FR_MMG EQ.4: *xy*-stress for FR_MMG EQ.5: *yz*-stress for FR_MMG EQ.6: *zx*-stress for FR_MMG EQ.7: Plastic strain for FR_MMG EQ.8: Internal energy for FR_MMG EQ.9: Bulk viscosity for FR_MMG

VARIABLE	DESCRIPTION	
	EQ.10:	Volume from previous cycle for FR_MMG
	GE.11.and.LE.20:	Other auxiliary variables for FR_MMG
	GE.21.and.LE.40:	Auxiliary variables for TO_MMG (*xx*-stress, …)
	EQ.41:	Mass for FR_MMG
	EQ.42:	Mass for TO_MMG
	EQ.43:	Volume fraction for FR_MMG
	EQ.44:	Volume fraction for TO_MMG
	EQ.45:	Material volume for FR_MMG
	EQ.46:	Material volume for TO_MMG
	EQ.47:	Time
	EQ.48:	Cycle
	EQ.49:	*x*-position of the ALE element center
	EQ.50:	*y*-position of the ALE element center
	EQ.51:	*z*-position of the ALE element center
	EQ.52:	*x*-position of the segment center
	EQ.53:	*y*-position of the segment center
	EQ.54:	*z*-position of the segment center
	EQ.55:	*x*-component of the segment normal
	EQ.56:	*y*-component of the segment normal
	EQ.57:	*z*-component of the segment normal
	GE.58.and.LE.65:	*x*-positions of the ALE nodes
	GE.66.and.LE.69:	*x*-positions of the segment nodes
	GE.70.and.LE.77:	*y*-positions of the ALE nodes
	GE.79.and.LE.81:	*y*-positions of the segment nodes
	GE.83.and.LE.89:	*z*-positions of the ALE nodes
	GE.90.and.LE.93:	*z*-positions of the segment nodes
	GE.94.and.LE.101:	*x*-velocities of the ALE nodes
	GE.102.and.LE.105:	*x*-velocities of the segment nodes
	GE.106.and.LE.113:	*y*-velocities of the ALE nodes
	GE.114.and.LE.117:	*y*-velocities of the segment nodes

VARIABLE	DESCRIPTION
	GE.118.and.LE.125: z-velocities of the ALE nodes
	GE.126.and.LE.129: z-velocities of the segment nodes
	GE.130.and.LE.137: x-accelerations of the ALE nodes
	GE.138.and.LE.141: x-accelerations of the segment nodes
	GE.142.and.LE.149: y-accelerations of the ALE nodes
	GE.150.and.LE.153: y-accelerations of the segment nodes
	GE.154.and.LE.161: z-accelerations of the ALE nodes
	GE.162.and.LE.165: z-accelerations of the segment nodes
	GE.166.and.LE.173: Masses of the ALE nodes
	GE.174.and.LE.177: Masses of the segment nodes
	EQ.178: Position of the variable updated by the function (See Remark 4)
	EQ.179: Index of the multi-material group in the set
	EQ.180: Time step

Remarks:

1. **Mapping.** The multi-material group sets that are specified by the fields FR_MMG and TO_MMG must be of the same length. Multi-material groups are switched so that, for instance, the fourth multi-material group in the set FR_MMG is mapped to the fourth multi-material group in the set TO_MMG. (see Example 1).

2. **Variable Specification.** The variables are presented to the function IDFUNC as floating point data. The order of the arguments appearing in the *DEFINE_FUNCTION should match the order of the variables, VAR*i*, specified on Card 2 (for this keyword). For example, when there is one card in format 2 containing "47, 48, 41, 42", then the time (47), the cycle (48), and the masses (41 & 42) should be the first, second, third, and fourth arguments to the function defined on the *DEFINE_FUNCTION keyword.

   If there is a blank column between 2 variables, the list between these 2 ranks is specified. For example, if the card contains "1, ,6", then the 6 stresses (1 through 6) are selected as arguments (see Example 2). In the case that there are several groups in the sets, if a variable is repeated, the corresponding variable will be defined in the function for the same number of groups as variable instances. For instance, if the sets have 3 groups and the volume fractions of the 2 first groups

in the set TO_MMG are required as arguments of the function, a card in format 2 should have "44,44".

3. **Variable Update for Several Groups.** If there is more than one group in the set, the function is called for each group. For a given group that has an index in the set > 1 (VAR*i* = 179), some variables, including the volume fraction, mass, and internal energy, may have been updated during the previous switches. If their original values are required, they can be obtained by setting the first field (VAR1) to 0.

4. **Variable Update by the User.** The variables can be updated by the user. If VAR*i* < 0 for some variables, the function is called again (after the switch) for each of these variables. VAR*i* = 178 gives the position of the variable for which the function is called. The function's return value is taken as the new value for this variable (instead of the fraction of material to switch). If the position given by VAR*i* = 178 is zero, it means that the function is called for the switch. Only the 46 first variables 1 < VAR*i* < 46, 58 < VAR*i* < 165 and VAR*i* = 180 can be modified.

Example 1:

The first example switches the material if the pressure is lower than a given value.

```
*comment
units: mks

Switch from the 3rd group to the 5th one if the pressure of the 3rd group
is lower than pc : pres < pc
Do the same for the switch from 4th to 7th
If the switch occurs, the function frac returns 1.0. So the whole
material is permuted.

xxsig : xx-stress of the groups in the 1st *set_multi-material_group_list
yysig : yy-stress of the groups in the 1st *set_multi-material_group_list
zzsig : zz-stress of the groups in the 1st *set_multi-material_group_list
pres  : pressure
pc    : pressure cutoff
*ALE_SWITCH_MMG
$#  fr_mmg    to_mmg    idfunc  idsegset  idsldset   ncycseg   ncycsld
         1         2        10
         1         2         3
*set_multi-material_group_list
1
3,4
*set_multi-material_group_list
2
5,7
*DEFINE_FUNCTION
10
float frac(float xxsig, float yysig, float zzsig)
{
  float pc;
  pres=-(xxsig+yysig+zzsig)/3.0;
```

```
  pc=-1000;
  if (pres<pc) {
    return 1.0;
  } else {
    return 0.0;
  }
 }
```

Example 2:

The second example switches the material if it goes through a segment.

```
*comment
units: mks

Switch the 1st group to the 2nd group if the ALE element center goes
through a segment of the set defined by idsegset=1.
The segment position is updated every cycle
A fraction of the material is switched.  This fraction depends on the
distance between the segment and element centers

time    : 47th variable
cycle   : 48th variable
xsld    : 49th variable (x-position of the element center)
ysld    : 50th variable (y-position of the element center)
zsld    : 51th variable (z-position of the element center)
xseg    : 52th variable (x-position of the segment center)
yseg    : 53th variable (y-position of the segment center)
zseg    : 54th variable (z-position of the segment center)
xn      : 55th variable (x-component of the segment normal)
yn      : 56th variable (y-component of the segment normal)
zn      : 57th variable (z-component of the segment normal)
volmat1 : 43th variable (material volume of the 1st group)
volfrac1: 45th variable (volume fraction of the 1st group)
segsurf : segment surface (given by 0.5*sqrt(xn*xn+yn*yn+zn*zn))
sldvol  : ALE element volume (given by volmat1/volfrac1)
segcharaclen: characteristic length for the segment
sldcharaclen: characteristic length for the solid
xseg2sld: x-component of the vector segment center to element center
yseg2sld: y-component of the vector segment center to element center
zseg2sld: z-component of the vector segment center to element center
distnormseg2sld: Distance segment-element projected on the normal
disttangseg2sld: Distance segment-element projected on the segment plane
*ALE_SWITCH_MMG
$#  fr_mmg    to_mmg    idfunc  idsegset  idsldset   ncycseg   ncycsld
         1         2        11         1                  1
        47                  57        43        45
*set_multi-material_group_list
1
1
*set_multi-material_group_list
2
2
*DEFINE_FUNCTION
11
float switchmmg(float time, float cycle,
                float xsld, float ysld, float zsld,
                float xseg, float yseg, float zseg,
```

```
                float xn, float yn,  float zn,
                float volmat1, float volfrac1)
{
  float segsurf, sldvol, segcharaclen, sldcharaclen;
  float xseg2sld, yseg2sld, zseg2sld, distnormseg2sld;
  float xtangseg2sld, ytangseg2sld, ztangseg2sld, disttangseg2sld;
  float frac;
  segsurf=sqrt(xn*xn+yn*yn+zn*zn);
  if (segsurf != 0.0) {
    xn=xn/segsurf;
    yn=yn/segsurf;
    zn=zn/segsurf;
  }
  segsurf=0.5*segsurf;
  sldvol=volmat1/volfrac1;
  segcharaclen=0.5*sqrt(segsurf);
  sldcharaclen=0.5*sldvol**(1.0/3.0);
  xseg2sld=xsld-xseg;
  yseg2sld=ysld-yseg;
  zseg2sld=zsld-zseg;
  distnormseg2sld=xseg2sld*xn+yseg2sld*yn+zseg2sld*zn;
  xtangseg2sld=xseg2sld-distnormseg2sld*xn;
  ytangseg2sld=yseg2sld-distnormseg2sld*yn;
  ztangseg2sld=zseg2sld-distnormseg2sld*zn;
  disttangseg2sld=xtangseg2sld*xtangseg2sld+
                  ytangseg2sld*ytangseg2sld+
                  ztangseg2sld*ztangseg2sld;
  disttangseg2sld=sqrt(disttangseg2sld);
  if (disttangseg2sld <= segcharaclen &&
      distnormseg2sld <= sldcharaclen) {
    sldcharaclen=2.0*sldcharaclen;
    frac=distnormseg2sld/sldcharaclen;
    frac=0.5-frac;
    return frac;
  } else {
    return 0.0;
  }
}
```

*ALE_TANK_TEST

Purpose: Control volume airbags (*AIRBAG_) only require two engineering curves to define the gas inflator, i.e. $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$; those two curves can be experimentally measured. However, the ALE inflator needs one additional state variable - the inlet gas velocity which is impractical to obtain. This keyword provides such a curve through an engineering approximation.

It takes two curves from the accompanying *SECTION_POINT_SOURCE as input. It assumes inflator gas under choking condition to generate the velocity curve. During this process, the original curves, $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, are modified accordingly.

This keyword complements and must be used together with the*SECTION_POINT_-SOURCE command. Please see *SECTION_POINT_SOURCE for additional information.

Card 1	1	2	3	4	5	6	7	8
Variable	MDOTLC	TANKV	PAMB	PFINAL	MACHL	VELMAX	AORIF	
Type	I	I	I	I	F	F	F	
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 2	1	2	3	4	5	6	7	8
Variable	AMGIDG	AMGIDA	NUMPNT					
Type	I	I	I					
Default	0	0	50					

VARIABLE	DESCRIPTION
MDOTLC	LCID for mass flow rate as a function of time. This may be obtained directly from the control-volume type input data.
TANKV	Volume of the tank used in a tank test from which the tank pressure is measured; $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ are computed from this tank pressure data.

VARIABLE	DESCRIPTION
PAMB	The pressure inside the tank before jetting (usually 1 bar).
PFINAL	The final equilibrated pressure inside the tank from the tank test.
MACHL	A limiting MACH number for the gas at the throat (MACH = 1 preferred).
VELMAX	Maximum allowable gas velocity across the inflator orifice (not preferred).
AORIF	Total inflator orifice area (optional, only needed if the *SECTION_POINT_SOURCE card is not used).
AMGIDG	The ALE multi-material group ID (AMMGID) of the gas.
AMGIDA	The ALE multi-material group ID (AMMGID) of the air.
NUMPNT	The number of points in $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ curves. If NUMPNT = 0, defaults to 50 points.

Remarks:

In an airbag inflator tank test, the tank pressure data is measured. This pressure is used to derive $\dot{m}(t)$ and to estimate $\overline{T}_{\text{gas}}(t)$, the stagnation temperature of the inflator gas. This is done by applying a lumped-parameter method to the system of conservation equations using an equation of state.

Together $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ provide enough information to model an airbag with the control volume method (see *AIRBAG cards). However, for an ALE or Eulerian fluid-structure interaction analysis, the gas velocity, $v(t)$, and density, $\rho(t)$, at the inlet must be computed. But, since only $\dot{m}(t)$ is known, additional assumptions must be made about the inlet conditions. If $v(t)$ and $\rho(t)$ are calculated outside of LS-DYNA, then LS-DYNA combines them with $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ to obtain $\overline{T}_{\text{gas corrected}}(t)$, $v(t)$ and $\rho(t)$ which are sufficient input for an ALE calculation.

The curves $v(t)$ and $\rho(t)$ need not be calculated outside of LS-DYNA as LS-DYNA features a method for calculating them itself. This card, *ALE_TANK_TEST, activates this capability. Thus, with the combination of this card and the *SECTION_POINT_SOURCE card, LS-DYNA can proceed directly from the control volume method input, $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, to an ALE or Eulerian fluid-structure interaction analysis. The user does not have to do the conversion himself.

If the *ALE_TANK_TEST card is present:

1. The definitions of the relative volume, $V(t)$, and the velocity, $v(t)$, curves in the *SECTION_POINT_SOURCE card will be ignored in favor of those computed by LS-DYNA.

2. The $\dot{m}(t)$curve is read in on *ALE_TANK_TEST card.

3. The $\overline{T}_{\text{gas}}(t)$ curve (stagnation temperature), as opposed to $\overline{T}_{\text{gas corrected}}(t)$, is read in on *SECTION_POINT_SOURCE card.

   There is a subtle, but important, distinction between the two temperatures. $\overline{T}_{\text{gas}}(t)$ is derived directly from the tank pressure data based on a lump-parameter approach, whereas $\overline{T}_{\text{gas corrected}}(t)$ is computed from $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ with additional isentropic and sonic flow assumptions for the maximum velocity at an orifice. $\overline{T}_{\text{gas corrected}}(t)$ is most appropriately interpreted as the static temperature. These assumptions provide a necessary and physically reasonable supplement to the governing equation,

   $$\dot{m}(t) = \rho(t)v(t)A$$

   in which only $\dot{m}(t)$ and A are known, leaving two parameters, $\rho(t)$ and $v(t)$, as unkown.

4. The inflator area is computed from the *SECTION_POINT_SOURCE card that has the AMMGID of the inflator gas in the *ALE_TANK_TEST card. If the *BOUNDARY_AMBIENT_EOS card is used instead of the *SECTION_POINT_-SOURCE card, then the area may be input in this *ALE_TANK_TEST card.

5. The reference density of the propellant "gas," ρ_0, is computed internally and automatically used for the calculation. The ρ_0 value from the *MAT_NULL card is ignored.

Example:

Consider a tank test model that consists of the inflator gas (PID 1) and the air inside the tank (PID 2). The following information from the control volume model is available:

- $\dot{m}(t)$ (LCID 1 is from control volume model input).
- $\overline{T}_{\text{gas}}(t)$ (LCID 2 is from control volume model input).
- Volume of the tank used in the inflator tank test.
- Final equilibrated pressure inside the tank.
- Ambient pressure in the air.

Also available are:

- The nodal IDs of the nodes defining the orifice holes through which the gas flows into the tank.
- The area associated with each hole (the node is assumed to be at the center of this area).
- The vector associated with each hole defining the direction of flow.

In the input below LCID 1 and 2 are $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, respectively. LCID 4 and 5 will be ignored when the *ALE_TANK_TEST card is present. If it is not present, all 3 curves in the *SECTION_POINT_SOURCE card will be used. When the *SECTION_POINT_-SOURCE card is present, the element formulation is equivalent to an ELFORM = 11.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*PART
inflator gas
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1         0         0         0         0         0
*PART
air inside the tank
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         2         2         2         0         0         0         0         0
*SECTION_SOLID
$    SECID    ELFORM       AET
         2        11         0
*ALE_MULTI-MATERIAL_GROUP
$      SID   SIDTYPE
         1         1
         2         1
*SECTION_POINT_SOURCE
$    SECID     LCIDT  LCIDVOLR   LCIDVEL      <= 3 curves in tempvolrvel.k file
         1         2         4         5
$   NODEID    VECTID      AREA
     24485         3    15.066
       ...
     24557         3    15.066
*ALE_TANK_TEST
$   MDOTLC     TANKV      PAMB    PFINAL     MACHL    VELMAX     AORIF
         1     6.0E7    1.0E-4  5.288E-4       1.0       0.0
$   AMGIDG    AMGIDA    NUMPNT
         1         2        80
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*ALE_UP_SWITCH

Purpose: For the simulation of airbag inflation process, this card allows the switching from an ALE or S-ALE computation to a control volume (CV) or uniform pressure (UP) method at a user-defined switch time.

Card 1	1	2	3	4	5	6	7	8
Variable	UPID	SWTIME						
Type	I	F						
Default	0	10^{16}						

Card 2	1	2	3	4	5	6	7	8
Variable	FSI_ID1	FSI_ID2	FSI_ID3	FSI_ID4	FSI_ID5	FSI_ID6	FSI_ID7	FSI_ID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Hybrid Card. Include this card if UPID = 0.

Card 3	1	2	3	4	5	6	7	8
Variable	SID	SIDTYPE	MMGAIR	MMGGAS				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
UPID	ID corresponding to an instantiation of *AIRBAG_HYBRID_ID card for use in an ALE or S-ALE method switching to CV method simulation. The simulation starts with the ALE or S-ALE computational method and then switches to a CV (or UP) method at some given time.

VARIABLE	DESCRIPTION
	EQ.0: The code constructs an equivalent *AIRBAG_HYBRID_ID card automatically internally (default). Card 3 is then a required input. NE.0: An ID points to a corresponding *AIRBAG_HYBRID_ID card which must be defined for use after the switch. If UPID is defined, do not define Card 3.
SWTIME	The time at which the computation does a switch from an ALE or S-ALE method to CV method.
FSI_ID1, …, FSI_ID8	Coupling IDs for one or more ALE fluid-structure-interaction (FSI) *CONSTRAINED_LAGRANGE_IN_SOLID_ID cards or S-ALE FSI *ALE_STRUCTURED_FSI_TITLE cards. These couplings are deleted during the CV computational phase.
SID	A set ID that defines the Lagrangian parts which make up the airbag
SIDTYPE	Set ID type for the above SETID (following the conventions in *AIRBAG_HYBRID card). EQ.0: SID is a segment set ID. NE.0: SID is a part set ID.
MMGAIR	The AMMG (ALE multi-material group) ID of the surrounding air
MMGGAS	The AMMG ID of inflator gas injected into the airbag

Example 1:

Consider an airbag model with a 2-phase simulation: an ALE calculation being switched to a CV method. During the CV phase, the simulation is defined by an *AIRBAG_HYBRID_ID card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_UP_SWITCH
$    UP_ID   SW_time
    100000    2.0000
$ FSI_ID_1  FSI_ID_2  FSI_ID_3  FSI_ID_4  FSI_ID_5  FSI_ID_6  FSI_ID_7  FSI_ID_8
         1         2
$--------------------------------------------------------------------------------
*AIRBAG_HYBRID_ID
$       ID
    100000
$      SID    SIDTYP      RBID      VSCA      PSCA      VINI       MWD      SPSF
         2         1         0       1.0       1.0       0.0       0.0       0.0
$ 2   ATMT      ATMP      ATMD        GC        CC
      293. 1.0130e-4  1.200E-9    8.3143        1.
```

```
$       C23     LCC23       A23     LCA23       CP23     LCP23       AP23    LCAP23

$       OPT     PVENT       NGAS
                                4
$bac LCIDM      LCIDT   NOTUSED         MW     INITM         A         B         C
      1001       1002            0.0288691       1.0     28.98
$    FMASS

$air LCIDM      LCIDT   NOTUSED         MW     INITM         A         B         C
      1600       1603            28.97E-3       0.0     26.38  8.178e-3 -1.612e-6
$    FMASS

$pyroLCIDM      LCIDT   NOTUSED         MW     INITM         A         B         C
      1601       1603            43.45E-3       0.0     32.87  2.127e-2 -5.193E-6
$    FMASS

$sto_LCIDM      LCIDT   NOTUSED         MW     INITM         A         B         C
      1602       1603            39.49E-3       0.0     22.41  2.865e-3 -6.995e-7
$    FMASS
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

Example 2:

Consider the same airbag model with the same 2-phase simulation. However, all the *AIRBAG_HYBRID_ID card definitions are extracted automatically from the ALE model. There is no need to define the *AIRBAG_HYBRID_ID card. Card 3 is required.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_UP_SWITCH
$    UP_ID   SW_time
$   100000    2.0000
          0    2.0000
$ FSI_ID_1  FSI_ID_2  FSI_ID_3  FSI_ID_4  FSI_ID_5  FSI_ID_6  FSI_ID_7  FSI_ID_8
          1         2
$    SETID    SETYPE   MMG_AIR   MMG_GAS
          2         1         2         1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*BOUNDARY

The keyword *BOUNDARY provides a way of defining imposed motions on boundary nodes. The keyword control cards in this section are defined in alphabetical order:

BOUNDARY_ACOUSTIC_COUPLING_*{OPTION}

*BOUNDARY_ACOUSTIC_COUPLING_SPECTRAL

*BOUNDARY_ACOUSTIC_FREE_SURFACE

*BOUNDARY_ACOUSTIC_IMPEDANCE

*BOUNDARY_ACOUSTIC_IMPEDANCE_COMPLEX

*BOUNDARY_ACOUSTIC_IMPEDANCE_MECHANICAL

*BOUNDARY_ACOUSTIC_INTERFACE

*BOUNDARY_ACOUSTIC_MAPPING

*BOUNDARY_ACOUSTIC_NON_REFLECTING

*BOUNDARY_ACOUSTIC_PRESCRIBED_MOTION

*BOUNDARY_ACOUSTIC_PRESSURE_SPECTRAL

*BOUNDARY_ALE_MAPPING

*BOUNDARY_AMBIENT

*BOUNDARY_AMBIENT_EOS

BOUNDARY_CONVECTION_*OPTION

*BOUNDARY_COUPLED

*BOUNDARY_CYCLIC

*BOUNDARY_DE_NON_REFLECTING

*BOUNDARY_FLUIDM

*BOUNDARY_FLUIDM_BOTTOM

*BOUNDARY_FLUIDM_FREE_SURFACE

*BOUNDARY_FLUIDM_INTERIOR

*BOUNDARY

BOUNDARY_FLUX_*OPTION

*BOUNDARY_FLUX_TRAJECTORY

*BOUNDARY_MCOL

*BOUNDARY_NON_REFLECTING

*BOUNDARY_NON_REFLECTING_2D

*BOUNDARY_PAP

BOUNDARY_PORE_FLUID_*OPTION

*BOUNDARY_PRECRACK

*BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID

*BOUNDARY_PRESCRIBED_FINAL_GEOMETRY

BOUNDARY_PRESCRIBED_MOTION_*OPTION

BOUNDARY_PRESCRIBED_ORIENTATION_RIGID_*OPTION

BOUNDARY_PRESSURE_OUTFLOW_*OPTION

BOUNDARY_PWP_*OPTION

*BOUNDARY_PZEPOT

BOUNDARY_RADIATION_*OPTION

*BOUNDARY_RADIATION_ENCLOSURE

*BOUNDARY_SALE_MESH_FACE

*BOUNDARY_SLIDING_PLANE

BOUNDARY_SPC_*OPTION

*BOUNDARY_SPC_SYMMETRY_PLANE

*BOUNDARY_SPH_FLOW

*BOUNDARY_SPH_HEAT_CONVECTION

*BOUNDARY_SPH_NON_REFLECTING

*BOUNDARY_SPH_NOSLIP

*BOUNDARY_SPH_SYMMETRY_PLANE

*BOUNDARY_SPH_TRACK_SURFACE

*BOUNDARY_SYMMETRY_FAILURE

BOUNDARY_TEMPERATURE_*OPTION

*BOUNDARY_TEMPERATURE_PERIODIC_SET

*BOUNDARY_TEMPERATURE_RSW

*BOUNDARY_TEMPERATURE_TRAJECTORY

*BOUNDARY_THERMAL_BULKFLOW

*BOUNDARY_THERMAL_BULKNODE

*BOUNDARY_THERMAL_WELD

*BOUNDARY_THERMAL_WELD_TRAJECTORY

*BOUNDARY_USA_SURFACE

*BOUNDARY_ACOUSTIC_COUPLING_{OPTION}

Purpose: This keyword provides fluid-structure coupling when the fluid is modeled with *MAT_ACOUSTIC and solid formulation 8 or 14, *and* the nodes of the fluid at the fluid-structure interface are not merged/shared with nodes of the structure.

If acoustic fluid exists on one side of a structural surface *and* the nodes of the fluid and structure are merged/shared at the interface, this keyword is not required. Coupling between an acoustic fluid and a structural surface occurs automatically when nodes are shared at the interface.

Available options for the keyword are:

<BLANK>

MISMATCH

If the nodes of the fluid are not coincident with the nodes of the structure at the fluid-structure interface, the MISMATCH option must be used. If the nodes are coincident, the <BLANK> option is sufficient. The input for this keyword is a segment set that defines the structural surface(s) to be coupled to the acoustic fluid.

If acoustic fluid exists on both sides of a structural surface comprised of shells, that is, the shell surface constitutes a physical break between the two fluid masses (call these fluid A and fluid B), fluid A should not share nodes with fluid B. In this case, coupling may be defined (1) wholly through use of this keyword if the structural shells do not share nodes with either fluid, or (2) by fluid A sharing nodes with the structural shells and by using this keyword to couple fluid B to the structural shells. Note that this concept of coupling two fluid masses to a structural surface does not apply to structural solids because a solid face can have direct contact with only one fluid.

If no keyword option is used (<BLANK>), then the nodes of the structural element segments must be coincident with, but not merged to, the nodes of the acoustic element faces. Each structural element face to be coupled, regardless of whether there is fluid on one or both sides of that face, requires only one corresponding segment to be included in the segment set, and the orientation of the segment normal is unimportant.

The MISMATCH option permits the coupling of acoustic fluid elements to a structural surface when the meshes of the fluid and structural elements are moderately mismatched, meaning the nodes of the fluid and structure at the interface are not necessarily coincident. The segments in the segment set should define the structural surface and, following the right-hand rule, the normal vector for the segments should point toward the fluid volume elements with which coupling is intended. If fluid-structure coupling is required on both sides of structural shell elements, duplicate segments with opposite normal vectors should be defined. Every segment in the segment set must couple with the fluid volume at some integration point, but not all integration points on the segment need to

couple with the fluid. The meshes do not have to be mismatched to use the MISMATCH option, as long as the fluid and structural nodes are not merged.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT. This set defines the structural segments being coupled to the acoustic elements.

Remarks:

1. **Stability Condition.** For the stability of the acoustic-structure coupling, the following condition must be satisfied:

$$\frac{2\rho_a D}{\rho_s t_s} < 5 \ ,$$

where ρ_a is the density of the acoustic medium, D is the total thickness of the acoustic elements adjacent to the structural element, ρ_s is the density, and t_s is the thickness of the structural shell element. If the structural element is a solid or thick shell element, then t_s should be half the thickness of the element. If coupling is on both sides of the structural elements, then t_s should also be half the thickness of the structural element.

2. **Mismatched Coupling and Orientation.** In mismatched coupling, free fluid faces are considered for coupling with the structural segments if they are near one another and if they face each other. Faces and segments that differ in orientation by more than 45 degrees are excluded. In regions of high curvature, the surfaces, therefore, need to be more similar than when the surfaces are flat. If a fluid face couples with any structural segment, then all four integration points on the fluid face must couple with some structural segment. Fluid faces may not be partially coupled. Structural segments are allowed to be partially coupled.

3. **Mismatched Coupling Output Files.** The mismatched coupling process dumps two LS-DYNA files that can be imported into LS-PrePost for review of the results of the coupling process. The file bac_str_coupling.dyn contains shell

elements where structural segments have coupled with the fluid and mass elements at structural integration points with coupling. When the messag file indicates that some structural segments have partial coupling, this file can be used to check the unconnected segment integration points. The file bac_flu_coupling.dyn contains shell elements where free fluid faces have coupled with the structural segments and mass elements at free fluid face integration points with coupling. These files are only for visualization of the coupling and serve no other purpose.

4. **Mismatched Coupling Excludes *MAT_ACOUSTIC.** The free surface given on *MAT_090 is excluded from the mismatched coupling logic. If there are multiple fluid regions with different free surfaces, then those regions should be given different material IDs.

*BOUNDARY_ACOUSTIC_COUPLING_SPECTRAL

Purpose: Specify the acoustic and structural surfaces that are to participate in fluid-structure coupling. This keyword is only for coupling the structure with spectral acoustic elements. *CONTROL_ACOUSTIC_SPECTRAL must also be in the input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	SSIDS	SSIDF						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
SSIDS	Segment set ID for the structural faces
SSIDF	Segment set ID for the fluid faces

Remarks:

1. **Segment Orientation.** The normal vectors of the segments in SSIDS and SSIDF should point toward each other.

*BOUNDARY_ACOUSTIC_FREE_SURFACE

Purpose: Assign properties to an acoustic free surface in an SSD or spectral analysis (see *CONTROL_IMPLICIT_SSD_DIRECT and CONTROL_ACOUSTIC_SPECTRAL). The surface may exhibit a zero pressure condition, or for very low frequency response, a small amplitude (linear) wave surface.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	GRAV						
Type	I	F						
Default	none	0.0						

VARIABLE	DESCRIPTION
SSID	Segment set ID of an acoustic surface
GRAV	Acceleration of gravity

Remarks:

1. **Zero Pressure Surface.** If GRAV = 0.0, then the acoustic surface (SSID) exhibits a zero pressure boundary condition.

2. **Small Amplitude Waves.** If GRAV ≠ 0.0, then the acoustic surface exhibits the small amplitude, linear wave condition of sloshing.

*BOUNDARY_ACOUSTIC_IMPEDANCE

Purpose: Define a segment set to prescribe the acoustic impedance of acoustic volume element (type 8 and type 14 solid elements) faces.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	ZEE						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT
ZEE	Value of the acoustic impedance ρc

Remarks:

1. **Stability of Solutions.** The effect of the boundary impedance on the acoustic cavity response is incorporated in the forcing vector. Solutions are conditionally stable; low values of impedance relative to the impedance of the *MAT_ACOUSTIC elements cause instabilities. Reducing the factor of safety on the time step extends the range of applicability; however, it is recommended that pressure release conditions be handled by leaving the boundary free rather than by providing a relatively low boundary acoustic impedance value. If the boundary impedance value is less than 25 percent of the *MAT_ACOUSTIC impedance, LS-DYNA displays a warning. A value less than 1 percent of the *MAT_ACOUSTIC impedance is considered to be an error.

2. **Non-Reflecting Entrant Boundary Condition.** A non-reflecting entrant boundary condition is assumed when on the same segments both *LOAD_SEGMENT sets pressures and *BOUNDARY_ACOUSTIC_IMPEDANCE is defined. The pressures in the LOAD_SEGMENT_SET definition are treated as incoming incident pressure. Pressure waves within the *MAT_ACOUSTIC domain striking this boundary will exit the model. In contrast, a *LOAD_SEGMENT_SET on *MAT_ACOUSTIC volume faces in the absence of *BOUNDARY_ACOUSTIC_IMPEDANCE acts as a time-dependent, total pressure constraint. Pressure waves within the *MAT_ACOUSTIC domain striking this boundary will be reflected back into the model.

*BOUNDARY_ACOUSTIC_IMPEDANCE_COMPLEX

Purpose: Assign complex impedance properties to an acoustic surface in an SSD analysis invoked with *CONTROL_IMPLICIT_SSD_DIRECT.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	ZR	ZI	LCIDR	LCIDI			
Type	I	F	F	I	I			
Default	none	0.0	0.0	0	0			

VARIABLE	DESCRIPTION
SSID	Segment set ID of an acoustic surface
ZR	Real part of the boundary impedance Z_R
ZI	Imaginary part of the boundary impedance Z_I
LCIDR	Frequency dependence of Z_R
LCIDI	Frequency dependence of Z_I

Remarks:

1. **Impedance**. On the boundary, the contribution to the damping matrix is:

$$[\bar{W}_f] = \frac{-Z_r}{(Z_r^2 + Z_i^2)} \int_S N_f^T N_f \, dS + \frac{iZ_i}{(Z_r^2 + Z_i^2)} \int_S N_f^T N_f \, dS \ .$$

2. **Frequency Dependence**. If the frequency dependence is not specified by a load curve, then the impedance is assumed to be constant.

*BOUNDARY_ACOUSTIC_IMPEDANCE_MECHANICAL

Purpose: Assign mechanical impedance properties to an acoustic surface in an SSD analysis invoked with *CONTROL_IMPLICIT_SSD_DIRECT.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	MPAREA	CPAREA	KPAREA				
Type	I	F	F	F				
Default	none	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
SSID	Segment set ID of an acoustic surface
MPAREA	Mass per unit area m
CPAREA	Damping per unit area c
KPAREA	Stiffness per unit area k

Remarks:

1. **Impedance**. On the boundary, the impedance is given by

$$[\bar{W}_f] = \frac{-Z_r}{(Z_r^2 + Z_i^2)} \int_S N_f^T N_f \, dS + \frac{iZ_i}{(Z_r^2 + Z_i^2)} \int_S N_f^T N_f \, dS$$

where

$$Z_r = c, \quad Z_i = \frac{k}{\omega} - \omega m \ .$$

*BOUNDARY_ACOUSTIC_INTERFACE

Purpose: Read the interface file generated by *INTERFACE_ACOUSTIC from a prior analysis to load an acoustic fluid model in a second SSD analysis (see *CONTROL_IMPLICIT_SSD_DIRECT).

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	IFID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT. This set defines the structural segments being coupled to the acoustic elements.
IFID	Interface ID from the file baifac.db created by *INTERFACE_ACOUSTIC. It will be used to drive the acoustic fluid boundary.

Remarks:

1. **Coupling.** The faces of the acoustic boundary defined by SSID do not need to identically match up with the structural segments of IFID from the interface file. The fluid faces are projected on the structural faces, so the acoustic volume may be meshed with tetrahedra and the structure with hexahedra or shells.

2. **Interpolation.** The displacements that drive the acoustic fluid boundary are linearly interpolated from the bracketing displacement states in the interface file.

*BOUNDARY_ACOUSTIC_MAPPING

Purpose: Define a set of elements or segments on a structure for mapping structural nodal velocity to acoustic volume boundary.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	STYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
SSID	Set or part ID
STYP	Set type: EQ.0: part set ID, see *SET_PART, EQ.1: part ID, see *PART, EQ.2: segment set ID, see *SET_SEGMENT.

Remarks:

If acoustic elements are not overlapping the structural elements, this keyword passes structural velocity to acoustic volume boundary, for subsequent frequency domain acoustic computation.

*BOUNDARY_ACOUSTIC_NON_REFLECTING

Purpose: Specify an absorbing acoustic boundary condition. This keyword differs from *BOUNDARY_NON_REFLECTING in the sense that higher order conditions are supported (see Remark 2).

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	NRBTYP	CRVOPT	DATA1	DATA2	DATA3		
Type	I	I	I	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
SSID	Segment set ID of an acoustic surface
NRBTYP	Absorbing boundary type: EQ.1: Plane wave absorbing boundary EQ.2: Curved wave absorbing boundary (see CRVOPT)
CRVOPT	Curvature specification option for NRBTYP = 2: EQ.1: Provide average curvature, $1/R$ EQ.2: Provide coordinates of the center of curvature, (X_c, Y_c, Z_c)
DATA1	CRVOPT.EQ.1: Average curvature, $1/R$ CRVOPT.EQ.2: Coordinate X_c
DATA2	Coordinate Y_c for CRVOPT = 2
DATA3	Coordinate Z_c for CRVOPT = 2

Remarks:

1. **Orientation.** The boundaries are most effective if the incident wave is roughly normal to the boundary. Sharp corners with three faces coming together at a point should be avoided.

2. **Equivalence.** NRBTYP = 1 is the same as *BOUNDARY_NON_REFLECTING and the 0^{th} order condition of Kallivokas, Bielak and McCamy. NRBTYP = 2 is equivalent to the 1^{st} order condition of Kallivokas, Bielak and McCamy.

*BOUNDARY_ACOUSTIC_PRESCRIBED_MOTION

Purpose: Impose a prescribed motion on the boundary of an acoustic volume.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	VAD	LCID	SF				
Type	I	I	I	F				
Default	none	none	none	1.0				

VARIABLE	DESCRIPTION
SSID	Segment set ID for the fluid boundary faces
VAD	Velocity/acceleration/displacement flag (see Remark 2): EQ.0: Velocity EQ.1: Acceleration EQ.2: Displacement
LCID	Load curve ID to describe motion as a function of time or frequency (see Remarks 1 and 2)
SF	Load curve scale factor

Remarks:

1. **SSD Analyses.** Fluid boundary velocities, accelerations, or displacements may be prescribed in SSD analyses (see *CONTROL_IMPLICIT_SSD_DIRECT). The abscissa of the load curve should then be in Hertz.

2. **Explicit Spectral Element Analyses.** Explicit simulations with *CONTROL_ACOUSTIC_SPECTRAL require the boundary motions be accelerations, VAD = 1. The abscissa of the load curve is in units of time.

*BOUNDARY_ACOUSTIC_PRESSURE_SPECTRAL

Purpose: Impose a varying pressure condition on the boundary of an acoustic volume.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	LCID	SF	TDEATH				
Type	I	I	F	F				
Default	none	none	1.0	10^{10}				

VARIABLE	DESCRIPTION
SSID	Segment set ID for the fluid boundary faces
LCID	Load curve ID to specify pressure as a function of time
SF	Load curve scale factor
TDEATH	Time when the pressure boundary condition should no longer be applied

*BOUNDARY_ALE_MAPPING

Purpose: This card maps ALE data histories from a previous run to a region of elements. Data are read or written in a mapping file called by the prompt "map=" on the command line (see Remarks 4 and 5). This keyword is supported for S-ALE and ALE. To map data at the initial time (not the histories) to the whole ALE domain (not just a region of elements), see *INITIAL_ALE_MAPPING

The following transitions are allowed:

1D → 2D	2D → 2D	3D → 3D
1D → 3D	2D → 3D	

Card Summary:

Card 1. This card is required.

ID	TYP	AMMSID	IVOLTYP	BIRTH	DEATH	DTOUT	INI

Card 2a. Include this card for |IVOLTYP| = 1.

THICK	RADIUS	X1	Y1	Z1			

Card 2b. Include this card for |IVOLTYP| = 2.

		X1	Y1	Z1	X2	Y2	Z2

Card 2c. Include this card for |IVOLTYP| = 3.

THICK	RADIUS	X1	Y1	Z1	X2	Y2	Z2

Card 2d. Include this blank card for |IVOLTYP| = 4.

Card 3. This card is required.

XO	YO	ZO	VECID				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYP	AMMSID	IVOLTYP	BIRTH	DEATH	DTOUT	INI
Type	I	I	I	I	F	F	F	I
Default	none	none	none	none	0.0	10^{20}	time step	0

VARIABLE	DESCRIPTION
ID	Part ID, part set ID, or element set ID
TYP	Type of "ID" (see Remark 1): EQ.0: Part set ID EQ.1: Part ID EQ.2: Shell set ID EQ.3: Solid set ID
AMMSID	Set ID of ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP. See Remark 1.
IVOLTYP	Type of volume containing the selected elements for the mapping. The absolute value of IVOLTYPE indicates the type of volume and the sign indicates whether the data is being read of written. **Volume Type** |IVOLTYP|.EQ.1: Spherical surface with thickness (THICK) |IVOLTYP|.EQ.2: Box |IVOLTYP|.EQ.3: Cylindrical surface with thickness (THICK) |IVOLTYP|.EQ.4: All the elements defined by ID **Read/Write** IVOLTYP.LT.0: Data from the mapping file are read for the elements of this volume. IVOLTYP.GT.0: Data from the elements of this volume are written in the mapping file.

VARIABLE	DESCRIPTION
BIRTH	Birth time to write or read the mapping file. If a mapping file is written, the next run reading this file will begin at time BIRTH if this parameter for this next run is not larger.
DEATH	Death time to write or read the mapping file. If a mapping file is written, the next run will stop reading this file at time DEATH if this parameter for this next run is not smaller.
DTOUT	Time interval between outputs in the mapping file. This parameter is only used to write in the mapping file.
INI	Flag to initialize all the ALE domain of the next run: EQ.0: No initialization EQ.1: Initialization. *INITIAL_ALE_MAPPING will have to be in the input deck of the next run to read the data from the mapping file. The initial time of the next run will be BIRTH.

Spherical Surface with Thickness Card. This card is included for |IVOLTYP| = 1.

Card 2a	1	2	3	4	5	6	7	8
Variable	THICK	RADIUS	X1	Y1	Z1			
Type	F	F	F	F	F			
Default	0.0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
THICK	Thickness for the element selection using surfaces
RADIUS	Radius
X1	x-coordinate of the sphere center
Y1	y-coordinate of the sphere center
Z1	z-coordinate of the sphere center

Box Card. This card is included for |IVOLTYP| = 2.

Card 2b	1	2	3	4	5	6	7	8
Variable			X1	Y1	Z1	X2	Y2	Z2
Type			F	F	F	F	F	F
Default			0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
X1	*x*-coordinate of the box's minimum point
Y1	*y*-coordinate of the box's minimum point
Z1	*z*-coordinate of the box's minimum point
X2	*x*-coordinate of the box's maximum point
Y2	*y*-coordinate of the box's maximum point
Z2	*z*-coordinate of the box's maximum point

Cylindrical Surface with Thickness Card. This card is included for |IVOLTYP| = 3.

Card 2c	1	2	3	4	5	6	7	8
Variable	THICK	RADIUS	X1	Y1	Z1	X2	Y2	Z2
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
THICK	Thickness for the element selection using surfaces
RADIUS	Radius
X1	*x*-coordinate of a point on the cylinder's axis
Y1	*y*-coordinate of a point on the cylinder's axis

VARIABLE	DESCRIPTION
Z1	z-coordinate of a point on the cylinder's axis
X2	x-coordinate of a vector parallel to the cylinder's axis
Y2	y-coordinate of a vector parallel to the cylinder's axis
Z2	z-coordinate of a vector parallel to the cylinder's axis

All Elements in Set Card. This blank card is included for |IVOLTYP| = 4.

Card 2d	1	2	3	4	5	6	7	8
Variable								
Type								

Card 3	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	VECID				
Type	F	F	F	I				
Default	0.0	0.0	0.0	none				

VARIABLE	DESCRIPTION
X0	Origin position in global x-direction. See Remark 2.
Y0	Origin position in global y-direction. See Remark 2.
Z0	Origin position in global z-direction. See Remark 2.
VECID	ID of the symmetric axis defined by *DEFINE_VECTOR. See Remark 3.

Remarks:

1. **Mapping of multi-material groups.** The routines of this card need to know which mesh will be initialized with the mapping data and more specifically which multi-material groups. The first 2 parameters (ID and TYP) defines the

mesh and the third one (AMMSID) refer to the *SET_MULTI-MATERIAL_-GROUP_LIST card. This card will define a list of material groups in the current run. The rank in this list should match the rank of the multi-material groups from the previous run (as a reminder the ranks of multi-material groups are defined by *ALE_MULTI-MATERIAL_GROUP). For instance, if the previous model has 3 groups, the current one has 5 groups, and the following mapping is wanted:

The 1st group (previous) ⇒ the 3rd group (current),
The 2nd group (previous) ⇒ the 5th group (current) and,
The 3rd group (previous) ⇒ the 4th group (current).

Then, the *SET_MULTI-MATERIAL_GROUP_LIST card should be as follows:

```
*SET_MULTI-MATERIAL_GROUP_LIST
300
3,5,4
```

2. **Origin.** The data can be mapped to different parts of the mesh by defining the origin of the coordinate system (X0, Y0, Z0).

3. **Orientation vector: VECID.** For a mapping file created by a previous axisymmetric model, the symmetric axis orientation in the current model is specified by VECID. For a mapping file created by a 3D or 1D spherical model, the vector VECID is read but ignored. The definitions of X0, Y0, Z0 and VECID change in the case of the following mappings:

   a) plane strain 2D (ELFORM = 13 in *SECTION_ALE2D) to plane strain 2D

   b) plane strain 2D to 3D

   While VECID still defines the *y*-axis in the 2D domain, the 3 first parameters in *DEFINE_VECTOR, additionally, define the location of the origin. The 3 last parameters define a position along the y-axis. For this case when 2D data is used in a 3D calculation the point X0, Y0, Z0 together with the vector, VECID, define the plane.

4. **Mapping file.** To make one mapping: only the command-line argument "map=" is necessary. If IVOLTYP is positive, the mapping file will be created, and ALE data histories will be written in this file. If IVOLTYP is negative the mapping file will be read, and ALE data histories will be used to interpolate the ALE variables of the selected elements. This file contains the following nodal and element data:

   - nodal coordinates
   - nodal velocities
   - part IDs

- element connectivities
- element centers
- densities
- volume fractions
- stresses
- plastic strains
- internal energies
- bulk viscosities
- relative volumes

5. **Successive mappings.** To make several successive mappings: the prompt "map1=" is necessary. If IVOLTYP is positive and the prompt "map1=" is in the command line, the ALE data are written to the mapping file given by "map1=". If IVOLTYP is negative and the prompt "map=" is in the command line, ALE data are read from the mapping file given by "map=".

6. **Comparison to *ALE_MAPPING.** Both *ALE_MAPPING nd *BOUNDARY_-ALE_MAPPING, can write multiple sets of data to a file at a given frequency between certain times. The big difference between these two keywords is reading the data. With *ALE_MAPPING, you select only one set of data to map from the previous run to the current run which is mapped at a time you select. With *BOUNDARY_ALE_MAPPING, all data output between t = BIRTH and t = DEATH from the previous run is mapped to the elements in the current run. The discrete histories from the previous run are interpolated.

*BOUNDARY_AMBIENT

Purpose: Define ALE "ambient" type element formulations (see Remarks 1, 2, and 5).

Card 1	1	2	3	4	5	6	7	8
Variable	SETID	MMG	AMBTYP	SIDR				
Type	I	I/A	I	I				
Default	none	none	none	0				

Optional Card. Additional optional card for AMBTYP = 4 with curves.

Card 2	1	2	3	4	5	6	7	8
Variable	LCID1	LCID2						
Type	I	I						
Default	Rem 1	Rem 1						

VARIABLE	DESCRIPTION
SETID	The ambient element set ID for which the thermodynamic state is being defined. The element set can be *SET_SOLID for a 3D ALE model, *SET_SHELL for a 2D ALE model, or *SET_BEAM for a 1D ALE model.
MMG	ALE multi-material group ID. In the case of S-ALE, the AMMG name (AMMGNM) can be used in place of the AMMGID. See Remark 6. EQ.0: Keep the already existing filling. Note that this is only available if the ambient element is prefilled with ALE materials.
AMBTYP	Ambient element type: EQ.4: Pressure inflow/outflow (see Remarks 1 and 2) EQ.5: Receptor for blast load (See *LOAD_BLAST_ENHANCED)

VARIABLE	DESCRIPTION
SIDR	Flag controlling the use of this keyword during dynamic relaxation: EQ.0: LS-DYNA applies this keyword only during the normal analysis phase. EQ.1: LS-DYNA applies this keyword during the dynamic relaxation phase but not the normal analysis phase. EQ.2: LS-DYNA applies this keyword during the dynamic relaxation and the normal analysis phases.
LCID1	A load curve ID for internal energy per unit reference volume (see Remark 4 and the *EOS section for details). If using *EOS_IDEAL_GAS, this ID refers to a temperature load curve ID.
LCID2	Load curve ID for relative volume, $v_r = (v/v_0 = \rho_0/\rho)$. See Remark 3 and the *EOS section for details.

Remarks:

1. **Ambient elements.** The term "ambient" refers to a medium with a predetermined thermodynamic state throughout the simulation. LS-DYNA resets the thermodynamic state of all "ambient" elements to this predetermined state every cycle. If using a *EOS card to determine this state, this predetermined thermodynamic state is constant throughout the simulation. For AMTYP = 4, if specifying the state with the load curves of Card 2, the thermodynamic state varies according to these defined load curves. The first and last abscissa values in these curves are this keyword's birth and death times. The literature sometimes refers to "ambient" elements as "reservoir" elements as they may be used to simulate a semi-infinite region.

2. **Thermodynamic state.** In general, two thermodynamic variables define a thermodynamic state of a non-reacting and no-phase-change material. By defining (a) an internal energy per unit reference volume load curve (or a temperature load curve if using *EOS_IDEAL_GAS) and (b) a relative volume load curve, LS-DYNA can computer the pressure as a function of time for this ambient part ID using the equation of state (*EOS_…).

3. **Reference specific volume.** A reference specific volume, $v_0 = 1/\rho_0$, is the inverse of a reference density, ρ_0. The reference density is the density at which the material is under a reference or nominal state. Refer to the *EOS section for a more complete explanation.

4. **Internal energy.** The internal energy per unit reference volume may be defined as

$$e_{\text{ipv0}} = \frac{C_v T}{v_0} \ .$$

The specific internal energy (or internal energy per unit mass) is defined as $C_v T$.

5. **Related cards.** This card does not require AET to be defined under *SECTION_SOLID or SECTION_ALE2D or SECTION_ALE1D card.

6. **AMMG name for S-ALE.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, you can only refer to each AMMG with their AMMGID. The order of appearance of the AMMGs in the input deck dictates the AMMGID for each AMMG. For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_MULTI-MATERIAL_GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the field AMMGNM. You can then refer to each AMMG defined with that keyword with either its name or its AMMGID (which is again based on order of appearance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, the AMMGIDs may change. Then, you must find all those references and alter them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

*BOUNDARY_AMBIENT_EOS

Purpose: This command defines the IDs of 2 load curves: (1) internal energy per unit reference volume (or temperature if using *EOS_IDEAL_GAS) and (2) relative volume. These 2 curves completely prescribe the thermodynamic state as a function of time for any ALE or Eulerian part with an "ambient" type element formulation (please see Remark 5). For a more specific form of this keyword that allows specification of ambient boundary conditions by element set, see *BOUNDARY_AMBIENT.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCID1	LCID2					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	The ambient part ID for which the thermodynamic state is being defined
LCID1	Load curve ID (see *DEFINE_CURVE or *DEFINE_CURVE_FUNCTION) for internal energy per unit reference volume (please read the beginning of the EOS section for details). If *EOS_IDEAL_GAS is being used, then this ID refers to a temperature load curve ID.
LCID2	Load curve ID (see *DEFINE_CURVE or *DEFINE_CURVE_FUNCTION) for relative volume, $$v_r = \left(\frac{v}{v_0} = \frac{\rho_0}{\rho}\right) .$$ (Please read the beginning of the EOS section for details.)

Remarks:

1. **Ambient Parts.** The term "ambient" refers to a medium that has predetermined thermodynamic state throughout the simulation. All "ambient" parts/elements will have their thermodynamic state reset back to this predetermined state every cycle. If this state is defined via the *EOS card, then this predetermined thermodynamic state is constant throughout the simulation. If it is defined via this card, *BOUNDARY_AMBIENT_EOS, then its thermodynamic state will vary

according to these defined load curves. "Ambient" part is sometimes also referred to as "reservoir" part as it may be used to simulate semi-infinite region.

2. **Thermodynamic Variables.** In general, a thermodynamic state of a non-reacting and no-phase-change material may be defined by 2 thermodynamic variables. By defining (a) an internal energy per unit reference volume load curve (or a temperature load curve if using *EOS_IDEAL_GAS) and (b) a relative volume load curve, the pressure as a function of time for this ambient part ID can be computed directly via the equation of state (*EOS_…).

3. **Reference Specific Volume.** A reference specific volume, $v_0 = 1/\rho_0$, is the inverse of a reference density, ρ_0. The reference density is defined as the density at which the material is under a reference or nominal state. Please refer to the *EOS section for additional explanation on this.

4. **Internal Energy.** The internal energy per unit reference volume may be defined as

$$e_{\text{ipv0}} = \frac{C_v T}{v_0}.$$

The specific internal energy (or internal energy per unit mass) is defined as $C_v T$.

5. **"Ambient" Elements.** This card is only to be used with an "ambient" element type as defined by the parameters under the *SECTION_SOLID card:

   a) ELFORM = 7,

   b) ELFORM = 11 and AET = 4, or

   c) ELFORM = 12 and AET = 4.

Example:

Consider an ambient ALE part ID 1 which has its internal energy per unit reference volume in a load curve ID 2 and relative volume load curve ID 3:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*BOUNDARY_AMBIENT_EOS
$      PID  e/T_LCID rvol_LCID
         1         2         3
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

***BOUNDARY_CONVECTION**_*OPTION*

Available options include:

SEGMENT

SET

Purpose: Apply a convection boundary condition on a SEGMENT or SEGMENT_SET for a thermal analysis. Two cards are defined for each option.

Card 1 for SET keyword option.

Card 1a	1	2	3	4	5	6	7	8
Variable	SSID	PSEROD						
Type	I	I						
Default	none	none						

Card 1 for SEGMENT keyword option.

Card 1b	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4				
Type	I	I	I	I				
Default	none	none	none	none				

Card 2	1	2	3	4	5	6	7	8
Variable	HLCID	HMULT	TLCID	TMULT	LOC			
Type	I	F	I	F	I			
Default	none	0.	none	0.	0			

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT.
PSEROD	Part set ID for updating boundary segments exposed to the environment as solid elements erode; see Remark 4.
N1, N2, ….	Node ID's defining segment.
HLCID	Convection heat transfer coefficient, h. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 2). When the reference is to a curve, HLCID has the following interpretation: GT.0: h is given as a function of time, t. The curve consists of $(t, h(t))$ data pairs. EQ.0: h is a constant defined by the value HMULT. LT.0: h is given as a function of temperature, T_{film}. The curve consists of (T_{film}, h) data pairs. |HLCID| references the *DEFINE_CURVE load curve ID. See Remark 1.
HMULT	Convection heat transfer coefficient, h, curve multiplier.
TLCID	Environment temperature, T_∞. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 3). When the reference is to a curve, TLCID has the following interpretation: GT.0: T_∞ is defined by a curve indexed by time consisting of $(t, T_\infty(t))$ data pairs. EQ.0: T_∞ is a constant defined by the value TMULT.
TMULT	Environment temperature, T_∞, curve multiplier.
LOC	For a thick thermal shell, the convection will be applied to the surface identified by LOC. See parameter, THSHEL, on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element EQ.1: upper surface of thermal shell element

Remarks:

1. **Film Temperature.** A convection boundary condition is calculated using $\dot{q}'' = h(T_{\text{surface}} - T_\infty)$ where h is the heat transfer coefficient, and $T_{\text{surface}} - T_\infty$ is a temperature potential. If h is a function of temperature, h is evaluated at the average or "film" temperature defined by

$$T_{film} = \frac{(T_{\text{surface}} + T_\infty)}{2}.$$

2. **Function Arguments for HLCID.** If HLCID references a *DEFINE_FUNCTION, the convective heat transfer coefficient can be a function of the segment centroid coordinates, the segment centroid velocity components, the segment centroid temperature, the environment temperature (T_∞), and the solution time, that is, "f(x, y, z, vx, vy, vz, temp, tinf, time)."

3. **Function Arguments for Environment Temperature.** If TLCID references a DEFINE_FUNCTION, the environment temperature can be a function of the segment centroid coordinates, the segment centroid velocity components, and the solution time, that is, "f(x, y, z, vx, vy, vz, time)."

4. **Erosion.** When convection is applied to a segment set associated with solid elements that erode, the convection condition on a deleted segment will not by default be transferred to newly exposed segments. It will instead cease to exist. By specifying a part set through the parameter PSEROD, any such new segment, *that is attached to an element in this part set*, will inherit this boundary condition, using the same data as prescribed for all original segments.

*BOUNDARY_COUPLED

Purpose: Define a boundary that is coupled with an external program. Two cards are required for each coupled boundary.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	SET	TYPE	PROG					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
ID	ID for this coupled boundary
TITLE	Descriptive name for this boundary
SET	Node set ID
TYPE	Coupling type: EQ.1: node set with force feedback EQ.2: node set for multiscale spotwelds
PROG	Program to couple to EQ.1: MPP-DYNA

Remarks:

This option is only available in the MPP version and allows for loose coupling with other MPI programs using a "multiple program" execution method. Currently it is only useful when linking with MPP-DYNA for the modeling of multiscale spotwelds (TYPE = 2,

PROG = 1). See *INCLUDE_MULTISCALE_SPOTWELD for information about using this capability.

*BOUNDARY_CYCLIC_{OPTION}

The *OPTION* allows an optional ID to be given that applies to each cyclic definition

ID

Purpose: Define nodes in boundary planes for cyclic symmetry.

These boundary conditions can be used to model a segment of an object that has rotational symmetry such as an impeller (see Figure 5-1). The segment boundary, denoted as a side 1 and side 2, may be curved or planar. In this section, a paired list of points is defined on the sides that are to be joined.

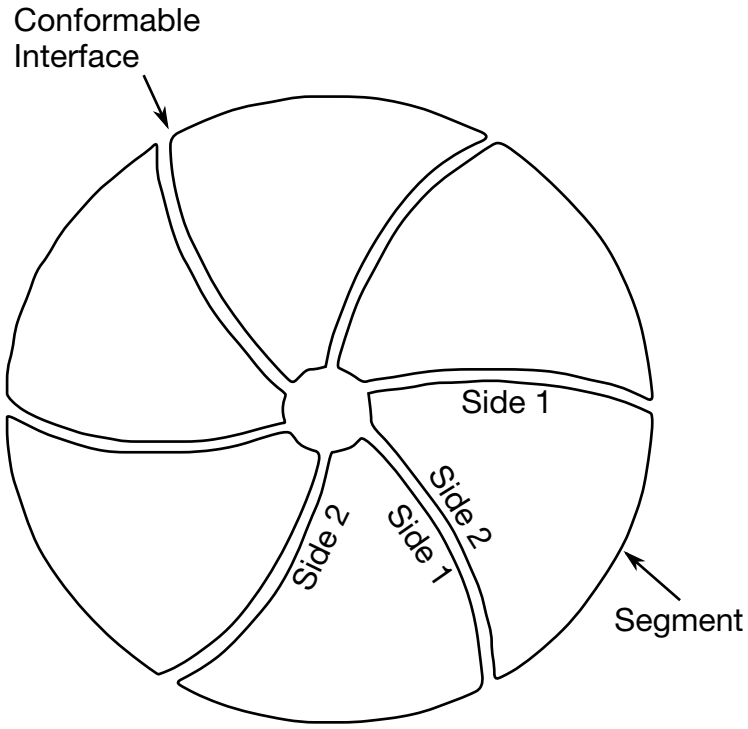


Figure 5-1. With axisymmetric cyclic symmetry, only one segment is modeled.

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	NSID1	NSID2	IGLOBAL	ISORT	CID
Type	F	F	F	I	I	I	I	I
Default	none	none	none	none	none	0	0	0

VARIABLE	DESCRIPTION
ID	ID for boundary definition
HEADING	Description of cyclic boundary
XC	x-component axis vector of axis of rotation (ignored if CID ≠ 0)
YC	y-component axis vector of axis of rotation (ignored if CID ≠ 0)
ZC	z-component axis vector of axis of rotation (ignored if CID ≠ 0)
NSID1	Node set ID for first boundary (side 1, see Figure 5-1).
NSID2	Node set ID for second boundary (side 2, see Figure 5-1). Each node in this set is constrained to its corresponding node in the first node set. Node sets NSID1 and NSID2 must contain the same number of nodal points. The shape of the two surfaces formed by the two node sets need not be planar, but the shapes should match.
IGLOBAL	Flag for repeating symmetry: EQ.0: Axisymmetric cyclic symmetry (default) EQ.1: Repeating symmetry in planes normal to global X EQ.2: Repeating symmetry in planes normal to global Y EQ.3: Repeating symmetry in planes normal to global Z
ISORT	Set to 1 for automatic sorting of nodes in node sets. See Remark 2.
CID	Coordinate system ID used to define the origin and axis of rotation for IGLOBAL = 0. The axis of rotation is the z-axis from CID. XC, YC, and ZC are ignored. The CID origin has no meaning for ISORT = 0 because the center of rotation is determined from the nodal sets. For ISORT = 1, however, the nodal sorting assumes that the rotation axis passes through the CID origin instead of the global

VARIABLE	DESCRIPTION
	origin.

Remarks:

1. **Node sets.** Each node set should generally be boundaries of the model.

2. **Node sorting.** Prior to version 970, it was assumed that the nodes are correctly ordered within each set, that is, the n^{th} node in NSID1 is equivalent to the n^{th} node in NSID2. In version 970 and later versions, if the ISORT flag is active, the nodes in NSID2 are automatically sorted to achieve equivalence, so the nodes can be picked by the quickest available method. However, for axisymmetric cyclic symmetry (IGLOBAL = 0), it is assumed that the axis passes through the global origin with a globally defined axis rotation or through the CID origin with the axis of rotation given by the z-axis of CID.

*BOUNDARY_DE_NON_REFLECTING

Purpose: Define a non-reflecting boundary for discrete element.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	CX	CY	CZ				
Type	I	F	F	F				
Default	none	1.0	1.0	1.0				
Remarks	1							

VARIABLE	DESCRIPTION
NSID	Node set ID.
CX	Decimal fraction of critical damping for translations in the x direction
CY	Decimal fraction of critical damping for translations in the y direction
CZ	Decimal fraction of critical damping for translations in the z direction

Remarks:

1. **Boundary Model.** Non-reflecting boundaries are used on the exterior boundaries of an analysis model of an infinite domain, such as a half-space to prevent artificial stress wave reflections generated at the model boundaries from re-entering the model and contaminating the results.

*BOUNDARY_FLUIDM

Purpose: Request the calculation of the external, fluid boundary mass on a structural surface in an inviscid, incompressible fluid. Unless accompanied by *BOUNDARY_FLUIDM_FREE_SURFACE or *BOUNDARY_FLUIDM_BOTTOM, the fluid is assumed to be infinite.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	RHOF	IEIGEN	ICURV	BESIMF			
Type	I	F	I	I	I			
Default	none	none	0	0	0			
Remarks	1, 2, 3		4	2	5			

VARIABLE	DESCRIPTION
SSID	Segment set ID (see *SET_SEGMENT) for the structural faces on which the fluid boundary mass is to be calculated
RHOF	Fluid mass density in the same units as the structural model
IEIGEN	Eigenvalue solution flag: EQ.0: Use the fluid mass directly in the calculation of the in-fluid structural modes of vibration. EQ.1: Write a superelement for the fluid-added mass matrix to file FLUIDM.DMIG, and then stop the solution. This is only advisable for very small matrices.
ICURV	Curvature usage flag: EQ.0: If no segment radii of curvature are included in the segment set, they are estimated from the angle between the neighboring segments. EQ.1: Disable estimating the radii of curvature.
BESIMF	Boundary element solution method for external fluid mass: EQ.0: LAPACK direct matrix solution EQ.1: Block low rank (BLR) direct solution without pivoting

VARIABLE	DESCRIPTION
	EQ.2: Pseudo-block GMRES iterative solution

Remarks:

1. **Normal vectors.** The outward normal vectors for the segments in the set should be directed into the external fluid.

2. **Segment curvatures.** Knowledge of local surface radii of curvature enhances the accuracy of the fluid boundary element formulation. These curvatures may be entered for individual segments in the fields DA1 and DA2 of the *SET_SEGMENT data cards. DA1 corresponds to the radius of curvature for the curve between nodes 1 and 2 while DA2 corresponds to the radius of curvature for the curve between nodes 1 and 4. If this information is not known, then it will be estimated from the change in the outward normal vectors for neighboring elements. You may disable this estimate with ICURV = 1.

3. **Eccentricity and back-to-back segments.** If the external fluid appears on both sides of a plate element, the fluid mass segments must be offset from the plane of the element. To do this, you specify the eccentricity of the segment in the DA3 field in *SET_SEGMENT. The eccentricity is half the plate thickness. If back-to-back segments are not offset, then a fatal, singularity error will occur.

4. **Eigenvalue solutions.** Currently, *BOUNDARY_FLUIDM may only be used in eigenvalue solutions and those which use eigenvectors as the basis of the solution, such as *FREQUENCY_DOMAIN solutions. Furthermore, the eigenvalue solution must use *CONTROL_IMPLICIT_EIGENVALUE method EIGMTH = 102 (LOBPCG).

5. **Boundary mass solution method.** The LAPACK solution method requires at least $NSEGS^2$ words of memory and is generally the most robust of the solution strategies. The BLR solution method is faster and requires less memory, typically $1/4^{th}$ of the memory for LAPACK. GMRES requires the least amount of memory but is the slowest method.

*BOUNDARY_FLUIDM_BOTTOM

Purpose: Includes the effects of a flat, arbitrarily oriented bottom in the calculation of the *BOUNDARY_FLUIDM mass matrix. A finite bottom impedance may be included with a normalized, reflection coefficient. *BOUNDARY_FLUIDM_BOTTOM can be combined with *BOUNDARY_FLUIDM_FREE_SURFACE in very shallow conditions.

Card 1	1	2	3	4	5	6	7	8
Variable	DISTBT	CXBT	CYBT	CZBT	BNORM			
Type	F	F	F	F	F			
Default	none	none	none	none	1.0			
Remarks	1							

VARIABLE	DESCRIPTION
DISTBT	Perpendicular distance from the plane of the bottom to the origin of the global coordinate system
CXBT	*X*-direction cosine for the normal vector of the bottom pointing into the bottom
CYBT	*Y*-direction cosine for the normal vector of the bottom pointing into the bottom
CZBT	*Z*-direction cosine for the normal vector of the bottom pointing into the bottom
BNORM	Low frequency bottom reflection coefficient ($0 < \text{BNORM} \leq 1$)

Remarks:

1. **Proximity Effect.** When nearby, a bottom will increase the added mass experienced by a submerged structure.

*BOUNDARY_FLUIDM_FREE_SURFACE

Purpose: Includes the effects of a flat, arbitrarily oriented free surface in the calculation of the *BOUNDARY_FLUIDM mass matrix. Keyword *BOUNDARY_FLUIDM_BOTTOM can be combined with *BOUNDARY_FLUIDM_FREE_SURFACE in very shallow conditions.

Card 1	1	2	3	4	5	6	7	8
Variable	DISTFS	CXFS	CYFS	CZFS				
Type	F	F	F	F				
Default	none	none	none	none				
Remarks	1, 2							

VARIABLE	DESCRIPTION
DISTFS	Perpendicular distance from the plane of the free surface to the origin of the global coordinate system
CXFS	*X*-direction cosine for the normal vector of the free surface pointing into the air
CYFS	*Y*-direction cosine for the normal vector of the free surface pointing into the air
CZFS	*Z*-direction cosine for the normal vector of the free surface pointing into the air

Remarks:

1. **Proximity Effect.** When nearby, a free surface will reduce the added mass experienced by a submerged structure.

2. **Elements Penetrating Free Surface.** Any FLUIDM segment which penetrates the free surface will be removed from the segment set.

*BOUNDARY_FLUIDM_INTERIOR

Purpose: Request calculating the added mass for an inviscid, incompressible fluid in a closed, internal volume/tank with a free surface. An input deck may contain multiple instantiations of this keyword when there are multiple separate internal volumes. It is intended for structural eigenvalue solutions and linear dynamic solutions based on modal superposition.

Card 1	1	2	3	4	5	6	7	8
Variable	TNAME							
Type	A6							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	FSTYPE	IEIGEN	SSID		SSIDBF	BESIMF		
Type	I	I	I		I	I		
Default	none	0	none		0	0		
Remarks		1, 5	2		2, 3	4		

Card 3	1	2	3	4	5	6	7	8
Variable	RHOF		DISTFS	CXFS	CYFS	CZFS		
Type	F		F	F	F	F		
Default	none		0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
TNAME	Name of the tank or volume. If forming a fluid mass superelement, this name is the file name.
FSTYPE	Free surface type: EQ.2: A quiescent zero pressure surface located in space by DISTFS, CXFS, CYFS, and CZFS.
IEIGEN	Eigenvalue solution flag: EQ.0: Use the fluid mass directly to calculate the in-fluid structural modes of vibration. EQ.1: Write a superelement for the fluid-added mass matrix to file [TNAME].DMIG, and then stop the solution. This is only advisable for very small matrices.
SSID	Segment set ID for the structural faces for which the fluid boundary mass is calculated, excluding the free surface.
SSIDBF	Segment set ID for internal baffle faces having fluid on both sides of plate elements.
BESIMF	Boundary element solution method for internal fluid mass: EQ.0: LAPACK direct matrix solution (default) EQ.1: Block low rank (BLR) direct solution without pivoting EQ.2: Pseudo-block GMRES iterative solution
RHOF	Fluid mass density in the same units as the structural model
DISTFS	Perpendicular distance from the plane of the free surface to the origin of the global coordinate system
CXFS	x-direction cosine for the normal vector of the free surface pointing into the air
CYFS	y-direction cosine for the normal vector of the free surface pointing into the air
CZFS	z-direction cosine for the normal vector of the free surface pointing into the air

Remarks:

1. **Consistency with *BOUNDARY_FLUIDM.** The same choice for IEIGEN should be made for all internal volumes and be consistent with the choice made for external fluid if it also exists.

2. **Normal vectors.** The outward normal vectors for the segments in the set should be directed out of the internal fluid volume.

3. **Baffle segments.** Only one side of the baffle should be specified with the segment set SSIDBF. Duplicate faces with opposite normal vectors are automatically generated. The thickness of the baffle must be specified with DA3 on *SET_SEGMENT.

4. **Boundary mass solution method.** The LAPACK solution method requires at least $NSEGS^2$ words of memory and is generally the most robust of the solution strategies. The BLR solution method is faster and requires less memory, typically $1/4^{th}$ of the memory for LAPACK. GMRESM requires the least amount of memory but is the slowest method.

5. **Eigenvalue solutions.** Currently, *BOUNDARY_FLUIDM_INTERIOR may only be used in eigenvalue solutions and solutions that use eigenvectors as the basis of the solution, such as *FREQUENCY_DOMAIN solutions. Furthermore, the eigenvalue solution must use *CONTROL_IMPLICIT_EIGENVALUE method EIGMTH = 102 (LOBPCG).

***BOUNDARY_FLUX**_OPTION_

Available options include:

SEGMENT

SET

Purpose: Apply a flux boundary condition (power/area) on a SEGMENT or SEGMENT_-SET for a thermal analysis. History variables can be associated with the boundary condition which will invoke a call to a user defined boundary flux subroutine for computing the flux.

Card Summary:

Card 1a. This card is included if and only if the SET option is used.

SSID	PSEROD						

Card 1b. This card is included if and only if the SEGMENT option is used.

N1	N2	N3	N4				

Card 2. This card is required.

LCID	MLC1	MLC2	MLC3	MLC4	LOC	NHISV	

Card 3. Include as many of this card as needed to define NHISV history variables.

HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8

Data Cards:

Card 1 for SET option.

Card 1a	1	2	3	4	5	6	7	8
Variable	SSID	PSEROD						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT
PSEROD	Part set ID for updating boundary segments exposed to the environment as solid elements erode; see Remark 5.

Card 1 for SEGMENT option.

Card 1b	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
N1, N2, …	Node IDs that define the segment

Card 2	1	2	3	4	5	6	7	8
Variable	LCID	MLC1	MLC2	MLC3	MLC4	LOC	NHISV	
Type	I	F	F	F	F	I	I	
Default	none	0.	0.	0.	0.	0	0	

VARIABLE	DESCRIPTION
LCID	This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 2) for heat flux. When the reference is to a curve, LCID has the following interpretation: GT.0: the flux is defined by a curve consisting of (time, flux) data pairs using the *DEFINE_CURVE keyword. The flux value applied to the nodal points is the curve value multiplied by the values MLC1, MLC2, MLC3, and MLC4, respectively.

VARIABLE	DESCRIPTION
	EQ.0: a constant flux is applied to each node defined by the values MLC1, MLC2, MLC3, and MLC4, respectively. LT.0: the flux is defined by a curve consisting of (temperature, flux) data pairs using the *DEFINE_CURVE keyword. The flux value applied to the nodal points is the curve value multiplied by the values MLC1, MLC2, MLC3, and MLC4. Enter |−LCID| on the *DEFINE_CURVE keyword.
MLC1	Curve multiplier at node N1.
MLC2	Curve multiplier at node N2.
MLC3	Curve multiplier at node N3.
MLC4	Curve multiplier at node N4.
LOC	For a thick thermal shell, the flux will be applied to the surface identified by LOC. See parameter, THSHEL, on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element EQ.1: upper surface of thermal shell element
NHISV	Number of history variables associated with the flux definition: GT.0: A user defined subroutine will be called to compute the flux. See Remark 3.

Define as many cards as necessary to initialize NHISV history variables.

Card 3	1	2	3	4	5	6	7	8
Variable	HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
HISV1	Initial value of history variable 1
HISV2	Initial value of history variable 2
⋮	⋮
HISV*n*	Initial value of history variable *n*, where *n* = NHISV

Remarks:

1. **Flux Direction.** The segment normal has no bearing on the flux. A negative flux transfers energy into the volume; a positive flux transfers energy out of the volume.

2. **Flux Definition with *DEFINE_FUNCTION.** If LCID references a *DEFINE_-FUNCTION, the heat flux can be a function of the segment centroid coordinates, the segment velocity components, the segment centroid temperature, the environment temperature, and the solution time, that is "f(x, y, z, vx, vy, vz, temp, tinf, time)."

3. **Flux Definition with User Subroutines.** When NHISV>0, the user subroutine

```
subroutine usrflux(fl, flp, ...)
```

   will be called to compute the heat flux (fl). For more details see Appendix S.

4. **SPH Elements.** This keyword is supported for SPH elements to define the flux boundary conditions for a thermal or coupled thermal/structural analysis. The values n_1, n_2, n_3, n_4 from the SPH particles or segments are used to define the flux segments.

5. **Erosion.** When flux is applied to a segment set associated with solid elements that erode, the flux condition on a deleted segment will not by default be transferred to newly exposed segments. It will instead cease to exist. By specifying a part set through the parameter PSEROD, any such new segment, *that is attached to an element in this part set*, will inherit this boundary condition, using the same data as prescribed for all original segments.

*BOUNDARY_FLUX_TRAJECTORY

Purpose: Define a moving surface heat source to model for example laser heated surfaces of solid or shell structures. Motion of the source is described by a nodal path and a prescribed velocity on this path. This keyword is applicable in coupled thermal-structural and thermal-only simulations.

Card Summary:

Card 1. This card is required.

SSID	PSEROD	NSID1	SPD1	NSID2	SPD2	RELVEL	

Card 2. This card is required.

EROD	LOC	LCROT	LCLAT				

Card 3. This card is required.

IFORM	LCTIM	Q	LCINC	ENFOR			

Card 4. This card is required.

P1	P2	P3	P4	P5	P6	P7	P8

Card 5. This card is included if and only if NSID2 = 0.

TX	TY	TZ					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	PSEROD	NSID1	SPD1	NSID2	SPD2	RELVEL	
Type	I	I	I	F	I	F	I	
Default	none	none	none	0.	none	0.	0	

VARIABLE	DESCRIPTION
SSID	Segment set ID containing segments that are potentially heated by surface heat source (flux).

VARIABLE	DESCRIPTION
PSEROD	Part set ID for updating boundary segments exposed to the environment as solid elements erode; see Remark 5.
NSID1	Node set defining the path of the weld source. The source travels along the path at speed SPD1. The nodes are traversed according to their ordering in the node set. See Remark 1.
SPD1	Speed of the heat source on the weld trajectory GT.0.0: constant speed LT.0.0: |SPD1| is a load curve ID defining weld speed as a function of time.
NSID2	Node or segment set containing information for the weld source aiming direction: GT.0: NSID2 together with SPD2 define a curve in the same way that NSID1 and SPD1 define a curve. Aiming direction is taken to be the vector pointing from the current position along NSID2 (for example your hand holding the torch) to the current position on NSID1 (the weld source). EQ.0: beam aiming direction is (TX, TY, TZ) input on Card 5.
SPD2	Speed of reference point in NSID2 (ignored unless NSID2 > 0) GT.0: constant speed LT.0: |SPD2| is a load curve ID defining weld speed as a function of time.
RELVEL	Defines if SPD1 and SPD2 are relative or absolute speeds in coupled simulations EQ.0: absolute speeds EQ.1: relative speeds with respect to underlying structure

Card 2	1	2	3	4	5	6	7	8
Variable	EROD	LOC	LCROT	LCLAT				
Type	I	I	I	I				
Default	0	none	none	none				

VARIABLE	DESCRIPTION
EROD	Flag for updating boundary segments exposed to the environment as solid elements (defined in PSEROD) erode; see Remark 5. EQ.0: no propagation onto new segments EQ.1: propagation onto new segments
LOC	For a thick thermal shell, the flux will be applied to the surface identified by LOC. See field THSHEL on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element EQ.1: upper surface of thermal shell element
LCROT	Load curve defining the rotation (angle in degrees) of weld source around the trajectory as function of time. See Remark 2.
LCLAT	Load curve for lateral offset of weld source as function of time. See Remark 2.

Card 3	1	2	3	4	5	6	7	8
Variable	IFORM	LCTIM	Q	LCINC	ENFOR			
Type	I	I	F	I	I			
Default	none	none	0.	none	0			

VARIABLE	DESCRIPTION
IFORM	Geometry description for energy rate density distribution (see Remark 3): EQ.1: double elliptic with constant density EQ.2: double elliptic with Gaussian distribution EQ.3: user defined distribution
LCTIM	Load curve ID for flux energy input rate multiplier $q_1(t)$ as a function of time, see Remark 4. EQ.0: use constant multiplier value $q_1(t) = 1.0$.
Q	Base energy input rate Q_b [energy/time], serves as base multiplier for the energy rate distributions defined by IFORM. See Remark 4 for details.
LCINC	Load curve ID for flux energy input rate multiplier $q_2(\alpha)$ as a function of inclination angle α (in degree), see Remark 4. EQ.0: use constant multiplier value $q_2(\alpha) = 1.0$.
ENFOR	Flag for heat input enforcement option (see Remark 4) EQ.0: no additional scaling of heat source EQ.1: account for inclination angle, α, of heat source by energy input rate multiplier $q_3(\alpha) = \cos(\alpha)$ EQ.2: scale the nodal fluxes by a multiplier $q_4(t)$ such that the resulting heat input equals the user input $Q(t) = Q_b \times q_1(t)$ EQ.3: apply option 1 and 2

Card 4	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
P*i*	Parameters defining flux geometry, depending on field IFORM. See Remark 3 for details.

Optional Weld Source Aiming Direction Card. Additional card for NSID2 = 0.

Card 5	1	2	3	4	5	6	7	8
Variable	TX	TY	TZ					
Type	F	F	F					
Default	0.	0.	0.					

VARIABLE	DESCRIPTION
TX, TY, TZ	Weld beam direction vector in global coordinates (NSID2 = 0 only)

Remarks:

1. **Heat Source Motion.** This card can be applied to coupled thermal-structure and thermal-only simulations. The source motion can be defined independently form the motion of nodes. This keyword applies to both solid and thermal thick shells.

   If NSID1 consists of work piece nodes, then the heat source will follow the motion of the work piece. By setting field RELVEL to 1 the velocity of the heat source can be defined relative to the motion of the nodes in NSID1 (the work piece).

2. **Heat Source Direction (Tilting the Torch).** There is a natural local coordinate system ($\mathbf{r}, \mathbf{s}, \mathbf{t}$) associated with the motion of the heat source. The relative velocity vector on the trajectory of the heat source defines the "forward" direction $\mathbf{r}$, so that material points that are approaching the heat source are in "front" of the beam. The weld source aiming direction, denoted by $\mathbf{t}$, determines the direction of the weld pool depth. The coordinate direction $\mathbf{s}$ is normal to the plane containing both the relative velocity and the aiming direction; note that $\mathbf{s} = \mathbf{t} \times \mathbf{r}$ to have a right handed coordinate system. It determines the direction of the weld pool width.

   The position and aiming direction of the heat source (for example tilting the torch) can be adjusted by rotating and translating the local coordinate system.

To do this, the system is first rotated around the vector **r** by a value given by the load curve LCROT, resulting in a new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. Then, the system is translated in direction $\mathbf{s}'$ using LCLAT, respectively.

3. **Energy Rate Surface Distributions.** Several different heat source geometries, described below, can be defined with this keyword. The local coordinate system needed for the description of the geometry is discussed in Remark 2.

For IFORM = 1 heat is generated in an elliptical region centered at the heat source with a constant energy rate surface density. The half-width a of the ellipse in the $\mathbf{s}'$-direction is given by input parameter P1. The half-widths of the ellipse in **r**-direction are different in front and behind the beam. The half-width in front of the beam is denoted by b_f (input parameter P2) and behind the beam is referred to as b_r (input parameter P3). Consequently, the flux value at a point p in the ellipse is given as

$$q = \frac{F}{\pi a b},$$

where:

$$F = \begin{cases} F_f \text{ if point } p \text{ is in front of beam} \\ F_r \text{ if point } p \text{ is behind beam} \end{cases}$$
$$b = \begin{cases} b_f \text{ if point } p \text{ is in front of beam} \\ b_r \text{ if point } p \text{ is behind beam} \end{cases}$$

It is expected that the sum of the weighting factors, F_f and F_r (defined as input parameters P4 and P5), equals 2.

The normalized surface density q for IFORM = 2 is assumed to be controlled by an exponential decay from the center of the ellipse to the boundary. The geometry of the source and its input correspond to IFORM = 1, but the energy rate surface density is here given as

$$q = \frac{nF}{\pi a b} \exp\left(\frac{-nx^2}{a^2}\right) \exp\left(\frac{-ny^2}{b^2}\right)$$

with the same assumptions for F and b as above and with (x, y) being the first two coordinates of point p with respect to the new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. The additional parameter n is defined by input parameter P6.

In contrast to the above, IFORM = 3 refers to a user defined function that determines the surface density q as function of the local coordinates (x, y), the current time t, and the temperature T. Consequently, the definition of an arbitrarily shape equivalent heat source may read as follows:

```
*DEFINE_FUNCTION
1, surface density distribution
q(r,s,time,temp)= …
```

To speed-up the computation, the function is evaluated for nodes within a distance less than a from the current center position. Consequently, the function ID and this distance are the only parameters needed in the input for Card 4.

Note that in all of the above cases, the normalized surface density q is multiplied by the energy rate $Q(t, \alpha)$, which is discussed in Remark 4.

IFORM	1	2	3
P1	a	a	FID
P2	b_f	b_f	a
P3	b_r	b_r	
P4	F_f	F_f	
P5	F_r	F_r	
P6		n	
P7			
P8			

4. **Energy Rate Input and Enforcement.** Naturally the calculation of the surface fluxes as discussed in Remark 3 requires the definition of a base energy input rate Q. It can be defined as a function of time, t, and inclination angle, α:

$$Q(t, \alpha) = Q_b \times q_1(t) \times q_2(\alpha) \times q_3(\alpha) \times q_4(t)$$

The multipliers $q_1(t)$ and $q_2(\alpha)$ are given as user-defined load curves. The multiplier $q_3(\alpha)$, if activated by a corresponding choice of ENFOR, account for the fact that a tilted heat source affects a larger area of the surface and is set to $q_3(\alpha) = \cos(\alpha)$. The value of α ranges from 0° (the segment normal and heat source point in the same direction) to 180° (the segment normal and heat source point in opposite directions). Naturally, the values of $q_2(\alpha)$ and $q_3(\alpha)$ vary among the individual segments.

For some applications it is required that the total heat flux from the boundary condition always agrees with the user input, meaning $Q_b \times q_1(t)$. The integration of the surface fluxes does not always meet this requirement, particularly for coarse meshes and highly curved structures. For that purpose, option ENFOR = 2 or 3 are implemented, which modify the nodal fluxes by a scaling factor $q_4(t)$. Note that this factor is not a user input but internally calculated and the same for all nodes with the particular flux boundary condition.

5. **Erosion.** When flux is applied to a segment set associated with solid elements that erode, the flux condition on a deleted segment will not by default be

transferred to newly exposed segments. It will instead cease to exist. By specifying a part set through the parameter PSEROD and activating the feature with EROD, any such new segment, *which is attached to an element in this part set*, will inherit this boundary condition, using the same data as prescribed for all original segments.

*BOUNDARY_MCOL

Purpose: Define parameters for MCOL coupling. The MCOL Program is a rigid body mechanics program for modeling the dynamics of ships. See Remark 1 for more information.

Card 1	1	2	3	4	5	6	7	8
Variable	NMCOL	MXSTEP	ETMCOL	TSUBC	PRTMCOL			
Type	I	I	F	F	F			
Default	2	none	0.0	0.0	none			
Remarks			2					

Ship Card. Include NMCOL cards, one for each ship.

Card 2	1	2	3	4	5	6	7	8
Variable	RBMCOL	MCOLFILE						
Type	I	A60						
Default		none						

VARIABLE	DESCRIPTION
NMCOL	Number of ships in MCOL coupling.
MXSTEP	Maximum number of time steps allowed in MCOL calculation. If the number of MCOL time steps exceeds MXSTEP, then LS-DYNA will terminate.
ETMCOL	Uncoupling termination time; see Remark 2 below. EQ.0.0: set to LS-DYNA termination time
TSUBC	Time interval for MCOL subcycling. EQ.0.0: no subcycling

VARIABLE	DESCRIPTION
PRTMCOL	Time interval for output of MCOL rigid body data.
RBMCOL	LS-DYNA rigid body material assignment for the ship.
MCOLFILE	Filename containing MCOL input parameters for the ship.

Remarks:

1. **MCOL Coupling.** The basis for MCOL is a convolution integral approach for simulating the equations of motion. A mass and inertia tensor are required as input for each ship. The masses are then augmented to include the effects of the mass of the surrounding water. A separate program determines the various terms of the damping/buoyancy force formulas which are also input to MCOL. The coupling is accomplished in a simple manner: at each time step LS-DYNA computes the resultant forces and moments on the MCOL rigid bodies and passes them to MCOL. MCOL then updates the positions of the ships and returns the new rigid body locations to LS-DYNA. A more detailed theoretical and practical description of MCOL can be found in a separate report (to appear).

2. **ETMCOL.** After the end of the LS-DYNA/MCOL calculation, the analysis can be continued using MCOL alone. ETMCOL is the termination time for this analysis. If ETMCOL is smaller than the LS-DYNA termination time, the uncoupled analysis will not be activated.

3. **Output Files.** The MCOL output is written to the files mcolout (ship position) and mcolenergy (energy breakdown). In LS-PrePost, mcolout can be plotted through the rigid body time history option and MCOLENERGY.

*BOUNDARY_NON_REFLECTING

Purpose: Define a non-reflecting boundary. This option applies to continuum domains modeled with solid or thick shell elements. For geomechanical problems this option is important for limiting the spatial extent of the finite element mesh and thus the number of solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	AD	AS					
Type	I	F	F					
Default	none	0.0	0.0					
Remarks	1, 2	3	3					

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT.
AD	Default activation flag for dilatational waves. EQ.0.0: on NE.0.0: off
AS	Default activation flag for shear waves. EQ.0.0: on NE.0.0: off

Remarks:

1. **Restrictions.** Non-reflecting boundaries defined with this keyword are only used with three-dimensional solid elements. Boundaries are defined as a collection of segments, and segments are equivalent to element faces on the boundary. Segments are defined by listing the corner nodes in either a clockwise or counterclockwise order.

2. **Model Description.** Non-reflecting boundaries are used on the exterior boundaries of an analysis model of an infinite domain, such as a half-space to prevent artificial stress wave reflections generated at the model boundaries from

reentering the model and contaminating the results. Internally, LS-DYNA computes an impedance matching function for all non-reflecting boundary segments based on an assumption of linear material behavior. Thus, the finite element mesh should be constructed so that all significant nonlinear behavior is contained within the discrete analysis model.

3. **Reflecting Waves.** With the two optional flags, the influence of reflecting waves can be studied.

4. **Dynamic Relaxation.** During the dynamic relaxation phase (optional), nodes on non-reflecting segments are constrained in the normal direction. Nodal forces associated with these constraints are then applied as external loads and held constant in the transient phase while the constraints are replaced with the impedance matching functions. In this manner, soil can be quasi-statically prestressed during the dynamic relaxation phase and dynamic loads (with non-reflecting boundaries) subsequently applied in the transient phase.

5. **Time Step Size.** In explicit analyses, TSSFAC in *CONTROL_TIMESTEP is set to 0.667 by default if not specified, in order to obtain a stable solution. This is typically necessary if three non-reflecting boundary planes meet at a corner. A larger value may be specified to obtain a shorter simulation time if it does not affect stability.

*BOUNDARY_NON_REFLECTING_2D

Purpose: Define a non-reflecting boundary. This option applies to continuum domains modeled with two-dimensional solid elements in the *xy* plane. For geomechanical problems, this option is important for limiting the size of the models. For ALE 2D models, including this keyword without data cards (Card 1) causes the non-reflecting boundary condition to be applied to all the boundary edges of the 2D ALE mesh.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	AD	AS					
Type	I	I	I					
Default	none	0	0					
Remarks	1, 2							

VARIABLE	DESCRIPTION
NSID	Node set ID; see *SET_NODE. See Figure 5-2. LT.0: |NSID| is the ID of *SET_SEGMENT.
AD	Default activation flag for dilatational waves: EQ.0.0: On NE.0.0: Off
AS	Default activation flag for shear waves: EQ.0.0: On NE.0.0: Off

Remarks:

1. **Boundary Definition.** Non-reflecting boundaries defined with this keyword are only used with two-dimensional solid elements in either plane strain or axisymmetric geometries. Boundaries are defined as a sequential string of nodes moving *counterclockwise* around the boundary.

2. **Model Assumptions.** Non-reflecting boundaries are used on the exterior boundaries of an analysis model of an infinite domain, such as a half-space to prevent artificial stress wave reflections generated at the model boundaries from re-entering the model and contaminating the results. Internally, LS-DYNA computes an impedance matching function for all non-reflecting boundary segments based on an assumption of linear material behavior. Thus, the finite element mesh should be constructed so that all significant nonlinear behavior in contained within the discrete analysis model.

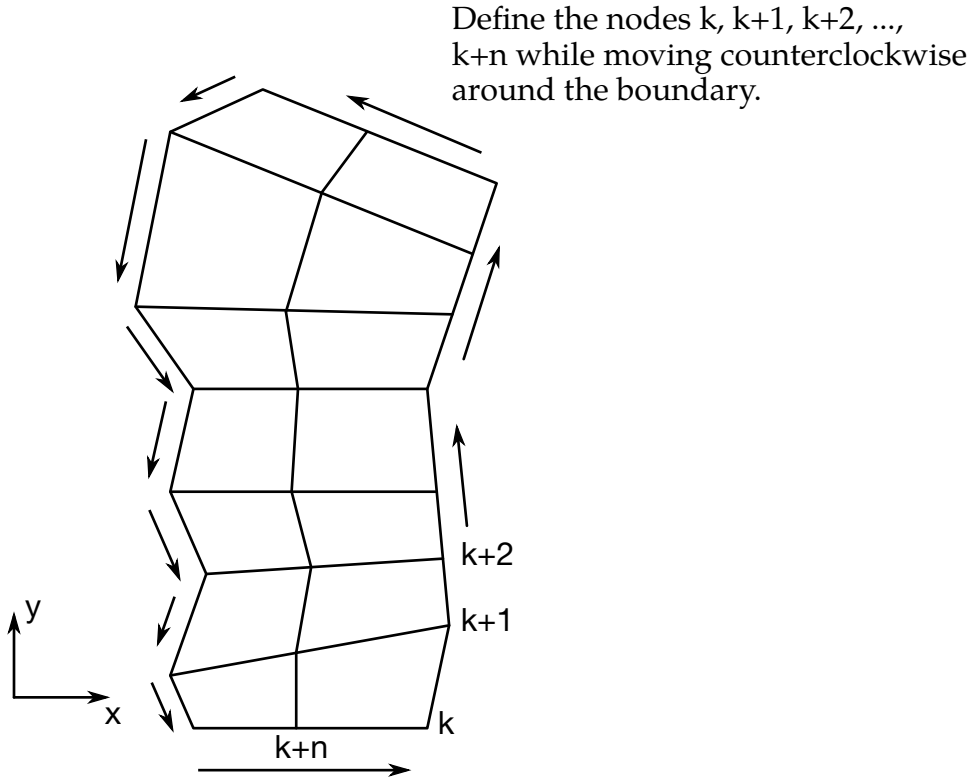


Figure 5-2. When defining a transmitting boundary in 2D define the node numbers in the node set in consecutive order while moving counterclockwise around the boundary.

*BOUNDARY_PAP

Purpose: Define pressure boundary conditions for pore air flow calculation, such as at the structure surface exposed to atmospheric pressure.

Card 1	1	2	3	4	5	6	7	8
Variable	SEGID	LCID	CMULT	CVMASS	BLOCK	TBIRTH	TDEATH	CVRPER
Type	I				F	F	F	F
Default	none	CMULT	none	none	0.0	0.0	10^{20}	1.0
Remarks				1, 2				3

VARIABLE	DESCRIPTION
SEGID	Segment set ID
LCID	Load curve giving pore air pressure as a function of time. EQ.0: constant pressure assumed equal to CMULT
CMULT	Factor on curve or constant pressure head if LCID = 0.
CVMASS	Initial mass of a control volume next to the segment set SEGID
BLOCK	Contact blockage effect, EQ.0: When all segments in SEGID are subject to the pressure defined by LCID and CMULT; EQ.1: When only elements in SEGID not involved in contact are subject to the pressure defined by LCID and CMULT.
TBIRTH	Time at which boundary condition becomes active
TDEATH	Time at which boundary condition becomes inactive
CVRPER	Permeability factor of cover material, where cover refers to a shell layer coating the surface of the solid. Default value is 1.0 when it is not defined. $0.0 \leq \text{CVRPER} \leq 1.0$

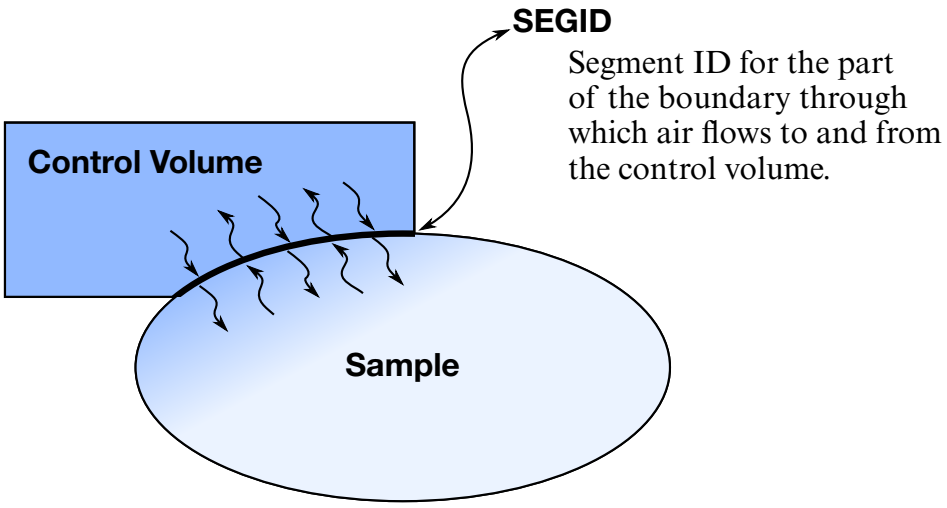


Figure 5-3. Air flows between the control volume and the sample. CVMASS specifies the control volume's initial mass, and CVMULT sets the initial pressure.

Remarks:

1. **Structure Surfaces.** All structure surfaces subject to specified pressure must be defined.

2. **Air Mass Transfer.** A non-zero CVMASS, together with a non-zero CMULT and an un-defined LCID, can be used to simulate air mass transfer between a control volume and a test specimen containing pore air. The control volume is assumed to have a fixed volume, an initial pressure of CMULT, and an initial mass of CVMASS. Air mass transfer happens between the control volume and its neighboring specimen. Such mass transfer results in pressure change in control volume and test specimen.

3. **Porosity Properties.** The porosity properties of the cover material can be modeled using CVRPER. If SEGID is covered by a material of very low permeability (e.g., coated fabric), CVRPER should be set to 0.0. In this case, P_c, the pressure calculated assuming no boundary condition, is applied to SEGID. If SEGID is not covered by any material, CVRPER should be set to 1.0, the default value. In this case, the applied pressure becomes P_b, the boundary pressure determined by CMULT and LCID.

BOUNDARY_PORE_FLUID_*OPTION

Available options include:

PART

SET

Purpose: Define parts that contain pore fluid. Defaults are given on *CONTROL_PORE_-FLUID.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	WTABLE	PF_RHO	ATYPE	PF_BULK	ACURVE	WTCUR	SUCLIM
Type	I	F	F	I	F	I	I	F
Default	none	*	*	*	*	0	0	0.

*Defaults are taken from *CONTROL_PORE_FLUID.

VARIABLE	DESCRIPTION
PID	Part ID or part set ID. All elements within the part must lie below the water table.
WTABLE	*Z*-coordinate at which pore pressure = 0 (water table)
PF_RHO	Density of pore water in soil skeleton (see Remark 2): EQ.0: Default density specified on *CONTROL_PORE_FLUID card is used.
ATYPE	Analysis type for parts: EQ.0: Default to value specified on *CONTROL_PORE_FLUID EQ.1: Undrained analysis EQ.2: Drained analysis EQ.3: Time dependent consolidation (coupled) EQ.4: Consolidate to steady state (uncoupled) EQ.5: Drained in dynamic relaxation, undrained in transient

VARIABLE	DESCRIPTION
PF_BULK	Bulk modulus of pore fluid: EQ.0: Default to value specified on *CONTROL_PORE_FLUID
ACURVE	Curve of analysis type as a function of time (see Remark 3)
WTCUR	Curve of water table (z-coordinate) as a function of time
SUCLIM	Suction limit (defined in head, that is, length units). It must not be negative. See Remark 4.

Remarks:

1. **Pore Water Parts.** This card must be present for all parts having pore water.

2. **Density.** The density on this card is used only to calculate pressure head. To ensure the correct gravity loading, the density of the soil material should be increased to include the mass associated with the pore water.

3. **Analysis Type Curve.** The y-axis values of the curve of analysis type as a function of time can only be 1, 2, 3, or 4, where these values have the same meanings as for ATYPE. During dynamic relaxation, the analysis type will be taken from the first value on the curve

4. **Suction.** The default for SUCLIM is zero, meaning that the pore fluid cannot generate suction. To allow unlimited suction, set this parameter to a large positive number.

*BOUNDARY_PRECRACK

Purpose: Define pre-cracks in XFEM shell formulations 52 or 54 for purposes of fracture analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	CTYPE	NP					
Type	I	I	I					
Default	none	1	none					

Pre-Crack Point Cards. Include NP cards, one for each point in the pre-crack.

Card 2	1	2	3	4	5	6	7	8
Variable	X	Y	Z					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID where the pre-crack is located.
CTYPE	Type of pre-crack: EQ.1: straight line
NP	Number of points defining the pre-crack.
X, Y, Z	Coordinates of the points defining the pre-crack. The points should be on or close to the shell surface and the line segments should not pass any nodal points in the part.

*BOUNDARY_PRESCRIBED_ACCELEROMETER_RIGID

Purpose: Prescribe the motion of a rigid body based on experimental data obtained from accelerometers affixed to the rigid body.

Note: This feature is available starting with LS-DYNA 971R3.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	SOLV						
Type	I	I						
Default	none	1						

Accelerometer Cards. Define one card for each accelerometer affixed to the rigid body. Input is terminated at the next keyword ("*") card. A minimum of three accelerometers are required (see Remarks below).

Card 2	1	2	3	4	5	6	7	8
Variable	NID	CID	LCIDX	LCIDY	LCIDZ			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
PID	Part ID for rigid body whose motion is prescribed.
SOLV	Solver type: EQ.1: Gaussian elimination (default) EQ.2: linear regression EQ.3: Reciprocal, inverse explicit time integration
NID	Node ID corresponding to the location of the accelerometer.

VARIABLE	DESCRIPTION
CID	Coordinate system ID describing the orientation of the accelerometer's local axes (see *DEFINE_COORDINATE_NODES). All nodes must reside on the same part. Set FLAG = 1.
LCIDX	Load curve ID containing the local *x*-acceleration time history from the accelerometer.
LCIDY	Load curve ID containing the local *y*-acceleration time history from the accelerometer.
LCIDZ	Load curve ID containing the local *z*-acceleration time history from the accelerometer.

Remarks:

1. **Time Histories.** Acceleration time histories from a minimum of three accelerometers each providing output from three channels are required. Load curves must have the same number of points and data must be uniformly spaced.

2. **Local Axes.** Local axes of the accelerometers must be orthogonal.

3. **SOLV=3.** The following is a theoretical description of SOLV=3.

   As a prerequisite we emphasize that the accelerations given by load curves LCIDX, LCIDY and LCIDZ in this keyword is in the *local* system CID which is affixed to the rigid body and therefore rotates with it accordingly. This is *exactly* the data that is output for the *first* node in *ELEMENT_SEATBELT_ACCELEROMETER and therefore this can be used as input here to reproduce a desired motion. It may be worth mentioning that the local system in that other keyword is the local system determined by the nodes involved in the accelerometer and should therefore match the initial configuration of CID in this keyword. In general, sample data from a physical test, or a simulation with a deformable body for that matter, will not provide data that corresponds to an exact rigid body motion. Therefore, the motion cannot be determined uniquely from the data, but SOLV=3 is an option that ensures that *if* the input is exactly a rigid body motion from a previous simulation with full range of sample data (essentially accelerations from every time step), it will reproduce the motion within round-off errors assuming the *same* time step is used in the latter simulation. Since accelerations are used for input, the sensitivity is high and small errors in the data may result in large errors in the resulting motion. It is therefore recommended to use several sample locations in space that span the rigid body, as this will presumably increase the likelihood of getting better results. With this in mind, the theory amounts to establish a relationship between a rigid body node

acceleration and the translational and rotational increments of the rigid body and solve for the unknowns.

An arbitrary point on the rigid body in question has the coordinate $\boldsymbol{x}$, and can be expressed as

$$\boldsymbol{x} = \boldsymbol{X} + \boldsymbol{Q}\boldsymbol{\xi}$$

where $\boldsymbol{X}$ is the coordinate of the rigid body center of mass, $\boldsymbol{Q}$ is the orientation of the rigid body, and $\boldsymbol{\xi}$ is a constant vector associated with the relative position of $\boldsymbol{x}$. Using the explicit time integration scheme for that node, we have

$$\begin{aligned}\boldsymbol{x}_{n+1} &= \boldsymbol{x}_n + \Delta t_{n+\frac{1}{2}} \boldsymbol{v}_{n+\frac{1}{2}} \\ \boldsymbol{v}_{n+\frac{1}{2}} &= \boldsymbol{v}_{n-\frac{1}{2}} + \Delta t_n \boldsymbol{a}_n\end{aligned}$$

where the right-hand side quantities are in general known and $\Delta t_n = \frac{\Delta t_{n+\frac{1}{2}} + \Delta t_{n-\frac{1}{2}}}{2}$. We also have for the rigid body translations

$$\boldsymbol{X}_{n+1} = \boldsymbol{X}_n + \Delta t_{n+\frac{1}{2}} \boldsymbol{V}_{n+\frac{1}{2}},$$

and for the rigid body rotations

$$\boldsymbol{Q}_{n+1} = \boldsymbol{\Omega}(\boldsymbol{\theta})\boldsymbol{Q}_n,$$

where $\boldsymbol{\Omega}(\boldsymbol{\theta})$ is the orthogonal Hughes-Winget matrix obtained from the rotational increment $\boldsymbol{\theta} = \boldsymbol{\omega}_{n+\frac{1}{2}} \Delta t_{n+\frac{1}{2}}$ of the rigid body. Using these expressions, the updated velocity of a rigid body node is given as

$$\begin{aligned}\boldsymbol{v}_{n+\frac{1}{2}} = \frac{\boldsymbol{x}_{n+1} - \boldsymbol{x}_n}{\Delta t_{n+\frac{1}{2}}} &= \frac{\boldsymbol{X}_{n+1} + \boldsymbol{Q}_{n+1}\boldsymbol{\xi} - \boldsymbol{X}_n - \boldsymbol{Q}_n\boldsymbol{\xi}}{\Delta t_{n+\frac{1}{2}}} = \boldsymbol{V}_{n+\frac{1}{2}} + \frac{\boldsymbol{Q}_{n+1} - \boldsymbol{Q}_n}{\Delta t_{n+\frac{1}{2}}}\boldsymbol{\xi} \\ &= \boldsymbol{V}_{n+\frac{1}{2}} + \frac{\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I}}{\Delta t_{n+\frac{1}{2}}}\boldsymbol{Q}_n\boldsymbol{\xi} = \boldsymbol{V}_{n+\frac{1}{2}} + \frac{\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I}}{\Delta t_{n+\frac{1}{2}}}(\boldsymbol{x}_n - \boldsymbol{X}_n)\end{aligned}$$

and therefore $\boldsymbol{a}_n$ is given by

$$\boldsymbol{a}_n = \frac{\boldsymbol{v}_{n+\frac{1}{2}} - \boldsymbol{v}_{n-\frac{1}{2}}}{\Delta t_n} = \frac{\boldsymbol{V}_{n+\frac{1}{2}} + \frac{\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I}}{\Delta t_{n+\frac{1}{2}}}(\boldsymbol{x}_n - \boldsymbol{X}_n) - \boldsymbol{v}_{n-\frac{1}{2}}}{\Delta t_n}.$$

The intention here is to establish $\boldsymbol{U} = \boldsymbol{V}_{n+\frac{1}{2}} \Delta t_{n+\frac{1}{2}}$ and $\boldsymbol{\theta}$ from at least three acceleration points on the rigid body. We first establish $\boldsymbol{U}$ as a function of $\boldsymbol{\theta}$ by rewriting the expression for $\boldsymbol{a}_n$ as

$$\boldsymbol{U} = \boldsymbol{a}_n \Delta t_n \Delta t_{n+\frac{1}{2}} + \boldsymbol{v}_{n-\frac{1}{2}} \Delta t_{n+\frac{1}{2}} - (\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I})(\boldsymbol{x}_n - \boldsymbol{X}_n).$$

Then assume we have sample points $i = 1,2,\ldots,k$, where k is at least 3, then we can use the mean of these equations to solve for $\boldsymbol{U}$

$$\boldsymbol{U} = \bar{\boldsymbol{a}}_n \Delta t_n \Delta t_{n+\frac{1}{2}} + \overline{\boldsymbol{v}}_{n-\frac{1}{2}} \Delta t_{n+\frac{1}{2}} - (\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I})(\bar{\boldsymbol{x}}_n - \boldsymbol{X}_n)$$

where the bar indicates the mean value of the quantity. Then we insert this expression into the other equations, $i = 1,2,\ldots k$

$$(\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I})(\boldsymbol{x}_n^i - \overline{\boldsymbol{x}}_n) - (\boldsymbol{a}_n^i - \overline{\boldsymbol{a}}_n)\Delta t_n \Delta t_{n+\frac{1}{2}} - \left(\boldsymbol{v}_{n-\frac{1}{2}}^i - \overline{\boldsymbol{v}}_{n-\frac{1}{2}}\right)\Delta t_{n+\frac{1}{2}} = \boldsymbol{0}$$

which is an overdetermined set of nonlinear equations for $\boldsymbol{\theta}$. So, form

$$\boldsymbol{f}^i(\boldsymbol{\theta}) = (\boldsymbol{\Omega}(\boldsymbol{\theta}) - \boldsymbol{I})(\boldsymbol{x}_n^i - \overline{\boldsymbol{x}}_n) - (\boldsymbol{a}_n^i - \overline{\boldsymbol{a}}_n)\Delta t_n \Delta t_{n+\frac{1}{2}} - \left(\boldsymbol{v}^i{}_{n-\frac{1}{2}} - \overline{\boldsymbol{v}}_{n-\frac{1}{2}}\right)\Delta t_{n+\frac{1}{2}}$$

and then minimize

$$f(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{k} \boldsymbol{f}^i(\boldsymbol{\theta})^T \boldsymbol{f}^i(\boldsymbol{\theta})\,.$$

The solution to this minimization problem is the extreme point given by

$$\boldsymbol{0} = \frac{\partial f}{\partial \boldsymbol{\theta}} = \sum_{i=1}^{k} \boldsymbol{f}^i(\boldsymbol{\theta})^T \frac{\partial \boldsymbol{f}^i}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta})$$

which is a nonlinear set of equations. This is solved by a Newton method, and the resulting $\boldsymbol{\theta}$ is then used to establish $\boldsymbol{U}$. These two increments are then used for determining the motion of the rigid body.

*BOUNDARY_PRESCRIBED_FINAL_GEOMETRY

Purpose: The final displaced geometry for a subset of nodal points is defined. The nodes of this subset are displaced from their initial positions specified in the *NODE input to the final geometry along a straight-line trajectory. A load curve defines a scale factor as a function of time that is bounded between zero and unity corresponding to the initial and final geometry, respectively. A unique load curve can be specified for each node, or a default load curve can apply to all nodes. The external work generated by the displacement field is included in the energy ratio calculation for the glstat file.

Card Summary:

Card 1. This card is required.

BPFGID	LCIDF	DEATHD	IBRTH				

Card 2a. Include this card if IBRTH = 0. Include as many instances of this card as desired. The next keyword ("*") card terminates this input.

NID	X	Y	Z	LCID	DEATH

Card 2b. Include this card if IBRTH = 1. Include as many instances of this card as desired. The next keyword ("*") card terminates this input.

NID	X	Y	Z	LCID	DEATH	BIRTH

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	BPFGID	LCIDF	DEATHD	IBRTH				
Type	I	I	F	I				
Default	0	0	infinity	0				

VARIABLE	DESCRIPTION
BPFGID	ID for this set of imposed boundary conditions
LCIDF	Default load curve ID. This curve ranges between zero and unity.
DEATHD	Default death time. At this time, the prescribed motion is inactive,

VARIABLE	DESCRIPTION
	and the nodal point is allowed to move freely.
IBRTH	Flag to allow node input with death and birth times. Note that IBRTH = 1 has a different format for Card 2. EQ.0: No birth time. Use card format Card 2a. EQ.1: Birth time included. Use card format Card 2b.

Node Cards for IBRTH = 0. The next keyword ("*") card terminates this input.

Card 2a	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z		LCID	DEATH	
Type	I	F		F		F		I	F	
Default	none	0.		0.		0.		LCIDF	DEATHD	

Node Cards for IBRTH = 1. The next keyword ("*") card terminates this input.

Card 2b	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z		LCID	DEATH	BIRTH
Type	I	F		F		F		I	F	F
Default	none	0.		0.		0.		LCIDF	DEATHD	0.

VARIABLE	DESCRIPTION
NID	GT.0: Node ID for which the final position is defined. Nodes defined in this section must also appear under the *NODE input. LT.0: |NID| is a node set ID for which the final projection plane normal to the global *z*-axis is defined. In this case, only the offset Z value is needed, and all the nodes in this node set are displaced from their initial positions to the projected points on the *xy*-plane with Z offset. They share the same LCID and DEATH.

VARIABLE	DESCRIPTION
X	*x*-coordinate of final geometry
Y	*y*-coordinate of final geometry
Z	*z*-coordinate of final geometry
LCID	Load curve ID for NID. If zero, the default curve ID, LCIDF, is used.
DEATH	Death time for NID. If zero, the default value, DEATHD, is used.
BIRTH	Birth time for NID. The default value is zero. The prescribed motion begins acting at time = BIRTH, but with the motion from the zero-abscissa value of the curve LCID. In other words, the abscissa values are shifted by an amount BIRTH, such that it has the same effect as setting OFFA = BIRTH in *DEFINE_CURVE.

***BOUNDARY_PRESCRIBED_MOTION**_*OPTION1_{OPTION2}_{OPTION3}*

Available options for *OPTION1* include:

NODE

SET

SET_BOX

SET_SEGMENT

RIGID

RIGID_LOCAL

SET_LINE

POINT_UVW

EDGE_UVW

FACE_UVW

FACE_XYZ

SET_POINT_UVW

SET_EDGE_UVW

SET_FACE_UVW

SET_FACE_XYZ

OPTION2 allows an optional ID to be given for the single node, node set, segment set, and rigid body options. If you define a heading, the ID and heading are included at the beginning of the ASCII file, bndout.

ID

OPTION3 is only supported for motion imposed on a rigid body. Thus, *OPTION1* must include RIGID in the name.

BNDOUT2DYNAIN

With this feature, the reaction force as a parameter is output to the dynain file for use in a subsequent simulation. With this output, an equivalent force can replace the motion between simulations without manual intervention. We only support this feature for

implicit analysis (IMFLAG = 1 on *CONTROL_IMPLICIT_GENERAL) and only for the lsda format of the dynain file (FTYPE = 3 on *INTERFACE_SPRINGBACK_LSDYNA).

Purpose: Define an imposed nodal motion (velocity, acceleration, or displacement) on a node or a set of nodes. You can also impose velocities and displacements on rigid bodies. If you activate the RIGID_LOCAL option, the motion is prescribed with respect to the local coordinate system for the rigid body; see fields LCO and CID of keywords *MAT_RIGID and *CONSTRAINED_NODAL_RIGID_BODY, respectively. This keyword allows translational nodal velocity and acceleration specifications for rigid body nodes. Remark 6 describes the application of these motions. For nodes on rigid bodies, use the NODE option. Do not use the NODE option in *r*-adaptive problems because the node IDs may change during the adaptive step.

The SET_LINE option causes the inclusion of a generated node set comprising existing nodes and new nodes created from *h*-adaptive mesh refinement along the straight line connecting two specified nodes in prescribed boundary conditions.

The POINT_UVW, EDGE_UVW, FACE_XYZ, SET_POINT_UVW, SET_EDGE_UVW, and SET_FACE_XYZ options are used when prescribed boundary conditions are applied to IGA shells. The FACE_UVW and SET_FACE_UVW options are used when prescribing boundary motion to IGA solids.

Card Summary:

Card ID. Include this card if using the ID keyword option.

ID	HEADING

Card 1. This card is required.

TYPEID	DOF	VAD	LCID	SF	VID	DEATH	BIRTH

Card 2. Include this card for the SET_BOX keyword option.

BOXID	TOFFSET	LCBCHK					

Card 3. Additional card that is expected if |DOF| = 9, 10, or 11 or VAD = 4 on Card 1; otherwise skip this card.

OFFSET1	OFFSET2	LRB	NODE1	NODE2			

Card 4. Include this card for the SET_LINE keyword option.

NBEG	NEND						

Card 5. Include this card for the BNDOUT2DYNAIN keyword option.

PRMR							

Card 6. Include this card for the POINT_UVW, EDGE_UVW, FACE_UVW, FACE_XYZ, SET_POINT_UVW, SET_EDGE_UVW, SET_FACE_UVW, and SET_FACE_XYZ keyword options.

FORM	SFD						

Data Card Definitions:

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Prescribed motion set ID to which this node, node set, segment set, rigid body, parametric point, parametric point set, parametric edge, parametric edge set, physical face, physical face set, parametric face, or parametric face set belongs. This ID does not need to be unique.
HEADING	An optional descriptor for the given ID that will be written into the d3hsp file and the bndout file.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPEID	DOF	VAD	LCID	SF	VID	DEATH	BIRTH
Type	I	I	I	I	F	I	F	F
Default	none	none	0	none	1.0	0	10^{28}	0.0

VARIABLE	DESCRIPTION
TYPEID	Node ID (NID in *NODE), nodal set ID (SID in *SET_NODE), segment set ID (SID in *SET_SEGMENT, see DOF = 12), part ID (PID in *PART) for a rigid body, parametric point ID (PID in *IGA_POINT_UVW), parametric edge ID (EID in *IGA_EDGE_UVW), physical face ID (FID in *IGA_FACE_XYZ), parametric face ID (FID in *IGA_FACE_UVW), parametric point set ID (SID in *SET_IGA_POINT_UVW), parametric edge set ID (SID in *SET_IGA_EDGE_UVW), physical face set ID (SID in *SET_IGA_FACE_XYZ) or parametric face set ID (SID in *SET_IGA_FACE_UVW).
DOF	Applicable degrees of freedom: EQ.1: x-translational degree of freedom for Cartesian systems (local or global). For a cylindrical or spherical system, this is the local radial degree of freedom r. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See Remark 9. EQ.2: y-translational degree of freedom for Cartesian systems (local or global). For a cylindrical system, this is the local circumferential degree of freedom φ, and for a spherical system, it is the local latitude degree of freedom θ. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See Remark 9. EQ.3: z-translational degree of freedom for Cartesian systems (local or global). For a cylindrical system, this is the local z-translational degree of freedom, and for a spherical system, it is the local longitude degree of freedom φ. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See Remark 9. EQ.4: Translational motion in the direction given by the VID. Movement on a plane normal to the vector is permitted. EQ.-4: Translational motion in the direction given by the VID. Movement on a plane normal to the vector is not permitted. In explicit analyses, this option only applies to rigid bodies if |CMO| = 2 on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY. EQ.5: x-rotational degree of freedom. For a cylindrical or spherical system, this is the local radial degree of freedom r. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See

VARIABLE	DESCRIPTION
	Remark 9.
	EQ.6: y-rotational degree of freedom. For a cylindrical system, this is the local circumferential degree of freedom φ, and for a spherical system, it is the local latitude degree of freedom θ. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See Remark 9.
	EQ.7: z-rotational degree of freedom. For a cylindrical system, this is the local z-translational degree of freedom, and for a spherical system, it is the local longitude degree of freedom φ. To specify a local coordinate system, set VID to the negative of the local coordinate system ID. See Remark 9.
	EQ.8: Rotational motion about an axis, parallel to vector VID, passing through the center of gravity of the node, node set, or rigid body (or passing through a specified point when |CMO| = 2 on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY). Rotation about the normal axes, namely, axes in a plane normal to VID, is permitted.
	EQ.-8: Rotational motion about an axis, parallel to vector VID, passing through the center of gravity of the node, node set, or rigid body (or, in explicit analyses, passing through a specified point when |CMO| = 2 on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY). Rotation about the normal axes is not permitted. This option only applies to rigid bodies if |CMO| = 2.
	EQ.9: Rotation with axis parallel to the x-axis and passing through the yz-plane at y = OFFSET1 and z = OFFSET2. Radial motion is NOT permitted. This option is not applicable to rigid bodies.
	EQ.-9: Rotation with axis parallel to the x-axis and passing through the yz-plane at y = OFFSET1 and z = OFFSET2. Radial motion is permitted. This option is not applicable to rigid bodies.
	EQ.10: Rotation with axis parallel to the y-axis and passing through the zx-plane at z = OFFSET1 and x = OFFSET2. Radial motion is NOT permitted. This option is not applicable to rigid bodies.
	EQ.-10: Rotation with axis parallel to the y-axis and passing

VARIABLE	DESCRIPTION
	through the zx-plane at $z =$ OFFSET1 and $x =$ OFFSET2. Radial motion is permitted. This option is not applicable to rigid bodies.
	EQ.11: Rotation with axis parallel to the z-axis and passing through the xy-plane at $x =$ OFFSET1 and $y =$ OFFSET2. Radial motion is NOT permitted. This option is not applicable to rigid bodies.
	EQ.-11: Rotation with axis parallel to the z-axis and passing through the xy-plane at $x =$ OFFSET1 and $y =$ OFFSET2. Radial motion is permitted. This option is not applicable to rigid bodies.
	EQ.12: Translational motion in the direction given by the normals to the segments. This option is available only when using the SET_SEGMENT keyword option.
VAD	Velocity/Acceleration/Displacement flag:
	EQ.0: Velocity (rigid bodies and nodes)
	EQ.1: Acceleration (rigid bodies and nodes)
	EQ.2: Displacement (rigid bodies and nodes; see Remark 2)
	EQ.3: Velocity as a function of displacement. This option only applies to rigid bodies with |CMO| $\neq 2$ on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY. See Remark 3.
	EQ.4: Relative displacement: This option only applies to rigid bodies with |CMO| $\neq 2$ on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY. See Remark 4.
LCID	Curve ID or function ID to describe motion value as a function of time; see *DEFINE_CURVE, *DEFINE_CURVE_FUNCTION, or *DEFINE_FUNCTION. If LCID refers to *DEFINE_FUNCTION, the function has four arguments: time and x, y and z coordinates of the node or rigid body, such as $f(t,x,y,z) = 10.0 \times t + \max(x - 100,0.)$. If VAD = 2, the function has one argument ,which is time, such as $f(t) = 10.0 \times t$ (see Remark 2). See BIRTH below.
SF	Load curve scale factor. The default is 1.0.
VID	Direction and update option for the constraint.
	GT.0: Vector ID for DOF values of 4 or 8; see *DEFINE_VEC-

VARIABLE	DESCRIPTION
	TOR. The direction of this vector is not updated with time. LT.0: -VID is the coordinate system ID (see *DEFINE_COORDINATE_NODES and *DEFINE_COORDINATE_SYSTEM) for DOF values 1 through 3 when using a local coordinate system. See Remark 9 and Figure 17-10.
DEATH	Removal time for the imposed motion/constraint. For alternatives to DEATH and BIRTH for specifying the transitory application of prescribed motion, see the SET_BOX option, *DEFINE_DEATH_TIMES, or *SENSOR_CONTROL. EQ.0.0: Set to 1028, the default value.
BIRTH	Activation time for the imposed motion/constraint. The prescribed motion begins acting at time = BIRTH, but with the motion from the zero abscissa value of the curve or function (*DEFINE_FUNCTION). In other words, the abscissae are shifted by an amount BIRTH, such that it has the same effect as setting OFFA = BIRTH in *DEFINE_CURVE. Warning: BIRTH is ignored if the LCID is defined as a function, as in *DEFINE_CURVE_FUNCTION.

For the SET_BOX keyword option, define the following additional card.

Card 2	1	2	3	4	5	6	7	8
Variable	BOXID	TOFFSET	LCBCHK					
Type	I	I	I					
Default	none	0	0					

VARIABLE	DESCRIPTION
BOXID	A box ID defining a box region in space in which the constraint is activated. The prescribed motion will *only* be applied to the nodes falling inside the box. If the LCBCHK field is not defined, the box volume is reevaluated every time step to determine the nodes for which the prescribed motion is active. This reevaluation of the volume is referred to as a "box-check."

VARIABLE	DESCRIPTION
TOFFSET	Time offset flag for the SET_BOX option: EQ.0: No time offset is applied to LCID. EQ.1: The time value of the load curve, LCID, will be offset by the time when the node enters the box.
LCBCHK	Optional load curve allowing more flexible and efficient use of SET_BOX option. Instead of performing box-check at every time step, discrete box-check times could be given as x-values of LCBCHK. LCBCHK's y-values specify corresponding death times. For example, a curve with points $(20, 30)$ and $(50, 70)$ will result in two box checks. The first will occur at 20, and the prescribed motion will be active from 20 to 30. The second will occur at 50, and the prescribed motion will be active from 50 to 70. A y-value of "0" means the prescribed motion will stay active until next box-check. For example, an additional 3rd point of $(90, 0)$ will lead to another box-check at 90, and the prescribed motion will be active from 90 until the end of the simulation.

Additional card that is expected if |DOF| = 9, 10, or 11 or VAD = 4 on Card 1; otherwise skip this card.

Card 3	1	2	3	4	5	6	7	8
Variable	OFFSET1	OFFSET2	LRB	NODE1	NODE2			
Type	F	F	I	I	I			
Default	0.	0.	0	0	0			

VARIABLE	DESCRIPTION
OFFSET1	Offset for DOF types 9 - 11 (y, z, x directions)
OFFSET2	Offset for DOF types 9 - 11 (z, x, y directions)
LRB	Lead rigid body for measuring the relative displacement (for VAD = 4).
NODE1	Optional orientation node, n_1, for relative displacement (for VAD = 4).

VARIABLE	DESCRIPTION
NODE2	Optional orientation node, n_2, for relative displacement (for VAD = 4).

For the SET_LINE keyword option, include the following card.

Card 4	1	2	3	4	5	6	7	8
Variable	NBEG	NEND						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
NBEG	Node ID of a starting node
NEND	Node ID of an ending node. All existing nodes and new nodes generated by *h*-adaptive mesh refinement along the straight line connecting NBEG and NEND will be included in the prescribed boundary motions.

For the BNDOUT2DYNAIN keyword option, include the following card.

Card 5	1	2	3	4	5	6	7	8
Variable	PRMR							
Type	A							
Default	none							

VARIABLE	DESCRIPTION
PRMR	String representing the name of the parameter to be output to the dynain file. Its value will be the reaction force from the constraint at the end of the simulation. PRMR is only an A9 string because when the dynain file is read the rules for parameters apply (see *PARAMETER for details). Thus, make sure to not use the first character position.

For the POINT_UVW, EDGE_UVW, FACE_XYZ, FACE_UVW, SET_POINT_UVW, SET_EDGE_UVW, SET_FACE_XYZ, and SET_FACE_UVW keyword options, include the following card (see Remark 8).

Card 6	1	2	3	4	5	6	7	8
Variable	FORM	SFD						
Type	I	F						
Default	0	1.0						

VARIABLE	DESCRIPTION
FORM	Formulation type: EQ.0: Penalty method
SFD	Scale factor for penalty stiffness

Remarks:

1. **Rotational degrees of freedom.** When DOF = 5, 6, 7, or 8, nodal rotational degrees of freedom are prescribed in the case of deformable nodes (*OPTION1* = NODE or SET), whereas body rotations are prescribed in the case of a rigid body (*OPTION1* = RIGID). In the case of a rigid body, the axis of prescribed rotation passes through the body's center of mass, unless |CMO| = 2 on *MAT_RIGID or *CONSTRAINED_NODAL_RIGID_BODY. For |DOF| = 8, the axis of the prescribed rotation is parallel to vector VID. To prescribe a body rotation of a set of deformable nodes, with the axis of rotation parallel to global axes x, y, or z, use *OPTION1* = SET with |DOF| = 9, 10, or 11, respectively. The load curve scale factor can be used for simple modifications or unit adjustments.

2. **Limitation for prescribed displacement.** VAD = 2 cannot be used if LCID references a *DEFINE_CURVE_FUNCTION which involves anything other than arithmetic functions or the variable "time". This limitation is because velocities must be computed from the displacement curve using an interpolation scheme, which requires data for a future time step; those future data are unknown if the curve function involves a simulation response.

3. **Velocity as a function of displacement.** For VAD = 3, the LCID gives velocity as a function of displacement for the rigid body in the direction specified by DOF.

4. **Relative displacement.** The relative displacement prescribed motion (VAD = 4) can be measured in either of two ways:

   a) Along a straight line between the mass centers of the rigid bodies.

   b) Along a vector beginning at node n_1 and terminating at node n_2.

   With the first option, a positive displacement will move the rigid bodies further apart, and, likewise, a negative motion will move the rigid bodies closer together. The mass centers of the rigid bodies must not be coincident when this option is used. With the second option, the relative displacement is measured along the vector, and the rigid bodies may be coincident. Note that the motion of the lead rigid body is not directly affected by VAD = 4, that is, no forces are generated on the lead rigid body.

5. **Activation time.** The activation time, BIRTH, is the time during the solution that the constraint begins to act. Until this time, the prescribed motion card is ignored. The function value of the load curves will be evaluated at the offset time given by the difference of the solution time and BIRTH (solution time - BIRTH). Relative displacements that occur before reaching BIRTH are ignored. Only relative displacements that occur after BIRTH are prescribed.

6. **Rigid body nodes.** When the node is on a rigid body, LS-DYNA imposes the translational motion without altering the angular velocity of the rigid body by calculating the appropriate translational velocity for the center of mass of the rigid body using the equation:

$$\mathbf{v}_{\text{cm}} = \mathbf{v}_{\text{node}} - \boldsymbol{\omega} \times (\mathbf{x}_{\text{cm}} - \mathbf{x}_{\text{node}}) .$$

   Here $\mathbf{v}_{\text{cm}}$ is the velocity of the center of mass, $\mathbf{v}_{\text{node}}$ is the specified nodal velocity, $\boldsymbol{\omega}$ is the angular velocity of the rigid body, $\mathbf{x}_{\text{cm}}$ is the current coordinate of the mass center, and $\mathbf{x}_{\text{node}}$ is the current coordinate of the nodal point. Extreme care must be used when prescribing motion of a rigid body node. Typically, for nodes on a given rigid body, the motion of no more than one node should be prescribed or unexpected results may be obtained.

7. **Rigid body rotations.** When the RIGID option is used to prescribe rotation of a rigid body in explicit analysis, the axis of rotation will, unless |CMO| = 2, be shifted such that it passes through the center of mass of the rigid body. To override the motion around the internally-calculated center-of-mass location, set |CMO| = 2 (see *MAT_RIGID and *CONSTRAINED_NODAL_RIGID_BODY) or use *PART_INERTIA or *CONSTRAINED_NODAL_RIGID_BODY_INERTIA. Note, |CMO| = 2 is not available for implicit simulations. Instead, constraints can be applied to any of the nodes of the rigid body and the rotation center is not limited to the center of mass.

When the RIGID_LOCAL option is invoked, the orientation of the local coordinate system rotates with time in accordance with rotation of the rigid body. The local coordinate system is specified with LCO and CID in *MAT_RIGID and *CONSTRAINED_NODAL_RIGID_BODY, respectively. If LCO/CID is 0, the local coordinate system defaults to the principal inertia directions of the rigid body (refer to "principal directions" in the mass summary section of the d3hsp file).

Angular displacements are applied in an incremental fashion; hence, a successive set of rotations about multiple axes cannot be correctly prescribed. In light of this, the command *BOUNDARY_PRESCRIBED_ORIENTATION_RIGID should be used for the purpose of prescribing the general orientation of a rigid body.

8. **Prescribed boundary conditions on IGA shells and solids.** The keyword options for boundary conditions on IGA shells (POINT_UVW, EDGE_UVW, FACE_XYZ, SET_POINT_UVW, SET_EDGE_UVW, and SET_FACE_XYZ) support DOF = 1 to 8. For IGA solids, the available keyword options (FACE_UVW and SET_FACE_UVW) only support DOF = 1 to 4 because solid IGA elements do not have rotational degrees of freedom. For both IGA shells and solids, the prescribed boundary motion is supported for VAD = 0 to 2. The penalty parameter is set to the Young modulus of the corresponding IGA shell or solid.

   For boundary conditions to be applied to IGA shells, the point and edge IDs specified in TYPEID must be included in the parametric point set (PSID) and parametric edge set (ESID) in *IGA_FACE_XYZ of the corresponding *IGA_SHELL. This does not apply to the edges already in *IGA_1D_BREP.

   To specify the prescribed motion boundary conditions on IGA solids with a set of parametric faces (SET_FACE_UVW), field FSID on *IGA_VOLUME_XYZ must refer to the parametric face set (*SET_IGA_FACE_UVW).

   When boundary conditions are prescribed for physical faces and parametric edges, the boundary conditions are directly imposed on the control points if these faces are untrimmed and these edges are the actual edges of the patch. In this case, FORM and SFD are ignored.

9. **Prescribing motion along a direction in a local coordinate system.** Applying translational and rotational prescribed motion along a direction in a local coordinate system is available for DOF = 1, 2, 3, 5, 6, and 7. Prescribing rotational nodal motion in cylindrical or spherical systems may only make sense in special cases and should be used with caution. The local coordinate system can be Cartesian, cylindrical, or spherical. -VID references the local coordinate system specified with *DEFINE_COORDINATE_NODES or *DEFINE_COORDINATE_SYSTEM. Field TYPE on those keywords sets the kind of coordinate system. The input for *DEFINE_COORDINATE_NODES and *DEFINE_COORDI-

NATE_SYSTEM defines a local Cartesian coordinate system as shown in Figure 17-10. TYPE specifies whether to cylindrical coordinates or spherical coordinates from this basis.

Examples:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *BOUNDARY_PRESCRIBED_MOTION_SET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A set of nodes is given a prescribed translational velocity in the
$  x-direction according to a specified vel-time curve (which is scaled).
$
*BOUNDARY_PRESCRIBED_MOTION_SET
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid       dof       vad      lcid        sf       vid     death
         4         1         0         8       2.0
$
$     nsid = 4    nodal set ID number, requires a *SET_NODE_option
$      dof = 1    motion is in x-translation
$      vad = 0    motion prescribed is velocity
$     lcid = 8    velocity follows load curve 8, requires a *DEFINE_CURVE
$       sf = 2.0  velocity specified by load curve is scaled by 2.0
$      vid        not used in this example
$    death        use default (essentially no death time for the motion)
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$

$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *BOUNDARY_PRESCRIBED_MOTION_RIGID
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A rigid body is given a prescribed rotational displacement about the
$  z-axis according to a specified displacement-time curve.
$
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid       dof       vad      lcid        sf       vid     death
        84         7         2         9                          14.0
$
$      pid = 84   apply motion to part number 84
$      dof = 7    rotation is prescribed about the z-axis
$      vad = 2    the prescribed motion is displacement (angular)
$     lcid = 9    rotation follows load curve 9, requires a *DEFINE_CURVE
$                     (rotation should be in radians)
$       sf        use default (sf = 1.0)
$      vid        not used in this example
$    death = 14   prescribed motion is removed at 14 ms (assuming time is in ms)
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

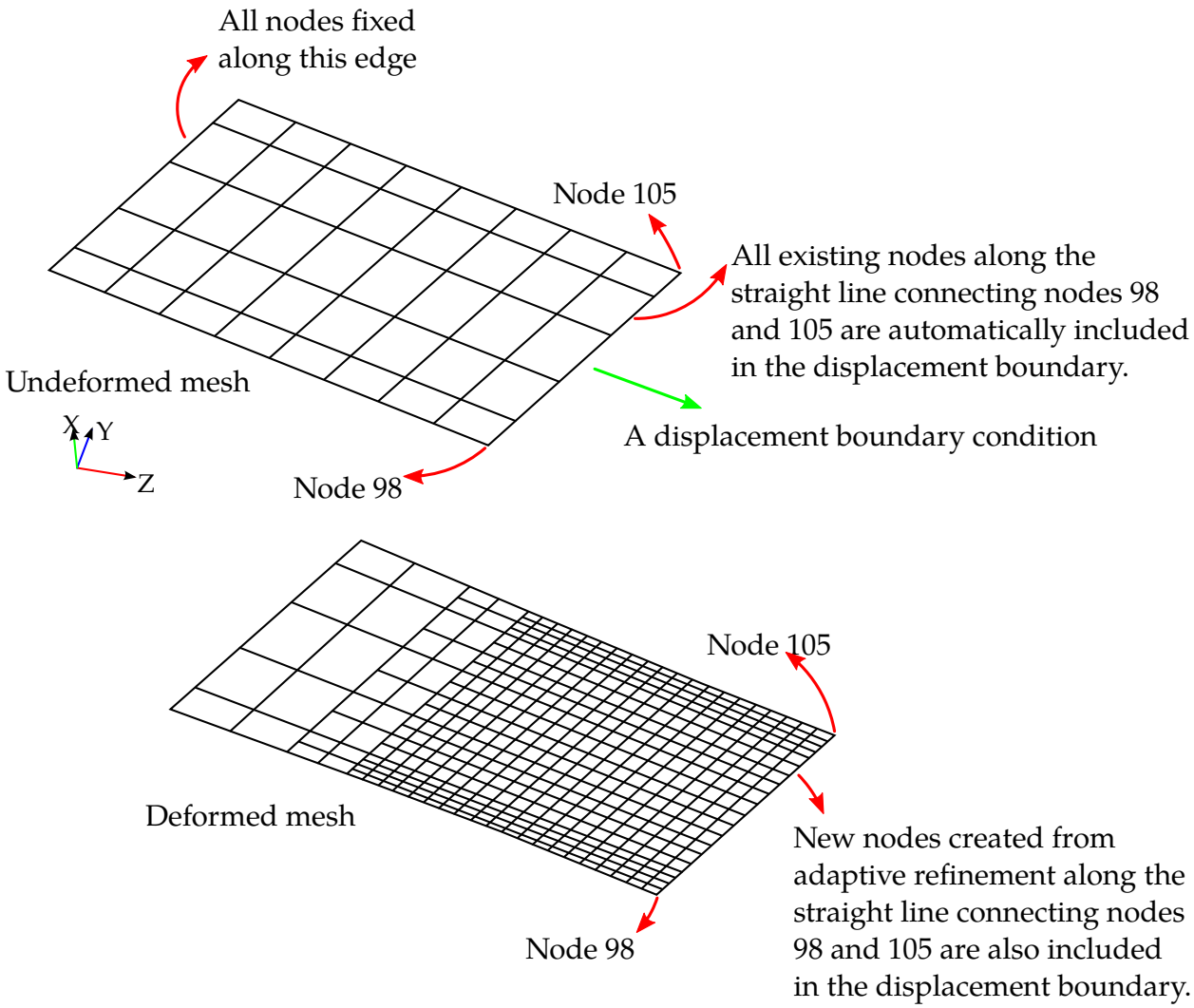


Figure 5-4. Example using the SET_LINE option.

An example using the SET_LINE option is shown below. The example is of a flat plate which is fixed on one edge and pulled using a velocity boundary condition on the opposite edge as indicated in Figure 5-4. The velocity boundary condition is applied to the line connecting nodes 98 and 105 which includes the nodes created from *h*-adaptive mesh refinement on that line.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
*BOUNDARY_PRESCRIBED_MOTION_SET_LINE
$#     nsid       dof       vad      lcid        sf       vid     death     birth
       122         3          0         2
$      NBEG     NEND
         98      105
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
*DEFINE_CURVE
2
                 0.0              &velo
             &endtime             &velo
               1000.0             &velo
```

BOUNDARY_PRESCRIBED_ORIENTATION_RIGID_*OPTION1_{OPTION2}

Available options for *OPTION1* include:

DIRCOS

ANGLES

EULERP

VECTOR

OPTION2 allows an optional ID:

ID

The defined ID can be referred to by *SENSOR_CONTROL.

Purpose: Prescribe the orientation of rigid body as a function of time.

Card Summary:

Card ID. Include this card if using the ID option.

ID	HEADING

Card 1. This card is required.

PIDB	PIDA	INTRP	BIRTH	DEATH	TOFFSET		

Card 2a.1. Include this card if using the DIRCOS option.

LCIDC11	LCIDC12	LCIDC13	LCIDC21	LCIDC22	LCIDC23	LCIDC31	LCIDC32

Card 2a.2. Include this card if using the DIRCOS option.

	LCIDC33						

Card 2b. Include this card if using the ANGLES option.

LCIDQ1	LCIDQ2	LCIDQ3	ISEQ	ISHFT	BODY		

Card 2c. Include this card if using the EULERP option.

LCIDE1	LCIDE2	LCIDE3	LCIDE4				

Card 2d. Include this card if using the VECTOR option.

LCIDV1	LCIDV2	LCIDV3	LCIDS	VALSPIN			

Data Cards:

ID Card. Optional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Optional ID for PRESCRIBED ORIENTATION that can be referred to by *SENSOR_CONTROL. When not defined, the sequential definition order will be used as ID when referred to by *SENSOR_CONTROL.
HEADING	An optional descriptor for the given ID

Card 1	1	2	3	4	5	6	7	8
Variable	PIDB	PIDA	INTRP	BIRTH	DEATH	TOFFSET		
Type	I	I	I	F	F	I		
Default	none	↓	1	0.	10^{20}	0		

VARIABLE	DESCRIPTION
PIDB	Part ID for rigid body B whose orientation is prescribed
PIDA	Part ID for rigid body A. The orientation of PIDB is measured with respect to the coordinate system of PIDA, as defined by LCO on *MAT_RIGID. If zero, then the orientation of PIDB is measured with respect to the global reference frame except for when BODY = 1 for the ANGLES option; see Remark 5.

VARIABLE	DESCRIPTION
INTRP	Interpolation method used on time history curves: EQ.1: Linear interpolation (default)
BIRTH	Prior to this time the body moves freely under the action of other agents.
DEATH	The body is freed at this time and subsequently allowed to move under the action of other agents.
TOFFSET	Time offset flag: EQ.0: No time offset is applied. EQ.1: The time value of all load curves will be offset by the birth time.

Cosine Card 1. Additional card for DIRCOS option.

Card 2a.1	1	2	3	4	5	6	7	8
Variable	LCIDC11	LCIDC12	LCIDC13	LCIDC21	LCIDC22	LCIDC23	LCIDC31	LCIDC32
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Cosine Card 2. Additional card for DIRCOS option.

Card 2a.2	1	2	3	4	5	6	7	8
Variable	LCIDC33							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
LCIDC*ij*	Load curve ID specifying direction cosine C_{ij} as a function of time. C_{ij} is defined as:

VARIABLE	DESCRIPTION
	$$C_{ij} = \mathbf{a}_i \cdot \mathbf{b}_j$$ where the $\{\mathbf{a}_i\}$ are mutually perpendicular unit vectors fixed in PIDA and the $\{\mathbf{b}_j\}$ are mutually perpendicular unit vectors fixed in PIDB. If PIDA = 0 then the $\{\mathbf{a}_i\}$ are unit vectors aligned with the global x, y, and z. See Remark 1.

Angles Card. Additional card for ANGLES option, see Remark 5.

Card 2b	1	2	3	4	5	6	7	8
Variable	LCIDQ1	LCIDQ2	LCIDQ3	ISEQ	ISHFT	BODY		
Type	I	I	I	I	I	I		
Default	none	none	none	none	1	0		

VARIABLE	DESCRIPTION
LCIDQ*i*	Load curve ID specifying the orientation angle q_i in radians as a function of time. See Remark 1.
ISEQ	Specifies the sequence in which the rotations are performed. In this first set of sequences, three unique axes are involved. This sequence is associated with what are commonly called Cardan or Tait-Bryan angles. *All angles must be in units of radians.* Whether these rotations are intrinsic or extrinsic is determined by the BODY field. EQ.123: The first rotation is performed about the x-axis through an angle of q_1, the second about the y-axis through an angle of q_2, and the third about the z-axis through an angle of q_3. EQ.231: The first rotation is performed about the y-axis through an angle of q_1, the second about the z-axis through an angle of q_2, and the third about the x-axis through an angle of q_3. EQ.312: The first rotation is performed about the z-axis through an angle of q_1, the second about the x-axis through an angle of q_2, and the third about the y-axis through an angle of q_3.

VARIABLE	DESCRIPTION

EQ.132: The first rotation is performed about the x-axis through an angle of q_1, the second about the z-axis through an angle of q_2, and the third about the y-axis through an angle of q_3.

EQ.213: The first rotation is performed about the y-axis through an angle of q_1, the second about the x-axis through an angle of q_2, and the third about the z-axis through an angle of q_3.

EQ.321: The first rotation is performed about the z-axis through an angle of q_1, the second about the y-axis through an angle of q_2, and the third about the z-axis through an angle of q_3.

The second set of sequences involve only two unique axes where the first and third are repeated. This sequence is associated with what are commonly called Euler angles.

EQ.121: The first rotation is performed about the x-axis through an angle of q_1, the second about the y-axis through an angle of q_2, and the third about the x-axis through an angle of q_3.

EQ.131: The first rotation is performed about the x-axis through an angle of q_1, the second about the z-axis through an angle of q_2, and the third about the x-axis through an angle of q_3.

EQ.212: The first rotation is performed about the y-axis through an angle of q_1, the second about the x-axis through an angle of q_2, and the third about the y-axis through an angle of q_3.

EQ.232: The first rotation is performed about the y-axis through an angle of q_1, the second about the z-axis through an angle of q_2, and the third about the y-axis through an angle of q_3.

EQ.313: The first rotation is performed about the z-axis through an angle of q_1, the second about the x-axis through an angle of q_2, and the third about the z-axis through an angle of q_3.

EQ.323: The first rotation is performed about the z-axis through an angle of q_1, the second about the y-axis through an angle of q_2, and the third about the z-axis through an angle of q_3.

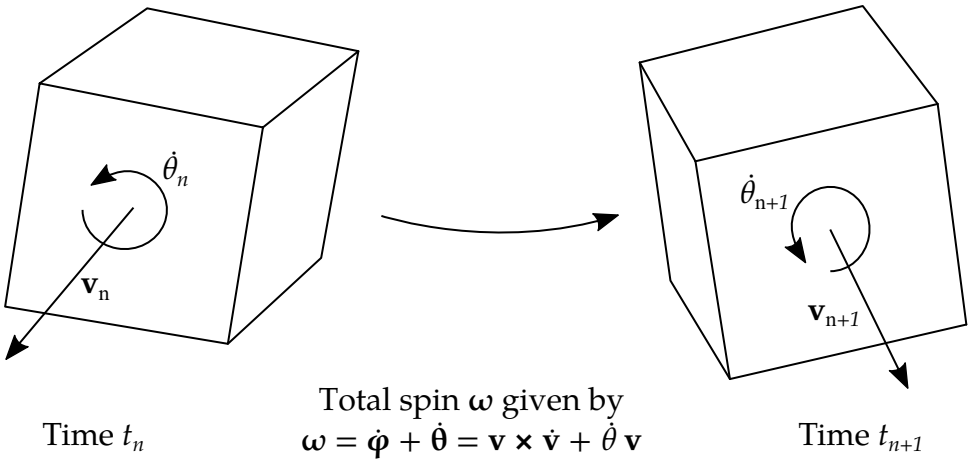


Figure 5-5. Orientation of rigid body computed using the VECTOR option.

VARIABLE	DESCRIPTION
ISHFT	Angle shift. EQ.1: Angle curves are unaltered. EQ.2: Shifts angle data in the LCIDQ*i* curves as necessary to eliminate discontinuities. If angles are confined to the range $[-\pi, \pi]$ and the data contains excursions exceeding π, then set ISHFT = 2.
BODY	Reference axes: EQ.0: Rotations are performed about axes fixed to PIDA (extrinsic rotation, default). The coordinate system in question is the one defined by LCO on *MAT_RIGID for body A. EQ.1: Rotations are performed about axes fixed to PIDB (intrinsic rotation). The coordinate system in question is the one defined by LCO on *MAT_RIGID for body B.

Euler Parameter Card. Additional card for EULERP option.

Card 2c	1	2	3	4	5	6	7	8
Variable	LCIDE1	LCIDE2	LCIDE3	LCIDE4				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
LCIDEi	Load curve ID specifying Euler parameter e_i as a function of time. See Remark 1. The Euler parameters are defined as follows: $$\varepsilon_i = \boldsymbol{\varepsilon} \cdot \mathbf{a}_i = \boldsymbol{\varepsilon} \cdot \mathbf{b}_i, \qquad (i = 1,2,3)$$ $$\varepsilon_4 = \cos\left(\frac{\theta}{2}\right)$$ where $\boldsymbol{\varepsilon}$ is the Euler vector, $\{\mathbf{a}_i\}$ and $\{\mathbf{b}_i\}$ are dextral sets of unit vectors fixed in PIDA and PIDB, respectively, and θ (in radians) is associated with the rotation of PIDB in PIDA about Euler vector. If PIDA = 0, then the $\{\mathbf{a}_i\}$ are unit vectors aligned, respectively, with the global x, y, and z-axes. See Remark 2.

Vector Card. Additional card for VECTOR option.

Card 2d	1	2	3	4	5	6	7	8
Variable	LCIDV1	LCIDV2	LCIDV3	LCIDS	VALSPIN			
Type	I	I	I	I	F			
Default	none	none	none	0	0.			

VARIABLE	DESCRIPTION
LCIDVi	Load curve ID specifying the vector measure number v_i as a function of time. See Remark 1. The vector measure numbers are defined as follows: $$v_i = \mathbf{v} \cdot \mathbf{n}_i, \qquad i = 1,2,3.$$ where $\mathbf{v}$ is a vector and $\{\mathbf{n}_i\}$ are unit vectors aligned, respectively, with the global axes x, y, and z-axes. Note that the vector $\mathbf{v}$ is attached to the body in question, so changing the direction of this vector will induce a rotation of the body defined by $\dot{\boldsymbol{\varphi}} = \mathbf{v} \times \dot{\mathbf{v}}$. See Figure 5-5.
LCIDS	Load curve ID which specifies the overlayed spin speed $\dot{\theta}$ of PIDB about the axis parallel to the vector $\mathbf{v}$. EQ.0: A constant spin speed as defined by VALSPIN is used. GT.0: Load curve for spin speed (radians per unit time)

VARIABLE	DESCRIPTION
VALSPIN	Value for the constant spin speed of PIDB (radians per unit time $\dot{\theta}$). This field is ignored if the load curve number defined above is non-zero.

Remarks:

1. **Load Curves.** All load curves must contain the same number of points and the data must be uniformly spaced.

2. **Euler Parameters.** The Euler parameters are defined as

$$\boldsymbol{\varepsilon} = \sin\left(\frac{\theta}{2}\right)\mathbf{n}$$

$$\varepsilon_4 = \cos\left(\frac{\theta}{2}\right)$$

where $\mathbf{n}$ is a unit vector defining the axis of rotation and θ is the angle with which the rotation occurs. Consequently, the four parameters are subjected to the condition

$$\boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} + \varepsilon_4^2 = 1\ .$$

It is therefore recommended that the control points of the curves already fulfil this or else LS-DYNA will internally normalize these values. From the Euler parameters at time t, a unique rotation matrix $\mathbf{Q}_t$ is computed that is used to determine the total orientation $\mathbf{Q}$. The rotation is performed with respect to the *reference state* $\mathbf{Q}_0$ given by the Euler parameters at time 0. In general, $\mathbf{Q}_0 \neq \boldsymbol{I}$ and the rotation of the rigid body is given by $\mathbf{Q} = \mathbf{Q}_t\mathbf{Q}_0^T$. If the parameters are initially $\boldsymbol{\varepsilon} = \mathbf{0}$ and $\varepsilon_4 = 1$, then the reference state is $\mathbf{Q}_0 = \mathbf{I}$ and $\mathbf{Q} = \mathbf{Q}_t$ defines the orientation of the rigid body.

For a nonzero PIDA, the rotation matrix $\mathbf{Q}$ as defined above is expressed in a system that is fixed in rigid body A. If this system is denoted $\mathbf{R}_t$ at time t, and assuming $\mathbf{R}_0 = \mathbf{I}$, the orientation with respect to a global system is $\mathbf{RQ}$.

3. **Coordinate System.** LC0 in *MAT_RIGID must be used to identify a coordinate system for each rigid body. The coordinate system must be defined with *DEFINE_COORDINATE_NODES and FLAG = 1. Nodes used in defining the coordinate system must reside on the same body.

4. **Incompatibilities.** This feature is incompatible with *DEFINE_CURVE_FUNCTION.

5. **ANGLES Option.** For the ANGLES option we assume that bodies A and B are defined by local systems attached to the bodies, $\mathbf{R}_a$ for body A and consequently

$\mathbf{R}_b$ for body B, and we intend to determine the latter. The initial configurations are determined by the parameter LCO on each respective *MAT_RIGID card. If this parameter is absent, the initial system is taken as the configuration of principal inertia directions, so we recommend defining it whenever PIDA or BODY are nonzero. We denote the initial ($t = 0$) configuration matrices for each body $\mathbf{R}_A$ and $\mathbf{R}_B$, respectively. We, also, define rotation matrices $\mathbf{Q}_i$, $i = 1, 2, 3$ as functions of time according to the formula

$$\mathbf{Q}_i = \cos q_i\left(\mathbf{u}_i\mathbf{u}_i^T + \mathbf{v}_i\mathbf{v}_i^T\right) + \sin q_i\left(\mathbf{v}_i\mathbf{u}_i^T - \mathbf{u}_i\mathbf{v}_i^T\right) + \mathbf{w}_i\mathbf{w}_i^T \ .$$

Here $\{\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i\}$ is an arbitrary orthonormal system with $\mathbf{w}_i$ being the rotation direction associated with the parameter ISEQ. *This system is fixed in time*. Furthermore, q_i is the rotation angle associated with LCIDQ*i and is thus time dependent*. More specifically, for a "1" in ISEQ we have $\mathbf{w}_i^T = (1 \quad 0 \quad 0)$, for a "2" we have $\mathbf{w}_i^T = (0 \quad 1 \quad 0)$, and for a "3" we have $\mathbf{w}_i^T = (0 \quad 0 \quad 1)$. Depending on the values of BODY and PIDA, we define the configuration of body B at time t as

$$\mathbf{R}_b = \begin{cases} \mathbf{Q}_3\mathbf{Q}_2\mathbf{Q}_1\mathbf{R}_B & \text{if BODY} = 0 \text{ and PIDA} = 0 \\ \mathbf{R}_a\mathbf{Q}_3\mathbf{Q}_2\mathbf{Q}_1\mathbf{R}_A^T\mathbf{R}_B & \text{if PIDA} \neq 0 \\ \mathbf{R}_B\mathbf{Q}_1\mathbf{Q}_2\mathbf{Q}_3 & \text{if BODY} = 1 \text{ and PIDA} = 0 \end{cases}$$

BOUNDARY_PRESSURE_OUTFLOW_*OPTION

Available options include:

SEGMENT

SET

Purpose: Define pressure outflow boundary conditions. These boundary conditions are attached to solid elements using the Eulerian ambient formulation (refer to ELFORM in *SECTION_SOLID_ALE) and defined to be pressure outflow ambient elements (refer to AET in *SECTION_SOLID_ALE).

Card 1 for SET option.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID							
Type	I							
Default	none							

Card 1 for SEGMENT option.

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
SSID	Segment set ID
N1, N2, …	Node ID's defining segment

***BOUNDARY_PWP**_*OPTION1_{OPTION2}*

Purpose: Define pressure boundary conditions for pore water, such as at soil surface. The TABLE/TABLE_SET option applies to a whole part/part set, while the other options apply to specified nodes.

OPTION1 is required. The available choices for *OPTION1* are:

NODE

SET

TABLE

TABLE_SET

OPTION2 is optional. It allows an optional ID to be given that applies to this boundary condition:

ID

If defined, the ID may be referenced by *SENSOR_CONTROL to activate and/or deactivate the boundary condition. The ID has no other function in LS-DYNA.

Card Summary:

Card ID. This card is included if *OPTION2* is ID

ID							

Card 1a. This card is included if *OPTION1* is NODE or SET.

NID	LC	CMULT	LCDR	TBIRTH	TDEATH		

Card 1b. This card is included if *OPTION1* is TABLE or TABLE_SET.

PID				TBIRTH	TDEATH		

Card 2a. This card is included if *OPTION1* is NODE or SET

IPHRE	ITOTEX	IDRFLAG		LCLEAK	CLEAK	LCPUMP	TYPPMP

Card 2b. This card is included if *OPTION1* is TABLE or TABLE_SET

	ITOTEX		TABLE				

Data Card Definitions:

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
ID	ID for this boundary condition

Node Card. This card is included if the keyword *OPTION1* is NODE or SET.

Card 1a	1	2	3	4	5	6	7	8
Variable	NID	LC	CMULT	LCDR	TBIRTH	TDEATH		
Type	I	F	F	I	F	F		
Default	none	none	0.0	none	0.0	10^{20}		

VARIABLE	DESCRIPTION
NID	Node ID (option = NODE) or node set ID (option = SET)
LC	Load curve or function giving pore water pressure head (length units) as a function of time (see Remark 5). EQ.0: Constant pressure head assumed equal to CMULT.
CMULT	Factor on curve or constant pressure head if LC = 0
LCDR	Load curve or function giving pore water pressure head during dynamic relaxation. EQ.0: During dynamic relaxation, use first pressure head value on LC.

VARIABLE	DESCRIPTION
TBIRTH	Time at which boundary condition becomes active
TDEATH	Time at which boundary condition becomes inactive

Table Card. This card is included if the keyword *OPTION1* is TABLE or TABLE_SET.

Card 1b	1	2	3	4	5	6	7	8
Variable	PID				TBIRTH	TDEATH		
Type	I				F	F		
Default	none				0.0	10^{20}		

VARIABLE	DESCRIPTION
PID	Part ID (option = TABLE) or part set ID (option = TABLE_SET)
TBIRTH	Time at which boundary condition becomes active
TDEATH	Time at which boundary condition becomes inactive

This card is included if the keyword *OPTION1* is NODE or SET.

Card 2a	1	2	3	4	5	6	7	8
Variable	IPHRE	ITOTEX	IDRFLAG		LCLEAK	CLEAK	LCPUMP	TYPPMP
Type	I	I	I		I	F	I	I
Default	0	0	0		0	0.0	optional	0

VARIABLE	DESCRIPTION
IPHRE	Phreatic behavior option: EQ.0: Default behavior EQ.1: For phreatic behavior (water can be removed by the boundary condition but not added, such as at a sloping free surface). See Remark 7.

VARIABLE	DESCRIPTION
ITOTEX	Flag for the type of pressure boundary condition (see Remark 8): EQ.0: Total head EQ.1: Excess head EQ.2: Hydraulic head EQ.4: z-coord where head = 0 (piezometric level)
IDRFLAG	Active flag: EQ.0: Active only in transient analysis EQ.1: Active only in dynamic relaxation EQ.2: Active in all analysis phases
LCLEAK	Optional load curve ID (see *DEFINE_CURVE) giving the area of the hole through which pore fluid leaks to the boundary condition. See Remark 9.
CLEAK	Discharge coefficient, applicable when LCLEAK is nonzero
LCPUMP	Optional load curve ID (see *DEFINE_CURVE) giving the volumetric outflow rate per node or the upper limit on the volume flow rate, depending on the choice of TYPPMP. The curve x-axis is time while the y-axis is in units of volume per unit time. See Remarks 11 and 12.
TYPPMP	Determines the action of LCPUMP (see Remarks 11 and 12): EQ.0: LCPUMP defines the volume flow rate to the boundary condition. LC, CVAL, and ITOTEX are ignored. EQ.1: LCPUMP defines an upper limit on volume flow rate to the boundary condition; only outflow is permitted. EQ.2: Same as 1, except that inflow is permitted as well as outflow.

This card is included if the keyword *OPTION1* is TABLE or TABLE_SET.

Card 2b	1	2	3	4	5	6	7	8
Variable		ITOTEX		TABLE				
Type		I		I				
Default		0		0				

VARIABLE	DESCRIPTION
ITOTEX	Flag for type of pressure boundary condition (see Remark 8): EQ.0: Total head EQ.1: Excess head EQ.2: Hydraulic head
TABLE	Table ID for TABLE/TABLE_SET option only. See Remark 6.

Remarks:

1. **Pressure Dimensions.** Pressure should be given as pressure head in length units. Pressure head is defined as pressure/ρg where ρ is the fluid density given on *BOUNDARY_PORE_FLUID and g is the acceleration due to gravity given on *CONTROL_PORE_FLUID.

2. **Applicability of NODE and SET options.** The NODE and SET options should be applied only to nodes belonging to parts whose analysis type (see ATYPE and ACURVE on *BOUNDARY_PORE_FLUID) is Undrained, Time-dependent consolidation or Steady-state consolidation. The NODE and SET options have no effect on the pore pressure in Drained parts.

3. **Applicability of TABLE and TABLE_SET options.** The TABLE and TABLE_SET options should be used *only* with Drained parts. They have no effect on the pore pressure in parts with other pore fluid analysis types.

4. **Nodal Boundary Conditions.** *BOUNDARY_PWP_NODE or SET overrides pressure head from *BOUNDARY_PWP_TABLE or TABLE_SET at nodes where both are present. This situation can arise at the boundary between Drained and Undrained parts.

5. **Input Arguments for LC Function.** If LC is a *DEFINE_FUNCTION, the input arguments are $(\text{time}, x, y, z, x0, y0, z0)$, where x, y, and z are the current coordinates and $x0$, $y0$, and $z0$ are the initial coordinates of the node.

6. **TABLE and TABLE_SET Options.** The table consists of a list of times in ascending order, followed immediately by curves of z-coordinate versus pore pressure head. Each curve represents the pore water pressure head distribution with z-coordinate at the corresponding time. There must be the same number of curves as time values, arranged immediately after the *DEFINE_TABLE and in the correct order to correspond to the time values. Each curve should be arranged in ascending order of z-coordinate – they look upside-down on the page. The z-coordinate is the x-axis of the curve, the pore water pressure head (in length units) is the y-axis. Each curve should have the same z-coordinates (x-values) and use the same value of LCINT. Ensure that the range of z-coordinates in the curve exceeds by at least 5% the range of z-coordinates of the nodes belonging to the parts to which the boundary condition is applied.

7. **IPHRE.** "Phreatic" means that water can be removed by the boundary condition but not added. The boundary condition enforces the pressure head being less than or equal to zero. This condition occurs when the free surface of the soil is sloping so that any water emerging from the soil runs away down the slope. When IPHRE is nonzero, the input fields LC, CMULT, LCDR and ITOTEX are ignored. See also Remark 9 below.

8. **ITOTEX.** Descriptions for the different types of pressure conditions are as follows:

   a) For ITOTEX = 0, the value from the curve or table is total head. This may be used with any pore pressure analysis type.

   b) For ITOTEX = 1, the value from the curve or table is excess head. Total head will be determined by adding the hydrostatic head. This option cannot be used with drained analysis, which sets excess head to zero.

   c) For ITOTEX = 2, the value from the curve or table is hydraulic head, to which excess head may be added due to volume change in the soil if the analysis type is not drained.

   d) For ITOTEX = 4, the curve value is the z-coordinate of the water surface; pore pressure head at any node in this boundary condition, $h_{BC,node}$, is given by,

   $$h_{BC,node} = z_{\text{surface}} - z_{\text{node}}$$

   This option allows a single boundary condition to be used for nodes at any depth, provided that the pressure distribution is hydrostatic below the

given surface. This option is not available for the TABLE and TABLE_SET options.

9. **Leakage through a hole.** If LCLEAK is nonzero, the boundary condition models flow through a hole with the pressure head outside the hole, h_{ext}, determined by LC, CVAL, and ITOTEX. The size of the hole and the pressure head difference across the hole determine the flow rate. When h_{ext} is zero or negative (e.g., if LC, CVAL, and ITOTEX are left blank), the space outside the hole is assumed to be dry. In this case, if the pressure head at the node is positive, fluid flows out to the boundary condition. However, if the pressure head at the node is negative, no inflow can occur, and the boundary condition has no effect. In contrast, when h_{ext} is greater than zero, the space outside the hole is assumed to be a fluid-filled reservoir. Thus, either outflow or inflow occurs, depending on whether the pressure head at the node, h, is greater than or less than h_{ext}.

   At time t, the y-axis value of LCLEAK, $A(t)$, is the area of the hole, while CLEAK is a non-dimensional discharge coefficient, C. LCLEAK should be defined with time on the x-axis and hole area on the y-axis. For the SET option, the same hole area, $A(t)$, and discharge coefficient, C, are applied to every node in the set. Therefore, the total leakage area is $nA(t)$ where n is the number of nodes in the set.

   When $h > h_{\text{ext}}$, the volume outflow rate from the node, $\dot{Q}$, is given by:

$$\dot{Q} = CA(t)\sqrt{2(h - h_{ext})g}$$

   This rate is subject to h being above the suction limit (see SUCLIM on *BOUNDARY_PORE_FLUID). If h is at or below the suction limit, then $\dot{Q} = 0$.

   When $h < h_{\text{ext}}$, the volume flow rate (with inflow being treated as negative) is:

$$\dot{Q} = \begin{cases} -CA(t)\sqrt{2(h_{\text{ext}} - h)g} & \text{if } h_{\text{ext}} > 0 \\ 0 & \text{if } h_{\text{ext}} \le 0 \end{cases}$$

10. **Activation by Sensor.** Optionally, *BOUNDARY_PWP_NODE_ID or *BOUNDARY_PWP _SET_ID can be activated by *SENSOR_CONTROL with TYPE set to BPWPN. In this case the sensor overrides TBIRTH and TDEATH.

11. **Using LCPUMP to model pumping water out.** When used with TYPPMP set to zero, LCPUMP represents situations where water is pumped out of the soil at a known rate. Instead of constraining the pore pressure, the boundary condition removes water from the nodes to which it applies at a prescribed rate. The y-axis of the curve gives the outflow rate (volume per unit time) at each node to which this boundary condition applies. Water is removed by the boundary condition only while the pressure head at the affected node is above the suction limit (see SUCLIM on *BOUNDARY_PORE_FLUID). For nodes at the suction limit,

the boundary condition has no effect. When TYPPMP is zero and LCPUMP is nonzero, LC, CVAL, and ITOTEX are ignored.

12. **Using LCPUMP to limit flow rate to the boundary condition.** When used with TYPPMP set to 1 or 2, LCPUMP defines an upper limit on the volume flow rate between each node and the boundary condition. The boundary condition pressure head, $h_{\mathrm{BC,\,node}}$, is defined by LC, CVAL, and ITOTEX. The pressure head at the node attempts to follow $h_{\mathrm{BC,\,node}}$ subject to (a) the maximum volume flow rate to or from the boundary condition not exceeding the current value of LCPUMP, and (b) the pressure head at the node not falling below the suction limit (see SUCLIM on *BOUNDARY_PORE_FLUID). For TYPPMP set to 1, only outflow is permitted. For TYPPMP set to 2, both outflow and inflow are permitted. The values on the y-axis of LCPUMP should be input as positive, and are treated as limiting the absolute value of the volume flow rate per node to or from the boundary condition.

*BOUNDARY_PZEPOT

Purpose: Define prescribed electric potential on a node set for a piezoelectric material; see *MAT_ADD_PZELECTRIC.

Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	NSID	LCID	SF				
Type	I	I	I	I				
Default	none	none	0	none				

VARIABLE	DESCRIPTION
ID	ID of this boundary condition, which can be referred to by *SENSOR_CONTROL with TYPE='PZBC' or *DEFINE_CURVE_FUNCTION with FUNCTION='ECHGBC'
NSID	Node set ID, see *SET_NODE
LCID	Load curve giving prescribed electric potential as a function of time
SF	Scale factor on curve or constant electric potential if LCID = 0.

***BOUNDARY_RADIATION**_*OPTION1_{OPTION2}_{OPTION3}*

Available values for *OPTION1* include:

SET

SEGMENT

ENCLOSURE

Available values for *OPTION2* include (available if *OPTION1* is **not** ENCLOSURE):

VF_READ

VF_CALCULATE

<BLANK>

Available values for *OPTION3* include (available if *OPTION1* is **not** ENCLOSURE):

RESTART

<BLANK>

OPTION1 specifies radiation boundary surface definition by a surface set (SET) or by a segment list (SEGMENT). The option ENCLOSURE can be used to transfer heat between segments in an enclosure. The ENCLOSURE option is available in LS-DYNA MPP only.

OPTION2 indicates the radiation boundary surface is part of an enclosure. When set to VF OPTION2 specifies the use of view factors. The suffix, READ, indicates that the view factors should be read from the file "viewfl". The suffix, CALCULATE, indicates that the view factors should be calculated. The Stefan Boltzmann constant must be defined for radiation in an enclosure on the *CONTROL_THERMAL_SOLVER keyword. The parameter DTVF entered on the CONTROL_THERMAL_SOLVER keyword defines the time interval between VF updates for moving geometries.

OPTION3 is the keyword suffix RESTART. This is only applicable in combination with the keyword VF_CALCULATE. In very long runs, it may be necessary to halt execution. This is accomplished by entering Ctrl-C followed by sw1. To restart the view factor calculation, add the suffix RESTART to all VF_CALCULATE keywords in the input file.

The status of an in-progress view factor calculation can be determined by using the sense switch. This is accomplished by first typing Control-C followed by:

sw1.	Stop run and save viewfl file for restart
sw2.	Viewfactor run statistics

A list of acceptable keywords are:

*BOUNDARY_RADIATION_ENCLOSURE

*BOUNDARY_RADIATION_SEGMENT

*BOUNDARY_RADIATION_SEGMENT_VF_READ

*BOUNDARY_RADIATION_SEGMENT_VF_CALCULATE

*BOUNDARY_RADIATION_SET

*BOUNDARY_RADIATION_SET_VF_READ

*BOUNDARY_RADIATION_SET_VF_CALCULATE

Remarks:

In models that include radiation boundary conditions, a thermodynamic temperature scale is required, i.e., zero degrees must correspond to absolute zero. The Kelvin and Rankine temperature scales meet this requirement whereas Celsius and Fahrenheit temperature scales do not.

*BOUNDARY_RADIATION_ENCLOSURE

Purpose: Define an enclosure by a list of segment sets to be used for a thermal radiation analysis. This keyword is only available in LS-DYNA MPP.

Card Summary:

Card 1. This card is required.

BRENCID	ENCNAME

Card 2. This card is required.

CALOPT	OUTOPT	CONOPT					

Card 3. This card is required.

FILENAME

Card 4. This card is required.

SMFLAG	SMMAXI	SMABST	SMRELT				

Card 5. This card is required.

STYPE	SLMAXI	SLABST	SLRELT	SLMLEV			

Card 6.1. Repeat pairs of Cards 6.1 and 6.2 until all the segment sets that form the enclosure are defined. This input ends at the next keyword ("*") card.

SSID							

Card 6.2. Repeat pairs of Cards 6.1 and 6.2 until all the segment sets that form the enclosure are defined. This input ends at the next keyword ("*") card.

NINT	BLOCK	SELCID	SEMULT	LOC			

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	BRENCID	ENCNAME						
Type	I	A70						
Default	none	none						

VARIABLE	DESCRIPTION
BRENCID	Boundary radiation ID for this enclosure
ENCNAME	Name of enclosure, used for output purposes

Card 2	1	2	3	4	5	6	7	8
Variable	CALOPT	OUTOPT	CONOPT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
CALOPT	Calculation option: EQ.0: view factors
OUTOPT	Output option: EQ.0: no output EQ.1: output in LSDA format
CONOPT	Control option: EQ.0: calculate view factors matrix and preform thermal analysis

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	<jobid>.viewfactor.<suffix>							

VARIABLE	DESCRIPTION
FILENAME	Name of view factor output file

Card 4	1	2	3	4	5	6	7	8
Variable	SMFLAG	SMMAXI	SMABST	SMRELT				
Type	I	I	F	F				
Default	0	500	10^{-10}	10^{-6}				

VARIABLE	DESCRIPTION
SMFLAG	View factor matrix smoothing flag: EQ.0: no smoothing EQ.1: smoothing
SMMAXI	Maximum number of iterations for view factor matrix smoothing (default = 500)
SMABST	Absolute convergence tolerance for view factor matrix smoothing (default = 10^{-10})
SMRELT	Relative convergence tolerance for view factor matrix smoothing (default = 10^{-6})

Card 5	1	2	3	4	5	6	7	8
Variable	STYPE	SLMAXI	SLABST	SLRELT	SLMLEV			
Type	I	I	F	F	I			
Default	0	500	10^{-10}	10^{-6}	0			

VARIABLE	DESCRIPTION
STYPE	Solver type: EQ.0: reverse conjugated gradient
SLMAXI	Maximum number of iterations for radiosity solver (default = 500)
SLABST	Absolute convergence tolerance for radiosity solver (default is 10^{-10})
SLRELT	Relative convergence tolerance for radiosity solver (default = 10^{-6})
SLMLEV	Radiosity solver message level: EQ.0: no output EQ.1: debug output level I EQ.2: debug output level II EQ.3: debug output level III

Segment Set Card. Repeat pairs of Cards 6.1 and 6.2 until all segment sets which form the enclosure are listed. This input ends at the next keyword ("*") card.

Card 6.1	1	2	3	4	5	6	7	8
Variable	SSID							
Type	I							
Default	none							

Segment Set Characteristics Card. Repeat pairs of Cards 6.1 and 6.2 until all segment sets which form the enclosure are listed. This input ends at the next keyword ("*") card.

Card 6.2	1	2	3	4	5	6	7	8
Variable	NINT	BLOCK	SELCID	SEMULT	LOC			
Type	I	I	I	F	I			
Default	none	0	0	0.	0			

VARIABLE	DESCRIPTION
SSID	SSID specifies the ID for a set of segments that comprise a portion of, or possibly, the entire enclosure. See *SET_SEGMENT.
NINT	Number of integration points for view factor calculation, $1 \le NINT \le 10$ EQ.0: LS-DYNA determines the number of integration points based on the segment size and separation distance.
BLOCK	Flag indicating if this surface blocks the view between any other 2 surfaces: EQ.0: no blocking (default) EQ.1: blocking
SELCID	Load curve ID for surface emissivity (see *DEFINE_CURVE): GT.0: surface emissivity as a function of time EQ.0: use constant multiplier value, SEMULT LT.0: surface emissivity as a function of temperature. The value of –SELCID must be an integer, and it is interpreted as a load curve ID.
SEMULT	Curve multiplier for surface emissivity; see *DEFINE_CURVE
LOC	Application of surface for thermal shell elements (see THSHEL in the *CONTROL_SHELL input): EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element

VARIABLE	DESCRIPTION
	EQ.1: upper surface of thermal shell element

Remarks:

1. **Temperature Scale.** In models that include radiation boundary conditions, a thermodynamic temperature scale is required, meaning zero degrees must correspond to absolute zero. The Kelvin and Rankine temperature scales meet this requirement whereas Celsius and Fahrenheit temperature scales do not.

*BOUNDARY_RADIATION_SEGMENT

Purpose: Apply a radiation boundary condition on a segment to transfer heat between the segment and the environment.

Include the following 2 cards for each segment. Setting TYPE = 1 on Card 1 below indicates that the segment transfers heat to the environment.

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	TYPE			
Type	I	I	I	I	I			
Default	none	none	none	none	1			

Card 2	1	2	3	4	5	6	7	8
Variable	FLCID	FMULT	TLCID	TMULT	LOC			
Type	I	F	I	F	I			
Default	none	0.	none	0.	0			

VARIABLE	DESCRIPTION
N1, N2, N3, N4	Node ID's defining segment
TYPE	Radiation type: EQ.1: Radiation to environment
FLCID	Radiation heat transfer coefficient, $f = \sigma\varepsilon F$, specification where σ is the Stefan Boltzmann constant, ε is the surface emissivity, F is the surface view factor. See Remark 1. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 2). When the reference is to a curve, FLCID has the following interpretation: GT.0: f is defined as a function of time, t, having points consisting of $(t, f(t))$ data pairs.

VARIABLE	DESCRIPTION
	EQ.0: f is a constant defined by the value FMULT. LT.0: f is defined as a function of temperature, T, by a curve consisting of $(T, f(T))$ data pairs. Enter |FLCID| on the *DEFINE_CURVE keyword.
FMULT	Radiation heat transfer coefficient, f, curve multiplier
TLCID	Environment temperature, T_∞, specification. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 3). When the reference is to a curve, TLCID has the following interpretation: GT.0: T_∞ is defined as a function of time, t, by a curve consisting of $(t, T_\infty(t))$ data pairs. EQ.0: T_∞ a constant defined by the value TMULT.
TMULT	Environment temperature, T_∞, curve multiplier
LOC	For a thick thermal shell, the radiation will be applied to the surface identified by LOC. See the parameter THSHEL on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element EQ.1: upper surface of thermal shell element

Remarks:

1. **Radiation Boundary Condition.** A radiation boundary condition is calculated using

$$\dot{q}'' = \sigma\varepsilon F\left(T_{\text{surface}}^4 - T_\infty^4\right) = f\left(T_{\text{surface}}^4 - T_\infty^4\right) \ ,$$

where f is the radiation heat transfer coefficient. If f is a function of temperature, f is evaluated at the surface temperature, $T_{surface}$.

2. **FLCID Function Arguments.** If FLCID references a *DEFINE_FUNCTION, the radiation heat transfer coefficient can be a function of the segment centroid coordinates, the segment centroid velocity components, the segment centroid temperature, the environment temperature (T_∞), and the solution time, that is, "f(x, y, z, vx, vy, vz, temp, tinf, time)."

3. **TLCID Function Arguments.** If TLCID references a *DEFINE_FUNCTION, the environment temperature, T_∞, can be a function of the segment centroid coordinates, the segment centroid velocity components, and the solution time, that is "f(x, y, z, vx, vy, vz, time)."

BOUNDARY_RADIATION_SEGMENT_VF_*OPTION

Available options include:

READ

CALCULATE

Purpose: Apply a radiation boundary condition on a SEGMENT to transfer heat between the segment and an enclosure surrounding the segment using view factors.

The file viewfl must be present for the READ option, whereas it will be created with the CALCULATE option. If the file viewfl exists when using the CALCULATE option, LS-DYNA will terminate with an error message to prevent overwriting the file. The file viewfl contains the surface-to-surface area × view factor products (that is, $A_i F_{ij}$). These products are stored by row and formatted as 5E16.0.

Include the following two cards for each segment. The enclosure is defined by additional segments using this keyword. Setting TYPE = 2 on Card 1 below specifies that the segment belongs to an enclosure.

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	TYPE	BLOCK	NINT	
Type	I	I	I	I	I	I	I	
Default	none	none	none	none	2	0	0	

Card 2	1	2	3	4	5	6	7	8
Variable	SELCID	SEMULT						
Type	I	F						
Default	SEMULT	0.						

VARIABLE	DESCRIPTION
N1, N2, N3, N4	Node ID's defining segment

VARIABLE	DESCRIPTION
TYPE	Radiation type: EQ.2: Radiation within an enclosure
BLOCK	Flag indicating if this surface blocks the view between any other 2 surfaces. EQ.0: no blocking (default) EQ.1: blocking
NINT	Number of integration points for viewfactor calculation: EQ.0: LS-DYNA determines the number of integration points based on the segment size and separation distance GT.0: User specified number between (and including) 1 and 10.
SELCID	Load curve ID for surface emissivity (see *DEFINE_CURVE) GT.0: surface emissivity as a function of time EQ.0: use constant multiplier value, SEMULT LT.0: surface emissivity as a function of temperature. The value of –SELCID must be an integer, and it is interpreted as a load curve ID.
SEMULT	Curve multiplier for surface emissivity; see *DEFINE_CURVE.

*BOUNDARY_RADIATION_SET

Purpose: Apply a radiation boundary condition on a SEGMENT_SET to transfer heat between the segment set and the environment.

Include the following 2 cards for each set. Setting TYPE = 1 on Card 1 below indicates that the segment transfers energy to the environment.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	TYPE					PSEROD	
Type	I	I					I	
Default	none	1					none	

Card 2	1	2	3	4	5	6	7	8
Variable	FLCID	FMULT	TLCID	TMULT	LOC			
Type	I	F	I	F	I			
Default	none	0.	none	0.	0			

VARIABLE	DESCRIPTION
SSID	SSID specifies the ID for a set of segments that comprise a portion of, or possibly, the entire enclosure. See *SET_SEGMENT.
TYPE	Radiation type: EQ.1: radiation to environment
PSEROD	Part set ID for updating boundary segments exposed to the environment as solid elements erode; see Remark 4.
FLCID	Radiation heat transfer coefficient, $f = \sigma\varepsilon F$, specification, where σ is the Stefan Boltzmann constant, ε is the surface emissivity, F is the surface view factor. See Remark 1.This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DE-

VARIABLE	DESCRIPTION
	FINE_FUNCTION and Remark 2). When the reference is to a curve, FLCID has the following interpretation: GT.0: f is defined as a function of time, t, by a curve consisting of $(t,f(t))$ data pairs. EQ.0: f is a constant defined by the value FMULT. LT.0: f is defined as a function of temperature, T, by a curve consisting of $(T,f(T))$ data pairs. Enter |FLCID| on the *DEFINE_CURVE keyword.
FMULT	Curve multiplier of f (see FLCID)
TLCID	Environment temperature, T_∞ , specification. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 3). When the reference is to a curve, TLCID has the following interpretation: GT.0: T_∞ is defined as a function of time, t, by a curve consisting of $(t, T_\infty(t))$ data pairs. EQ.0: T_∞ a constant defined by the value TMULT
TMULT	Curve multiplier for T_∞
LOC	For a thick thermal shell, the radiation will be applied to the surface identified by LOC. See the parameter THSHEL on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0.: middle surface of thermal shell element EQ.1: upper surface of thermal shell element

Remarks:

1. **Radiation Boundary Condition.** A radiation boundary condition is calculated using

$$\dot{q}'' = \sigma\varepsilon F\left(T_{\text{surface}}^4 - T_\infty^4\right) = f\left(T_{\text{surface}}^4 - T_\infty^4\right) ,$$

where f is the radiation heat transfer coefficient. If f is a function of temperature, f is evaluated at the surface temperature, $T_{surface}$.

2. **FLCID Function Arguments.** If FLCID references a *DEFINE_FUNCTION, the radiation heat transfer coefficient can be a function of the segment centroid

coordinates, the segment centroid velocity components, the segment centroid temperature, the environment temperature (T_∞), and the solution time, that is, "f(x, y, z, vx, vy, vz, temp, tinf, time)."

3. **TLCID Function Arguments.** If TLCID references a *DEFINE_FUNCTION, the environment temperature can be a function of the segment centroid coorodinates, the segment centroid velocity components, and the solution time, that is, "f(x, y, z, vx, vy, vz, time)."

4. **Erosion.** When radiation is applied to a segment set associated with solid elements that erode, the radiation condition on a deleted segment will not by default be transferred to newly exposed segments but ceases to exist. By specifying a part set through the parameter PSEROD, any such new segment, *that is attached to an element in this part set*, will inherit this boundary condition, using the same data as prescribed for all original segments.

BOUNDARY_RADIATION_SET_VF_*OPTION

Available options include:

READ

CALCULATE

Include the following 2 cards for each set. This keyword applies a radiation boundary condition on a segment set to transfer heat using view factors between the segment set and an enclosure surrounding the segments. Segments contained in the segment set may form the enclosure. Setting TYPE = 2 on Card 1 below specifies that the segment set belongs to an enclosure.

The file "viewfl" must be present for the READ option. The file "viewfl" will be created for the CALCULATE option. If the file "viewfl" exists when using the CACULATE option, LS-DYNA will terminate with an error message to prevent overwriting the file. The file "viewfl" contains the surface-to-surface area × view factor products (i.e. $A_i F_{ij}$). These products are stored by row and formatted as 5E16.0.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	TYPE	RAD_GRP	FILE_NO	BLOCK	NINT		
Type	I	I	I	I	I	I		
Default	none	2	0	0	0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	SELCID	SEMULT						
Type	I	F						
Default	none	0.						

VARIABLE	DESCRIPTION
SSID	SSID specifies the ID for a set of segments that comprise a portion of, or possibly, the entire enclosure. See *SET_SEGMENT.

VARIABLE	DESCRIPTION
TYPE	Radiation type: EQ.2: Radiation within an enclosure
RAD_GRP	Radiation enclosure group ID. The segment sets from all radiation enclosure definitions with the same group ID are augmented to form a single enclosure definition. If RAD_GRP is not specified or set to zero, then the segments are placed in group zero. All segments defined by the SEGMENT option are placed in set zero.
FILE_NO	File number for view factor file. FILE_NO is added to "viewfl_" to form the name of the file containing the view factors. For example, if FILE_NO is specified as 22, then the view factors are read from viewfl_22. For radiation enclosure group zero FILE_NO is ignored, and view factors are read from viewfl. The same file may be used for different radiation enclosure group definitions.
BLOCK	Flag indicating if this surface blocks the view between any other 2 surfaces. EQ.0: No blocking (default) EQ.1: Blocking
NINT	Number of integration points for view factor calculation EQ.0: LS-DYNA determines the number of integration points based on the segment size and separation distance GE.11: Not allowed
SELCID	Load curve ID for surface emissivity, see *DEFINE_CURVE GT.0: Function of time EQ.0: Use constant multiplier value, SEMULT LT.0: Function of temperature. |SELCID| is a load curve ID.
SEMULT	Curve multiplier for surface emissivity

Remarks:

Multiple enclosures can be modeled when using view factors. Consider the following example input. The order of segments in the view factor file follows the order the sets are assigned to the boundary radiation definition.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *BOUNDARY_RADIATION_SET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Make boundary enclosure radiation groups 8 and 9.
$
*BOUNDARY_RADIATION_SET_VF_READ
*       SSID       TYPE    RAD_GRP    FILE_NO
         15          2          9         10
        1.0        1.0
*BOUNDARY_RADIATION_SET_VF_READ
*       SSID       TYPE    RAD_GRP    FILE_NO
         12          2          9         10
        1.0        1.0
*BOUNDARY_RADIATION_SET_VF_READ
*       SSID       TYPE    RAD_GRP    FILE_NO
         13          2          8         21
        1.0        1.0
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Enclosure radiation group 9 is composed of all the segments in segment set 15 followed by those in segment set 12. The view factors are stored in the file viewfl_10. Enclosure radiation group 8 is composed of the segments in segment set 13. The view factors are stored in the file viewfl_21.

*BOUNDARY_SALE_MESH_FACE

Purpose: Define boundary conditions at S-ALE mesh faces. This keyword serves as a macro for other keywords, simplifying the specification of boundary conditions. As a result, you can use a single keyword to apply boundary conditions.

The keyword reader translates this keyword into a combination of several different keywords. Those keywords include *SET_NODE_GENERAL (with SALEFAC option), *SET_SEGMENT_GENERAL (with SALEFAC option), *BOUNDARY_SPC_SET, *BOUNDARY_NON_REFLECTING, *BOUNDARY_PRESCRIBED_MOTION, *LOAD_SEGMENT_SET, and *BOUNDARY_AMBIENT.

Include as many cards as needed with each card representing one boundary condition. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	BCTYPE	MSHID	NEGX	POSX	NEGY	POSY	NEGZ	POSZ
Type	A	I	I	I	I	I	I	I
Default	none	none	0	0	0	0	0	0

VARIABLE	DESCRIPTION
BCTYPE	Available boundary conditions:
	EQ.FIXED: All nodes at the face are fixed in all directions. (Applied with *BOUNDARY_SPC_SET.)
	EQ.NOFLOW: No flow allowed through the face. (Applied with *BOUNDARY_SPC_SET.)
	EQ.SYM: The face is a symmetric plane (same as NOFLOW). (Applied with *BOUNDARY_SPC_SET.)
	EQ.NONREFL: Non-reflective boundary condition. (Applied with *BOUNDARY_NON_REFLECTING.)
	EQ.FLOWVEL: Nodes constrained by time-dependent velocities. (Applied with *BOUNDARY_PRESCRIBED_MOTION.)
	EQ.PRES: Faces loaded by time-dependent pressures. (Applied with *LOAD_SEGMENT_SET.)
	EQ.AMBIENT: Ambient elements attached to the faces. See

VARIABLE	DESCRIPTION
	Remarks 1 and 2. (Applied with *BOUNDARY_AMBIENT.)
MSHID	S-ALE Mesh ID
NEG[X,Y,Z], POS[X,Y,Z]	Determine where the boundary condition is applied to the mesh. NEGX, POSX, NEGY, POSY, NEGZ, or POSZ means the mesh faces with an outward normal vector in the local -*x*, +*x*, -*y*, +*y*, -*z*, or +*z*-directions, respectively. For BCTYPE ≠ PRES, FLOWVEL, or AMBIENT: EQ.0: The boundary condition is *not* applied to faces with this outward normal. EQ.1: The boundary condition is applied to faces with this outward normal. For BCTYPE = PRES or FLOWVEL: EQ.0: The boundary condition is *not* applied to faces with this outward normal. GT.0: The boundary condition is applied to faces with this outward normal and is controlled by a time-dependent curve (*DEFINE_CURVE), whose ID is referred to with this curve. For BCTYPE = AMBIENT: EQ.0: The boundary condition is *not* applied to the elements connected to the faces with this outward normal. GT.0: The boundary condition is applied to the elements connected to the faces with this outward normal. The value sets the reservoir material as described in Remark 1.

Remarks:

1. **AMBIENT.** If AMBIENT is applied to mesh faces, the elements attached to these faces are turned into "ambient" elements controlled by *BOUNDARY_AMBIENT with AMBTYP = 4. If [NEG,POS][X,Y,Z] = 1, MMG on *BOUNDARY_AMBIENT is set to 0 and Card 2 on *BOUNDARY_AMBIENT is not used. Any other nonzero value of [NEG,POS][X,Y,Z] should have the following form: MMG + 100 × LCID1 + 10000 × LCID2, where MMG, LCID1, and LCID2 are the fields on *BOUNDARY_AMBIENT providing the "reservoir" material, internal energy load curve, and the relative volume load curve. If these elements have

their pressures initialized by *INITIAL_ALE_HYDROSTATIC, *ALE_AMBIENT_HYDROSTATIC controls the ambient elements.

2. **Boundary conditions combined with AMBIENT.** If AMBIENT and NONREFL or AMBIENT and FLOWVEL are applied to the same mesh faces, the elements attached to these faces are turned into ambient elements. The faces connecting the ambient elements to the mesh are selected to apply NONREFL. The nodes connecting the ambient elements to the mesh are selected to apply FLOWVEL.

Example:

The following input specifies no-flow boundary conditions on faces with normal vectors in the -*y* and -*z*-directions and non-reflecting boundary conditions on the faces with normal vectors in the -*x*, +*x*, +*y*, and +*z*-directions.

```
*BOUNDARY_SALE_MESH_FACE
$    option     mshid      NEGX      POSX      NEGY      POSY      NEGZ      POSZ
    NOFLOW          1                              1                 1
   NONREFL          1         1         1                 1                   1
```

Internally, the above input is translated to the following keywords:

```
*BOUNDARY_SPC_SET
         1                   0         1         0
*BOUNDARY_SPC_SET
         2                   0         0         1
*SET_NODE_GENERAL
$       SID
         1
$   OPTION      MSHID       XMN       XMX       YMN       YMX       ZMN       ZMX
   SALEFAC          1                              1
*SET_NODE_GENERAL
$       SID
         2
$   OPTION      MSHID       XMN       XMX       YMN       YMX       ZMN       ZMX
   SALEFAC          1                                                   1
*BOUNDARY_NON_REFLECTING
$       SID
        11
*SET_SEGMENT_GENERAL
$       SID
        11
$   OPTION      MSHID       XMN       XMX       YMN       YMX       ZMN       ZMX
   SALEFAC          1         1         1                 1                   1
```

In comparison to the translated version, *BOUNDARY_SALE_MESH_FACE provides a more streamlined, intuitive approach to facilitate the boundary condition setup for S-ALE models. This method is particularly useful when the mesh is tilted since the local coordinate system is already assumed.

*BOUNDARY_SLIDING_PLANE

Purpose: Define a sliding symmetry plane. This option applies to continuum domains modeled with solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	VX	VY	VZ	COPT			
Type	I	F	F	F	I			
Default	none	0	0	0	0			

VARIABLE	DESCRIPTION
NSID	Nodal set ID, see *SET_NODE
VX	x-component of vector defining normal or vector
VY	y-component of vector defining normal or vector
VZ	z-component of vector defining normal or vector
COPT	Option: EQ.0: node moves on normal plane. EQ.1: node moves only in vector direction.

Remarks:

Any node may be constrained to move on an arbitrarily oriented plane or line depending on the choice of COPT. Each boundary condition card defines a vector originating at (0,0,0) and terminating at the coordinates defined above. Since an arbitrary magnitude is assumed for this vector, the specified coordinates are non-unique and define only a direction. Use of *BOUNDARY_SPC is preferred over *BOUNDARY_SLIDING_PLANE as the boundary conditions imposed using the latter have been seen to break down somewhat in lengthy simulations owing to numerical roundoff.

***BOUNDARY_SPC**_*OPTION1_{OPTION2}_{OPTION3}*

OPTION1 is required since it specifies whether the SPC applies to a single node or to a set. The two choices are:

NODE

SET

OPTION2 allows optional birth and death times to be assigned the single node or node set:

BIRTH_DEATH

This option requires one additional line of input. The BIRTH_DEATH option is inactive during the dynamic relaxation phase, which allows the SPC to be removed during the subsequent normal analysis phase. The BIRTH_DEATH option can be used only once for any given node and if used, no other *BOUNDARY_SPC commands can be used for that node.

OPTION3 allows an optional ID to be given that applies either to the single node or to the entire set:

ID

If a heading is defined with the ID, then the ID with the heading will be written at the beginning of the ASCII file, spcforc. A node which appears in more than one SPC set containing an ID will only be associated with one of those IDs in spcforc.

Purpose: Define nodal single point constraints. Do not use this option in r-adaptive problems since the nodal point ID's change during the adaptive step. If possible use CONSTRAINED_GLOBAL instead.

ID Card. Additional card for the ID keyword option.

Optional	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	NID/NSID	CID	DOFX	DOFY	DOFZ	DOFRX	DOFRY	DOFRZ
Type	I	I	I	I	I	I	I	I
Default	none	0	0	0	0	0	0	0

Birth/Death Card. Additional card for the BIRTH_DEATH keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH						
Type	F	F						
Default	0.0	10^{20}						

VARIABLE	DESCRIPTION
ID	Optional SPC set ID to which this node or node set belongs. This ID does not need to be unique
HEADING	An optional SPC descriptor that will be written into the d3hsp file and the spcforc file.
NID/NSID	Node ID or nodal set ID, see *SET_NODE.
CID	Coordinate system ID, see *DEFINE_COORDINATE_SYSTEM.
DOFX	Insert 1 for a translational constraint in the local x-direction for a Cartesiaon system or the local radial degree of freedom r for a cylindrical or spherical system.
DOFY	Insert 1 for a translational constraint in the local y-direction for a Cartesian system, the direction of the local circumferential degree of freedom φ for a cylindrical system, or the direction of the local latitude degree of freedom θ for a spherical system.
DOFZ	Insert 1 for a translational constraint in the local z-direction for a Cartesian system, the direction of the local z-translational degree of freedom for a cylindrical system, or the direction of the local

VARIABLE	DESCRIPTION
	longitude degree of freedom φ for a spherical system.
DOFRX	Insert 1 for rotational constraint about the local x-axis for a Cartesian system or about the axis of the local radial degree of freedom r for a cylindrical or spherical system.
DOFRY	Insert 1 for rotational constraint about the local y-axis for a Cartesian system, about the axis of the local circumferential degree of freedom φ for a cylindrical system, or about the axis of the local latitude degree of freedom θ for a spherical system.
DOFRZ	Insert 1 for rotational constraint about the local z-axis for a Cartesian system, about the axis of the local z-translational degree of freedom for a cylindrical system, or about the axis of the local longitude degree of freedom φ for a spherical system.
BIRTH	Activation time for SPC constraint. The birth time is ignored during dynamic relaxation.
DEATH	Deactivation time for the SPC constraint. The death time is ignored during dynamic relaxation.

Remarks:

Constraints are applied if for each DOF*ij* field set to 1. A value of zero means no constraint. No attempt should be made to apply SPCs to nodes belonging to rigid bodies (see *MAT_RIGID for application of rigid body constraints).

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *BOUNDARY_SPC_NODE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Make boundary constraints for nodes 6 and 542.
$
*BOUNDARY_SPC_NODE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       nid       cid      dofx      dofy      dofz     dofrx     dofry     dofrz
          6         0         1         1         1         1         1         1
        542         0         0         1         0         1         0         1
$
$       Node 6 is fixed in all six degrees of freedom (no motion allowed).
$
$       Node 542 has a symmetry condition constraint in the x-z plane,
$         no motion allowed for y translation, and x & z rotation.
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*BOUNDARY_SPC_SYMMETRY_PLANE_{OPTION}

Purpose: Constrain nodes that are within some distance (a tolerance) of a plane to have no motion orthogonal to that plane. This keyword is usually used for a geometric symmetry plane, so that the full geometry does not have to be modeled. This keyword supports both *h*- and *r*-adaptivity. Related keywords are: *BOUNDARY_SPC_SET, *CONSTRAINED_GLOBAL, and *CONSTRAINED_LOCAL. Note this keyword should not be used on any node that also has *BOUNDARY_SPC applied. This keyword is supported for shells, thick shells, solids, and beams.

Available options include:

<BLANK>

SET

Card Sets. For each symmetry plane input one pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	IDSP	PID/PSID	X	Y	Z	VX	VY	VZ
Type	I	I	F	F	F	F	F	F
Default	none	none	0.0	0.0	0.0	0.0	0.0	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	TOL							
Type	F							
Default	0.0/0.2							

VARIABLE	DESCRIPTION
IDSP	Identification number of the constraint. Must be unique.

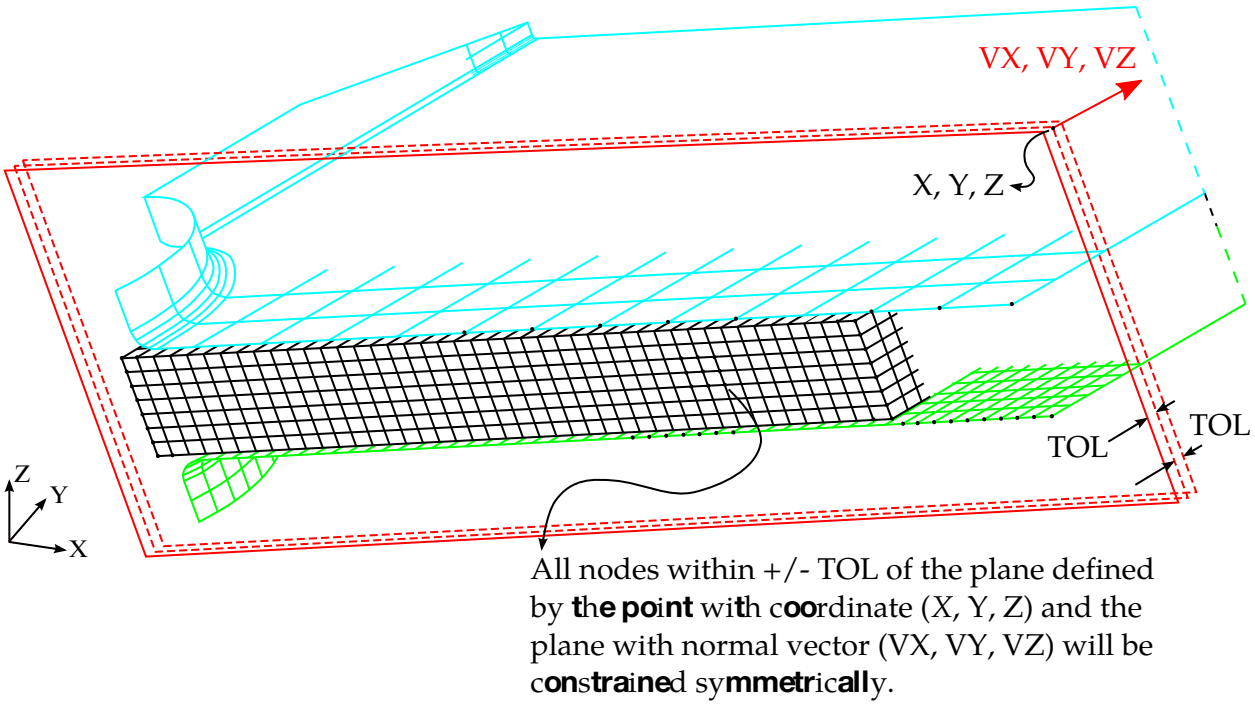


Figure 5-6. The nodes within a tolerance of the plane shown above are constrained by this keyword.

VARIABLE	DESCRIPTION
PID/PSID	When the option is blank, PID is a part ID specifying the part on which the constraint will be imposed. When the SET option is active, this field must contain a part set ID specifying the parts on which the constraint will be imposed.
X, Y, Z	Position coordinates on the symmetry plane.
VX, VY, VZ	Vector components of the symmetry plane's normal.
TOL	A distance tolerance value within which the nodes on the deformable part will be constrained. For shell elements, the default tolerance is 0.2.

Remarks:

1. **Keyword restrictions.** An input deck can contain multiple symmetry planes. However, multiple instantiations of this keyword cannot appear in the in the input deck with identical symmetry planes, meaning planes with the same position coordinates and normal vectors. Thus, to have two parts with the same symmetry plane, use the SET keyword option and include them in the same part set specified with PSID.

2. **Plane definition.** As shown in Figure 5-6, a symmetry plane's position is defined using a point's coordinates X, Y, and Z, and the plane's orientation is defined by the vector component VX, VY, and VZ. All nodes within a specified tolerance TOL have constrained motion in the direction orthogonal to the plane.

3. **Tailor-welded blanks.** The option SET allows for symmetry boundary conditions to be applied on tailor-welded blanks (TWB), where two pieces of sheet metal with different thickness or properties are welded in the butt joint configuration.

Example:

The following input creates symmetric constraints on nodes (from PID 11) within distance of 0.1 mm from the defined symmetry plane having a normal vector (1.0,1.0,1.0) passing through point $(10.5, 40.0, 20.0)$.

```
*BOUNDARY_SPC_SYMMETRY_PLANE
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     IDSP       PID         X         Y         Z        VX        VY        VZ
         1        11      10.5      40.0      20.0       1.0       1.0       1.0
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      TOL
      0.10

.
```

*BOUNDARY_SPH_FLOW

Purpose: Define a flow of particles. This option applies to continuum domains modeled with SPH elements.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	STYP	DOF	VAD	LCID	SF	DEATH	BIRTH
Type	I	I	I	I	I	F	F	F
Default	none	none	none	0	none	1.0	10^{20}	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	NODE	VID						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
NSID, PID	Nodal set ID (NSID) or part ID (PID).
STYP	Set type: EQ.1: part set ID, see *SET_PART, EQ.2: part ID, see *PART, EQ.3: node set ID, see *SET_NODE.
DOF	Applicable degrees-of-freedom: EQ.1: x-translational degree-of-freedom, EQ.2: y-translational degree-of-freedom, EQ.3: z-translational degree-of-freedom, EQ.4: translational motion in direction given by VID. Movement on plane normal to the vector is permitted.

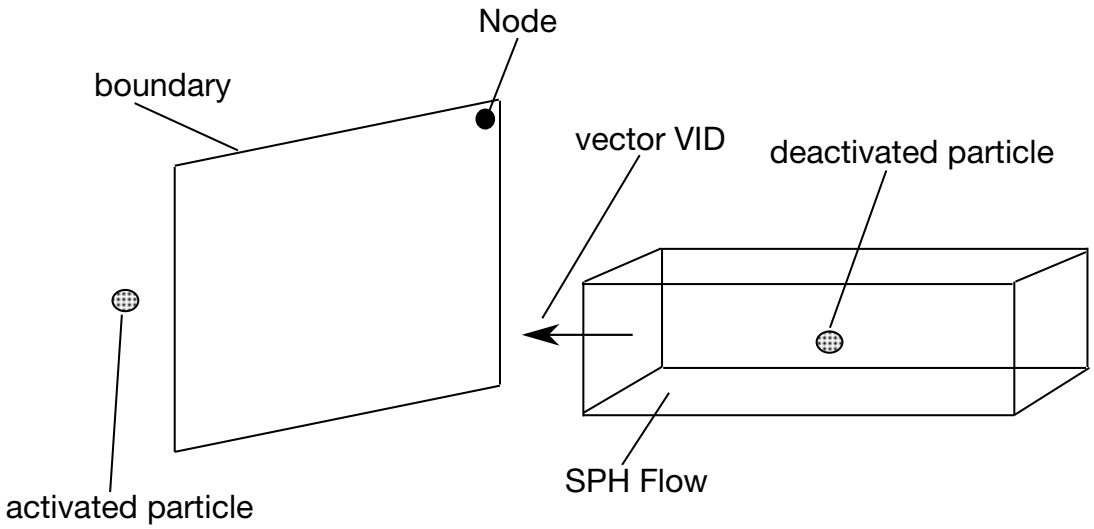


Figure 5-7. Vector VID determines the orientation of the SPH flow

VARIABLE	DESCRIPTION
VAD	Velocity / Acceleration / Displacement flag applied to SPH elements before activation: EQ.0: velocity, EQ.1: acceleration, EQ.2: displacement.
LCID	Load curve ID to describe motion value as a function of time; see *DEFINE_CURVE.
SF	Load curve scale factor.
DEATH	Time imposed motion / constraint is removed: EQ.0.0: default set to 10^{20}.
BIRTH	Time imposed motion / constraint is activated.
NODE	Node fixed in space which determines the boundary between activated particles and deactivated particles.
VID	Vector ID for defining the orientation of the SPH flow; see *DEFINE_VECTOR.

Remarks:

Initially, the user defines the set of particles that are representing the flow of particles during the simulation. At time t = 0, all the particles are deactivated which means that

no particle approximation is calculated. The boundary of activation is a plane determined by the NODE and normal to the vector VID. The particles are activated when they reached the boundary. When they are activated, particle approximation begins.

*BOUNDARY_SPH_HEAT_CONVECTION

Purpose: Apply heat convection to surface particles of the specifed SPH parts. Surface tracking is required to identify surface particles. If a part ID specified here is missing from those in *BOUNDARY_SPH_TRACK_SURFACE, then surface tracking is activated for them using default values. This is only supported in MPP with IDIM = 3 and FORM ≠ 13 in *CONTROL_SPH.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDTYP						
Type	I	I						
Default	none	0						

Card 2	1	2	3	4	5	6	7	8
Variable	HLCID	HMULT	TLCID	TMULT				
Type	I	F	I	F				
Default	none	0.0	none	0.0				

VARIABLE	DESCRIPTION
ID	Part or part set ID
IDTYP	ID type: EQ.0: Part ID (see *PART) EQ.1: Part set ID (see *SET_PART)
HLCID	Load curve ID for the heat convection coefficient: LT.0: The heat convection coefficient depends on film temperature. It is calculated as the average between the ambient and particle temperatures. EQ.0: The heat convection coefficient is constant and defined with HMULT.

VARIABLE	DESCRIPTION
	GT.0: The heat convection coefficient depends on time.
HMULT	Scale factor on load curve |HLCID| if HLCID ≠ 0 or constant heat convection coefficient if HLCID = 0
TLCID	Load curve ID for ambient temperature: EQ.0: Temperature is constant. GT.0: Temperature depends on time.
TMULT	Scale factor on load curve |TLCID| if TLCID ≠ 0 or constant ambient temperature if TLCID = 0

*BOUNDARY_SPH_NON_REFLECTING

Purpose: Define a non-reflecting boundary plane for SPH. This option applies to continuum domains modeled with SPH elements.

Card 1	1	2	3	4	5	6	7	8
Variable	VTX	VTY	VTZ	VHX	VHY	VHZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
VTX	x-coordinate of tail of a normal vector originating on the wall (tail) and terminating in the body (head); that is, the vector points from the non-reflecting boundary plane to the body.
VTY	y-coordinate of tail
VTZ	z-coordinate of tail
VHX	x-coordinate of head
VHY	y-coordinate of head
VHZ	z-coordinate of head

Remarks:

1. **Non-reflecting Boundary Plane Orientation.** The non-reflecting boundary plane has to be normal to either the x, y or z direction.

2. **Non-reflecting Boundary Plane Location.** The non-reflecting boundary plane has to be placed $h/2$ away from the SPH part, where h is the interparticle distance.

*BOUNDARY_SPH_NOSLIP

Purpose: Impose a no-slip boundary condition for continuum domains modeled with SPH elements. This is accomplished by doing the following to dummy particles: (1) adding a user-specified fictitious velocity to the (possibly non-zero) existing velocity, (2) ensuring the density stays constant throughout the simulation, (3) interpolating pressure using only the neighboring particles that are not part of the no-slip boundary condition, (4) zeroing out forces due to other SPH phenomena, and (5) subtracting the fictitious velocity at the end of the time step.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDTYP	CID	VID	LCID			
Type	I	I	I	I	I			
Default	none	0	0	none	none			

VARIABLE	DESCRIPTION
ID	Node set, node, part set, or part ID used to identify dummy particles.
IDTYP	ID type: EQ.0: Node set ID (see *SET_NODE) EQ.1: Node ID (see *NODE) EQ.2: Part set ID (see *SET_PART) EQ.3: Part ID (see *PART)
CID	Local coordinate system ID used to define the components of the velocity vector; see *DEFINE_COORDINATE_SYSTEM for example.
VID	Vector ID for defining the fictitious velocity applied to no-slip particles; see *DEFINE_VECTOR.
LCID	Load curve ID to describe the scaling factor for the velocity vector as a function of time; see *DEFINE_CURVE.

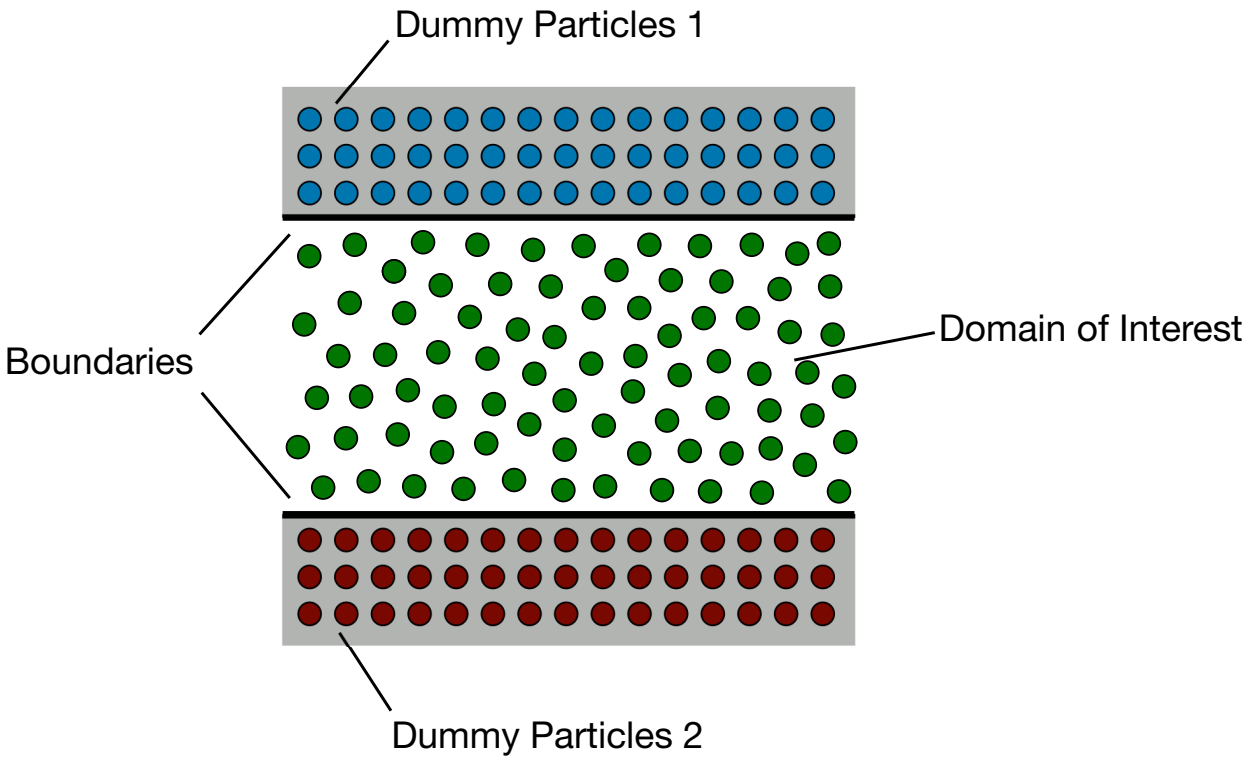


Figure 5-8. Neighboring particles (dummy particles 1 and 2) beyond the boundary must be defined in the input file

Remarks:

1. **SPH elements.** The SPH elements used to impose the no-slip boundary conditions are dummy particles that extend beyond the domain of interest and must be defined in the input file. We recommend using three to four layers of dummy particles as shown in Figure 5-8.

2. **Velocity.** The velocity specified here is added to any other imposed velocity when calculating internal forces that involve the dummy particles but omitted when computing displacement of said dummy particles. Hence, this keyword can be used in combination with others that assign a velocity to SPH particles.

*BOUNDARY_SPH_PERIODIC

Purpose: Impose a periodic boundary condition for domains modeled with SPH elements. To apply this condition, two inward-pointing vectors must be provided with *DEFINE_VECTOR to identify the periodic boundary planes. The vector tails are treated as local origins on their corresponding planes. Particles that cross one of the two planes are shifted to the equivalent position relative to the other plane (and within the SPH domain). Additionally, ghost particles are created for each interior particle that is near a periodic plane. The ghost particles are placed outside of the domain and at a position equivalent to the parent particle, but relative to the other plane. If an interior particle next to the edge moves sufficiently into the domain, its equivalent ghost particle is removed. Symmetric boundary planes do *not* need to be parallel but note that the relative rotation is calculated with the Rodrigues rotation formula. Thus, no in-plane rotations are accounted for.

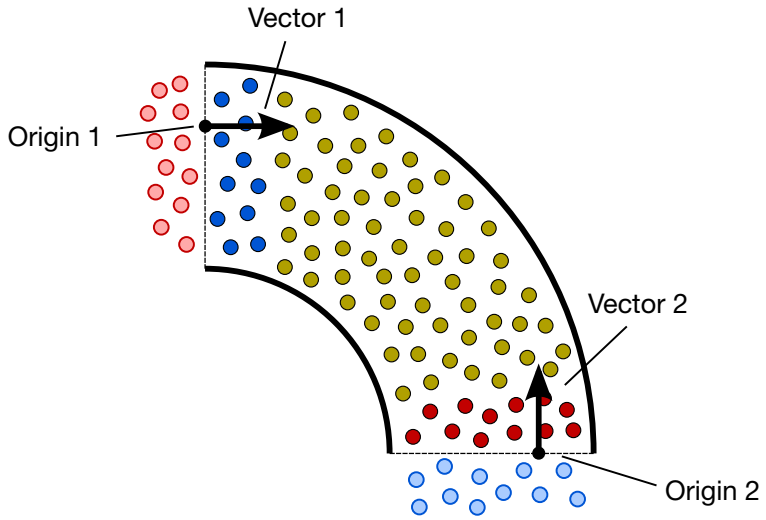


Figure 5-9. SPH particles inside of an annular domain with periodic boundaries identified by their corresponding vectors and origins. A ghost particle is generated for each particle near one of the boundary planes at the equivalent location relative to the opposite plane. In this figure, the exterior, light blue particles next to the second surface are the ghost particles for the interior, dark blue particles next to the first surface. Similarly, the light red particles are the ghost particle equivalents of the dark red particles.

Card 1	1	2	3	4	5	6	7	8
Variable	VID1	VID2						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
VID1, VID2	Vector IDs defining the inward-pointing normal vectors of the periodic boundary planes. See *DEFINE_VECTOR. The vector's tail is taken as a reference point for the given plane.

Remarks:

1. **Incompressible SPH.** Periodic boundary conditions are not supported for ISPH (FORM = 13 in *CONTROL_SPH).

*BOUNDARY_SPH_SYMMETRY_PLANE

Purpose: Define a symmetry plane for SPH. This option applies to continuum domains modeled with SPH elements.

Card 1	1	2	3	4	5	6	7	8
Variable	VTX	VTY	VTZ	VHX	VHY	VHZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
VTX	x-coordinate of tail of a normal vector originating on the wall (tail) and terminating in the body (head) (that is, vector points from the symmetry plane into the body).
VTY	y-coordinate of tail
VTZ	z-coordinate of tail
VHX	x-coordinate of head
VHY	y-coordinate of head
VHZ	z-coordinate of head

Remarks:

1. **SPH Elements.** A plane of symmetry is assumed for all SPH elements defined in the model.

2. **Plane Orientation.** The plane of symmetry has to be normal to either the x, y or z-direction.

3. **Plane Location.** The plane of symmetry is typically placed $h/2$ away from the SPH part, where h is the interparticle distance.

4. **Axisymmetric SPH.** For axisymmetric SPH analysis, IDIM = -2 (see *CONTROL_SPH), a plane of symmetry centered at the global origin and normal to x-direction is automatically created by LS-DYNA.

*BOUNDARY_SPH_TRACK_SURFACE

Purpose: Activate detection of surface particles by comparing both the number of neighbors and the center of mass for each neighborhood relative to each particle. You can control the lower bound for the ratio of the number of neighbors and the center of mass relative to the average values in order to be considered a surface particle. Additionally, you can request that only surface particles be considered in contact interfaces. To view the boundary tags, use command line argument d3dat=5 and look for "BC Tag" in the fringe plot. Positive values indicate particles tagged as "surface particles." The resulting tags heavily depend on the input below. This feature is only supported in MPP with IDIM = 3 and FORM ≠ 13 in *CONTROL_SPH.

Card 1	1	2	3	4	5	6	7	8
Variable	NBCK	NEIMOD	GNEI	GDIST				
Type	I	I	I	I				
Default	200	0	0.8	0.25				

Card 2	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	MAXNEI	MINDIST	ICONT			
Type	I	I	I	I	I			
Default	none	none	0.0	0.0	0			

VARIABLE	DESCRIPTION
NBCK	Cycle frequency to update surface particle tagging
NEIMOD	Compute the average number of neighbors: EQ.0: Using all SPH parts for which this condition applies. In other words, the average is computed using all the SPH parts specified in ID. EQ.1: Per SPH part. In other words, each SPH part has its own average value.

VARIABLE	DESCRIPTION
GNEI	Global maximum number of neighbors allowed to be considered a surface particle. This value is used when MAXNEI is not provided.
GDIST	Global minimum distance of the center of mass for a particle to be considered a boundary particle. This value is used when MINDIST is not provided.
ID	Part or part set ID for which this detection applies.
IDTYP	ID type: EQ.0: Part ID (see *PART) EQ.1: Part set ID (see *SET_PART)
MAXNEI	Maximum number of neighbors allowed for a particle to be considered a surface particle for this part or part set.
MINDIST	Minimum distance of center of mass for a particle to be considered a boundary particle for this part or part set.
ICONT	Feature to only include surface particles in contact interfaces: EQ.0: Off EQ.1: On

*BOUNDARY_SYMMETRY_FAILURE

Purpose: Define a symmetry plane with a failure criterion. This option applies to continuum domains modeled with solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	FS	VTX	VTY	VTZ	VHX	VHY	VHZ
Type	I	F	F	F	F	F	F	F
Default	none	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT
FS	Tensile failure stress > 0.0. The average stress in the elements surrounding the boundary nodes in a direction perpendicular to the boundary is used.
VTX	x-coordinate of tail of a normal vector originating on the wall (tail) and terminating in the body (head), that is, vector points from the symmetry plane into the body.
VTY	y-coordinate of tail
VTZ	z-coordinate of tail
VHX	x-coordinate of head
VHY	y-coordinate of head
VHZ	z-coordinate of head

Remarks:

A plane of symmetry is assumed for the nodes on the boundary at the tail of the vector given above. Only the motion perpendicular to the symmetry plane is constrained. After failure the nodes are set free.

BOUNDARY_TEMPERATURE_*OPTION

Available options include:

NODE

SET

Purpose: Define temperature boundary conditions for a thermal or coupled thermal/structural analysis. In a structural-only analysis, use *LOAD_THERMAL_OPTION to define temperatures (see also SOLN in *CONTROL_SOLUTION).

Card 1	1	2	3	4	5	6	7	8
Variable	NID	TLCID	TMULT	LOC	TDEATH	TBIRTH		
Type	I	I	F	I	F	F		
Default	none	0	0.	0	10^{20}	0.		

VARIABLE	DESCRIPTION
NID	Node or node set ID
TLCID	Temperature, T, specification. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 2). When the reference is to a curve, TLCID has the following interpretation: GT.0: T is defined by a curve consisting of (t, T) data pairs. EQ.0: T is a constant defined by the value TMULT.
TMULT	Temperature, T, curve multiplier.
LOC	Application of surface for thermal shell elements, see parameter, THSHEL, in the *CONTROL_SHELL input: EQ.-1: Lower surface of thermal shell element EQ.0: Middle surface of thermal shell element EQ.1: Upper surface of thermal shell element
TDEATH	Deactivation time for temperature boundary condition. At this point in time the temperature constraint is removed.

VARIABLE	DESCRIPTION
TBIRTH	Activation time for temperature boundary condition. Before this point in time the temperature constraint is ignored.

Remarks:

1. **SPH and ISPH particles.** This keyword can be used to apply temperature boundary conditions to SPH and ISPH particles.

2. **Temperature function.** If TLCID references a *DEFINE_FUNCTION, temperature may be a function of the nodal point coordinates, the nodal point velocity components, and the solution time, that is, "f(x,y,z,vx,vy,vz,time)."

*BOUNDARY_TEMPERATURE_PERIODIC_SET

Purpose: Specify different kinds of periodic boundary conditions for temperature.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID1	PTYPE	SSID2	TDLCID	AXE	NID	ANGLE	
Type	I	I	I	I	I	I	F	
Default	none	none	none	0	none	none	none	

VARIABLE	DESCRIPTION
SSID1	First segment set on which the periodic temperature boundary condition will be applied. See Remark 1.
PTYPE	Type of periodic boundary condition: EQ.1: Rotation boundary condition defined by an axis, an origin point and a rotation angle EQ.2: Reflective boundary condition defined by an axis and origin point EQ.3: Sliding boundary condition
SSID2	Second segment set on which the periodic temperature boundary condition will be applied.
TDCLID	Optional load curve specifying the temperature drop, T_{drop}, between the two surfaces in the periodic boundary condition as a function of time. Note that $T_{drop} = T_1 - T_2$ where T_1 is the temperature of the surface specified with SSID1 and T_2 is the temperature of the surface specified with SSID2. EQ.0: No temperature drop between that surfacs, that is, $T_{drop} = 0.0$
AXE	Axis for PTYPE = 1 or 2: EQ.1: X-axis EQ.2: Y-axis EQ.3: Z-axis

VARIABLE	DESCRIPTION
	Flag for meaning of ANGLE for PTYPE = 3. Setting AXE = 1 means that ANGLE is the contact distance. Otherwise, it is a scale factor on the contact distance search.
NID	Node ID giving the origin point coordinates
ANGLE	Rotation angle if PTYPE = 1. Scaling factor on contact distance search if PTYPE = 3 (default applies a scale factor of 0.3 on local element size). If AXE = 1 and PTYPE = 3, then ANGLE becomes the contact distance.

Remarks:

1. **Constraint-based method.** LS-DYNA applies these boundary conditions with a constraint-based method. For each node in SSID1 LS-DYNA finds a host face in SSID2 and applies correct weights to define a new constraint to add to the system. For PTYPE = 1 and 2, the meshes on both sides do not have to be identical, but we recommend matching meshes. PTYPE = 3 can be used to define a "perfect" thermal contact between two solids.

*BOUNDARY_TEMPERATURE_RSW

Purpose: Define temperature boundary conditions within an ellipsoidal region of the solid or shell structure. Temperatures are prescribed to nodes found in a region defined by this keyword. The boundary condition is tailored to represent the so-called weld nuggets evolving during resistive spot welding (RSW) processes. It is applicable to a thermal or coupled thermal/structural analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	OPTION	NID1	NID2	TDEATH	TBIRTH	LOC	
Type	I	I	I	I	F	F	I	
Default	none	0	none	none	10^{20}	0.	0	

Geometry and Temperature Card.

Card 2	1	2	3	4	5	6	7	8
Variable	DIST	H1	H2	R	TEMPC	TEMPB	LCIDT	
Type	F	F	F	F	F	F	I	
Default	0.	0.	0.	0.	0.	0.	none	

Heat Affected Zone Card. Additional card for OPTION = 1.

Card 3	1	2	3	4	5	6	7	8
Variable	HZ1	HZ2	RZ	TEMPZB				
Type	R	F	F	R				
Default	0.	0.	0.	0.				

VARIABLE	DESCRIPTION
SID	Node Set ID; see *SET_NODE_*OPTION*. Nodes in the set will be checked to see if they are in the nugget or heat affected zone. If

VARIABLE	DESCRIPTION
	they are, the boundary condition will be applied. The boundary condition will not be applied to nodes in these regions if they are not included in the set.
NID1	Node defining the tail of the orientation vector (axis of rotation of the ellipsoidal region) and the base for positioning of the nugget. See Remarks 1 and 2.
NID2	Node defining the head of the orientation vector (axis of rotation of the ellipsoidal region). See Remarks 1 and 2.
OPTION	Option for heat affected zone around the weld nugget: EQ.0: No heat affected zone EQ.1: Ellipsoidal region considered
TDEATH	Deactivation time for temperature boundary condition. At this point in time the temperature constraint is removed.
TBIRTH	Activation time for temperature boundary condition. Before this point in time the temperature constraint is ignored.
LOC	Application of surface for thermal shell elements; see parameter, THSHEL, in the *CONTROL_SHELL input: EQ.-1: Lower surface of thermal shell element EQ.0: Middle surface of thermal shell element EQ.1: Upper surface of thermal shell element
DIST	Position of center of nugget on the axis of rotation. Parameter defines the distance to NID1 along the orientation vector. See Remark 1.
H1	Half width h_1 of nugget in the lower half, meaning in direction to NID1. See Remark 2.
H2	Half width h_2 of nugget in the upper half, meaning in direction to NID2. See Remark 2.
R	Radius r_{weld} of the nugget in surface normal to orientation vector. See Remark 2.
TEMPC	Base temperature at the center of the nugget. See Remark 3.
TEMPB	Base temperature at the boundary of the nugget. See Remark 3.

VARIABLE	DESCRIPTION
LCIDT	|LCIDT| refers to the load curve ID prescribing the temperature evolution in the nugget as a function of time. The abscissa of the load curve will be normalized between the birth and death times of the boundary condition. GT.0: The ordinate values of the load curve scale the respective base temperature of a particular point. EQ.0: No temperature evolution. Base temperatures are used. LT.0: The ordinate values of the load curve are used to define a linear combination between the temperature at the birth time and the base temperature of a particular point. Load curve ordinate values should range between 0.0 and 1.0. We recommend LCIDT < 0 to ensure a smooth temperature evolution. See Remark 3.
HZ1	Half width h_{z1} of heat affected zone in the lower half, meaning in direction to NID1. Only active for OPTION = 1. See Remark 4.
HZ2	Half width h_{z2} of heat affected zone in the upper half, meaning in direction to NID1. Only active for OPTION = 1. See Remark 4.
RZ	Radius r_{haz} of the heat affected zone in surface normal to orientation vector. See Remark 4.
TEMPBZ	Base temperature at the boundary of the heat affected zone for OPTION = 1. See Remark 4.

Remarks:

1. **Positioning.** The position of the center of the nugget is defined starting from a base node (NID1). It is then translated by a distance DIST along the orientation vector **v** of the nugget. Vector **v** points from NID1 to NID2. See Figure 5-10.

2. **Geometry of the nugget.** Orientation vector **v** defines the axis of rotational symmetry for the ellipsoidal region, which reflects the weld nugget. The radius around this axis is defined by r_{weld}. The nugget consists of two half ellipsoids of possibly different height. The lower half, i.e. the half between NID1 and the center, has a height of h_1, wheras the upper half has a height of h_2. See Figure 5-10.

3. **Prescribing temperature values.** You can prescribe the base temperatures at the boundary and the center of the nugget with TEMPC and TEMPB,

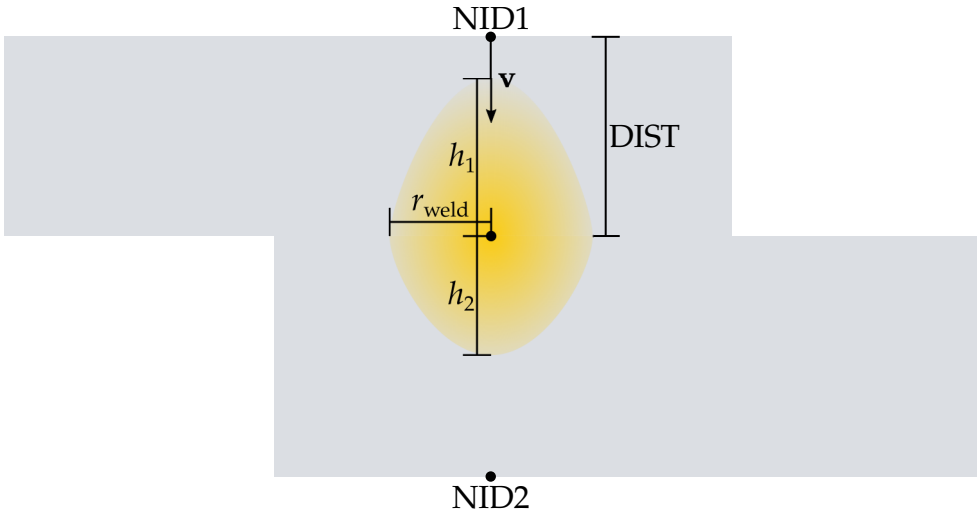


Figure 5-10. Example of geometry for OPTION = 0

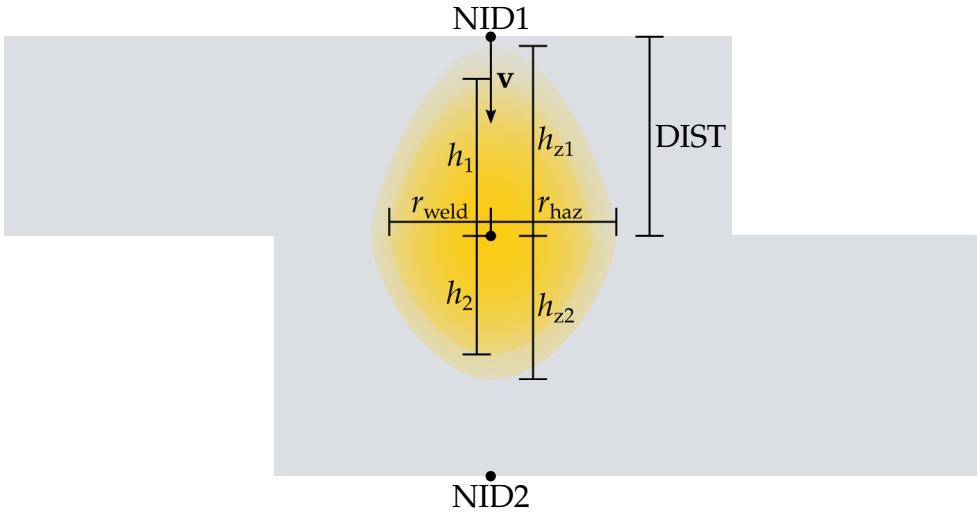


Figure 5-11. Example of geometry with OPTION = 1 (heat affected zone)

respectively. In between LS-DYNA calculates the base temperature in the nugget, $T_{\text{nug},b}(x,y,z)$, with a quadratic interpolation.

For this boundary condition you must provide activation and deactivation times, t_{ac} and t_{deac}, respectively. With LCIDT, you can specify the temperatures as a function, $\theta(\tilde{t})$, of the normalized time, $\tilde{t} = (t - t_{\text{ac}})/(t_{\text{deac}} - t_{\text{ac}})$, within this time span. For positive values of field LCIDT, a simple scaling is applied:

$$T_{\text{nug}}(x,y,z,t) = T_{\text{nug},b}(x,y,z) \times \theta(\tilde{t}(t)) \quad .$$

For negative values of LCIDT, a smooth evolution of the temperature from the birth time is ensured, so we recommend this type of load curve. The temperature is given by:

$$T_{\text{nug}}(x,y,z,t) = T_{\text{ac}}(x,y,z) + \Big(T_{\text{nug},b}(x,y,z) - T_{\text{ac}}(x,y,z)\Big) \times \theta(\tilde{t}) \quad .$$

Here T_{ac} corresponds to the temperature of a particular point right before the birth time.

4. **Heat affected zone.** With input field OPTION set to 1, the boundary condition expands to a heat affected zone that has the same general shape definition as the weld nugget described in Remark 2. The dimensions are here given by r_{haz}, h_{z1}, and h_{z2}. See Figure 5-11.

   You can prescribe the base temperature value at the boundary of the heat affected zone. LS-DYNA finds the base temperature in the heat affected zone, $T_{HAZ,b}(x,y,z)$ with a linear interpolation between the base boundary temperature of the nugget and the base boundary temperature of the heat affected zone. The load curve LCIDT governs the temperature evolution of the heat affected zone as described in Remark 3.

*BOUNDARY_TEMPERATURE_TRAJECTORY

Purpose: Apply a temperature boundary condition on nodes enclosed in either a cylindrical or rectangular prism volume moving along a trajectory. The center of the volume moves along the trajectory defined by a nodal path at a prescribed velocity. This keyword applies only to solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PTYP	NSID1	SPD1	NSID2	SPD2		RELVEL
Type	I	I	I	F	I	F		I
Default	none	1	none	none	none	none		0

Card 2	1	2	3	4	5	6	7	8
Variable	IFORM	LCID	TMULT	LCROT	LCMOV	LCLAT		
Type	I	I	F	I	I	I		
Default	none	none	none	none	none	none		

Card 3	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

Optional Volume Orientation Card. Additional card for NSID2 = 0.

Card 4	1	2	3	4	5	6	7	8
Variable	TX	TY	TZ					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID or part set ID to which the temperature boundary condition will be applied
PTYP	PID type: EQ.1: Part ID EQ.2: Part set ID
NSID1	Node set defining the path of the moving volume. The moving volume travels along the path at speed SPD1. The nodes are traversed according to their order in the node set. See Remark 1.
SPD1	Speed of the moving volume on the trajectory: GT.0.0: Constant speed LT.0.0: |SPD1| is a load curve ID defining the speed as a function of time.
NSID2	Node or segment set that specifies the orientation of the moving volume's center axis. GT.0: NSID2 together with SPD2 define a curve in the same way that NSID1 and SPD1 define a curve. Orientation of the moving volume's center axis is defined as a vector pointing from the current position on NSID2 to the current position on NSID1. EQ.0: The moving volume's center axis is oriented as (TX, TY, TZ) input on Card 4.

VARIABLE	DESCRIPTION
	LT.0: |NSID2| specifies a segment set. The moving volume's center axis is aligned with normals to segments in this set. To ensure that the axis orientation can be unambiguously determined at each point of the nodal path, LS-DYNA requires that each pair of consecutive nodes in NSID1 must both be in at least one segment of |NSID2|. When the center of the moving volume is on a node that is part of more than one segment in |NSID2|, the direction is determined by averaging the adjacent segment normals.
SPD2	Speed of reference point in NSID2 (ignored unless NSID2 > 0): GT.0: Constant speed LT.0: |SPD2| is a load curve ID defining the speed as a function of time.
RELVEL	Flag for whether SPD1 and SPD2 are relative or absolute speeds in a thermo-mechanical coupled analysis: EQ.0: Absolute speeds EQ.1: Relative speeds with respect to underlying structures
IFORM	Geometric description of the moving volume. See Remark 3 for details.
LCID	Load curve ID for temperature as a function of time. EQ.0: Temperature is a constant defined by the value TMULT.
TMULT	Curve multiplier for temperature
LCROT	Load curve defining the rotation angle (in degrees) of the moving volume around the trajectory as a function of time. See Remark 2.
LCMOV	Load curve defining the offset of the moving volume along its center axis as a function of time. See Remark 2.
LCLAT	Load curve defining the lateral offset of the moving volume as a function of time. See Remark 2.
P*i*	Parameters defining the moving volume's geometry. The meaning of each parameter depends on field IFORM. See Remark 3 and Figure 5-12 for details.

VARIABLE	DESCRIPTION
TX, TY, TZ	Orientation vector of the moving volume's center axis in global coordinates (NSID2 = 0 only)

Remarks:

1. **Volume Motion.** If NSID1 consists of nodes belonging to a certain part, the moving volume will follow the motion of the part. By setting field RELVEL to 1 the velocity of the moving volume can be defined relative to the motion of the nodes in NSID1.

2. **Volume Orientation.** There is a natural local coordinate system $(\mathbf{r}, \mathbf{s}, \mathbf{t})$ associated with the motion of the moving volume. The relative velocity vector on the trajectory of the moving volume defines the "forward" direction $\mathbf{r}$. The orientation vector aligned with the moving volume's center axis is denoted by $\mathbf{t}$. $\mathbf{s}$ is defined as $\mathbf{s} = \mathbf{t} \times \mathbf{r}$ by the right-handed rule and denotes the lateral direction.

   The position and orientation of the moving volume can be adjusted by rotating and translating the local coordinate system. Note that the system is always first rotated around the vector $\mathbf{r}$ by a value given by the load curve LCROT, resulting in a new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. Then, the system is translated in directions $\mathbf{t}'$ and $\mathbf{s}'$ according to values specified by LCMOV and LCLAT, respectively.

3. **Volume Geometry.** The moving volume can be either a cylinder or rectangular prism, depending on the values defined in field IFORM (see Figure 5-12).

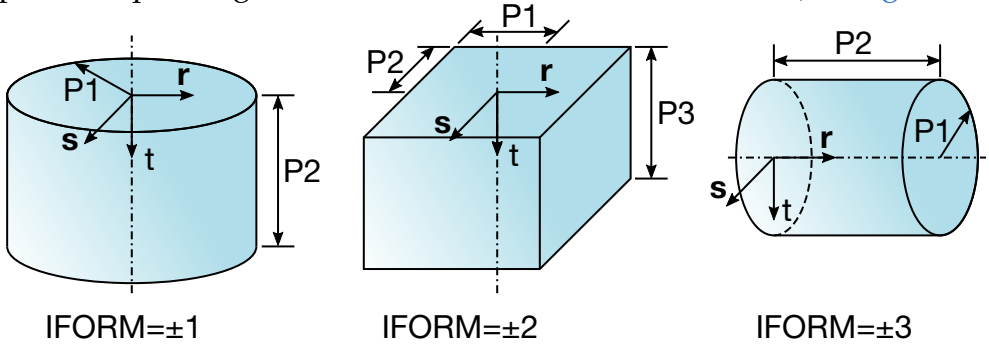


Figure 5-12. Definition of the geometry parameters

   a) *Cylinder (IFORM = 1* or *IFORM = -1).* P1 denotes the radius of the cylinder and P2 denotes the height of the cylinder. In the case that *IFORM = -1*, P1 and P2 are load curves that define the cylinder radius and height as functions of time.

b) *Rectangular Prism (IFORM = 2* or *IFORM = -2).* P1 denotes half the length of the rectangular prism, P2 denotes half width of the rectangular prism, and P3 denotes the height of the rectangular prism. In the case that *IFORM = -2*, P1, P2 and P3 are load curves that define the prism's half length, half width and heig ht as functions of time.

c) *Radially Oriented Cylinder (IFORM = 3* or *IFORM = -3).* P1 denotes the radius of the cylinder and P2 denotes the height of the cylinder. In the case that *IFORM = -3*, P1 and P2 are load curves that define the cylinder radius and height as functions of time. This cylinder differs from that defined by IFORM = ±1 in that the orientation vector, **t**, is along a radial direction instead of the length of the cylinder.

***BOUNDARY_THERMAL_BULKFLOW**_*OPTION1_OPTION2*

Purpose: Used to define bulk fluid flow elements.

OPTION1 is required since it specifies whether the BULKFLOW applies to an element or set.

ELEMENT

SET

OPTION2 if used turns on the fluid upwind algorithm

UPWIND

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID	LCID	MDOT					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
EID / SID	Beam element ID (EID) for ELEMENT option Beam set ID (SID) for SET option
LCID	Load Curve ID for mass flow rate versus time.
MDOT	Mass flow rate (e.g. kg/sec).

*BOUNDARY_THERMAL_BULKNODE

Purpose: Used to define thermal bulk nodes.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	PID	NBNSEG	VOL	LCID	H	AEXP	BEXP
Type	I	I	I	F	I	F	F	F
Default	none	none	0	none	0	0.	0.	0.

Bulk Node Cards. Include NBNSEG cards, one for each bulk node segment.

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4				
Type	I	I	I	I				

VARIABLE	DESCRIPTION
NID	Bulk node number.
PID	Bulk node part ID.
VOL	Bulk node volume.
NBNSEG	Number of element surface segments that transfer heat with this bulk node.
N1, N2, N3, N4	Nodal point numbers
LCID	Load curve ID for H
H	Heat transfer coefficient
AEXP	*a* exponent
BEXP	*b* exponent

Remarks:

The heat flow between a bulk node (T_B) and a bulk node segment (T_S) is given by

$$q = h(T_B^a - T_B^a)^b$$

1. For convection, set $a = b = 1$.

2. For radiation, set $a = 4,\ b = 1$.

3. For flux, set $a = b = 0$. Mathematically, anything to the 0 power is 1. This produces the expression, $\left(T_B^0 - T_S^0\right)^0 = (1-1)^0 = 0^0 = 1$. However, some computer operating systems don't recognize 0^0. It is safer to set $a = b =$ very small number.

*BOUNDARY_THERMAL_WELD

Purpose: Define a moving heat source to model welding. Only applicable for coupled thermal-structural simulations in which the weld source or work piece is moving.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PTYP	NID	NFLAG	X0	Y0	Z0	N2ID
Type	I	I	I	I	F	F	F	I
Default	none	1	↓	1	↓	↓	↓	none

Card 2	1	2	3	4	5	6	7	8
Variable	a	b	cf	cr	LCID	Q	Ff	Fr
Type	F	F	F	F	I	F	F	F
Default	none	none	none	none	Q	none	none	none

Beam Aiming Direction Card. Additional card for N2ID = -1.

Card 3	1	2	3	4	5	6	7	8
Variable	TX	TY	TZ					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID or Part Set ID to which weld source is applied
PTYP	PID type: EQ.1: PID defines a single part ID EQ.2: PID defines a part set ID

VARIABLE	DESCRIPTION
NID	Node ID giving location of weld source EQ.0: location defined by $(X0, Y0, Z0)$ below
NFLAG	Flag controlling motion of weld source EQ.1: source moves with node NID EQ.2: source is fixed in space at original position of node NID
X0, Y0, Z0	Coordinates of weld source, which remains fixed in space (ignored if NID is nonzero)
N2ID	Second node ID for weld beam aiming direction GT.0: beam is aimed from N2ID to NID, moves with these nodes EQ.-1: beam aiming direction is (TX, TY, TZ) input on Card 3
a	Weld pool radius (i.e., half width)
b	Weld pool depth (in beam aiming direction)
cf	Weld pool forward direction
cr	Weld pool rearward direction
LCID	Load curve ID for weld energy input rate as a function of time EQ.0: use constant multiplier value Q.
Q	Curve multiplier for weld energy input rate [energy/time, e.g., Watt] LT.0.0: use absolute value and accurate integration of heat
Ff	Forward distribution function
Fr	Rear distribution function (Note:$F_f + F_r = 2.0$)
TX, TY, TZ	Weld beam direction vector in global coordinates (N2ID = -1 only)

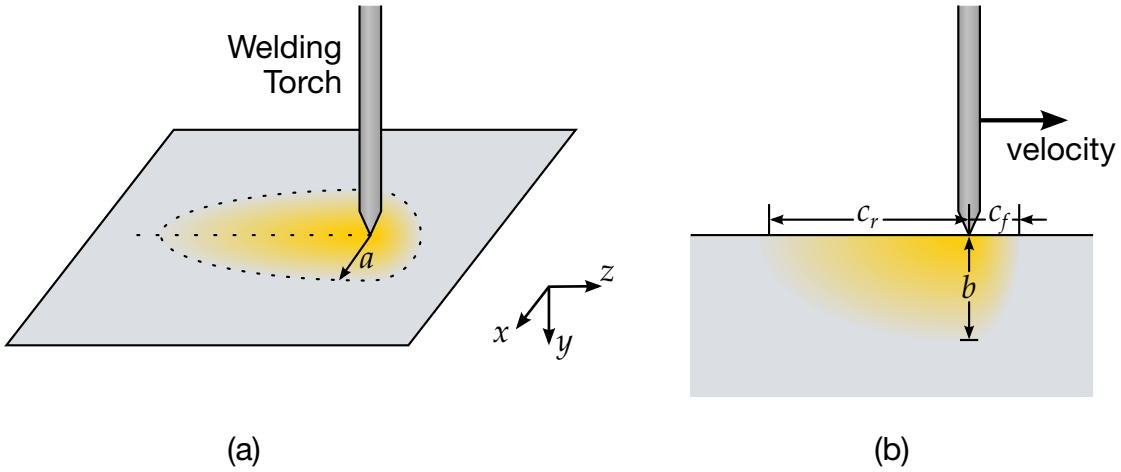


Figure 5-13. Schematic illustration of welding with moving torch. The left figure (a) shows the surface of the material from above, while the right figure (b) shows a slice along the dotted line in the y-z plane.

Remarks:

1. **Boundary Condition Model.** This boundary condition allows simulation of a moving weld heat source, following the work of Goldak, Chakravarti, and Bibby [1984]. Heat is generated in an ellipsoidal region centered at the weld source, and decaying exponentially with distance according to:

$$q = \frac{6\sqrt{3}FQ}{\pi\sqrt{\pi}abc}\exp\left(\frac{-3x^2}{a^2}\right)\exp\left(\frac{-3y^2}{b^2}\right)\exp\left(\frac{-3z^2}{c^2}\right),$$

where

$$\begin{aligned} q &= \text{weld source power density} \\ (x,y,z) &= \text{coordinates of point } p \text{ in weld material} \\ F &= \begin{cases} \mathrm{F_f} \text{ if point } p \text{ is in front of beam} \\ \mathrm{F_r} \text{ if point } p \text{ is behind beam} \end{cases} \\ c &= \begin{cases} \mathrm{c_f} \text{ if point } p \text{ is in front of beam} \\ \mathrm{c_r} \text{ if point } p \text{ is behind beam} \end{cases} \end{aligned}$$

A local coordinate system center at the heat source is constructed as shown in Figure 5-13. The relative velocity vector of the heat source defines the "forward" direction, so material points that are approaching the heat source are in "front" of the beam. The beam aiming direction is used to compute the weld pool depth. The weld pool width is measured normal to the relative velocity - aiming direction plane.

If Q is defined as a negative quantity in the input, then the formula above uses the absolute value of Q, and a more accurate integration of the heat source is performed with some additional cost in CPU time.

2. **Fixed Welding Torch Simulation.** To simulate a welding process during which the welding torch is fixed in space, NID and N2ID must be set to 0 and -1 respectively. The X0, Y0, and Z0 fields specify the global coordinates of the welding torch, and the TX, TY, and TZ fields specify the direction of the welding beam. The motion of the work piece is prescribed using the *BOUNDARY_PRESCRIBED_MOTION keyword.

3. **Fixed Work Piece Simulation.** To simulate a welding process for which the work piece fixed in space, NID and N2ID specify both the beam source location and direction. The X0, Y0, Z0, TX, TY, and TZ fields *are ignored*. The motion of welding source is prescribed with using the *BOUNDARY_PRESCRIBED_MOTION keyword applied to the two nodal points specified in the NID and N2ID fields.

*BOUNDARY_THERMAL_WELD_TRAJECTORY

Purpose: Define a moving heat source to model welding of solid or shell structures. Motion of the source is described by a nodal path and a prescribed velocity on this path. This keyword is applicable in coupled thermal-structural and thermal-only simulations (see Remark 1) and also supports thermal dumping. Different equivalent heat source descriptions are implemented.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PTYP	NSID1	SPD1	NSID2	SPD2	NCYC	RELVEL
Type	I	I	I	F	I	F	I	I
Default	none	1	none	none	none	none	1	0

Card 2	1	2	3	4	5	6	7	8
Variable	IFORM	LCID	Q	LCROT	LCMOV	LCLAT	DISC	ENFOR
Type	I	I	F	I	I	I	F	I
Default	none	none	none	none	none	none	none	0

Card 3	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F

Optional Weld Source Aiming Direction Card. Additional card for NSID2 = 0.

Card 4	1	2	3	4	5	6	7	8
Variable	TX	TY	TZ					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID or part set ID of solid and/or shell elements that are heated by the weld source. In the case of shell elements present in the heated structure, the thermal thick element formulation must be activated. See field THSHEL on *CONTROL_SHELL or field ITHELFM on *SECTION_SHELL_THERMAL.
PTYP	PID type: EQ.1: Part ID EQ.2: Part set ID
NSID1	Node set defining the path of the weld source. The source travels along the path at speed SPD1. The nodes are traversed according to their ordering in the node set. See Remark 1.
SPD1	Speed of the heat source on the weld trajectory: GT.0.0: Constant speed LT.0.0: |SPD1| is a load curve ID defining weld speed as a function of time.
NSID2	Node or segment set containing information for the weld source aiming direction: GT.0: NSID2 together with SPD2 define a curve in the same way that NSID1 and SPD1 define a curve. Aiming direction is taken to be the vector pointing from the current position along NSID2 (for example your hand holding the torch) to the current position on NSID1 (the weld source). EQ.0: Beam aiming direction is (TX, TY, TZ), input on Card 4. LT.0: |NSID2| specifies a segment set. The heat source is aimed perpendicularly to segments in this set. To ensure that a

VARIABLE	DESCRIPTION
	direction can be unambiguously associated with each point of the weld path, LS-DYNA requires that each pair of consecutive nodes in NSID1 must *both* be in at least one segment of |NSID2|. When the source is on a node that part of more than one segment in |NSID2|, then the direction is determined by averaging the adjacent segment normal vectors.
SPD2	Speed of reference point in NSID2 (ignored unless NSID2 > 0) GT.0: Constant speed LT.0: |SPD2| is a load curve ID defining weld speed as a function of time.
NCYC	Number of sub-steps for sub-cycling in evaluation of boundary condition. Allows thermal dumping. See Remark 3.
RELVEL	Defines if SPD1 and SPD2 are relative or absolute speeds in coupled simulations EQ.0: Absolute speeds EQ.1: Relative speeds with respect to underlying structure
IFORM	Geometry description for energy rate density distribution (see Remark 4): EQ.1: Goldak-type heat source EQ.2: Double ellipsoidal heat source with constant density EQ.3: Double conical heat source with constant density EQ.4: Frustum-shaped heat source with constant density EQ.5: User-defined function
LCID	Load curve ID for weld energy input rate as a function of time EQ.0: Use constant multiplier value *Q*.
Q	Curve multiplier for weld energy input rate [energy/time] LT.0: Take absolute value and accurate integration of heat using integration cells with edge length DISC.
LCROT	Load curve defining the rotation (angle in degrees) of weld source around the trajectory as function of time. See Remark 2.

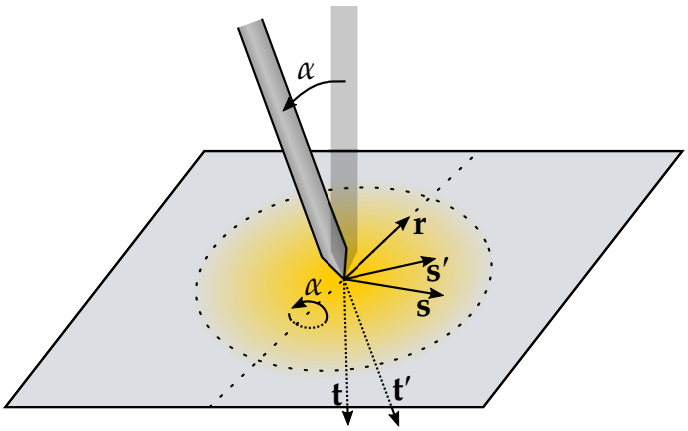


Figure 5-14. Local coordinates of the welding torch. These coordinates can be rotated about the **r**-axis to tilt the welding torch, forming a new set of local coordinates.

VARIABLE	DESCRIPTION
LCMOV	Load curve for offset of weld source in direction of the weld beam as function of time. See Remark 2.
LCLAT	Load curve for lateral offset of weld source as function of time. See Remark 2.
DISC	Resolution for accurate integration. Parameter defines edge length for integration cubes. Default is 5% of weld pool depth.
ENFOR	Flag for heat input enforcement option. If set, the nodal heat input is scaled such that the resulting heat inputs equals the user input as given by Q and LCID.
P*i*	Parameters defining for weld pool geometry, depending on field IFORM. See Remark 4 and Table 5-1 for details.
TX, TY, TZ	Weld beam direction vector in global coordinates (NSID2 = 0 only)

Remarks:

1. **Heat source motion.** As with the *BOUNDARY_THERMAL_WELD card this card can be applied to coupled thermal-structure simulations. Additionally, since the source motion can be defined independently form the motion of nodes, this card can be applied to thermal-only simulations. This keyword applies to both solid and thermal thick shells.

   If NSID1 consists of work piece nodes, then the heat source will follow the motion of the work piece. By setting field RELVEL to 1 the velocity of the heat

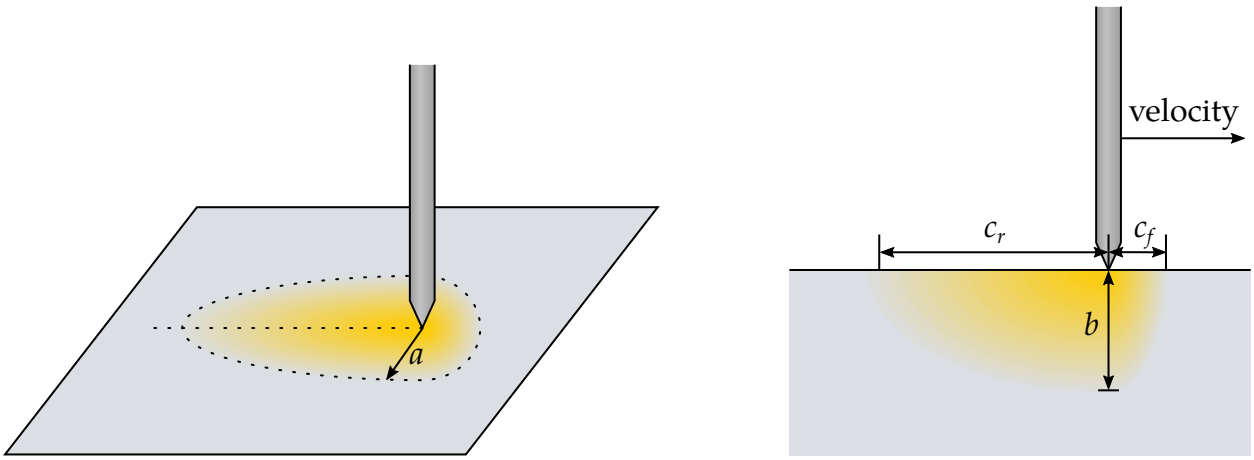


Figure 5-15. Heat source distribution geometry for IFORM = 1 and IFORM = 2

source can be defined relative to the motion of the nodes in NSID1 (the work piece).

2. **Heat source direction (tilting the torch).** There is a natural local coordinate system $(\mathbf{r}, \mathbf{s}, \mathbf{t})$ associated with the motion of the heat source. The relative velocity vector on the trajectory of the heat source defines the "forward" direction $\mathbf{r}$, so that material points that are approaching the heat source are in "front" of the beam. The weld source aiming direction, denoted by $\mathbf{t}$, determines the direction of the weld pool depth. The coordinate direction $\mathbf{s}$ is normal to the plane containing both the relative velocity and the aiming direction; note that $\mathbf{s} = \mathbf{t} \times \mathbf{r}$ to have a right handed coordinate system. It determines the direction of the weld pool width.

   The position and aiming direction of the heat source (for example tilting the torch) can be adjusted by rotating and translating the local coordinate system. To do this, the system is first rotated around the vector $\mathbf{r}$ by a value given by the load curve LCROT, resulting in a new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. Then, the system is translated in directions $\mathbf{t}'$ and $\mathbf{s}'$ using LCMOV and LCLAT, respectively. See Figure 5-14.

3. **Weld source sub-cycling.** The sub-cycling method introduces an individual time step size for the weld source evaluation. During each step of the heat transfer solver, NCYC steps are used to determine the energy rate distribution of the weld source. In each sub-step, the geometry of the weld pool is updated. Therefore, even with larger thermal time steps a relatively smooth temperature field around the weld source can be obtained and a jumping heat source across elements can be suppressed.

4. **Energy rate distribution geometries.** Several different heat source geometries that can be used with this keyword are described below. The local coordinate system needed for the description of the geometry is discussed in Remark 2.

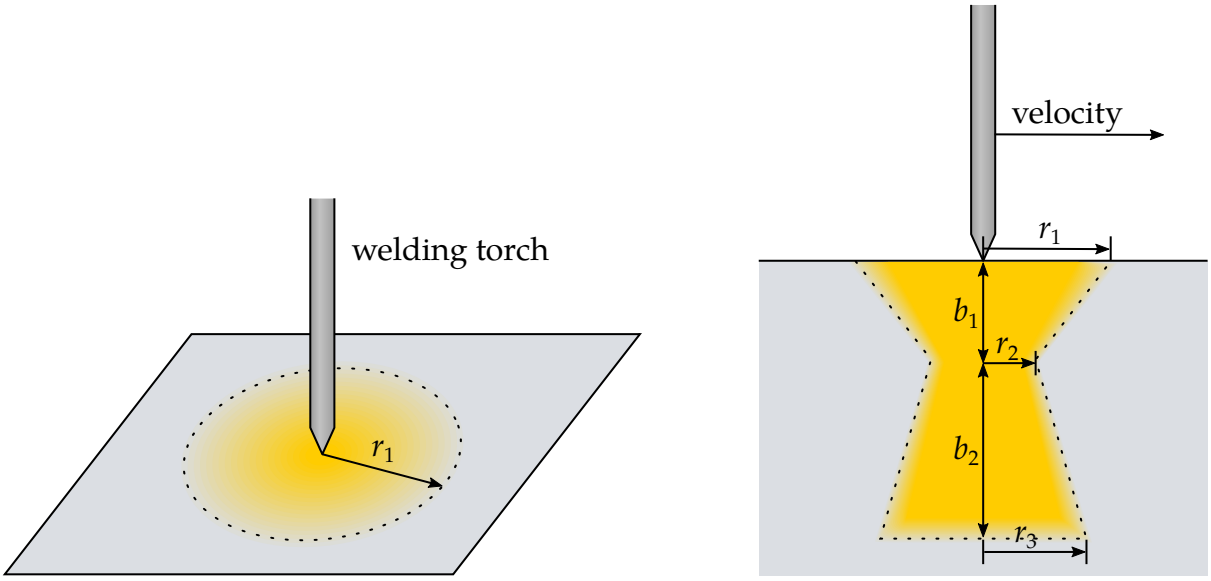


Figure 5-16. Heat source distribution geometry for IFORM = 3

For IFORM = 1 heat is generated in an ellipsoidal region centered at the weld source and decaying exponentially with distance according to the work of Goldak, Chakravarti, and Bibby [1984]. Energy rate distribution is governed by

$$q = \frac{2n\sqrt{n}FQ}{\pi\sqrt{\pi}abc}\exp\left(\frac{-nx^2}{a^2}\right)\exp\left(\frac{-ny^2}{b^2}\right)\exp\left(\frac{-nz^2}{c^2}\right) ,$$

where:

$$F = \begin{cases} \mathrm{F_f} \text{ if point } p \text{ is in front of beam} \\ \mathrm{F_r} \text{ if point } p \text{ is behind beam} \end{cases}$$
$$c = \begin{cases} \mathrm{c_f} \text{ if point } p \text{ is in front of beam} \\ \mathrm{c_r} \text{ if point } p \text{ is behind beam} \end{cases}$$

The local coordinates of point p are denoted by (x, y, z) and it is expected that the sum of the weighting factors $F_\mathrm{f}, F_\mathrm{r}$ equals 2. The half-width of the ellipsoid is given by a, the welding depth by b. See Figure 5-15. The complete set of parameters $(a, b, c_\mathrm{f}, c_\mathrm{r}, F_\mathrm{f}, F_\mathrm{r}, n)$ is input in the fields P1 to P7; see Table 5-1.

The energy rate density q for IFORM = 2 is assumed to be constant in the double ellipsoidal region as defined for Goldak-type heat sources. Its value is given by

$$q = \frac{3FQ}{2\pi abc}$$

with the same assumptions for F and c as above. The set of parameters consequently reduces to $(a, b, c_\mathrm{f}, c_\mathrm{r}, F_\mathrm{f}, F_\mathrm{r})$ which are input in fields P1 to P6.

In contrast to the above, IFORM = 3 defines an equivalent heat source with a constant energy rate density on a double conical region. The shape is defined

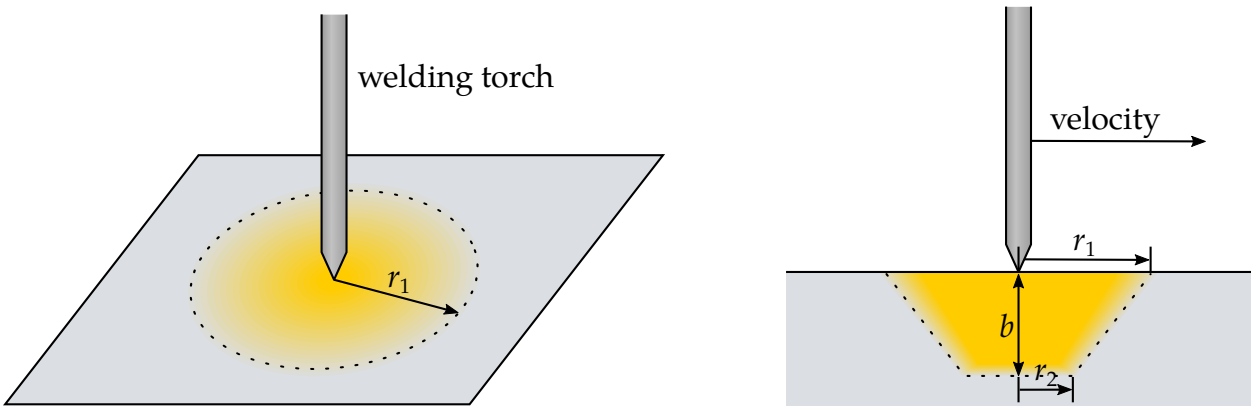


Figure 5-17. Heat source distribution geometry for IFORM = 4

by three radii $r_1, r_2, r_3$and two values b_1, b_2 defining the heights of the two parts of the shape. See Figure 5-16. The respective power densities of the parts are given by

$$q_i = \frac{3F_iQ}{2\pi b_i(r_i^2 + r_{i+1}^2 + r_i r_{i+1})}, \qquad i = 1, 2\,.$$

The parameter set $(r_1, r_2, r_3, b_1, b_2, F_1, F_2)$ is input in fields P1 to P7.

The last pre-defined form is IFORM = 4, which defines a constant power density over a frustum shaped region. See Figure 5-17. The density and the shape can easily be described using three geometrical parameters P1 to P3 corresponding to the radii r_1(at the heat source origin) and r_2 and the height b:

$$q = \frac{3Q}{\pi b(r_1^2 + r_2^2 + r_1 r_2)}$$

Finally, with IFORM = 5 the user can define an arbitrarily shaped equivalent heat source with the *DEFINE_FUNCTION keyword. The input of the function comprises the coordinates in the local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$, as discussed in Remark 2, the time and the weld speed. It may thus read as follows:

```
*DEFINE_FUNCTION
1,user heat source
heat(r,s,t,time,weldspd)= …
```

To speed-up the computation, the function is evaluated for nodes within a distance less than a from the current center position. Consequently, the function ID and this distance are the only parameters needed in Card 3 of the input.

IFORM	1	2	3	4	5
P1	a	a	r_1	r_1	FID
P2	b	b	r_2	r_2	a
P3	c_f	c_f	r_3	b_1	
P4	c_r	c_r	b_1		
P5	F_f	F_f	b_2		
P6	F_r	F_r	F_1		
P7	n		F_2		
P8					

Table 5-1. Parameters defining each weld pool geometry

*BOUNDARY_USA_SURFACE

Purpose: Define a surface for coupling with the USA code [DeRuntz 1993]. The outward normal vectors should point into the fluid media. The coupling with USA is operational in explicit transient and in implicit natural frequency analyses.

Card	1	2	3	4	5	6	7	8
Variable	SSID	WETDRY	NBEAM					
Type	I	I	I					
Default	none	0	0					

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT
WETDRY	Wet surface flag: EQ.0: Dry, no coupling for USA DAA analysis, or Internal fluid coupling for USA CASE analysis EQ.1: Wet, coupled with USA for DAA analysis, or External fluid coupling for USA CASE analysis
NBEAM	The number of nodes touched by USA Surface-of-Revolution (SOR) elements. It is not necessary that the LS-DYNA model has beams where USA has beams (i.e., SOR elements), merely that the LS-DYNA model has nodes to receive the forces that USA will return.

Remarks:

The underwater shock analysis code is an optional module. To determine availability, contact Ansys sales (https://www.ansys.com/contact-us).

The wet surface of 3 and 4-noded USA general boundary elements is defined in LS-DYNA with a segment set of 4-noded surface segments, where the fourth node can duplicate the third node to form a triangle. The segment normal vectors should be directed into the USA fluid. If USA overlays are going to be used to reduce the size of the DAA matrices, the user should nonetheless define the wet surface here as if no overlay were being used.

If Surface-of -Revolution elements (SORs) are being used in USA, then NBEAM should be non-zero on one and only one card in this section.

The wet surface defined here can cover structural elements or acoustic fluid volume elements, but it cannot touch both types in one model.

When running a coupled problem with USA, the procedure requires an additional input file of USA keyword instructions. These are described in a separate USA manual. The name of this input file is identified on the command line with the usa = flag:

LSDYNA.USA i=**inf** usa=**uin**

where **uin** is the USA keyword instruction file.

BOUNDARY_ELEMENT_METHOD_*OPTION

Available options include:

CONTROL

FLOW

NEIGHBOR

SYMMETRY

WAKE

Purpose: Define input parameters for boundary element method analysis of incompressible fluid dynamics or fluid-structure interaction problems.

The boundary element method (BEM) can be used to compute the steady state or transient fluid flow about a rigid or deformable body. The theory which underlies the method (see the LS-DYNA Theory Manual) is restricted to inviscid, incompressible, attached fluid flow. The method should not be used to analyze flows where shocks or cavitation are present.

In practice the method can be successfully applied to a wider class of fluid flow problems than the assumption of inviscid, incompressible, attached flow would imply. Many flows of practical engineering significance have large Reynolds numbers (above 1 million). For these flows the effects of fluid viscosity are small if the flow remains attached, and the assumption of zero viscosity may not be a significant limitation. Flow separation does not necessarily invalidate the analysis. If well-defined separation lines exist on the body, then wakes can be attached to these separation lines and reasonable results can be obtained. The Prandtl-Glauert rule can be used to correct for non-zero Mach numbers in a gas, so the effects of aerodynamic compressibility can be correctly modeled (as long as no shocks are present).

The BOUNDARY_ELEMENT_METHOD_FLOW card turns on the analysis, and is mandatory.

This capability is available only with shared memory parallel (SMP) LS-DYNA executables.

*BOUNDARY_ELEMENT_METHOD_CONTROL

Purpose: Control the execution time of the boundary element method calculation. Using the CONTROL keyword option is highly recommended. The BEM calculations can easily dominate the total execution time of an LS-DYNA run unless the parameters on this card (especially DTBEM and/or IUPBEM) are used appropriately.

The field DTBEM is used to increase the time increment between calls to the BEM routines. This can usually be done with little loss in accuracy since the characteristic times of the structural dynamics and the fluid flow can differ by several orders of magnitude. The characteristic time of the structural dynamics in LS-DYNA is given by the size of the smallest structural element divided by the speed of sound of its material. For a typical problem this characteristic time might be equal to 1 microsecond. Since the fluid in the boundary element method is assumed to be incompressible (infinite speed of sound), the characteristic time of the fluid flow is given by the streamwise length of the smallest surface in the flow divided by the fluid velocity. For a typical problem this characteristic time might be equal to 10 milliseconds. For this example DTBEM might be set to 1 millisecond with little loss of accuracy. Thus, for this example, the boundary element method would be called only once for every 1000 LS-DYNA iterations, saving an enormous amount of computer time.

The field IUPBEM is used to increase the number of times the BEM routines are called before the matrix of influence coefficients is recomputed and factored (these are time-consuming procedures). If the motion of the body is entirely rigid body motion, there is no need to ever recompute and factor the matrix of influence coefficients after initialization, and the execution time of the BEM can be significantly reduced by setting IUPBEM to a very large number. For situations where the structural deformations are modest an intermediate value, such as 10, for IUPBEM can be used.

Card 1	1	2	3	4	5	6	7	8
Variable	LWAKE	DTBEM	IUPBEM	FARBEM				
Type	I	F	I	F				
Default	50	0.	100	2.0				
Remarks	1			2				

VARIABLE	DESCRIPTION
LWAKE	Number of elements in the wake of lifting surfaces. Wakes must be defined for all lifting surfaces.
DTBEM	Time increment between calls to the boundary element method. The fluid pressures computed during the previous call to the BEM will continue to be used for subsequent LS-DYNA iterations until a time increment of DTBEM has elapsed.
IUPBEM	The number of times the BEM routines are called before the matrix of influence coefficients is recomputed and refactored.
FARBEM	Non-dimensional boundary between near-field and far-field calculation of influence coefficients.

Remarks:

1. **Wakes.** Wakes convect with the free-stream velocity. The number of elements in the wake should be set to provide a total wake length equal to 5-10 times the characteristic streamwise length of the lifting surface to which the wake is attached. Note that each wake element has a streamwise length equal to the magnitude of the free stream velocity multiplied by the time increment between calls to the boundary element method routines. This time increment is controlled by DTBEM.

2. **FARBEM.** The most accurate results will be obtained with FARBEM set to 5 or more, while values as low as 2 will provide slightly reduced accuracy with a 50% reduction in the time required to compute the matrix of influence coefficients.

*BOUNDARY_ELEMENT_METHOD_FLOW

Purpose: Turn on the boundary element method calculation, specify the set of shells which define the surface of the bodies of interest, and specify the onset flow.

The *BOUNDARY_ELEMENT_METHOD_FLOW card turns on the BEM calculation. This card also identifies the shell elements which define the surfaces of the bodies of interest, and the properties of the onset fluid flow. The onset flow can be zero for bodies which move through a fluid which is initially at rest.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	VX	VY	VZ	RO	PSTATIC	MACH	
Type	I	F	F	F	F	F	F	
Default	none	none	none	none	none	0.	0.	
Remark	1					2	3	

VARIABLE	DESCRIPTION
SSID	Shell set ID for the set of shell elements which define the surface of the bodies of interest (see *SET_SHELL). The nodes of these shells should be ordered so that the shell normal vectors point into the fluid.
VX, VY, VZ	x, y, and z components of the free-stream fluid velocity
RO	Fluid density
PSTATIC	Fluid static pressure
MACH	Free-stream Mach number

Remarks:

1. **Recommended Model Setup.** Using the NULL material (see *MAT_NULL) for the shell segments in the SSID set is recommended. Fluid pressures can then be displayed in the post-processor. For triangular shells the 4th node number should be the same as the 3rd node number. For fluid-structure interaction problems the boundary element shells should use the same nodes and be coincident

with the structural shell elements (or the outer face of solid elements) which define the surface of the body. This approach guarantees that the boundary element segments will move with the surface of the body as it deforms.

2. **Fluid Static Pressure.** A pressure of PSTATIC is applied uniformly to all segments in the segment set. If the body of interest is hollow, then PSTATIC should be set to the free-stream static pressure minus the pressure on the inside of the body.

3. **Subsonic Compressibility.** The effects of subsonic compressibility on gas flows can be included using a non-zero value for MACH. The pressures which arise from the fluid flow are increased using the Prandtl-Glauert compressibility correction. MACH should be set to zero for water or other liquid flows.

*BOUNDARY_ELEMENT_METHOD_NEIGHBOR

Purpose: Define the neighboring elements for a given boundary element segment.

The pressure at the surface of a body is determined by the gradient of the doublet distribution on the surface (see the LS-DYNA Theory Manual). The "Neighbor Array" is used to specify how the gradient is computed for each boundary element segment. Ordinarily, the Neighbor Array is set up automatically by LS-DYNA, and no user input is required. The NEIGHBOR option is provided for those circumstances when the user desires to define this array manually.

Elements Cards. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NELEM	NABOR1	NABOR2	NABOR3	NABOR4			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
NELEM	Element number
NABOR1	Neighbor for side 1 of NELEM.
NABOR2	Neighbor for side 2 of NELEM.
NABOR3	Neighbor for side 3 of NELEM.
NABOR4	Neighbor for side 4 of NELEM.

Remarks:

Each boundary element has 4 sides (Figure 6-1). Side 1 connects the 1st and 2nd nodes, side 2 connects the 2nd and 3rd nodes, etc. The 4th side is null for triangular elements.

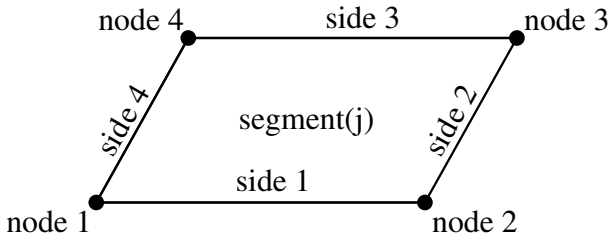


Figure 6-1. Each segment has 4 sides.

For most elements the specification of neighbors is straightforward. For the typical case a quadrilateral element is surrounded by 4 other elements, and the neighbor array is as shown in Figure 6-2.

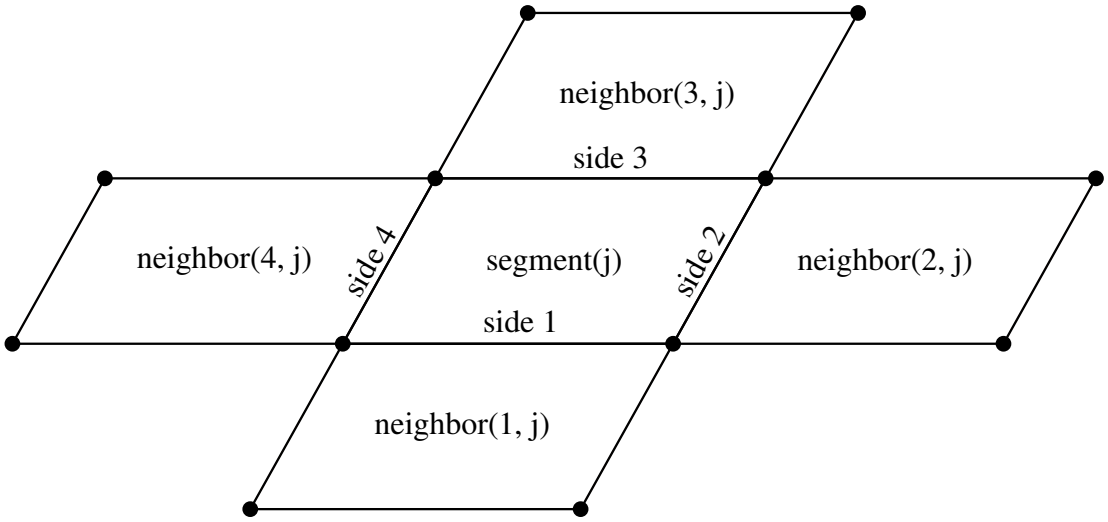


Figure 6-2. Typical neighbor specification.

There are several situations for which the user may desire to directly specify the neighbor array for certain elements. For example, boundary element wakes result in discontinuous doublet distributions, and neighbors which cross a wake should not be used. Figure 6-3 illustrates a situation where a wake is attached to side 2 of segment j. For this situation two options exist. If neighbor(2,j) is set to zero, then a linear computation of the gradient in the side 2 to side 4 direction will be made using the difference between the doublet strengths on segment j and segment neighbor(4,j). This is the default setup used by LS-DYNA when no user input is provided. By specifying neighbor(2,j) as a negative number a more accurate quadratic curve fit will be used to compute the gradient. The curve fit will use segment j, segment neighbor(4,j), and segment -neighbor(2,j); which is located on the opposite side of segment neighbor(4,j) as segment j.

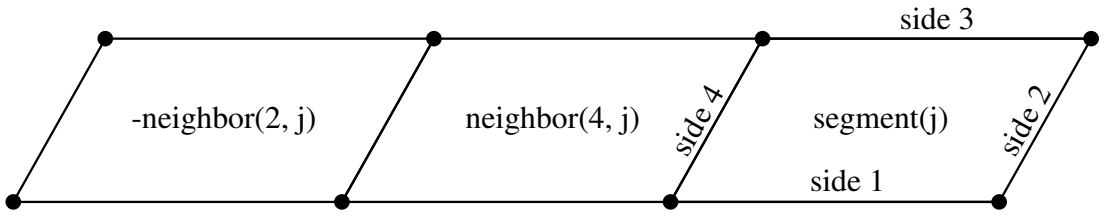


Figure 6-3. If neighbor(2,j) is a negative number, it is assumed to lie on the opposite side of neighbor(4,j) as segment j.

Another possibility is that no neighbors at all are available in the side 2 to side 4 direction. In this case both neighbor(2,j) and neighbor(4,j) can be set to zero, and the gradient in that direction will be assumed to be zero. This option should be used with caution, as the resulting fluid pressures will not be accurate for three-dimensional flows. However, this option is occasionally useful where quasi-two dimensional results are desired. All of the above options apply to the side 1 to side 3 direction in the obvious ways.

For triangular boundary elements side 4 is null. Gradients in the side 2 to side 4 direction can be computed as described above by setting neighbor(4,j) to zero for a linear derivative computation (this is the default setup used by LS-DYNA when no user input is provided) or to a negative number to use the segment on the other side of neighbor(2,j) and a quadratic curve fit. There may also be another triangular segment which can be used as neighbor(4,j) (see Figure 6-4).

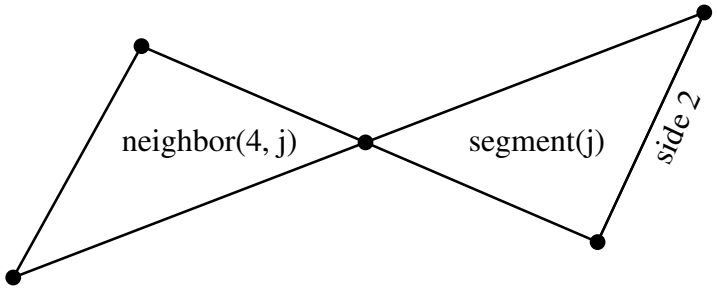


Figure 6-4. Sometimes another triangular boundary element segment can be used as neighbor (4,j).

The rules for computing the doublet gradient in the side 2 to side 4 direction can be summarized as follows (the side 1 to side 3 case is similar):

NABOR2	NABOR4	Doublet Gradient Computation
GT.0	GT.0	Quadratic fit using elements j, NABOR2, and NABOR4.
LT.0	GT.0	Quadratic fit using elements j, -NABOR2, and NABOR4. -NABOR2 is assumed to lie on the opposite side of NABOR4 as segment j (see Fig. 6-3).
GT.0	LT.0	Quadratic fit using elements j, NABOR2, and -NABOR4. -NABOR4 is assumed to lie on the opposite side of NABOR2 as segment j.
EQ.0	GT.0	Linear fit using elements j and NABOR4.
GT.0	EQ.0	Linear fit using elements j and NABOR2.
EQ.0	EQ.0	Zero gradient.

Table 6-5. Surface pressure computation for element j.

*BOUNDARY_ELEMENT_METHOD_SYMMETRY

Purpose: To define a plane of symmetry for the boundary element method. The SYMMETRY option can be used to reduce the time and memory required for symmetric configurations. For these configurations the reduction in the number of boundary elements by a factor of 2 will reduce the memory used by the boundary element method by a factor of 4 and will reduce the compute time required to factor the matrix of influence coefficients by a factor of 8. Only 1 plane of symmetry can be defined.

Card 1	1	2	3	4	5	6	7	8
Variable	BEMSYM							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
BEMSYM	Defines symmetry plane for boundary element method. EQ.0: no symmetry plane is defined EQ.1: $x = 0$ is a symmetry plane EQ.2: $y = 0$ is a symmetry plane EQ.3: $z = 0$ is a symmetry plane

*BOUNDARY_ELEMENT_METHOD_WAKE

Purpose: To attach wakes to the trailing edges of lifting surfaces. Wakes should be attached to boundary elements at the trailing edge of a lifting surface (such as a wing, propeller blade, rudder, or diving plane). Wakes should also be attached to known separation lines when detached flow is known to exist (such as the sharp leading edge of a delta wing at high angles of attack). Wakes are required for the correct computation of surface pressures for these situations. As described above, two segments on opposite sides of a wake should never be used as neighbors.

Element Cards. (The next "*" card terminates the input.)

Card 1	1	2	3	4	5	6	7	8
Variable	NELEM	NSIDE						
Type	I	I						
Default	none	none						
Remarks	1							

VARIABLE	DESCRIPTION
NELEM	Element number to which a wake is attached.
NSIDE	The side of NELEM to which the wake is attached (see Figure 6-1). This should be the "downstream" side of NELEM.

Remarks:

1. **Elements and Wake.** Normally two elements meet at a trailing edge (one on the "upper" surface and one on the "lower" surface). The wake can be attached to either element, but not to both.

*CASE

The *CASE command provides a way of running multiple LS-DYNA analyses (or cases) sequentially by submitting a single input file. When *CASE commands are used to define multiple cases, some portions of the input will be shared by some or all of the cases and other portions will be unique to each case. Because the cases are run sequentially, the results from one case, e.g., a dynain file, can be used in the analysis of a different, subsequent case. Each case creates a unique set of output file names by prepending "case***n**." to the default file name, e.g., case101.d3plot, case102.glstat.

When the *CASE keyword appears in an input deck, it becomes necessary to append the word "CASE" to the LS-DYNA execution line. For example, an SMP LS-DYNA execution line might look something like

path_to_ls-dyna i=**input.k** ncpu=**-4** CASE

An MPP LS-DYNA execution line might look something like

mpirun –np 4 path_to_mpp971 i=**input.k** CASE

The default behavior is to run all the cases defined in the input deck sequentially in increasing numeric order. If any of the cases fail to run to normal termination, the subsequent cases will not be run.

To run a subset of the cases defined in the input deck, specify the case ID numbers following the word "CASE" on the execution line, for example, "CASE = 10,20,47" will only run case IDs 10, 20 and 47, and skip any other cases. There can be NO spaces before or after " = " or ",". The specified cases will be run in increasing numeric order and not necessarily in the order given on the command line.

*__CASE__*_{OPTION}*

Available options include:

<BLANK>

BEGIN_*N*

END_*N*

Purpose: Define a series of cases and perhaps subcases. The options *CASE_BEGIN_*n* and *CASE_END_*n* appear in pairs and *n* is a numeric ID of a subcase. Subcase IDs may be referenced by the *CASE command in defining a case. In other words, a case may consist of one or more subcases. All keywords appearing between *CASE_BEGIN_*n* and *CASE_END_*n* comprise subcase *n*. If no *CASE command is defined, then subcases defined by *CASE_BEGIN_*n* and *CASE_END_*n* then become cases. *CASE_BE-

GIN/*CASE_END can be nested, overlapped, and disjointed. Examples below demonstrate the use of these options.

An alternative way of defining subcases is by appending the string "CID = *n*" to the end of any keyword command. Any keyword so tagged will then be active only for those cases that reference subcase *n*. There can be more than one space between the keyword and "CID = *n*".

Any keyword in the input deck not associated with a subcase is active for all cases.

The following input syntax applies only to the *CASE command, not to *CASE_BEGIN/*CASE_END.

Card 1	1	2	3	4	5	6	7	8
Variable	CASEID	JOBID						
Type	I	C						
Default	none	none						

Command Line Argument Cards. Command line cards set additional command line arguments for the case CASEID (see Card 1, above). Include as many as needed, or as few as none. Command line cards end when the first character of the next card is numeric.

Card 2	1	2	3	4	5	6	7	8
Variable	COMMANDS							
Type	A							
Default	Not Required							

Subcase ID Cards. Define active subcase IDs for case CASEID (see Card 1, above). These cards continue until the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	SCID1	SCID2	…	…	…	…	…	…
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
CASEID	Identification number for case.
JOBID	Optional string (no spaces) to be used as the jobid for this case. If no JOBID is specified, the string CASEXX is used, where XX is the CASEID in field 1
COM-MANDS	Command line arguments.
SCIDn	Subcase ID active for case CASEID.

Remarks:

1. If no *CASE keyword appears, subcases defined with *CASE_BEGIN/*CASE_-END commands become cases and *CASE_BEGIN can optionally be followed by extra command line arguments.

2. If no *CASE keyword appears, it is an error to append "CID = n" to any keyword.

3. If multiple *CASE or *CASE_BEGIN keywords appear that have the same ID, their command line arguments and active commands are merged.

4. The *CASE or *CASE_BEGIN keywords cannot be used within an include (*INCLUDE) file.

Example 1:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ Define case 101 which includes subcase 1.
$ Define case 102 which includes subcase 4.
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*CASE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7
$   CASEID
       101 JOBID_FOR_CASE101
MEMORY=20M
1
$
*CASE
$   CASEID
       102
MEMORY=20M  NCYCLE=1845
4
$
*TITLE  CID=1
THIS IS THE TITLE FOR CASE 101
$
*TITLE  CID=4
THIS IS THE TITLE FOR CASE 102
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Example 2:

```
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  Illustrate overlapping subcases.
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*CASE_BEGIN_5
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>
*DATABASE_BINARY_D3THDT
1.e-5
*CASE_BEGIN_3
*DATABASE_NODOUT
1.e-5
*CASE_END_5
*DATABASE_ELOUT
1.e-5
*CASE_END_3
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Example 2 above will generate d3thdt and nodout for CID = 5, and nodout and elout for CID = 3.

***COMMENT**_*{OPTION}*

Available options include:

<BLANK>

NOECHO

All input that falls between a *COMMENT command and the subsequent line of input that has an asterisk in the first column thereby signaling the start of another keyword command, is not acted on by LS-DYNA. This provides a convenient way to interject multiple, successive lines of commentary anywhere inside an input deck.

*COMMENT also provides a convenient way to comment out an existing keyword command and all its associated input data as shown in an example below.

Unless you append keyword option NOECHO, lines of input that are deactivated by *COMMENT are echoed on the screen and to the messag and d3hsp files.

Card 1	1	2	3	4	5	6	7	8
Variable	COMMENT							
Type	A							
Default	none							

VARIABLE	DESCRIPTION
COMMENT	Any comment line.

Example:

In this excerpt from an input deck, 5 lines of comments including blank lines, are added to the input deck.

```
*KEYWORD
*COMMENT
Units of this model are mks.

Input prepared by John Doe.
Input checked by Jane Doe.

*CONTROL_TERMINATION
1.E-02
⋮
```

Example 2:

In this excerpt from an input deck, a contact is disabled by inserting *COMMENT command before the contact keyword command.

```
⋮
*COMMENT  *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_ID
$#     cid                                                                title
         1
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
1,2,0,3
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
0.2
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf

$#    soft    sofscl    lcidab    maxpar     sbopt     depth     bsort    frcfrq
         2
*SET_SEGMENT
$#     sid       da1       da2       da3       da4    solver
         1     0.000     0.000     0.000     0.000MECH
$#      n1        n2        n3        n4        a1        a2        a3        a4
      2842       626      3232      3242     0.000     0.000     0.000     0.000
      2846      2842       627      2843     0.000     0.000     0.000     0.000
⋮
```

*COMPONENT

The keyword *COMPONENT provides a way of incorporating specialized components and features. The keyword control cards in this section are defined in alphabetical order:

COMPONENT_GEBOD_*OPTION

COMPONENT_GEBOD_JOINT_*OPTION

*COMPONENT_HYBRIDIII

COMPONENT_HYBRIDIII_JOINT_*OPTION

***COMPONENT_GEBOD**_*OPTION*

Purpose: Generate a rigid body dummy based on dimensions and mass properties from the GEBOD database. The motion of the dummy is governed by equations integrated within LS-DYNA separately from the finite element model. Default joint characteristics (stiffnesses, stop angles, etc.) are set internally and should give reasonable results, however, they may be altered using the *COMPONENT_GEBOD_JOINT command. Contact between the segments of the dummy and the finite element model is defined using the *CONTACT_GEBOD command. The use of a positioning file is essential with this feature; see Appendix N for further details.

OPTION specifies the human subject type. The male and female types represent adults while the child is genderless.

MALE

FEMALE

CHILD

Card 1	1	2	3	4	5	6	7	8
Variable	DID	UNITS	SIZE					
Type	I	I	F					
Default	none	none	none					

Card 2	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	GX	GY	GZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
DID	Dummy ID. A unique number must be specified.

VARIABLE	DESCRIPTION
UNITS	System of units used in the finite element model. EQ.1: lbf × s^2 / in - in - s EQ.2: kg - m - s EQ.3: kgf × s^2 / mm - mm - s EQ.4: metric ton - mm - s EQ.5: kg - mm – ms
SIZE	Size of the dummy. This represents a combined height and weight percentile ranging from 0 to 100 for the male and female types. For the child the number of months of age is input with an admissible range from 24 to 240.
VX, VY, VZ	Initial velocity of the dummy in the global *x*, *y* and *z* directions.
GX, GY, GZ	Global *x*, *y*, and *z* components of gravitational acceleration applied to the dummy.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *COMPONENT_GEBOD_MALE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A 50th percentile male dummy with the ID number of 7 is generated in the
$  lbf*sec^2-inch-sec system of units.  The dummy is given an initial velocity of
$  616 in/sec in the negative x direction and gravity acts in the negative z
$  direction with a value 386 in/sec^2.
$
*COMPONENT_GEBOD_MALE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       did     units      size
          7         1        50
$        vx        vy        vz        gx        gy        gz
       -616         0         0         0         0      -386
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

COMPONENT_GEBOD_JOINT_*OPTION

Purpose: Alter the joint characteristics of a GEBOD rigid body dummy. Setting a joint parameter value to zero retains the default value set internally. See Appendix N for further details.

The following options are available.

PELVIS	RIGHT_ELBOW
WAIST	LEFT_HIP
LOWER_NECK	RIGHT_HIP
UPPER_NECK	LEFT_KNEE
LEFT_SHOULDER	RIGHT_KNEE
RIGHT_SHOULDER	LEFT_ANKLE
LEFT_ELBOW	RIGHT_ANKLE

Card 1	1	2	3	4	5	6	7	8
Variable	DID	LC1	LC2	LC3	SCF1	SCF2	SCF3	
Type	F	I	I	I	F	F	F	

VARIABLE	DESCRIPTION
DID	Dummy ID, see *COMPONENT_GEBOD_*OPTION*.
LC*i*	Load curve ID specifying the loading torque as a function of rotation (in radians) for the i^{th} degree of freedom of the joint.
SCF*i*	Scale factor applied to the load curve of the i^{th} joint degree of freedom.

Card 2	1	2	3	4	5	6	7	8
Variable	C1	C2	C3	NEUT1	NEUT2	NEUT3		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
C*i*	Linear viscous damping coefficient applied to the i^{th} DOF of the joint. Units are torque × time/radian, where the units of torque and time depend on the choice of UNITS in Card 1 of *COMPONENT_GEBOD_*OPTION*.
NEUT*i*	Neutral angle (degrees) of the joint's i^{th} DOF.

Card 3	1	2	3	4	5	6	7	8
Variable	LOSA1	HISA1	LOSA2	HISA2	LOSA3	HISA3		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
LOSA*i*	Value of the low stop angle (degrees) for the i^{th} DOF of this joint.
HISA*i*	Value of the high stop angle (degrees) for the i^{th} DOF of this joint.

Card 4	1	2	3	4	5	6	7	8
Variable	UNK1	UNK2	UNK3					
Type	F	F	F					
Default	0.	0.	0.					

VARIABLE	DESCRIPTION
UNK*i*	Unloading stiffness (torque/radian) for the i^{th} degree of freedom of the joint. This must be a positive number. Units of torque depend on the choice of UNITS in Card 1 of *COMPONENT_GEBOD_*OPTION*.

Example 1:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

```
$$$$   *COMPONENT_GEBOD_JOINT_LEFT_SHOULDER
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  The damping coefficients applied to all three degrees of freedom of the left
$  shoulder of dummy 7 are set to 2.5.  All other characteristics of this joint
$  remain set to the default value.
$
*COMPONENT_GEBOD_JOINT_LEFT_SHOULDER
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      did       lc1       lc2       lc3      scf1      scf2      scf3
         7         0         0         0         0         0         0
$       c1        c2        c3     neut1     neut2     neut3
       2.5       2.5       2.5         0         0         0
$    losa1     hisa1     losa2     hisa2     losa3     hisa3
         0         0         0         0         0         0
$     unk1      unk2      unk3
         0         0         0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Example 2:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *COMPONENT_GEBOD_JOINT_WAIST
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Load curve 8 gives the torque versus rotation relationship for the 2nd DOF
$  (lateral flexion) of the waist of dummy 7.  Also, the high stop angle of the
$  1st DOF (forward flexion) is set to 45 degrees.  All other characteristics
$  of this joint remain set to the default value.
$
*COMPONENT_GEBOD_JOINT_WAIST
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      did       lc1       lc2       lc3      scf1      scf2      scf3
         7         0         8         0         0         0         0
$       c1        c2        c3     neut1     neut2     neut3
         0         0         0         0         0         0
$    losa1     hisa1     losa2     hisa2     losa3     hisa3
         0        45         0         0         0         0
$     unk1      unk2      unk3
         0         0         0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*COMPONENT_HYBRIDIII

Purpose: Define a HYBRID III dummy. The motion of the dummy is governed by equations integrated within LS-DYNA separately from the finite element model. The dummy interacts with the finite element structure through contact interfaces. Joint characteristics (stiffnesses, damping, friction, etc.) are set internally and should give reasonable results, however, they may be altered using the *COMPONENT_HYBRIDIII_JOINT command. Joint force and moments can be written to an ASCII file (see *DATABASE_H3OUT).

Card 1	1	2	3	4	5	6	7	8
Variable	DID	SIZE	UNITS	DEFRM	VX	VY	VZ	
Type	I	I	I	I	F	F	F	
Default	none	none	none	1	0.	0.	0.	

Card 2	1	2	3	4	5	6	7	8
Variable	HX	HY	HZ	RX	RY	RZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
DID	Dummy ID. A unique number must be specified.
SIZE	Size of dummy. EQ.1: 5th percentile adult EQ.2: 50th percentile adult EQ.3: 95th percentile adult

NOTE: If negative, then the best of currently available joint properties is applied.

VARIABLE	DESCRIPTION
UNITS	System of units used in the finite element model. EQ.1: lbf × s^2 / in - in - s EQ.2: kg - m - s EQ.3: kgf × s^2 / mm - mm - s EQ.4: metric ton - mm - s EQ.5: kg - mm – ms
DEFRM	Deformability type: EQ.1: all dummy segments entirely rigid EQ.2: deformable abdomen (low density foam, mat #57) EQ.3: deformable jacket (low density foam, mat #57) EQ.4: deformable headskin (viscoelastic, mat #6) EQ.5: deformable abdomen/jacket EQ.6: deformable jacket/headskin EQ.7: deformable abdomen/headskin EQ.8: deformable abdomen/jacket/headskin
VX, VY, VZ	Initial velocity of the dummy in the global *x*, *y* and *z* directions.
HX, HY, HZ	Initial global *x*, *y*, and *z* coordinate values of the H-point.
RX, RY, RZ	Initial rotation of dummy about the H-point with respect to the global *x*, *y*, and *z* axes (degrees).

Example 1:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *COMPONENT_HYBRIDIII
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A 50th percentile adult rigid HYBRID III dummy with an ID number of 7 is defined
$  in the lbf*sec^2-inch-sec system of units.  The dummy is assigned an initial
$  velocity of 616 in/sec in the negative x direction.  The H-point is initially
$  situated at (x,y,z)=(38,20,0) and the dummy is rotated 18 degrees about the
$  global x-axis.
$
*COMPONENT_HYBRIDIII
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      did    . size     units     defrm        vx        vy        vz
         7         2         1         1     -616.        0.        0.
```

```
$       hx        hy        hz        rx        ry        rz
       38.       20.        0.       18.        0.        0.
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

***COMPONENT_HYBRIDIII_JOINT**_OPTION

Purpose: Alter the joint characteristics of a HYBRID III dummy. Setting a joint parameter value to zero retains the default value set internally. Joint force and moments can be written to an ASCII file (see *DATABASE_H3OUT). Further details pertaining to the joints are found in the Hybrid III Dummies section of Appendix N.

The following options are available:

LUMBAR	RIGHT_ELBOW	RIGHT_KNEE
LOWER_NECK	LEFT_WRIST	LEFT_ANKLE
UPPER_NECK	RIGHT_WRIST	RIGHT_ANKLE
LEFT_SHOULDER	LEFT_HIP	STERNUM
RIGHT_SHOULDER	RIGHT_HIP	LEFT_KNEE_SLIDER
LEFT_ELBOW	LEFT_KNEE	RIGHT_KNEE_SLIDER

Card 1	1	2	3	4	5	6	7	8
Variable	DID	Q1	Q2	Q3	FRIC			
Type	F	F	F	F	F			

Card 2	1	2	3	4	5	6	7	8
Variable	C1	ALO1	BLO1	AHI1	BHI1	QLO1	QHI1	SCLK1
Type	F	F	F	F	F	F	F	F

Leave blank if joint has only one degree of freedom.

Card 3	1	2	3	4	5	6	7	8
Variable	C2	ALO2	BLO2	AHI2	BHI2	QLO2	QHI2	SCLK2
Type	F	F	F	F	F	F	F	F

Leave blank if the joint has only two degrees of freedom.

Card 4	1	2	3	4	5	6	7	8
Variable	C3	ALO3	BLO3	AHI3	BHI3	QLO3	QHI3	SCLK3
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
DID	Dummy ID, see *COMPONENT_HYBRIDIII
Q*i*	Initial value of the joint's i^{th} degree of freedom. Units of degrees are defined for rotational DOF. See Appendix N for a listing of the applicable DOF.
FRIC	Friction load on the joint.
C*i*	Linear viscous damping coefficient applied to the i^{th} DOF of the joint.
ALO*i*	Linear coefficient for the low regime spring of the joint's i^{th} DOF.
BLO*i*	Cubic coefficient for the low regime spring of the joint's i^{th} DOF.
AHI*i*	Linear coefficient for the high regime spring of the joint's i^{th} DOF.
BHI*i*	Cubic coefficient for the high regime spring of the joint's i^{th} DOF.
QLO*i*	Value at which the low regime spring definition becomes active.
QHI*i*	Value at which the high regime spring definition becomes active.
SCLK*i*	Scale value applied to the stiffness of the joint's i^{th} DOF (default = 1.0).

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *COMPONENT_HYBRIDIII_JOINT_LEFT_ANKLE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  The damping coefficients applied to all three degrees of freedom of the left
$  ankle of dummy 7 are set to 2.5.  All other characteristics of this joint
$  remain set to the default value.  The dorsi-plantar flexion angle is set to
$  20 degrees.
```

```
$
*COMPONENT_HYBRIDIII_JOINT_LEFT_ANKLE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      did        q1        q2        q3      fric
         7         0       20.         0         0         0
$       c1      alo1      blo1      ahi1      bhi1      qlo1      qhi1
       2.5         0         0         0         0         0         0
$       c2      alo2      blo2      ahi2      bhi2      qlo2      qhi2
       2.5         0         0         0         0         0         0
$      2.5      alo3      blo3      ahi3      bhi3      qlo3      qhi3
```

*CONSTRAINED

The keyword *CONSTRAINED provides a way of constraining degrees of freedom to move together in some way. The keyword cards in this section are defined in alphabetical order:

*CONSTRAINED_ADAPTIVITY

*CONSTRAINED_BEAM_IN_SOLID

*CONSTRAINED_BUTT_WELD

*CONSTRAINED_COORDINATE

*CONSTRAINED_EULER_IN_EULER

CONSTRAINED_EXTRA_NODES_*OPTION

CONSTRAINED_GENERALIZED_WELD_*WELDTYPE

*CONSTRAINED_GLOBAL

*CONSTRAINED_IMMERSED_IN_SPG

*CONSTRAINED_INTERPOLATION

*CONSTRAINED_INTERPOLATION_SPOTWELD

CONSTRAINED_JOINT_*TYPE

CONSTRAINED_JOINT_COOR_*TYPE

CONSTRAINED_JOINT_STIFFNESS_*OPTION

*CONSTRAINED_JOINT_USER_FORCE

*CONSTRAINED_LAGRANGE_IN_SOLID

*CONSTRAINED_LINEAR_GLOBAL

*CONSTRAINED_LINEAR_LOCAL

*CONSTRAINED_LOCAL

*CONSTRAINED_MULTIPLE_GLOBAL

*CONSTRAINED_NODAL_RIGID_BODY

*CONSTRAINED

*CONSTRAINED_NODE_INTERPOLATION

*CONSTRAINED_NODE_SET

*CONSTRAINED_NODE_TO_NURBS_PATCH

*CONSTRAINED_POINTS

*CONSTRAINED_RIGID_BODIES

*CONSTRAINED_RIGID_BODY_INSERT

*CONSTRAINED_RIGID_BODY_STOPPERS

*CONSTRAINED_RIVET

*CONSTRAINED_SHELL_IN_SOLID

*CONSTRAINED_SHELL_TO_SOLID

*CONSTRAINED_SOIL_PILE

*CONSTRAINED_SOLID_IN_SOLID

*CONSTRAINED_SPLINE

*CONSTRAINED_SPOTWELD_*{OPTION}*_*{OPTION}*

*CONSTRAINED_SPR2

*CONSTRAINED_TIEBREAK

*CONSTRAINED_TIED_NODES_FAILURE

*CONSTRAINED_ADAPTIVITY

Purpose: Constrains a node created during *h*-adaptivity of three-dimensional shells to the midpoint along an edge of an element. This keyword is automatically created by LS-DYNA during an *h*-adaptive simulation involving three-dimensional shells. See the theory manual for details about this type of adaptivity.

Card 1	1	2	3	4	5	6	7	8
Variable	DNID	NID1	NID2					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
DNID	Dependent node. This is the node constrained at the midpoint of an edge of an element.
NID1	Node at one end of an element edge
NID2	Node at the other end of that same element edge

Remarks:

Not all nodes created during *h*-adaptivity are constrained to an element edge. It only occurs when a node on the edge of an element created during an adaptive step is *not* on the boundary of the mesh and is *not* a node belonging to an element on the other side of the edge. If the element on the other side of the edge is adapted, then this constraint may be deleted.

Figure 10-1 illustrates which elements become dependent after adapting a single element mesh. The left element is adapted to create the elements on the right. In this figure, the red dots are dependent nodes while the blue dots are the edge nodes constraining the dependent nodes. For instance, node 2 is a dependent node constrained at the midpoint of the edge between nodes 1 and 3 and node 5 is a dependent node constrained at the midpoint of the edge between nodes 4 and 6.

Suppose that the element to the left of node 2 is adapted once. Then node 2 is no longer a dependent node and node 3 is no longer a constraining node as shown in Figure 10-2. These nodes are purple dots to illustrate that they are no longer part of a *CON-

STRAINED_ADAPTIVITY definition. However, node 1 continues to constrain the node to its right and now constrains the node to its left (node 7) with node 8.

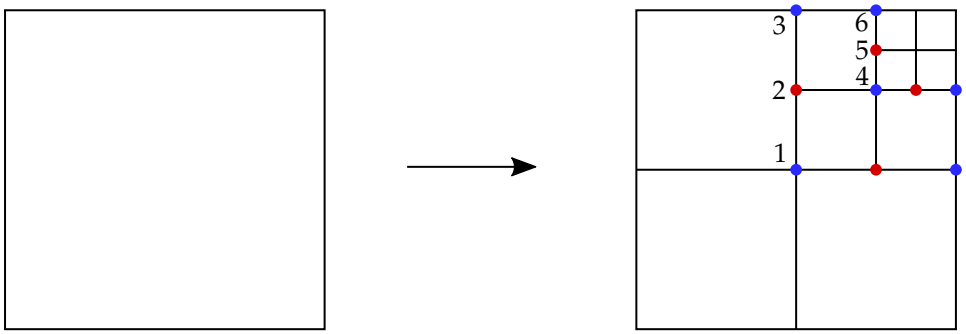


Figure 10-1. Illustration of which nodes are constrained after adaptive steps. The red nodes are dependent while the blue nodes constrain the dependent node that shares the edge.

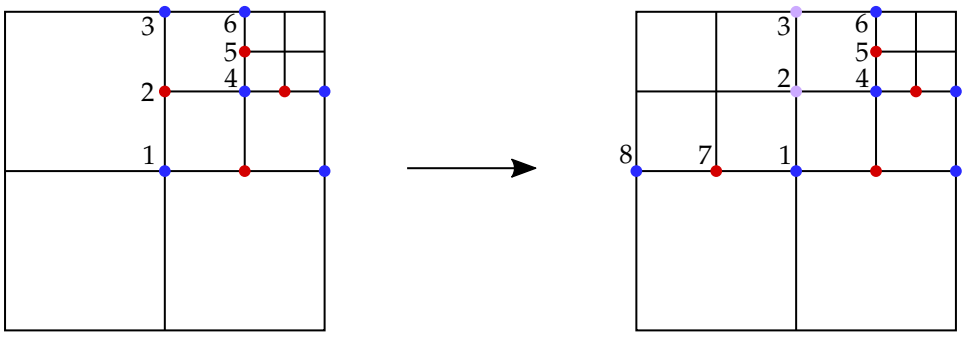


Figure 10-2. Illustration of which nodes are constrained after the element to the left of Node 2 is adapted. The purple nodes indicate which ones are no longer part of constraints.

*CONSTRAINED_BEAM_IN_SOLID_{OPTION1}_{OPTION2}

This keyword could take the following two forms:

*CONSTRAINED_BEAM_IN_SOLID

*CONSTRAINED_BEAM_IN_SOLID_PENALTY

To define an ID and heading, the following options are available:

ID

TITLE

Purpose: This keyword provides either constraint-based or penalty-based coupling between beams embedded in solid or thick shell elements. For shells embedded in solid or thick shell elements, see *CONSTRAINED_SHELL_IN_SOLID.

*CONSTRAINED_BEAM_IN_SOLID invokes constraint-based coupling. It constrains beam structures to move with Lagrangian solids/tshells. Both acceleration and velocity are constrained. This feature is intended to sidestep certain limitations in the CTYPE = 2 implementation of *CONSTRAINED_LAGRANGE_IN_SOLID. Notable features of this keyword include:

1. **CDIR = 1 feature.** With the CDIR = 1 option, coupling occurs only in the normal directions. This option allows for releasing the constraints along the beam axial direction and provides a means to model the debonding process.

2. **Axial coupling force.** Debonding processes can be modeled if a shear force-slip relationship is given and CDIR is set to 1 to release the axial constraints. There are three ways to specify the bond force-slip relationship:

   a) *A function defined with *DEFINE_FUNCTION*. To activate this, set the AXFOR flag to a negative integer which refers to the *DEFINE_FUNCTION ID.

   b) *A user-defined subroutine*. You provide a subroutine giving the axial shear force based on the slip between rebar nodes and concrete solid elements. To enable this, you set AXFOR to a number greater than 1000, and the user must modify the subroutine `rebar_bondslip_get_force()` in the user-defined subroutine Fortran files to add in one or more debonding laws, each tagged with a "lawid." An AXFOR value greater than 1000 will call the user subroutine and pass AXFOR in as "lawid."

   c) *A general built-in force-slip relationship*. Starting with R14.0, AXFOR = 999 invokes a general bond force-slip relationship based on Bulletin 65 of the fib Model Code for Concrete Structures 2010.

3. **NCOUP feature.** NCOUP is used to invoke coupling at both beam nodes and coupling points between the two beam nodes of each beam element. This implementation resolves the errors in energy balance observed when using the alternative implementation, *CONSTRAINED_LAGRANGE_IN_SOLID, CTYPE = 2.

4. **Tetrahedral and pentahedral solid elements.** These element shapes are supported in this keyword and are no longer treated as degenerated hexahedra as in the *CONSTRAINED_LAGRANGE_IN_SOLID CTYPE = 2 implementation.

5. **Constraints on nodes.** The *CONSTRAINED_LAGRANGE_IN_SOLID CTYPE = 2 implementation does not constrain beam nodes embedded in elements whose nodes have prescribed motion or other constraints.

6. **R-adaptivity support.** Severely deformed solid elements could terminate the simulation prematurely. One way to address this problem is to use the 3D tetrahedral *r*-adaptivity method to remesh and restart. This keyword supports *r*-adaptivity and can be used to model manufacturing processes with severe deformations, such as compression molding with embedded fibers.

7. **Implicit support.** Both SMP and MPP are supported.

8. **Optimized sorting.** The sorting subroutine is optimized for larger problems to achieve better performance with less memory usage.

9. **Thermal support.** In case the implicit thermal solver is invoked to perform either a thermal analysis (SOLN = 1 in *CONTROL_SOLUTION) or a coupled structural thermal analysis (SOLN = 2 in *CONTROL_SOLUTION), the temperature field is also constrained between beam and solid nodes.

*CONSTRAINED_BEAM_IN_SOLID_PENALTY invokes penalty-based coupling. A penalty spring is attached between coupling points on the beam and in the solid/tshell element. Penalty spring stiffness is calculated based on the geometric mean of the beam and the solid's bulk moduli. The magnitude of this coupling force can be controlled through field PSSF (penalty spring stiffness scale factor). This penalty coupling conserves kinetic energy much better in transient problems like blast loading. Please note that the CDIR and AXFOR fields only work with constraint-based coupling and are not available with penalty-based coupling.

Card Summary:

Card ID. This card is included if the TITLE or ID keyword option is used.

COUPID	TITLE

Card 1. This card is required.

BSID	SSID	BSTYP	SSTYP			NCOUP	CDIR

Card 2. This card is required.

START	END		AXFOR		PSSF		XINT

Card 3. This card is included if AXFOR = 999.

BONDC	BAREA	FCM	S1	S2	CLEAR	ALPHA	

Data Card Definitions:

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, an internal coupling ID is assigned based on the order of appearance in the input deck.
TITLE	A description of this coupling definition

Card 1	1	2	3	4	5	6	7	8
Variable	BSID	SSID	BSTYP	SSTYP			NCOUP	CDIR
Type	I	I	I	I			I	I
Default	none	none	0	0			0	0

VARIABLE	DESCRIPTION
BSID	Part or part set ID of the Lagrangian beam structure (see *PART and *SET_PART).
SSID	Part or part set ID of the Lagrangian solid elements or thick shell elements (see *PART and *SET_PART).
BSTYP	Set type of BSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
SSTYP	Set type of SSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
NCOUP	Number of coupling points generated in one beam element. If set to 0, coupling only happens at beam nodes. Otherwise, coupling is done at both the beam nodes and those automatically generated coupling points. NCOUP > 0 works only in explicit analysis; it is automatically set to zero in implicit analysis.
CDIR	Coupling direction. Only available in constraint form. EQ.0: Constraint applied along all directions. EQ.1: Constraint only applied along normal directions; along the beam axial direction, there is no constraint.

Card 2	1	2	3	4	5	6	7	8
Variable	START	END		AXFOR		PSSF		XINT
Type	F	F		I		F		F
Default	0	10^{20}		0		0.1		10^{16}

VARIABLE	DESCRIPTION
START	Start time to activate the coupling: LT.0.0: Start time is set to |START|. When negative, start time is followed during the dynamic relaxation phase of the

VARIABLE	DESCRIPTION
	calculation. After the completion of dynamic relaxation, coupling is activated regardless of the value of END. EQ.0.0: Start time is inactive, meaning coupling is always active GT.0.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, if END > 0, the start time applies during and after dynamic relaxation.
END	End time to deactivate the coupling: LT.0.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, negative END indicates that coupling is inactive during dynamic relaxation. After dynamic relaxation, the start and end times are followed and set to |START| and |END|, respectively. EQ.0.0: END defaults to 10^{20}. GT.0.0: END sets the time at which the coupling is deactivated.
AXFOR	Function that calculates the coupling force in the beam axial direction. Only available in constraint form with CDIR = 1. GE.0.and.LT.999: Off LT.0: |AXFOR| is a function ID; see *DEFINE_FUNCTION. See Example 1. EQ.999: General bond stress-slip relationship, which follows Bulletin 65 of fib Model Code for Concrete Structures 2010. GT.1000: Debonding law ID, "lawid," in the user-defined subroutine `rebar_bond-slip_get_force()`. See Example 2.
PSSF	Penalty spring stiffness scale factor. Only available in penalty form.
XINT	Interval distance. This field is designed to deal with beam elements with wide length variations. Coupling points are generated at an interval of length equal to XINT. Hence the number of coupling points in a beam element is no longer a fixed number (NCOUP) but instead variable, depending on the length of the beam element.

VARIABLE	DESCRIPTION
	This field can be used together with NCOUP. In that case, we will take the larger number of coupling points from these two options in each element.

General Bond Stress-Slip Relationship Card. Additional card if AXFOR = 999.

Card 3	1	2	3	4	5	6	7	8
Variable	BONDC	BAREA	FCM	S1	S2	CLEAR	ALPHA	
Type	I	F	F	F	F	F	F	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
BONDC	Bond condition flag. BONDC consists of a two-digit integer, BONDC = [BA]: $$\text{BONDC} = B \times 10 + A$$ The ones digit controls the transverse reinforcement: A.EQ.0: Unconfined A.EQ.1: Stirrups The tens digit controls the bond condition: B.EQ.0: Good bond condition B.EQ.1: All other bond condition
BAREA	Beam area
FCM	Concrete compression strength
S1	Slip at maximum stress (pull-out)
S2	Slip at the end of stress plateau (pull-out)
CLEAR	Clearance distance between ribs
ALPHA	α, see Bulletin 65 of fib Model Code for Concrete Structures 2010

Examples:

1. **Function for modeling debonding.** The example below shows how to define and use a function to prescribe the debonding process. User-defined functions are supported. The function computes the debonding force and has two internally calculated arguments: slip and leng. Slip is the relative axial displacement between the beam node (or coupling point) and the material in which the beam is embedded. Leng is the tributary length of the beam node or coupling point. Implicit calculations require a third argument which is output by the function: stiff. Stiff is the debonding spring stiffness. The asterisk in front of stiff (*stiff) is required to indicate that it is called by reference, meaning that its value is returned after the function is evaluated. Please note that this asterisk cannot be placed in the first column of the function body because the keyword reader assumes asterisks start new keywords.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*CONSTRAINED_BEAM_IN_SOLID
$#    bsid      ssid     bstyp     sstyp                          ncoup      cdir
         2         1         1         1                              2         1
$#   start       end               axfor
     0.000     0.000                 -10
*DEFINE_FUNCTION
        10
float force(float slip,float leng, float *stiff)
{
  float force,pi,d,area,shear,pf;
  pi=3.1415926;
  d=0.175;
  area=pi*d*leng;
  pf=1.0;
  if (slip<0.25) {
    shear=slip*pf;
  } else {
    shear=0.25*pf;
  }
  force=shear*area;
  *stiff=pf*area;
  return force;
}
```

2. **User subroutine for modeling debonding.** The example below shows how to define a user subroutine and use it to prescribe the debonding process.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*CONSTRAINED_BEAM_IN_SOLID
$#    bsid      ssid     bstyp     sstyp                          ncoup      cdir
         2         1         1         1                              2         1
$#   start       end               axfor
     0.000     0.000                1001
*CONSTRAINED_BEAM_IN_SOLID
$#    bsid      ssid     bstyp     sstyp                          ncoup      cdir
         3         1         1         1                              2         1
$#   start       end               axfor
     0.000     0.000                1002
*USER_LOADING
$    parm1     parm2     parm3     parm4     parm5     parm6     parm7     parm8
       1.0       6.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

The user debonding law subroutine:

```
      subroutine rebar_bondslip_get_force(slip,dl,force,hsv,
     .  userparm,lawid)
      real hsv
      dimension hsv(12),cm(8),userparm(*)
c
c in this subroutine user defines debonding properties and
c call his debonding subroutine to get force
      cm(1)=userparm(1)
      cm(2)=userparm(2)
      cm(3)=2.4*(cm(2)/5.0)**0.75
      cm(8)=0.
c
      pi=3.1415926
      d=0.175
      area=pi*0.25*d*d*dl
      pf=1.0
c
      if (lawid.eq.1001) then
        if (slip.lt.0.25) then
          shear=slip*pf
        else
          shear=0.25*pf
        endif
        force=sign(1.0,slip)*shear*area
      elseif (lawid.eq.1002) then
        if (slip.lt.0.125) then
          shear=slip*pf
        else
          shear=0.125*pf
        endif
      endif
      return
      end
```

*CONSTRAINED_BUTT_WELD

Purpose: Define a line of coincident nodes that represent a structural butt weld between two parts defined by shell elements. Failure is based on nodal plastic strain for ductile failure and stress resultants for brittle failure. This input is much simpler than the alternative approach for defining butt welds, *CONSTRAINED_GENERALIZED_WELD_BUTT. With this keyword, LS-DYNA automatically determines the local coordinate system, the effective length, and thickness for each pair of butt welded nodes. For *CONSTRAINED_GENERALIZED_WELD_BUTT, these quantities must be defined in the input.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID1	NSID2	EPPF	SIGF	BETA			
Type	I	I	F	F	F			
Default	none	none	0.	10^{16}	1.0			
Remarks		1, 2	3, 4	3	3			

VARIABLE	DESCRIPTION
NSID1	Node set ID for one side of the butt weld, see *SET_NODE_*OPTION*.
NSID2	Node set ID for the other side of the butt weld
EPPF	Plastic strain at failure
SIGF	σ_f, stress at failure for brittle failure
BETA	β, failure parameter for brittle failure

Remarks:

1. **Butt Weld Geometry and Node Sets.** Nodes in NSID1 and NSID2 must be given in the order they appear as one moves along the edge of the surface. An equal number of coincident nodes must be defined in each set. In a line weld the first and last node in a string of nodes can be repeated in the two sets. If the first and last pair of nodal points are identical, LS-DYNA assume the geometry

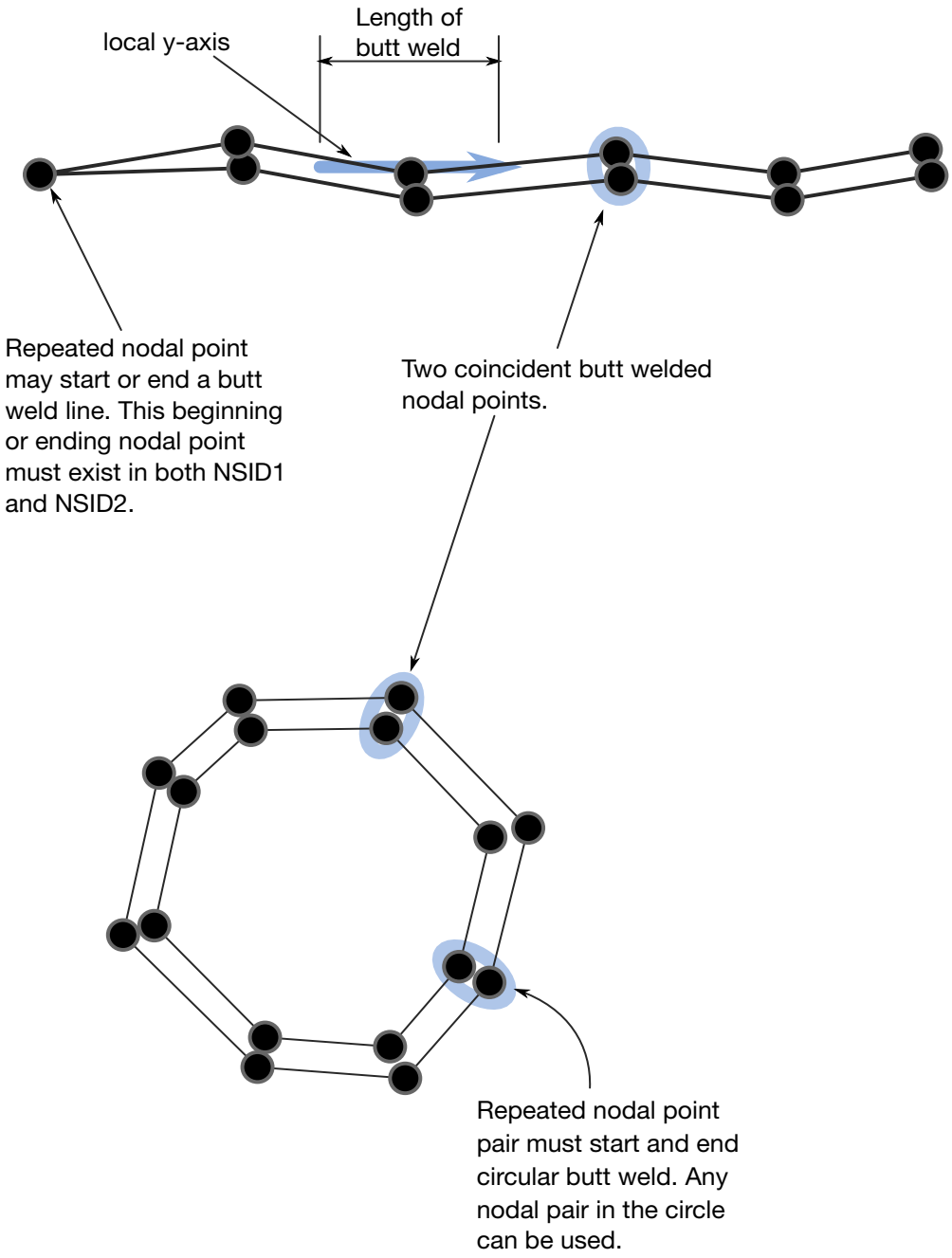


Figure 10-3. Definition of butt welds are shown above. The butt weld can be represented by a line of nodal points or by a closed loop

of the butt weld is circular or a closed loop. See Figure 10-3, where the line butt weld and closed loop weld are illustrated.

2. **Restrictions.** Butt welds may not cross. For complicated welds, this keyword can be combined with the input in *CONSTRAINED_GENERALIZED_WELD_-BUTT to handle the case where crossing occurs. Nodes in a butt weld must not be members of rigid bodies.

3. **Failure Model.** If the average volume-weighted effective plastic strain in the shell elements adjacent to a node exceeds the specified plastic strain at failure, the node is released. Brittle failure of the butt welds occurs when:

$$\beta\sqrt{\sigma_n^2 + 3(\tau_n^2 + \tau_t^2)} \geq \sigma_f$$

where,

σ_n = normal stress (local x)
τ_n = shear stress in direction of weld (local y)
τ_t = shear stress normal to weld (local z)
σ_f = failure stress
β = failure parameter

The component σ_n is nonzero for tensile values only. The nodes on each side of the butt weld must coincide.

The local coordinates at each set of coincident nodes are determined as follows. The normal vector (local x) is found by summing the unit normal vectors of all the shell elements on the *NSID2 side* sharing the butt welded node. The direction of the normal vector at the node is chosen so that the local x-direction points towards the elements on the NSID1 side in order to identify tensile versus compressive stresses. The local y-axis is taken as the vector in the direction of a line connecting the mid-points of the line segments lying on either side of the node in NSID2 (see Figure 10-3). The local z-axis at a node in NSID2 is normal to the plane of the butt weld at the node determined by looking at the *NSID2 side*.

The thickness and length of the butt weld are needed to compute the stress values. The thickness is based on the average thickness of the shell elements that share the butt welded nodal pair, and the chosen length of the butt weld is shown in Figure 10-3.

4. **Failure along a Line.** Butt welds may be used to simulate the effect of failure along a predetermined line, such as a seam or structural joint. When the failure criterion is reached at a nodal pair, the nodes begin to separate. As this effect propagates, the weld will appear to "unzip," thus simulating failure of the connection.

*CONSTRAINED_COORDINATE_{OPTION}

To define constraints based on position coordinates the following options are available:

<BLANK>

LOCAL

Purpose: Define constraints on position coordinates for springback simulations. With the frequent application of an adaptive mesh in stamping simulations, nodes needed for springback constraints are often unavailable until the last process simulation before springback is complete, making defining springback constraints at the beginning of the simulation process setup impossible. However, if the nodes are available, their positions may not be exactly at the desired locations required for springback constraints. With this keyword, it is possible to define constraints in both scenarios.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	PID	IDIR	X1	Y1	Z1	CID	
Type	I	I	I	F	F	F	I	
Default	none	none	none	0.0	0.0	0.0	0	

VARIABLE	DESCRIPTION
ID	Identification number of the constraint
PID	Part or part set to be constrained: GT.0: Part ID LT.0: |PID| is a part set ID.
IDIR	Constraint flag. IDIR > 0 works for implicit (not explicit dynamic) simulations only with no active adaptive refinement (adaptive blank is acceptable); the magnitude of IDIR defines the degrees of freedom below to be constrained at (X1,Y1,Z1). IDIR < 0 works for both implicit and explicit simulations with active adaptive refinement; the magnitude of IDIR determines the degrees of freedom below to be constrained at the node nearest to (X1,Y1,Z1). |IDIR|.EQ.1: x translational degree of freedom |IDIR|.EQ.2: y translational degree of freedom

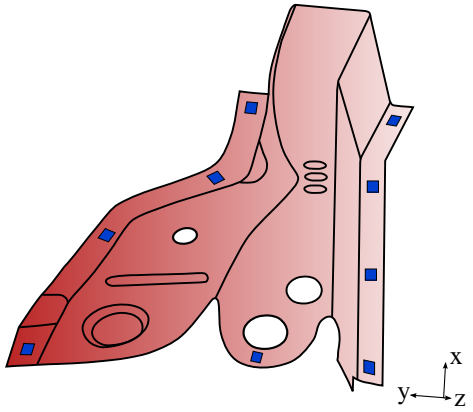


Figure 10-4. Constrained locations of a trim panel (NUMISHEET 2005 cross member).

VARIABLE	DESCRIPTION
	|IDIR|.EQ.3: z translational degree of freedom |IDIR|.EQ.11: x rotational degree of freedom |IDIR|.EQ.12: y rotational degree of freedom |IDIR|.EQ.13: z rotational degree of freedom |IDIR|.EQ.14: x and y rotational degrees of freedom |IDIR|.EQ.15: y and z rotational degrees of freedom |IDIR|.EQ.16: z and x rotational degrees of freedom |IDIR|.EQ.17: x, y and z rotational degrees of freedom
X1, Y1, Z1	X, Y, Z coordinates of the location being constrained
CID	Local coordinate system ID

Remarks:

1. **Constraint IDs.** The identification number of a constraint must be unique; the IDs must be unique even for two constraints with the same (X, Y, Z) position that constrains different degrees of freedom. When the LOCAL option is invoked, a local coordinate system ID (see *DEFINE_COORDINATE_*{OPTION}*) should be provided in the CID field.

2. **Moving parts.** The constraints defined by this keyword take effect after the deformable part is moved by *PART_MOVE. This is necessary, for example, in the case of a formed panel springback supported by a checking fixture, where the panel is automatically positioned by *CONTROL_FORMING_AUTOPOSITION_PARAMETER.

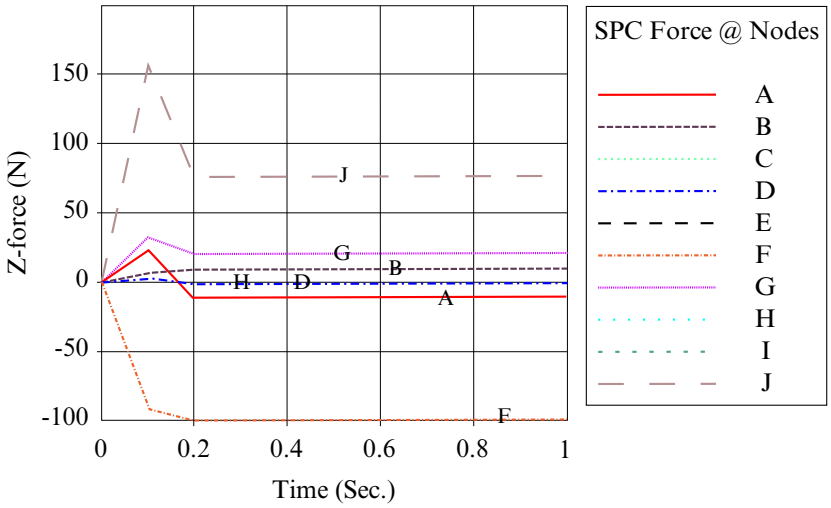


Figure 10-5. SPC Z-forces at 10 nodes

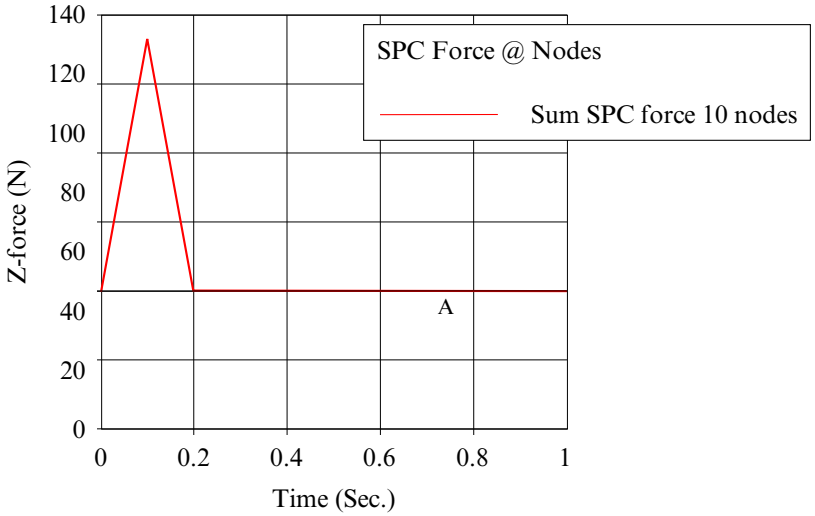


Figure 10-6. SPC Z-force summation of the 10 nodes

3. **SPC forces.** It is possible to output SPC forces on the coordinates constrained. For each position coordinate set, an extra node will be generated and SPC forces are calculated and output to the SPCFORC file. The frequency of the output is specified with the keyword *DATABASE_SPCFORC. For example, Figure 10-4 shows the Z-constrained locations on the trimmed panel (half with symmetric conditions at the smaller end) of the NUMISHEET 2005 cross member. SPC forces in the Z-direction of these 10 locations were recovered after a multi-step static implicit springback with this over-constrained boundary condition (see Figure 10-5). Figure 10-6 shows the summation of these Z-forces. The summation approaches zero as the residual stresses are balanced out by the springback shape absent of gravity.

Example:

A partial keyword input deck is shown below. A part with PID 18 is constrained in 6 locations in a local coordinate system with ID 9. Constrained DOFs are indicated by IDIR.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONSTRAINED_COORDINATE
$       ID      IDPT      IDIR         x         y         z       CID
         1        18         2  -555.128      86.6   1072.29         9
         2        18         3  -555.128      86.6   1072.29         9
         3        18         3  -580.334    -62.15   1068.32         9
         4        18         1   568.881   81.2945   1033.72         9
         5        18         2   568.881   81.2945   1033.72         9
         6        18         3   568.881   81.2945   1033.74         9
*DEFINE_COORDINATE_SYSTEM
$      CID        X0        Y0        Z0        XL        YL        ZL
         9       0.0       0.0       0.0       0.0      10.0       0.0
$       XP        YP        ZP
      10.0      10.0       0.0
```

*CONSTRAINED_EULER_IN_EULER

Purpose: Define the coupling interaction between Eulerian materials in two overlapping, geometrically similar, multi-material Eulerian mesh sets. The command allows a frictionless "contact" between two or more different Eulerian materials. To specify Eulerian meshes, use S-ALE or ALE. Note that the implementation of this feature for S-ALE differs from ALE.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID1	PSID2	PFAC					
Type	I	I	F					
Default	0	0	0.1					

VARIABLE	DESCRIPTION
PSID1	Part set ID of the 1st ALE or Eulerian set of mesh(es)
PSID2	Part set ID of the 2nd ALE or Eulerian set of mesh(es)
PFAC	A penalty factor for the coupling interaction between the two PSIDs

Remarks:

1. **Model description.** The two meshes must be of Eulerian formulation (the meshes are fixed in space, not moving). Consider two overlapping Eulerian meshes. Each Eulerian mesh contains two physical materials, say a vacuum and a metal. This keyword provides a frictionless "contact" or interaction between the two metals since each resides in a different Eulerian mesh system. Due to its restrictive nature, this option is currently only an experimental feature.

2. **Contact pressure.** Contact pressure is built up in two overlapping Eulerian elements if their combined material fill fraction exceeds 1.0 (penalty formulation).

3. **Materials.** This feature needs to be combined with *MAT_VACUUM (element formulation 11).

4. **S-ALE setup.** Two or more S-ALE meshes can be used; their overlapping domain(s) must coincide exactly. A straightforward way to ensure this alignment

is to use the same set of *ALE_STRUCTURED_MESH_CONTROL_POINTS across all S-ALE meshes. Each individual mesh can then be trimmed using *ALE_STRUCTURED_MESH_TRIM to reduce its computational domain.

5. **MPP support.** The ALE implementation of this feature does not support MPP, but the S-ALE one does.

Example:

Consider an ALE/Eulerian multi-material model (ELFORM = 11) consisting of:

PID 1 = *MAT_NULL (material 1)

PID 2 = *MAT_VACUUM ⇒ PID 1 is merged at its boundary to PID 2.

PID 3 = *MAT_NULL (material 3)

PID 4 = *MAT_VACUUM ⇒ PID 3 is merged at its boundary to PID 4.

The mesh set containing PID 1 and 2 intersects or overlaps with the mesh set containing PID 3 and 4. PID 1 is given an initial velocity in the positive x-direction. This will cause material 1 to contact material 3 (note that materials 2 and 4 are void). The interaction between materials 1 and 3 is possible by defining this coupling command. In this case material 1 can flow within the mesh region of PID 1 and 2 only, and material 3 can flow within the mesh region of PID 3 and 4 only.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_MULTI-MATERIAL_GROUP
$       SID   SIDYTPE
          1         1
          2         1
          3         1
          4         1
*CONSTRAINED_EULER_IN_EULER
$     PSID1     PSID2     PENAL
         11        12       0.1
*SET_PART_LIST
         11
          1         2
*SET_PART_LIST
         12
          3         4
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

***CONSTRAINED_EXTRA_NODES**_*OPTION*

Available options include:

NODE

SET

Purpose: Define extra nodes for rigid body.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NID/NSID	IFLAG					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
PID	Part ID of rigid body to which the nodes will be added, see *PART.
NID / NSID	Node (keyword option: NODE) or node set ID (keyword option: SET), see *SET_NODE, of added nodes.
IFLAG	This flag is meaningful if and only if the inertia properties of the Part ID are defined in PART_INERTIA. If set to unity, the center-of-gravity, the translational mass, and the inertia matrix of the PID will be updated to reflect the merged nodal masses of the node or node set. If IFLAG is defaulted to zero, the merged nodes will not affect the properties defined in PART_INERTIA since it is assumed the properties already account for merged nodes.

Remarks:

Extra nodes for rigid bodies may be placed anywhere, even outside the body, and they are assumed to be part of the rigid body. They have many uses including:

1. The definition of draw beads in metal forming applications by listing nodes along the draw bead.

2. Placing nodes where joints will be attached between rigid bodies.

3. Defining a node where point loads are to be applied or where springs may be attached.

4. Defining a lumped mass at a particular location.

The coordinates of the extra nodes are updated according to the rigid body motion.

Examples:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_EXTRA_NODES_NODE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Rigidly attach nodes 285 and 4576 to part 14. (Part 14 MUST be a rigid body.)
$
*CONSTRAINED_EXTRA_NODES_NODE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid       nid
        14       285
        14      4576
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$


$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_EXTRA_NODES_SET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Rigidly attach all nodes in set 4 to part 17. (Part 17 MUST be a rigid body.)
$
$  In this example, four nodes from a deformable body are attached
$  to rigid body 17 as a means of joining the two parts.
$
*CONSTRAINED_EXTRA_NODES_SET
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid      nsid
        17         4
$
$
*SET_NODE_LIST
$      sid
         4
$     nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
       665       778       896       827
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

CONSTRAINED_GENERALIZED_WELD_*WELDTYPE_{OPTION}

CONSTRAINED_GENERALIZED_WELD_*WELDTYPE is a family of keywords all sharing a common set of data cards and option flags. The available weld variants are:

SPOT

FILLET

BUTT

CROSS_FILLET

COMBINED

To define an ID for the weld use the option:

ID

Purpose: Define spot, fillet, butt, and other types of welds. Coincident nodes are permitted if the local coordinate ID is defined. For the spot weld a local coordinate ID is not required if the nodes are offset. Failures can include both the plastic and brittle failures. These can be used either independently or together. Failure occurs when either criteria is met. The welds may undergo large rotations since the equations of rigid body mechanics are used to update their motion. Weld constraints between solid element nodes are not supported.

Card Summary:

Card ID. Include this card if the ID keyword option is used.

WID							

Card 1. This card is required.

NSID	CID	FILTER	WINDOW	NPR	NPRT		

Card 2a. Include this card if the weld type is SPOT.

TFAIL	EPSF	SN	SS	N	M		

Card 2b. Incldue this card if the weld type is FILLET.

TFAIL	EPSF	SIGF	BETA	L	W	A	ALPHA

Card 2c. Include this card if the weld type is BUTT.

TFAIL	EPSF	SIGY	BETA	L	D		

Card 2d.1. Include this card if the weld type is CROSS_FILLET.

TFAIL	EPSF	SIGY	BETA	L	W	A	ALPHA

Card 2d.2. Include this card if the weld type is CROSS_FILLET. Include NPR of this card.

NODEA	NODEB	NCID					

Card 2e.1. Include this card if the weld type is COMBINED. Read in NPR pairs of Cards 2e.1 and 2e.2 for a total of 2 × NPR cards.

TFAIL	EPSF	SIGY	BETA	L	W	A	ALPHA

Card 2e.2. Include this card if the weld type is COMBINED. Read in NPR pairs of Cards 2e.1 and 2e.2 for a total of 2 × NPR cards.

NODEA	NODEB	NCID	WTYP				

Data Card Definitions:

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	WID							
Type	I							

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	CID	FILTER	WINDOW	NPR	NPRT		
Type	I	I	I	I	I	I		
Default	none	0	0	0	↓	0		

VARIABLE	DESCRIPTION
WID	Optional weld ID
NSID	Nodal set ID (see *SET_NODE)
CID	Coordinate system ID for output of spot weld data to SWFORC in local system; see *DEFINE_COORDINATE_*OPTION*. CID is not required for spot welds if the nodes are not coincident.
FILTER	Number of force vectors saved for filtering. This option can eliminate spurious failures due to numerical force spikes; however, memory requirements are significant since 6 force components are stored with each vector. LE.1: No filtering GE.2: Simple average of force components divided by FILTER or the maximum number of force vectors that are stored for the time window option below.
WINDOW	Time window for filtering. This option requires the specification of the maximum number of steps (defined by FILTER) which can occur within the filtering time window. A time weighted average is used. If the time step decreases too far, then the filtering time window will be ignored and the simple average is used. EQ.0: Time window is not used.
NPR	Number of individual nodal pairs in the cross fillet or combined general weld.
NPRT	Print option in file rbdout. EQ.0: Default from the control card, *CONTROL_OUTPUT, is used; see variable name IPRTF. EQ.1: Data is printed. EQ.2: Data is not printed.

Spot Weld Card. Additional card required for the SPOT keyword option.

Card 2a	1	2	3	4	5	6	7	8
Variable	TFAIL	EPSF	SN	SS	N	M		
Type	F	F	F	F	F	F		
Default	10^{16}	10^{16}	10^{16}	10^{16}	2.0	2.0		

VARIABLE	DESCRIPTION
TFAIL	Failure time for constraint set, t_f
EPSF	Effective plastic strain at failure, ε_{fail}^{p}, for ductile failure
SN	S_n, normal force at failure; only for the brittle failure of spot welds
SS	S_s, shear force at failure; only for the brittle failure of spot welds
N	n, exponent for normal force; only for the brittle failure of spot welds
M	m, exponent for shear force; only for the brittle failure of spot welds

Remarks:

Spot weld failure due to plastic straining occurs when the effective nodal plastic strain exceeds the input value, ε_{fail}^{p}. This option can model the tearing out of a spot weld from the sheet metal since the plasticity is in the material that surrounds the spot weld, not the spot weld itself. A least squares algorithm is used to generate the nodal values of plastic strains at the nodes from the element integration point values. The plastic strain is integrated through the element and the average value is projected to the nodes using a least square fit. This option should only be used for the material models related to metallic plasticity and can result in slightly increased run times.

Brittle failure of the spot welds occurs when:

$$\left[\frac{\max(f_n, 0)}{S_n}\right]^n + \left[\frac{|f_s|}{S_s}\right]^m \geq 1 \ ,$$

where f_n and f_s are the normal and shear interface forces. Component f_n contributes for tensile values only. When the failure time, t_f, is reached the nodal rigid body becomes inactive, and the constrained nodes may move freely. In Figure 10-7 the ordering of the

nodes is shown for two node and three node spot welds. This order is with respect to the local coordinate system where the local z-axis determines the tensile direction. The nodes in the spot weld may coincide. The failure of a three node spot weld may occur gradually with first one node failing and later the second node may fail. For n noded spot welds the failure is progressive starting with the outer nodes (1 and n) and then moving inward to nodes 2 and $n-1$. Progressive failure is necessary to preclude failures that would create new rigid bodies.

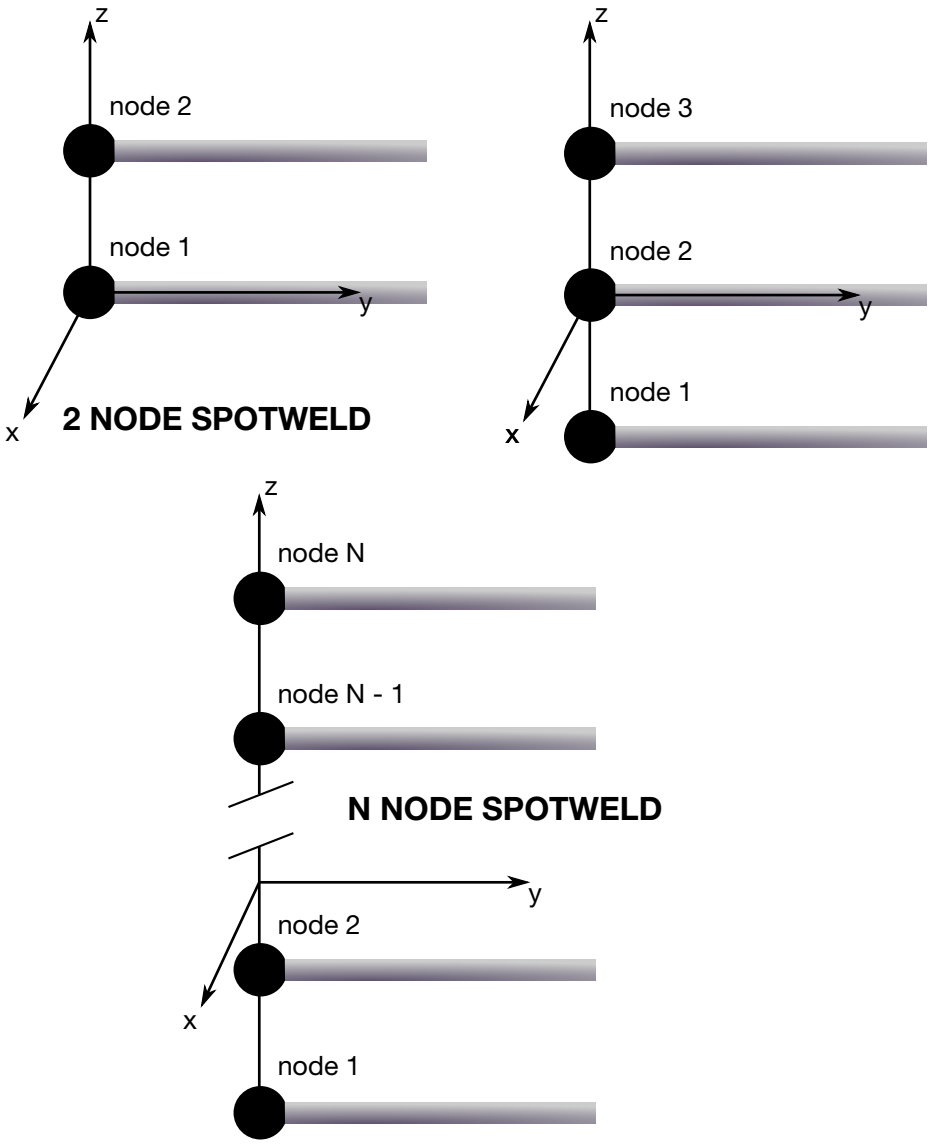


Figure 10-7. Nodal ordering and orientation of the local coordinate system is important for determining spotweld failure.

Fillet Weld Card. Additional card required for the FILLET keyword option.

Card 2b	1	2	3	4	5	6	7	8
Variable	TFAIL	EPSF	SIGF	BETA	L	W	A	ALPHA
Type	F	F	F	F	F	F	F	F
Default	10^{16}	10^{16}	10^{16}	1.0	none	none	none	none

VARIABLE	DESCRIPTION
TFAIL	Failure time for constraint set, t_f
EPSF	Effective plastic strain at failure, ε^p_{fail}, for ductile failure
SIGF	σ_f, stress at failure for brittle failure
BETA	β, failure parameter for brittle failure
L	L, length of fillet weld (see Figure 10-8)
W	w, separation of parallel fillet welds (see Figure 10-8)
A	a, fillet weld throat dimension (see Figure 10-8)
ALPHA	α, weld angle (see Figure 10-8) in degrees

Remarks:

Ductile fillet weld failure, due to plastic straining, is treated identically to spot weld failure. Brittle failure occurs when the following weld stress condition is met on the narrowest fillet weld cross section (across the throat):

$$\beta\sqrt{\sigma_n^2 + 3(\tau_n^2 + \tau_t^2)} \geq \sigma_f \ ,$$

where

σ_n = normal stress
τ_n = shear stress in the local zx-plane
τ_t = shear stress in the local y-direction
σ_f = failure stress
β = failure parameter

The component σ_n is nonzero for tensile values only. When the failure time, t_f, is reached, the nodal rigid body becomes inactive, and the constrained nodes may move freely.

Figure 10-8 shows the ordering of the nodes for two node and three node fillet welds. This order is with respect to the local coordinate system where the local z-axis determines the tensile direction. The initial orientation of the local coordinate system is defined by CID. If CID = 0, then the global coordinate system is used. The local coordinate system is updated according to the rotation of the rigid body representing the weld. The failure of a three node fillet weld may occur gradually with first one node failing and later the second node may fail.

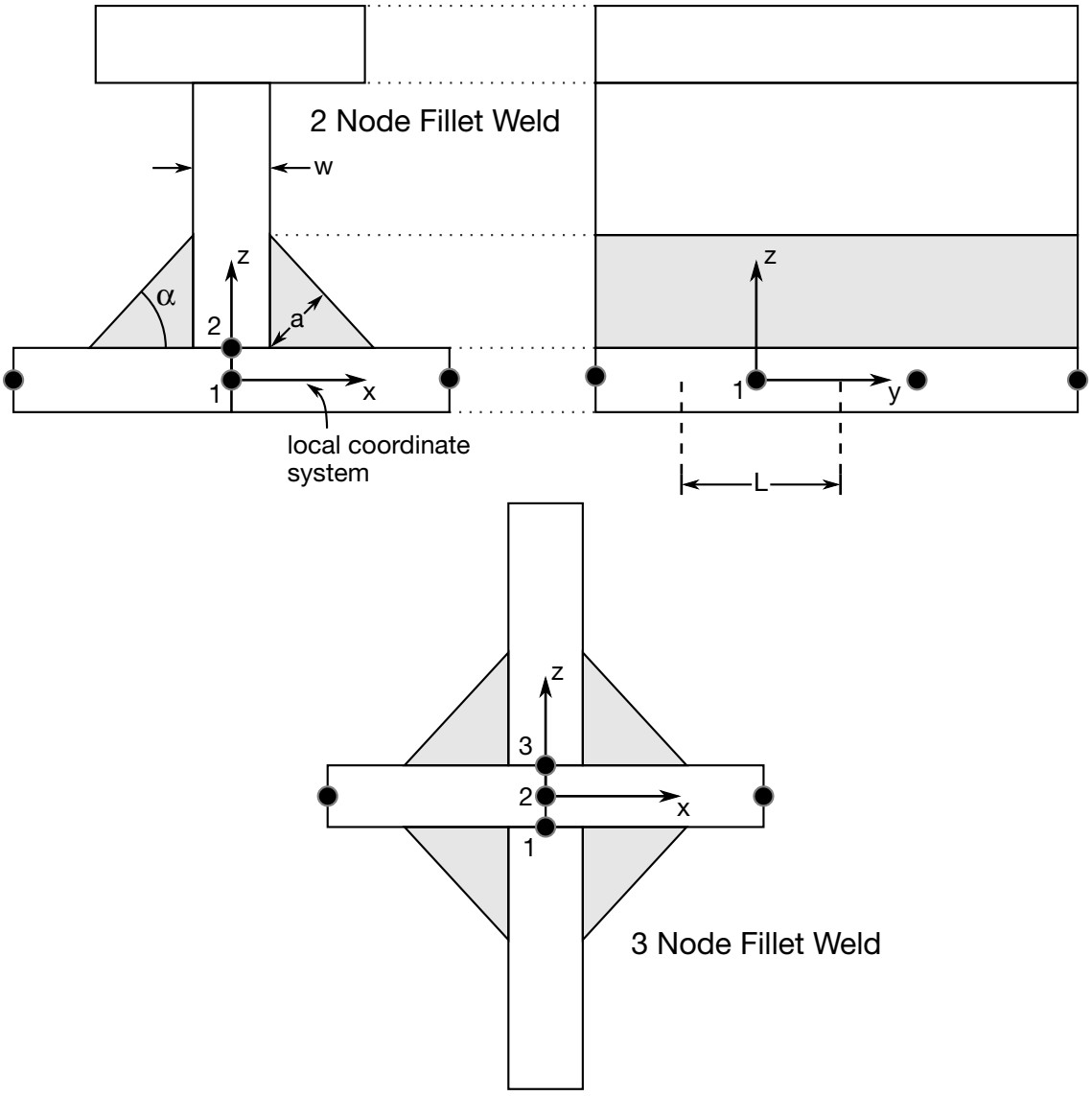


Figure 10-8. Nodal ordering and orientation of the local coordinate system is shown for fillet weld failure. The angle is defined in degrees.

Butt Weld Card. Additional card required for the BUTT keyword option.

Card 2c	1	2	3	4	5	6	7	8
Variable	TFAIL	EPSF	SIGY	BETA	L	D		
Type	F	F	F	F	F	F		
Default	10^{16}	10^{16}	10^{16}	1.0	none	none		

VARIABLE	DESCRIPTION
TFAIL	Failure time for constraint set, t_f
EPSF	Effective plastic strain at failure, ε^p_{fail}, for ductile failure
SIGY	σ_f, stress at failure for brittle failure
BETA	β, failure parameter for brittle failure
L	L, length of butt weld (see Figure 10-9)
D	d, thickness of butt weld (see Figure 10-9)

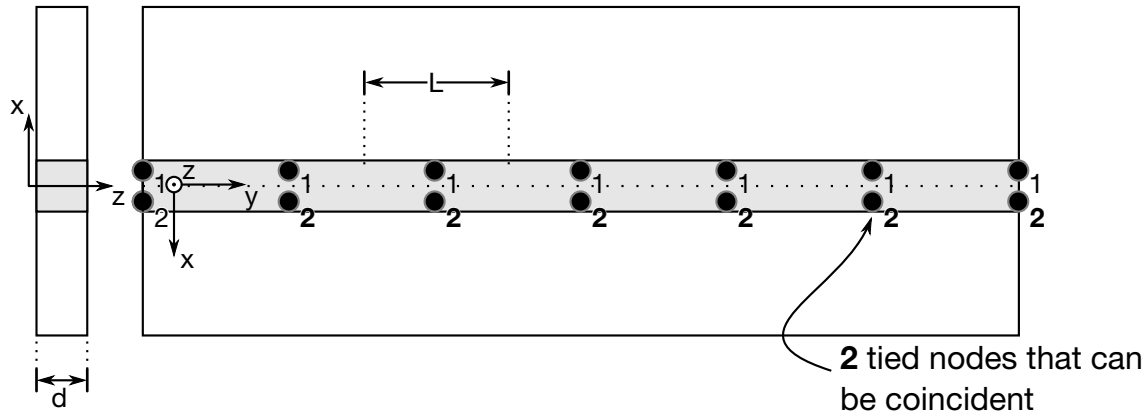


Figure 10-9. Orientation of the local coordinate system and nodal ordering is shown for butt weld failure.

Remarks:

Ductile butt weld failure, due to plastic straining, is treated identically to spot weld failure. Brittle failure of the butt welds occurs when:

$$\beta\sqrt{\sigma_n^2 + 3(\tau_n^2 + \tau_t^2)} \geq \sigma_f \ ,$$

where

σ_n = normal stress
τ_n = shear stress in the direction of the weld (local y)
τ_t = shear stress normal to the weld (local z)
σ_f = failure stress
β = failure parameter

The component σ_n is nonzero for tensile values only. When the failure time, t_f, is reached, the nodal rigid body becomes inactive, and the constrained nodes may move freely. The nodes in the butt weld may coincide.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_GENERALIZED_WELD_BUTT
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Weld two plates that butt up against each other at three nodal pair
$  locations.  The nodal pairs are 32-33, 34-35 and 36-37.
$
$  This requires 3 separate *CONSTRAINED_GENERALIZED_WELD_BUTT definitions,
$  one for each nodal pair.  Each weld is to have a length (L) = 10,
$  thickness (D) = 2, and a transverse length (Lt) = 1.
$
$  Failure is defined two ways:
$    Ductile failure if effective plastic strain exceeds 0.3
$    Brittle failure if the stress failure criteria exceeds 0.25
$       - scale the brittle failure criteria by beta = 0.9.
$    Note: beta > 1 weakens weld          beta < 1 strengthens weld
$
*CONSTRAINED_GENERALIZED_WELD_BUTT
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid       cid
        21
$    tfail      epsf      sigy      beta         L         D        Lt
                0.3     0.250       0.9      10.0       2.0       1.0
$
$
*CONSTRAINED_GENERALIZED_WELD_BUTT
$     nsid       cid
        23
$    tfail      epsf      sigy      beta         L         D        Lt
                0.3     0.250       0.9      10.0       2.0       1.0
$
$
*CONSTRAINED_GENERALIZED_WELD_BUTT
$     nsid       cid
        25
$    tfail      epsf      sigy      beta         L         D        Lt
                0.3     0.250       0.9      10.0       2.0       1.0
$
$
*SET_NODE_LIST
$      sid
```

```
         21
$      nid1      nid2
         32        33
*SET_NODE_LIST
         23
         34        35
*SET_NODE_LIST
         25
         36        37
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Cross Fillet Weld Card. Additional card for the CROSS_FILLET keyword option.

Card 2d.1	1	2	3	4	5	6	7	8
Variable	TFAIL	EPSF	SIGY	BETA	L	W	A	ALPHA
Type	F	F	F	F	F	F	F	F
Default	10^{16}	10^{16}	10^{16}	1.0	none	none	none	none

Nodal Pair Cards. Read NPR additional cards for the CROSS_FILLET keyword option.

Card 2d.2	1	2	3	4	5	6	7	8
Variable	NODEA	NODEB	NCID					
Type	I	I	I					

VARIABLE	DESCRIPTION
TFAIL	Failure time for constraint set, t_f
EPSF	Effective plastic strain at failure, ε^p_{fail}, for ductile failure
SIGY	σ_f, stress at failure for brittle failure
BETA	β, failure parameter for brittle failure
L	L, length of fillet weld (see Figure 10-8)
W	w, separation of parallel fillet welds (see Figure 10-8)
A	a, throat dimension of fillet weld (see Figure 10-8)
ALPHA	α, weld angle (see Figure 10-8) in degrees
NODEA	Node ID, A, in the weld pair. See Figure 10-10.
NODEB	Node ID, B, in the weld pair
NCID	Local coordinate system ID

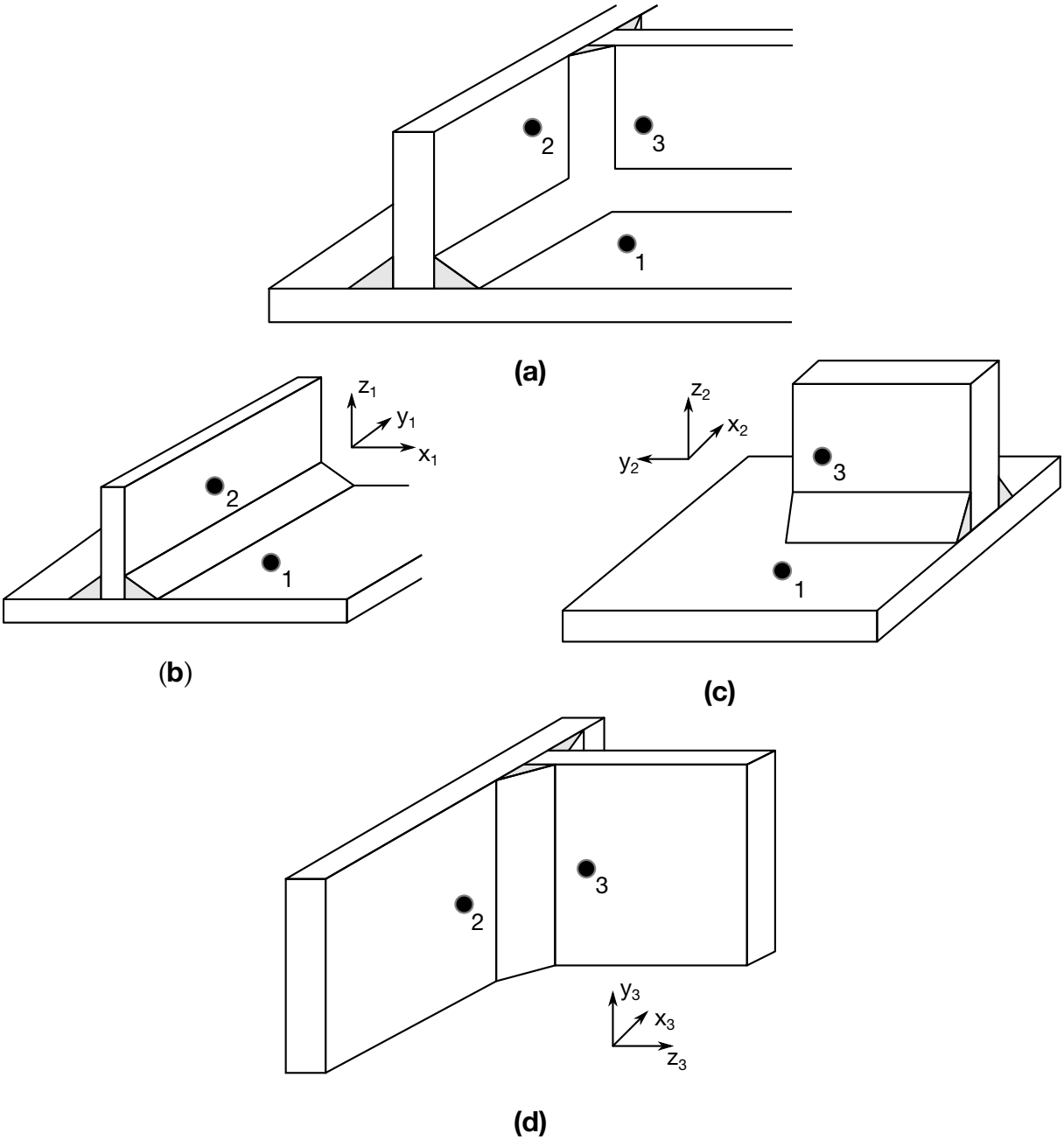


Figure 10-10. A simple cross fillet weld illustrates the required input. Here NPR = 3 with nodal pairs (A = 2, B = 1), (A = 3, B = 1), and (A = 3, B = 2). The local coordinate axes are shown. These axes are fixed in the rigid body and are referenced to the local rigid body coordinate system which tracks the rigid body rotation.

Combined Weld Cards:
Additional cards for the COMBINED keyword option. Read in NPR pairs of Cards 2e.1 and 2e.2 for a total of 2 × NPR cards.

Card 2e.1	1	2	3	4	5	6	7	8
Variable	TFAIL	EPSF	SIGY	BETA	L	W	A	ALPHA
Type	F	F	F	F	F	F	F	F
Default	10^{16}	10^{16}	10^{16}	1.0	none	none	none	none

Card 2e.2	1	2	3	4	5	6	7	8
Variable	NODEA	NODEB	NCID	WTYP				
Type	I	I	I	I				

VARIABLE	DESCRIPTION
TFAIL	Failure time, t_f, for the constraint set
EPSF	Effective plastic strain at failure, ε^p_{fail}, for ductile failure
SIGY	σ_f, stress at failure for brittle failure
BETA	β, failure parameter for brittle failure
L	L, length of fillet/butt weld (see Figure 10-8 and 10-9)
W	w, width of flange (see Figure 10-8)
A	a, width of fillet weld (see Figure 10-8)
ALPHA	α, weld angle (see Figure 10-8) in degrees
NODEA	Node ID, A, in the weld pair
NODEB	Node ID, B, in the weld pair
NCID	Local coordinate system ID

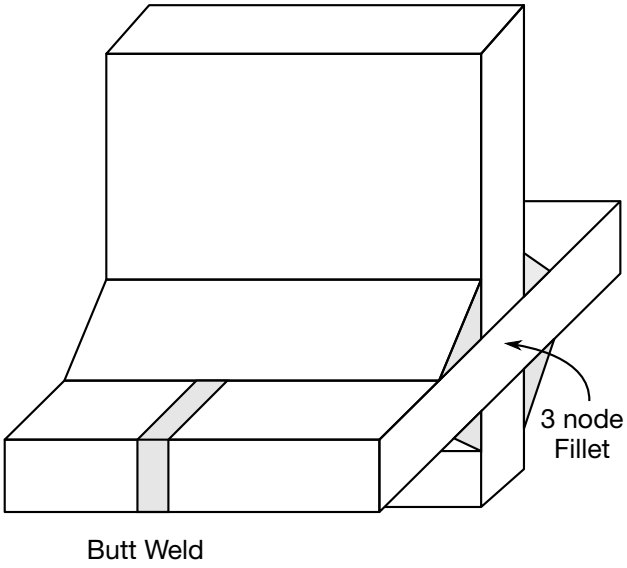


Figure 10-11. A combined weld is a mixture of fillet and butt welds.

VARIABLE	DESCRIPTION
WTYPE	Weld pair type (see Figure 10-11): EQ.0: Fillet weld EQ.1: Butt weld

*CONSTRAINED_GLOBAL

Purpose: Define a global boundary constraint plane. This card implements the trivial constraint equation; namely, constrained degrees of freedom are held fixed in time.

Card 1	1	2	3	4	5	6	7	8
Variable	TC	RC	DIR	X	Y	Z	TOL	
Type	I	I	I	F	F	F	F	
Default	0	0	none	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
TC	Translational Constraint: EQ.0: no translational constraints EQ.1: constrained x translation EQ.2: constrained y translation EQ.3: constrained z translation EQ.4: constrained x and y translations EQ.5: constrained y and z translations EQ.6: constrained x and z translations EQ.7: constrained x, y, and z translations
RC	Rotational Constraint: EQ.0: no rotational constraints EQ.1: constrained x-rotation EQ.2: constrained y-rotation EQ.3: constrained z-rotation EQ.4: constrained x and y rotations EQ.5: constrained y and z rotations EQ.6: constrained z and x rotations EQ.7: constrained x, y, and z rotations

VARIABLE	DESCRIPTION
DIR	Direction of normal for constraint plane: EQ.1: global x EQ.2: global y EQ.3: global z
X	Global x-coordinate of a point on the constraint plane
Y	Global y-coordinate of a point on the constraint plane
Z	Global z-coordinate of a point on the constraint plane
TOL	User-defined tolerance in length units. If non-zero, the internal mesh-size dependent tolerance gets replaced by this value.

Remarks:

Nodes within a mesh-size-dependent tolerance are constrained on a global plane. This option is recommended for use with *r*-method adaptive remeshing where nodal constraints are lost during the remeshing phase. See *CONSTRAINED_LOCAL for specifying constraints to nodes lying on a local plane.

*CONSTRAINED_IMMERSED_IN_SPG

Purpose: Define coupling of beams and SPG solid elements through immerse method.

Card 1	1	2	3	4	5	6	7	8
Variable	SPGPID							
Type	I							
Default	none							

FEM Beams Card. Include this card as many times as needed. Input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	IPID1	IPID2	IPID3	IPID4	IPID5	IPID6	IPID7	IPID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
SPGPID	Part ID of SPG solids into which where FEM beams are immersed
IPID*i*	Part IDs of FEM beams

*CONSTRAINED_INTERPOLATION_{OPTION}

Available options include:

<BLANK>

LOCAL

Purpose: Define an interpolation constraint. With this constraint type, the motion of a single dependent node is interpolated from the motion of a set of independent nodes.

This option is useful for the redistribution of a load applied to the dependent node by the surrounding independent nodes. This load may be a translational force or a rotational moment. This keyword is typically used to model shell-brick and beam-brick interfaces.

The mass and rotary inertia of the dependent nodal point is also redistributed. This constraint is applied in the global coordinate system unless the option LOCAL is active. *One *CONSTRAINED_INTERPOLATION card is required for each constraint definition.* The input list of independent nodes is terminated when the next keyword ("*") card is found.

In explicit calculations, the independent and dependent nodes cannot be dependent nodes in other constraints such as nodal rigid bodies; however, implicit calculations are not bound by this limitation. See Remark 2.

Card 1	1	2	3	4	5	6	7	8
Variable	ICID	DNID	DDOF	CIDD	ITYP	IDNSW	FGM	
Type	I	I	I	I	I	I	I	
Default	none	none	123456	global	0	1	0	

Independent Node Card Sets:

If LOCAL option is not set, for each independent node include the following card; if the LOCAL keyword option is set, include the following *pair* of cards. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	INID	IDOF	TWGHTX	TWGHTY	TWGHTZ	RWGHTX	RWGHTY	RWGHTZ
Type	I	I	F	F	F	F	F	F
Default	0	123456	1.0	TWGHTX	TWGHTX	TWGHTX	TWGHTX	TWGHTX

Local Coordinate Card. Additional card for the LOCAL keyword option to be paired with Card 2.

Card 3	1	2	3	4	5	6	7	8
Variable	CIDI							
Type	I							
Default	global							

VARIABLE	DESCRIPTION
ICID	Interpolation constraint ID.
DNID	Dependent node ID. This node should not be a member of a rigid body, or elsewhere constrained in the input.
DDOF	Dependent degrees-of-freedom. The list of dependent degrees-of-freedom consists of a number with up to six digits, with each digit representing a degree of freedom. For example, the value 1356 indicates that degrees of freedom 1, 3, 5, and 6 are controlled by the constraint. The default is 123456. DDOF are in the global coordinate system regardless of whether the LOCAL option is used or not. Digit degree of freedom ID's: EQ.1: x

VARIABLE	DESCRIPTION
	EQ.2: y
	EQ.3: z
	EQ.4: Rotation about x axis
	EQ.5: Rotation about y axis
	EQ.6: Rotation about z axis
CIDD	Local coordinate system ID if LOCAL option is active. If blank, the global coordinate system is assumed.
ITYP	Specifies the meaning of INID.
	EQ.0: INID is a node ID
	EQ.1: INID is a node set ID
INDSW	Switch for controlling the explicit solution when an independent (or dependent) node is deleted.
	EQ.0: Default to option 1.
	EQ.1: Terminate the explicit analysis when an independent node or the dependent node is deleted.
	EQ.2: Continue the explicit analysis with the constraints unchanged.
FGM	Flag for special treatment of this constraint for implicit problems only:
	EQ.0: Use standard constraint processing for implicit.
	EQ.1: Use special processing for this constraint for implicit only; see Remarks.
INID	Independent node ID or node set ID
IDOF	Independent degrees-of-freedom using the same form as for the dependent degrees-of-freedom, DDOF, above.
TWGHTX	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the x-translational component. It is normally sufficient to define only TWGHTX even if its degree-of-freedom is inactive since the other factors are set equal to this input value as the default. There is no requirement on the values that are chosen as the weighting factors, that is, that they sum to unity. The default value for the weighting factor is unity.

VARIABLE	DESCRIPTION
TWGHTY	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the y-translational component.
TWGHTZ	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the z-translational component.
RWGHTX	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the x-rotational component.
RWGHTY	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the y-rotational component.
RWGHTZ	Weighting factor for node INID with active degrees-of-freedom IDOF. This weight scales the z-rotational component.
CIDI	Local coordinate system ID if LOCAL option is active. If blank the global coordinate system is assumed.

Remarks:

1. **Dependent node not attached to the finite element model.** This constraint is sometimes used with a dependent node that is not attached to the finite element model. If point loads or discrete masses are put on this dependent node, then this constraint transfers the point load or the mass to the set of independent nodes. In another case, a dependent node can track a particular point in the geometry through this constraint as the model deforms and rotates, such as a point on the axis of rotation of a turbine. For these cases, where the dependent node is not attached to the finite element model but is an artificial node used to distribute forces and/or masses or to track a geometric position, we recommend setting FGM to 1.

2. **Limitations in enforcing constraints with the explicit solver.** As mentioned previously, dependent and independent nodes in this constraint cannot be dependent nodes for other constraints when using the explicit solver. LS-DYNA sequentially enforces constraints in explicit. Thus, there is no guarantee that LS-DYNA will evaluate the constraints in the desired order. For instance, suppose we have the following two constraints:

$$\text{constraint 1:} \quad x_1 = x_2$$
$$\text{constraint 2:} \quad x_2 = x_3$$

Then, at time step $n+1$, $x_1^{n+1} = x_2^n$ and $x_2^{n+1} = x_3^{n+1}$ because of the order of evaluation. Thus, we get a result that doesn't match the desired result of $x_1^{n+1} = x_2^{n+1} = x_3^{n+1}$.

Note that between different types of constraints (such as a rigid body constraint and a *CONSTRAINED_INTERPOLATION), the last evaluated constraint overwrites the previous results. Thus, one constraint may have no influence on the results and be effectively ignored.

Example 1:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_INTERPOLATION  (Beam to solid coupling)
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Tie a beam element to a solid element.
$
$  The node of the beam to be tied does not share a common node with the solids.
$  If the beam node is shared, for example, then set ddof=456.
$
*CONSTRAINED_INTERPOLATION
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     icid      dnid      ddof
         1        45    123456
$     inid      idof    twghtx    twghty    twghtz    rwghtx    rwghty    rwghtz
        22       123
        44       123
        43       123
$
*.........
$
```

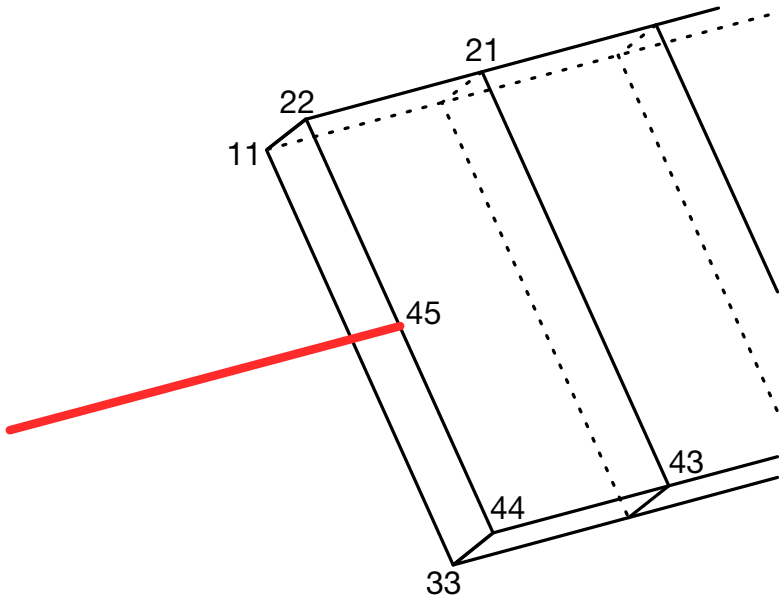


Figure 10-12. Illustration of Example 1.

Example 2:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_INTERPOLATION  (Load redistribution)
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Moment about normal axis of node 100 is converted to an equivalent load by
$  applying x-force resultants to the nodes lying along the right boundary
$
*DEFINE_CURVE
1,0,0.,0.,0.,0.,0
0.,0.
.1,10000.
*LOAD_NODE_POINT
100,6,1,1.0
$
*CONSTRAINED_INTERPOLATION
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     icid      dnid      ddof
         1       100         5
$     inid      idof    twghtx    twghty    twghtz    rwghtx    rwghty    rwghtz
        96       1
        97       1
        98       1
        99       1
       177       1
       178       1
       179       1
       180       1
$
*.........
$
```

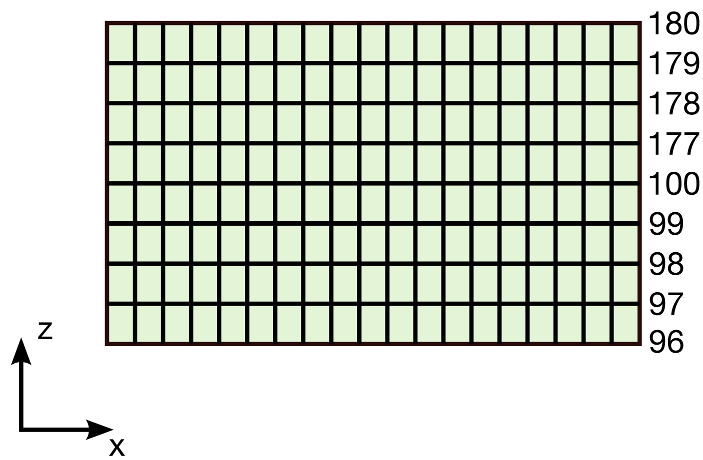


Figure 10-13. Illustration of Example 2.

*CONSTRAINED_INTERPOLATION_SPOTWELD_{OPTION}

(prior notation *CONSTRAINED_SPR3 still works)

Available options include:

<BLANK>

TITLE

Purpose: Define a spot weld with failure. This feature uses a plasticity-damage model that reduces the force and moment resultants to zero as the spot weld fails. A single node specifies the location of the spot weld at the center of two connected sheets. A radius approximately equal to the spot weld's radius gives the domain of influence. LS-DYNA performs a normal projection from the two sheets to the spot weld node and locates all nodes within the domain of influence. The numerical implementation of this feature is similar to the SPR2 model (*CONSTRAINED_SPR2).

Starting with R13.1, this approach can connect not only shell element parts but also solid elements. However, the new capability is still experimental and should be treated with care. For instance, it is advantageous if the solid element parts still have some kind of obvious planar structure.

Card Summary:

Card Title. Include this card if the TITLE keyword option is used.

TITLE

Card 1. This card is required.

PID1	PID2	NSID	THICK	R	STIFF	ALPHA1	MODEL

Card 2. This card is required.

RN	RS	BETA1	LCF	LCUPF	LCUPR	DENS	INTP

Card 3. Include this card when MODEL = 2, 12, or 22.

UPFN	UPFS	ALPHA2	BETA2	UPRN	UPRS	ALPHA3	BETA3

Card 4. Include this card when MODEL = 2, 12, or 22.

MRN	MRS						

Card 5. This card is read only when MODEL = 1, 3, 11, 21, 31, or 41. It is optional.

STIFF2	STIFF3	STIFF4	LCDEXP	GAMMA	SROPT	PIDVB	

Card 6. This card is read only when MODEL = 1, 3, 11, 21, 31, or 41. It is optional.

SCARN	SCARS	DAMP	SPROFF		FAILBM		

Card 7. This card is read only when MODEL = 1, 11, 21, 31, or 41. It is optional.

FFN	FFB	FFS	EXFC	STIFP	MFSFC	DEFC	NPFC

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

VARIABLE	DESCRIPTION
TITLE	Name or description of the SPR3 defined in this keyword. This name will be used as the part title for the visualization beams, such as in the d3plot database. If undefined, the part title is "SPR3_-NSID_...".

Card 1	1	2	3	4	5	6	7	8
Variable	PID1	PID2	NSID	THICK	R	STIFF	ALPHA1	MODEL
Type	I	I	I	F	F	F	F	F
Default	none	none	none	none	none	none	none	1.0

VARIABLE	DESCRIPTION
PID1	Part ID or part set ID of the first sheet:

VARIABLE	DESCRIPTION
	GT.0: Part ID LT.0: |PID1| is part set ID (for in-plane composed sheets, such as tailored blanks).
PID2	Part ID or part set ID of the second sheet. PID2 can be identical to PID1 if the spot weld location nodes from NSID lie in between the shell elements that should be self-connected. GT.0: Part ID LT.0: |PID2| is part set ID (for in-plane composed sheets, such as tailored blanks).
NSID	Node set ID of spot weld location nodes
THICK	Total thickness of both sheets
R	Spot weld radius
STIFF	Elastic stiffness. Function ID if MODEL > 10 (see Remark 2). GT.0: Constant value LT.0: |STIFF| refers to separate material data in *MAT_CONSTRAINED_SPR3 (*MAT_265).
ALPHA1	Scaling factor α_1. Function ID if MODEL > 10 (see Remark 2).
MODEL	Material behavior and damage model (see Remarks 3, 4, and 5). EQ.1: SPR3 (default) EQ.2: SPR4 EQ.3: SPR3 with strongly objective formulation EQ.11: Same as 1 with selected material parameters as functions EQ.12: Same as 2 with selected material parameters as functions EQ.21: Same as 11 with slight modification EQ.22: Same as 12 with slight modification EQ.31: Same as 11 but with 12 more material parameters as functions EQ.41: Same as 31 with slight modification

Card 2	1	2	3	4	5	6	7	8
Variable	RN	RS	BETA1	LCF	LCUPF	LCUPR	DENS	INTP
Type	F	F	F	I	I	I	F	F
Default	none	none	none	none	↓	↓	none	0

VARIABLE	DESCRIPTION
RN	Tensile strength factor, R_n. GT.0.0: Constant value unless MODEL > 10. Function ID if MODEL > 10 (see Remark 2). LT.0.0: Load curve with ID |RN| giving R_n as a function of peel ratio (see Remark 5)
RS	Shear strength factor, R_s. Function ID if MODEL > 10 (see Remark 2).
BETA1	Exponent for plastic potential β_1. Function ID if MODEL > 10 (see Remark 2).
LCF	Load curve or table ID. Load curve ID describing force as a function of plastic displacement, that is, $F^0(\bar{u}^{\text{pl}})$. Table ID describing force as a function of mode mixity (table values) and plastic displacement (curves), that is, $F^0(\bar{u}^{\text{pl}}, \kappa)$.
LCUPF	Load curve ID describing plastic initiation displacement as a function of mode mixity, that is, $\bar{u}_0^{\text{pl}}(\kappa)$. Only for MODEL = 1, 11, or 21. For MODEL = 1, LCUPF can also be a table ID giving plastic initiation displacement as a function of peel ratio (table values) and mode mixity (curves). See Remark 5.
LCUPR	Load curve ID describing plastic rupture displacement as a function of mode mixity, that is, $\bar{u}_f^{\text{pl}}(\kappa)$. Only for MODEL = 1, 11, or 21. For MODEL = 1, LCUPF can also be a table ID giving plastic initiation displacement as a function of peel ratio (table values) and mode mixity (curves). See Remark 5.
DENS	Spot weld density (necessary for time step calculation).
INTP	Flag for interpolation. INTP is interpreted digit-wise, namely as,

VARIABLE	DESCRIPTION
	$\text{INTP} = [LK] = K + 10 \times L$
	The first digit, K, controls the interpolation method:
	K.EQ.0: Linear (default)
	K.EQ.1: Uniform (not recommended for asymmetrical arrangement of the upper and lower nodes)
	K.EQ.2: Inverse distance weighting
	K.EQ.3: Quadratic
	The second digit, L, controls if the nodal area is considered when computing the weight:
	L.EQ.0: Nodal area is not considered (default),
	L.EQ.1: Nodal area is considered.

Additional Card for MODEL = 2, 12, or 22.

Card 3	1	2	3	4	5	6	7	8
Variable	UPFN	UPFS	ALPHA2	BETA2	UPRN	UPRS	ALPHA3	BETA3
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
UPFN	Plastic initiation displacement in the normal direction, $\bar{u}^{\text{pl},n}_{0,\text{ref}}$.
UPFS	Plastic initiation displacement in the shear direction, $\bar{u}^{\text{pl},s}_{0,\text{ref}}$.
ALPHA2	Plastic initiation displacement scaling factor, α_2.
BETA2	Exponent for plastic initiation displacement, β_2.
UPRN	Plastic rupture displacement in the normal direction, $\bar{u}^{\text{pl},n}_{f,\text{ref}}$.
UPRS	Plastic rupture displacement in shear direction, $\bar{u}^{\text{pl},s}_{f,\text{ref}}$.
ALPHA3	Plastic rupture displacement scaling factor, α_3.

VARIABLE	DESCRIPTION
BETA3	Exponent for plastic rupture displacement, β_3.

Additional Card for MODEL = 2, 12, or 22.

Card 4	1	2	3	4	5	6	7	8
Variable	MRN	MRS						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
MRN	Proportionality factor for dependency, m_{R_n}.
MRS	Proportionality factor for dependency, m_{R_s}.

Optional Card for MODEL = 1, 3, 11, 21, 31, or 41.

Card 5	1	2	3	4	5	6	7	8
Variable	STIFF2	STIFF3	STIFF4	LCDEXP	GAMMA	SROPT	PIDVB	
Type	F	F	F	F	F	F	F	
Default	STIFF	STIFF	STIFF	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
STIFF2	Elastic shear stiffness. Function ID if MODEL > 30 (see Remark 2). GT.0: Constant value or function ID. LT.0: |STIFF2| is a load curve ID for the curve giving shear stiffness as a function of load angle (ranging from 0 for pure shear to $\pi/2$ for pure tension).
STIFF3	Elastic bending stiffness. Function ID if MODEL > 30 (see Remark 2).

VARIABLE	DESCRIPTION
STIFF4	Elastic torsional stiffness. Function ID if MODEL > 30 (see Remark 2).
LCDEXP	Load curve for damage exponent as a function of mode mixity
GAMMA	Scaling factor, γ_1. Function ID if MODEL > 30 (see Remark 2).
SROPT	Shear rotation option that defines local kinematics system: EQ.0: Pure shear does not create a normal component (default). EQ.1: Pure shear creates a normal component.
PIDVB	Part ID for beams used to represent SPR3 in post-processing. EQ.0: Part ID automatically set (default). GT.0: PIDVB defines the part ID. LT.0: |PIDVB| defines the part ID, but an alternative type of beam representation (deformation) is invoked. Also, beams are deleted after failure.

Optional Card for MODEL = 1, 3, 11, 21, 31, 41.

Card 6	1	2	3	4	5	6	7	8
Variable	SCARN	SCARS	DAMP	SPROFF		FAILBM		
Type	F	F	F	F		F		
Default	1.0	1.0	0.0	0.0		0.0		

VARIABLE	DESCRIPTION
SCARN	Scale factor for tensile strength factor RN. This option also scales the displacements in LCF, LCUPF, and LCUPR so that the shape of the force-displacement curve stays similar. LT.0.0: |SCARN| refers to a load curve ID, giving the scale factor as a function of the state of the neighboring SPR3.See Remark 6.
SCARS	Scale factor for tensile strength factor RS. This option also scales the displacements in LCF, LCUPF, and LCUPR so that the shape of

VARIABLE	DESCRIPTION
	the force-displacement curve stays similar. LT.0.0: |SCARS| refers to a load curve ID, giving the scale factor as a function of the state of the neighboring SPR3. See Remark 6.
DAMP	Viscous coefficient to model stiffness-proportional damping effects. We recommend setting DAMP to a value between 0.05 and 0.15.
SPROFF	Specifies the relative location of the spot weld between PID1 and PID2. The value must be between -1.0 and 1.0. A value of -1.0 places the spot weld on PID1, while a value of 1.0 places it on PID2. By default (SPROFF = 0.0), the spot weld is located at the geometric center between PID1 and PID2
FAILBM	Erosion of the base material if the spot weld fails flag: EQ.0.0: If the spot weld fails, no erosion occurs in either sheet. EQ.1.0: If the spot weld fails, erosion occurs in the first sheet's nearest shell or solid element to the spot weld. EQ.2.0: If the spot weld fails, erosion occurs in the second sheet's nearest shell or solid element to the spot weld. EQ.3.0: If the spot weld fails, erosion occurs in the nearest shell or solid element in each sheet. This flag is for depicting nugget pullout and potential crack initiation.

Optional Card for MODEL = 1, 11, 21, 31, 41. See Remark 3.

Card 7	1	2	3	4	5	6	7	8
Variable	FFN	FFB	FFS	EXFC	STIFP	MFSFC	DEFC	NPFC
Type	F	F	F	F	F	F	F	F
Default	optional	optional	optional	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
FFN	Resultant normal force at failure. FFN is a function ID if MODEL > 30 (see Remark 2).
FFB	Resultant bending force at failure. FFB is a function ID if MODEL > 30 (see Remark 2).
FFS	Resultant shear force at failure. FFS is a function ID if MODEL > 30 (see Remark 2).
EXFC	Failure function exponent. EXFC is a function ID if MODEL > 30 (see Remark 2).
STIFP	Plastic stiffness. If greater than zero, this replaces LCF by a simple linear hardening law: $$F^0\left(\bar{u}^{\mathrm{pl}}\right) = 1.0 + \mathrm{STIFP} \times \bar{u}^{\mathrm{pl}} .$$ STIFP is a function ID if MODEL > 30 (see Remark 2).
MFSFC	Scaling factor for torsion term in resultant shear force. MFSC is a function ID if MODEL > 30 (see Remark 2).
DEFC	Fading energy for damage. DEFC is a function ID if MODEL > 30 (see Remark 2).
NPFC	Plastic displacement offset for damage initiation. NPFC is a function ID if MODEL > 30 (see Remark 2).

Remarks:

1. **Drilling rotation constraint method.** With this keyword, we recommend using the drilling rotation constraint method for the connected components in explicit analysis. To do this, specify DRPSID of *CONTROL_SHELL to include all shell parts involved in INTERPOLATION_SPOTWELD connections.

2. **Function inputs.** If MODEL is chosen to be greater than 10 (but less than 30), five variables must be defined as function IDs: STIFF, ALPHA1, RN, RS, and BETA1. These functions incorporate the following input values: thicknesses of both weld partners (t1, t2) and maximum engineering yield stresses, also called necking points, (sm1, sm2). For ALPHA1 = 100 such a function could look like,

```
*DEFINE_FUNCTION
       100
func(t1,t2,sm1,sm2)=sm1/sm2
```

(This function is only a demonstration; it does not make any physical sense.) For MODEL = 11 or 12, the first sheet (PID1) is the first weld partner represented by

t1 and sm1. For MODEL = 21 or 22, the thinner part is the first weld partner. Since material parameters must be identified from both weld partners during initialization, this feature is only available for a subset of material models, namely, materials 3, 24, 36, 81, 120, 123, 124, 133, 187, 224, 251, and 324.

If MODEL is chosen to be greater than 30, seventeen variables must be defined as function IDs: STIFF, ALPHA1, RN, RS, BETA1, STIFF2, STIFF3, STIFF4, GAMMA, FFN, FFB, FFS, EXFC, STIFP, MFSFC, DEFC, and NPFC. These functions incorporate the following input values: thicknesses of both weld partners (t1, t2); maximum engineering yield stresses, also called necking points, (sm1, sm2); initial yield stresses at plastic strain of 0.002 (sy1, sy2); resultant velocity (v); normal/bending/shear term in failure function (fn, fb, fs); and normal/tangential velocities (vn, vt). For ALPHA1 = 100 such a function could look like:

```
*DEFINE_FUNCTION
       100
func(t1,t2,sm1,sm2,sy1,sy2,v,fn,fb,fs,vn,vt)=sm1/sm2
```

For MODEL = 31, the first sheet (PID1) is the first weld partner represented by t1, sm1, and sy1. For MODEL = 41, the thinner part is the first weld partner.

3. **Model = 1, 3, 11, 21, 31, or 41 ("SPR3").** This numerical model is similar to the self-piercing rivet model SPR2 (see *CONSTRAINED_SPR2) but with some differences to make it more suitable for spot welds. The first difference is the symmetric behavior of the spot weld connection; that is, there is no distinction between an upper sheet and a lower sheet. This symmetry is done by averaging the normals of both parts and always distributing the balance moments equally to both sides.

   The second difference is that three (or four if STIFF4 > 0) quantities describe the kinematics instead of two. These quantities are the normal relative displacement, δ_n; the tangential relative displacement, δ_t; and the relative rotation, ω_b (and the relative twist, ω_t). These quantities are all with respect to the plane-of-maximum opening. We can define a relative displacement vector as:

$$\mathbf{u} = (\delta_n, \delta_t, \omega_b) \ .$$

   The third difference is the underlying material model. With the described kinematic quantities, LS-DYNA first computes an elastic effective force vector:

$$\tilde{\mathbf{f}} = (f_n, f_t, m_b) = \text{STIFF} \times \mathbf{u} = \text{STIFF} \times (\delta_n, \delta_t, \omega_b) \ .$$

   Or, if individual stiffnesses are defined on optional Card 5, the solver treats each component separately like this (also allows torsional stiffness to be added):

$$f_n = \text{STIFF} \times \delta_n$$
$$f_t = \text{STIFF2} \times \delta_t$$
$$m_b = \text{STIFF3} \times \omega_b$$
$$m_t = \text{STIFF4} \times \omega_t$$

From that, the algorithm computes two resultant forces for the normal and tangential directions (shear), respectively, using

$$F_n = \langle f_n \rangle + \alpha_1 m_b, \qquad F_s = f_t + \gamma_1 |m_t| \ .$$

A yield function is defined for plastic behavior

$$\phi(\tilde{\mathbf{f}}, \bar{u}^{\text{pl}}) = P(\tilde{\mathbf{f}}) - F^0(\bar{u}^{\text{pl}}) \leq 0$$

with the potential

$$P(\tilde{\mathbf{f}}) = \left[\left(\frac{F_n}{R_n} \right)^{\beta_1} + \left(\frac{F_s}{R_s} \right)^{\beta_1} \right]^{1/\beta_1} ,$$

and isotropic hardening described by load curve LCF (see Figure 10-14):

$$F^0 = F^0(\bar{u}^{\text{pl}}) \ .$$

Therefore, the relative plastic displacement, $\bar{u}^{\text{pl}}$, is calculated using the following relation:

$$P(\tilde{\mathbf{f}})\, \bar{u}^{\text{pl}} = \sum_i F_i \, u_i^{\text{pl}}, \qquad i = \{n, s\} \ .$$

In addition, if values on Card 7 are not defined, a linear softening evolution is incorporated, where damage is defined as:

$$d = \frac{\bar{u}^{\text{pl}} - \bar{u}_0^{\text{pl}}(\kappa)}{\bar{u}_f^{\text{pl}}(\kappa)}, \qquad 0 < d < 1$$

with mode mixity

$$\kappa = \frac{2}{\pi} \arctan\left(\frac{F_n}{F_s} \right), \qquad 0 < \kappa < 1 \ .$$

Alternatively, nonlinear damage evolution can be invoked by using LCDEXP, where the damage exponent m is defined as a function of the mode mixity:

$$d = \frac{1 - \exp\left(-m \dfrac{\bar{u}^{\text{pl}} - \bar{u}_0^{\text{pl}}(\kappa)}{\bar{u}_f^{\text{pl}}(\kappa)} \right)}{1 - \exp(-m)}, \qquad 0 < d < 1 \ .$$

Finally, the nominal force is computed as:

$$\mathbf{f} = (1 - d)\tilde{\mathbf{f}} \ .$$

If the parameters on Card 7 are defined, meaning at least FFN, FFB, and FFS are nonzero, a force-based failure criterion with an energy-based damage evolution is used. The failure criterion is given as:

$$f_c = \left(\frac{\langle f_n \rangle}{\text{FFN}}\right)^{\text{EXFC}} + \left(\frac{m_b}{\text{FFB}}\right)^{\text{EXFC}} + \left(\frac{f_t + \text{MFSFC} \times m_t}{\text{FFS}}\right)^{\text{EXFC}} - 1 \ .$$

The damage is defined as:

$$d = \frac{(f_n \delta_n + f_t \delta_t)}{\text{DEFC}}.$$

The implicit solver automatically uses a strongly objective formulation for the spot weld. To activate the strongly objective formulation in explicit, set MODEL to 3. For the explicit solver, this formulation tends to improve stability and accuracy when the elastic constants are high or when the model undergoes large rigid body rotation. Not all options available within MODEL = 1 are compatible with this formulation. For further details, regarding MODEL = 3, contact Synopsys or your local distributor.

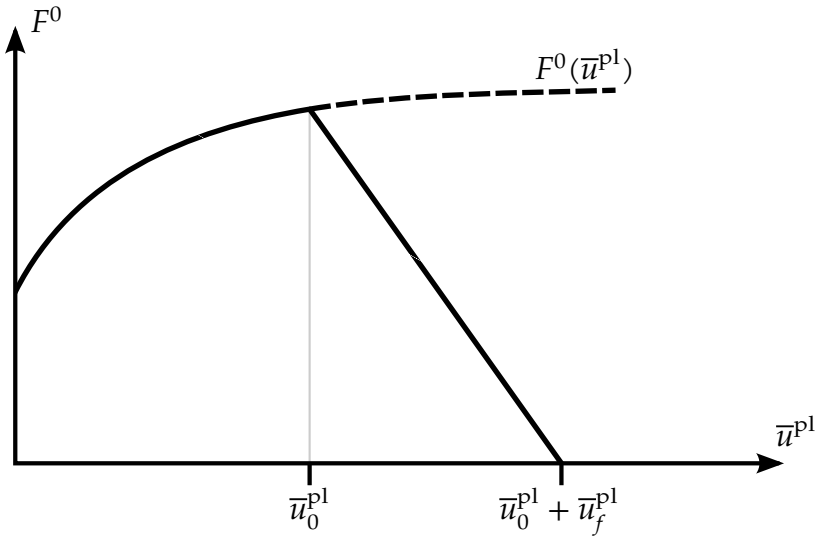


Figure 10-14. Force-displacement curve: plasticity and linear damage

4. **MODEL = 2, 12, or 22 (“SPR4”).** In this approach, the relative displacement vector is defined as in Model = 1:

$$\mathbf{u} = (\delta_n, \delta_t) \ .$$

The elastic effective force vector is computed using the elastic stiffness STIFF

$$\tilde{\mathbf{f}} = (f_n, f_t) = \text{STIFF} \times \mathbf{u} = \text{STIFF} \times (\delta_n, \delta_t) \ .$$

A yield function is defined for plastic behavior as

$$\phi(\tilde{\mathbf{f}}, \bar{u}^{\text{pl}}) = P(\tilde{\mathbf{f}}) - F^0(\bar{u}^{\text{pl}}) \leq 0$$

with relative plastic displacement $\bar{u}^{\text{pl}}$ and potential

$$P(\tilde{\mathbf{f}}) = \left[\left(\frac{f_n}{\tilde{R}_n}\right)^{\beta_1} + \left(\frac{f_t}{\tilde{R}_s}\right)^{\beta}\right]^{1/\beta_1}$$

in which $\tilde{R}_n$ and $\tilde{R}_s$ represent the load capacity in the normal and tangential directions. They are calculated using the values of R_n and R_s as follows with the influence of relative rotation angle, ω_b, scaled by α_1:

$$\tilde{R}_s = R_s$$
$$\tilde{R}_n = R_n(1 - \alpha_1\omega_b)$$

In addition, a linear softening evolution is incorporated, where damage is defined as:

$$d = \frac{\bar{u}^{\text{pl}} - \bar{u}_0^{\text{pl}}}{\bar{u}_f^{\text{pl}}}, \qquad 0 < d < 1 \ .$$

The values of $\bar{u}_0^{\text{pl}}$ and $\bar{u}_f^{\text{pl}}$ are calculated by solving the following equations

$$\left\{\left[\frac{\bar{u}_0^{\text{pl},n}}{\bar{u}_{0,\text{ref}}^{\text{pl},n}(1-\alpha_2\omega_b)}\right]^{\beta_2} + \left(\frac{\bar{u}_0^{\text{pl},s}}{\bar{u}_{0,\text{ref}}^{\text{pl},s}}\right)^{\beta_2}\right\}^{1/\beta_2} - 1 = 0, \qquad \begin{aligned}\bar{u}_0^{\text{pl},n} &= \sin(\varphi)\,\bar{u}_0^{\text{pl}}\\ \bar{u}_0^{\text{pl},s} &= \cos(\varphi)\,\bar{u}_0^{\text{pl}}\end{aligned}$$

$$\left\{\left[\frac{\bar{u}_f^{\text{pl},n}}{\bar{u}_{f,\text{ref}}^{\text{pl},n}(1-\alpha_3\omega_b)}\right]^{\beta_3} + \left(\frac{\bar{u}_f^{\text{pl},s}}{\bar{u}_{f,\text{ref}}^{\text{pl},s}}\right)^{\beta_3}\right\}^{1/\beta_3} - 1 = 0, \qquad \begin{aligned}\bar{u}_f^{\text{pl},n} &= \sin(\varphi)\,\bar{u}_f^{\text{pl}}\\ \bar{u}_f^{\text{pl},s} &= \cos(\varphi)\,\bar{u}_f^{\text{pl}}\end{aligned}$$

in which the load angle is

$$\varphi = \arctan\left(\frac{f_n}{f_s}\right) \ .$$

For rate-dependent behavior, a plastic deformation rate $\dot{\bar{u}}^{\text{pl}}$ is defined by

$$\dot{\bar{u}}^{\text{pl}} = \frac{\Delta\bar{u}^{\text{pl}}}{\Delta t}$$

where $\Delta\bar{u}^{\text{pl}}$ is the plastic increment in the current time step and Δt is the time step size. If MRN and MRS are defined, $\tilde{R}_n$ and $\tilde{R}_s$ are calculated as

$$\tilde{R}_n\left(\dot{\bar{u}}^{\text{pl}}\right) = \left(R_n + m_{R_n}\dot{\bar{u}}^{\text{pl}}\right)(1 - \alpha_1\,\omega_b)$$
$$\tilde{R}_s\left(\dot{\bar{u}}^{\text{pl}}\right) = R_s + m_{R_s}\dot{\bar{u}}^{\text{pl}}$$

A detailed description of the SPR4 approach (MODEL = 2) is given in Bier and Sommer [2013], where this model is called "SPR3_IWM".

5. **Peel ratio.** For MODEL = 1, you can optionally define R_n (RN), $\bar{u}_0^{\text{pl}}$ (LCUPF), and $\bar{u}_f^{\text{pl}}$ (LCUPR) to be dependent on the peel ratio. The peel ratio is the ratio of

the bending moment to the resultant axial force. This value is kept constant when plastic yield is reached.

6. **State of the neighboring SPR3.** The scale factors defined with SCARN and SCARS can depend on the state of the neighboring SPR3. SCARN and SCARS less than 0 enable this feature. The state measures how close the neighboring SPR3 is to failing. It is a value between 0 and 3. A value between 0 and 1 means the elastic regime, 1 and 2 means the plastic regime, and 2 and 3 means the damage (softening) regime. Thus, as the state approaches 3, the neighboring SPR3 approaches failure. Having the scale factor depend upon the state applies when the material neighboring the SPR3 weakens as the SPR3 fails.

7. **History variables output.** If NEIPB = 7 is defined on *DATABASE_EXTENT_BINARY, then the following history variables are written to the d3plot database for SPR3 beams:

History Variable	Description
1	Load state (0,…, 1 = elastic, 1,…, 2 = plastic, 2,…, 3 = damage) $$\text{SPR state} = \begin{cases} \dfrac{P(\tilde{\mathbf{f}})}{F^0(\bar{u}^{\text{pl}})} & \text{if } \phi < 0 \\ 1 + \dfrac{\bar{u}^{\text{pl}}}{\bar{u}_0^{\text{pl}}} & \text{if } d = 0 \\ 2 + d & \text{if } d > 0 \end{cases}$$
2	Resultant normal force, F_n
3	Resultant shear force, F_s
4	Normal relative displacement
5	Tangential relative displacement
6	Relative rotation (bending)
7	Relative twist (torsion), only if STIFF4 > 0

***CONSTRAINED_JOINT**_*TYPE_{OPTION}_{OPTION}_{OPTION}*

Purpose: Define a joint between two rigid bodies.

CONSTRAINED_JOINT_*TYPE is a family of keywords all sharing a common set of data cards and option flags. The available joint variants are (one is mandatory):

*CONSTRAINED_JOINT_SPHERICAL

*CONSTRAINED_JOINT_REVOLUTE

*CONSTRAINED_JOINT_CYLINDRICAL

*CONSTRAINED_JOINT_PLANAR

*CONSTRAINED_JOINT_UNIVERSAL

*CONSTRAINED_JOINT_TRANSLATIONAL

*CONSTRAINED_JOINT_LOCKING

*CONSTRAINED_JOINT_TRANSLATIONAL_MOTOR

*CONSTRAINED_JOINT_ROTATIONAL_MOTOR

*CONSTRAINED_JOINT_GEARS

*CONSTRAINED_JOINT_RACK_AND_PINION

*CONSTRAINED_JOINT_CONSTANT_VELOCITY

*CONSTRAINED_JOINT_PULLEY

*CONSTRAINED_JOINT_SCREW

*CONSTRAINED_JOINT_HARMONIC

If the force output data is to be transformed into a local coordinate, use the option:

LOCAL

to define a joint ID and heading the following option is available:

ID

and to define failure for penalty-based joints (LMF = 0 in *CONTROL_RIGID) use:

FAILURE

The ordering of the bracketed options is arbitrary.

For the first seven joint types above, excepting the Universal joint, the nodal points within the nodal pairs (1, 2), (3, 4), and (5, 6) (see Figures 10-16 through 10-22) should coincide in the initial configuration, and the nodal pairs should be as far apart as possible to obtain the best behavior. For the Universal joint, the nodes within the nodal pair (3, 4) do not coincide, but the lines drawn between nodes (1, 3) and (2, 4) must be perpendicular.

For the gear joint, the nodes within the nodal pair (1, 2) must not coincide, but for implicit the nodes within the pair (5, 6) must coincide and be located at the contact point of the gear teeth for correct results.

For cylindrical joints, by setting node 3 to zero, it is possible to use a cylindrical joint to join a node that is not on a rigid body (node 1) to a rigid body (nodes 2 and 4).

Harmonic (strain wave) gear joints use a third reference rigid body defined by nodes 7, 8, and 9. The nodal triplets (1, 2, 7) and (3, 4, 8) must coincide. Harmonic gear joints require the Lagrange multiplier formulation (LMF = 1 on *CONTROL_RIGID).

Card Summary:

Card ID. This card is included if and only if the ID keyword option is used.

JID	HEADING

Card 1. This card is required.

N1	N2	N3	N4	N5	N6	RPS	DAMP

Card 2. This card is required for joint type HARMONIC.

N7	N8	N9					

Card 3. This card is required for joint types: MOTOR, GEARS, RACK_AND_PINION, PULLEY, SCREW, and HARMONIC.

PARM	LCID	TYPE	R1	H_ANGLE			

Card 4. Include this card if the LOCAL keyword option is used.

RAID	LST						

Card 5. Include this card if the FAILURE keyword option is used.

CID	TFAIL	COUPL					

Card 6. Include this card if the FAILURE keyword option is used.

NXX	NYY	NZZ	MXX	MYY	MZZ		

Data Card Definitions:

ID Card. Additional card for ID keyword option. The heading is picked up by some of the peripheral LS-DYNA codes to aid in post-processing.

Card ID	1	2	3	4	5	6	7	8
Variable	JID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
JID	Joint ID. This must be a unique number.
HEADING	Joint descriptor. It is suggested that unique descriptions be used.

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	RPS	DAMP
Type	I	I	I	I	I	I	F	F
Default	0	0	0	0	0	0	1.0	1.0

VARIABLE	DESCRIPTION
N1	Node 1 in rigid body A. Define for all joint types.
N2	Node 2 in rigid body B. Define for all joint types.
N3	Node 3 in rigid body A. Define for all joint types except SPHERICAL.
N4	Node 4 in rigid body B. Define for all joint types except SPHERICAL.

VARIABLE	DESCRIPTION
N5	Node 5 in rigid body A. Define for joint types TRANSLATIONAL, LOCKING, ROTATIONAL_MOTOR, CONSTANT_VELOCITY, GEARS, RACK_AND_PINION, PULLEY, SCREW, and HARMONIC. For implicit GEARS, N5 should coincide with N6 and be located at the contact point of the gear teeth.
N6	Node 6 in rigid body B. Define for joint types TRANSLATIONAL, LOCKING, ROTATIONAL_MOTOR, CONSTANT_VELOCITY, GEARS, RACK_AND_PINION, PULLEY, SCREW, and HARMONIC. For implicit GEARS, N6 should coincide with N5 and be located at the contact point of the gear teeth.
RPS	Relative penalty stiffness (default = 1.0): GT.0.0: Constant value, LT.0.0: Time-dependent value given by load curve ID = -RPS (only for SPHERICAL, REVOLUTE, and CYLINDRICAL).
DAMP	Damping scale factor on the default damping value. (Revolute and Spherical Joints): EQ.0.0: Default is set to 1.0, GT.0.0.AND.LE.0.01: No damping is used.

Harmonic Joint Card. Additional card for joint type HARMONIC.

Card 2	1	2	3	4	5	6	7	8
Variable	N7	N8	N9					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
N7	Node 7 in rigid body C
N8	Node 8 in rigid body C
N9	Node 9 in rigid body C

Rotational Properties Card. Additional card for joint types MOTOR, GEARS, RACK_-AND_PINION, PULLEY, SCREW, and HARMONIC.

Card 3	1	2	3	4	5	6	7	8
Variable	PARM	LCID	TYPE	R1	H_ANGLE			
Type	F	I	I	F	F			
Default	none	0	none	↓	0.0			

VARIABLE	DESCRIPTION
PARM	Parameter, which is a function of joint type: Gears: Define R_2/R_1 Rack and Pinion: Define h Pulley: Define R_2/R_1 Screw: Define $\dot{x}/\omega$ Motors: Leave blank Harmonic: Define gear ratio (see Remark 5)
LCID	Load curve ID for MOTOR joints
TYPE	Flag for MOTOR joints as follows: EQ.0: Translational/rotational velocity EQ.1: Translational/rotational acceleration EQ.2: Translational/rotational displacement
R1	Radius, R_1, for the gear and pulley joint types. If undefined, nodal points 5 and 6 are assumed to be on the outer radius. The values of R1 and R2 affect the output reaction forces. The forces are calculated from the moments by dividing them by the radii.
H_ANGLE	Helix angle in degrees. This is only necessary for the gear joint if the gears do not mesh tangentially, such as worm gears. See Remark 2 below for a definition. See Remark 4 for a discussion of which joint formulation to use.

Local Card. Additional card required for LOCAL keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	RAID	LST						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
RAID	Rigid body or accelerometer ID. The force resultants are output in the local system of the rigid body or accelerometer.
LST	Flag for local system type: EQ.0: Rigid body EQ.1: Accelerometer

Failure Card 1. Additional card for FAILURE keyword option.

Card 5	1	2	3	4	5	6	7	8
Variable	CID	TFAIL	COUPL					
Type	I	F	F					
Default	global	0.	0.					

Failure Card 2. Additional card for FAILURE keyword option.

Card 6	1	2	3	4	5	6	7	8
Variable	NXX	NYY	NZZ	MXX	MYY	MZZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
CID	Coordinate ID for resultants in the failure criteria. EQ.0: Global coordinate system
TFAIL	Time for joint failure. EQ.0.0: Joint never fails.
COUPL	Coupling between the force and moment failure criteria. LE.0.0: The failure criteria are identical to those of spot welds. GT.0.0: The force and moment results are considered independently. See Remark 1 below.
NXX	Axial force resultant N_{xx_F} at failure. EQ.0.0: Failure due to this component is not considered.
NYY	Force resultant N_{yy_F} at failure. EQ.0.0: Failure due to this component is not considered.
NZZ	Force resultant N_{zz_F} at failure. EQ.0.0: Failure due to this component is not considered.
MXX	Torsional moment resultant M_{xx_F} at failure. EQ.0.0: Failure due to this component is not considered.
MYY	Moment resultant M_{yy_F} at failure. EQ.0.0: Failure due to this component is not considered.
MZZ	Moment resultant M_{zz_F} at failure. EQ.0.0: Failure due to this component is not considered.

Remarks:

1. **Failure criteria.** The moments for the revolute, cylindrical, planar, translational, and locking joints are calculated at the midpoint of nodes N1 and N3. The moments for the spherical, universal, constant velocity, gear, pulley, and rack and pinion joints are calculated at node N1. When COUPL is less than or equal to zero, the failure criterion is

$$\left(\frac{N_{xx}}{N_{xx_F}}\right)^2+\left(\frac{N_{yy}}{N_{yy_F}}\right)^2+\left(\frac{N_{zz}}{N_{zz_F}}\right)^2+\left(\frac{M_{xx}}{M_{xx_F}}\right)^2+\left(\frac{M_{yy}}{M_{yy_F}}\right)^2+\left(\frac{M_{zz}}{M_{zz_F}}\right)^2-1=0\ .$$

Otherwise, it consists of both

$$\left(\frac{N_{xx}}{N_{xx_F}}\right)^2+\left(\frac{N_{yy}}{N_{yy_F}}\right)^2+\left(\frac{N_{zz}}{N_{zz_F}}\right)^2-1=0\ ,$$

and

$$\left(\frac{M_{xx}}{M_{xx_F}}\right)^2+\left(\frac{M_{yy}}{M_{yy_F}}\right)^2+\left(\frac{M_{zz}}{M_{zz_F}}\right)^2-1=0\ .$$

2. **Helix angle for gear joints.** For a gear joint, the relative direction and magnitude of rotation between the two gears is determined by the *helix angle*. Let $\mathbf{e}_1$ be the unit vector directed from node 2 to 4, which corresponds to the second gear's rotation axis. See Figure 10-25. Let $\mathbf{e}_2$ be defined as the positively oriented tangent vector to motion of the teeth when spun about the $\mathbf{e}_1$ axis (the gear's axis). See Figure 10-15. The helix angle α characterizes the deviation of the teeth axis from the gear axis. In particular, α is defined as the angle between the direction of teeth, called $\mathbf{e}_3$, and the axis of the gear $\mathbf{e}_1$,

$$\mathbf{e}_3=\cos\alpha\mathbf{e}_1+\sin\alpha\mathbf{e}_2\ .$$

The gears are assumed to be setup so that the teeth initially fit having matching $\mathbf{e}_3$ directions. A nonzero helix angle is typically used to model worm gears.

3. **Penalty formulation.** When the penalty formulation is used (see LMF on *CONTROL_RIGID), at each time step, the relative penalty stiffness is multiplied by a function dependent on the step size to give the maximum stiffness that will not destroy the stability of the solution. Instabilities can result in the explicit time integration scheme if the penalty stiffness is too large. If instabilities occur, we recommend decreasing the time step or reducing the penalty scale factor to eliminate these problems.

4. **Joint formulation for gear joints with a helix angle.** If a gear joint requires a helix angle (such as a worm joint), we recommend using the Lagrange multiplier formulation (LMF = 1 on *CONTROL_RIGID) instead of the default penalty formulation (LMF = 0). The penalty formulation does not account for the helix angle, leading to incorrect results.

5. **Joint formulation harmonic gear.** Harmonic, or strain wave, gear joints use three rigid bodies: the Wave Generator (input), the Flex Spline (output), and the Circular Spline. See Figure 10-30. PARAM specifies the gear ratio that relates the relative input angular velocity, $\omega_{\text{in}}^{\text{rel}}$ to the relative output angular velocity, $\omega_{\text{out}}^{\text{rel}}$:

$$\text{PARAM} = -\frac{\omega_{\text{in}}^{\text{rel}}}{\omega_{\text{out}}^{\text{rel}}}$$

$$\omega_{\text{in}}^{\text{rel}} = \omega_{\text{wave generator}} - \omega_{\text{circular spline}}$$

$$\omega_{out}^{\text{rel}} = \omega_{\text{flex spline}} - \omega_{\text{circular spline}}$$

For example, PARAM = 100 means that the relative output angular velocity is 1/100 of the input velocity and in the opposite direction.

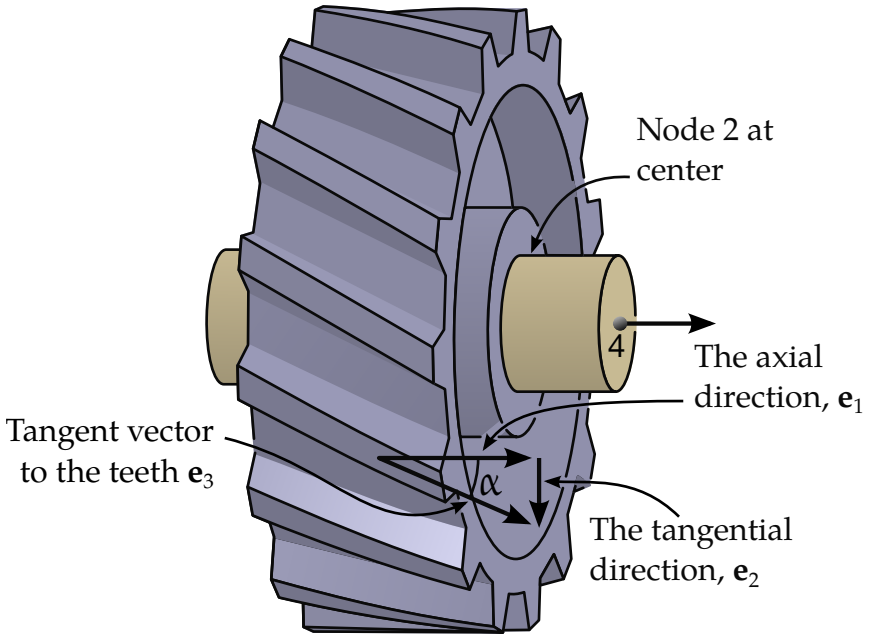


Figure 10-15. Helix angle α definition, gear #2 viewed from the extension of node n_2 to node n_6.

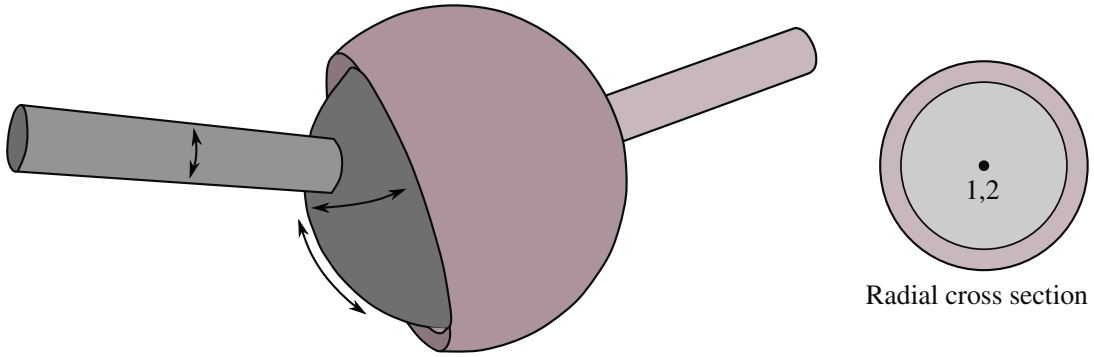


Figure 10-16. *Spherical joint.* The relative motion of the rigid bodies is constrained so that nodes which are initially coincident remain coincident. In the above figure the socket's node is not interior to the socket—LS-DYNA does not require that a rigid body's nodes be interior to the body.

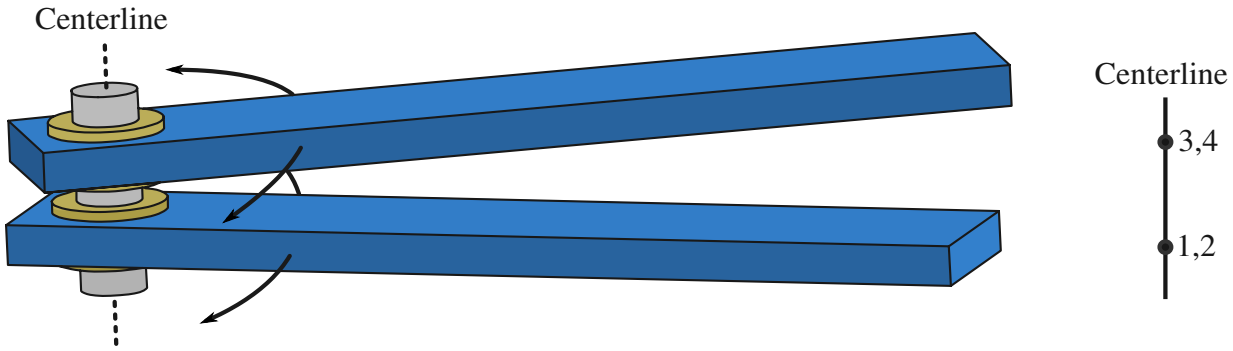


Figure 10-17. *Revolute Joint.* Nodes 1 and 2 are coincident; nodes 3 and 4 are coincident. Nodes 1 and 3 belong to rigid body A; nodes 2 and 4 belong to rigid body B. The relative motion of the two rigid bodies is restricted to rotations about the axis formed by the two pairs of coincident nodes. This axis is labeled the "centerline."

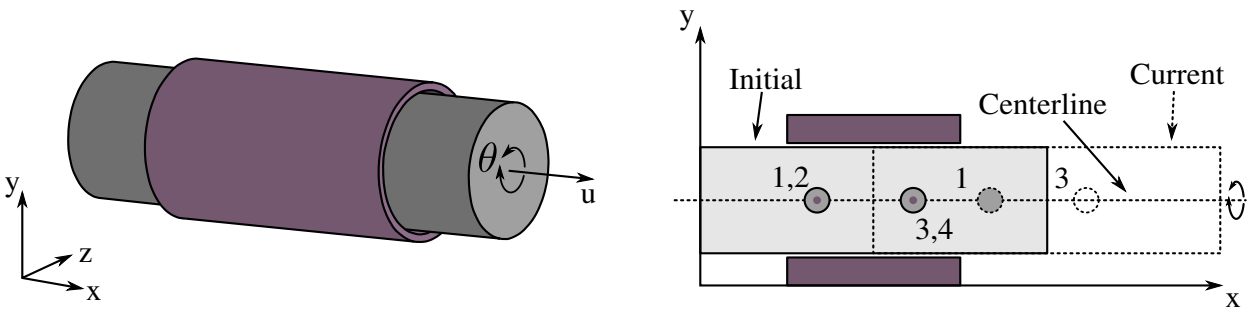


Figure 10-18. *Cylindrical Joint.* This joint is derived from the rotational joint by relaxing the constraints along the centerline. This joint admits relative rotation and translation along the centerline.

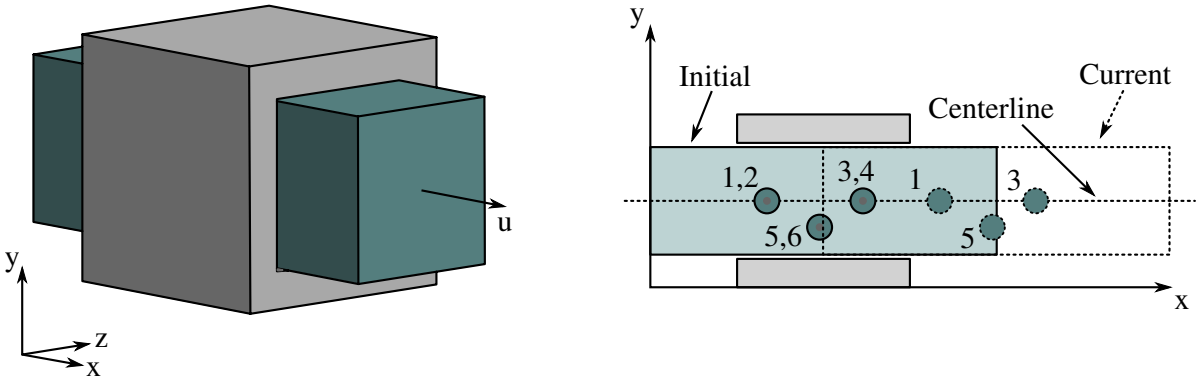


Figure 10-19. *Translational joint*. This is a cylindrical joint with a third pair of off-centerline nodes which restrict rotation. Aside from translation along the centerline, the two rigid bodies are stuck together.

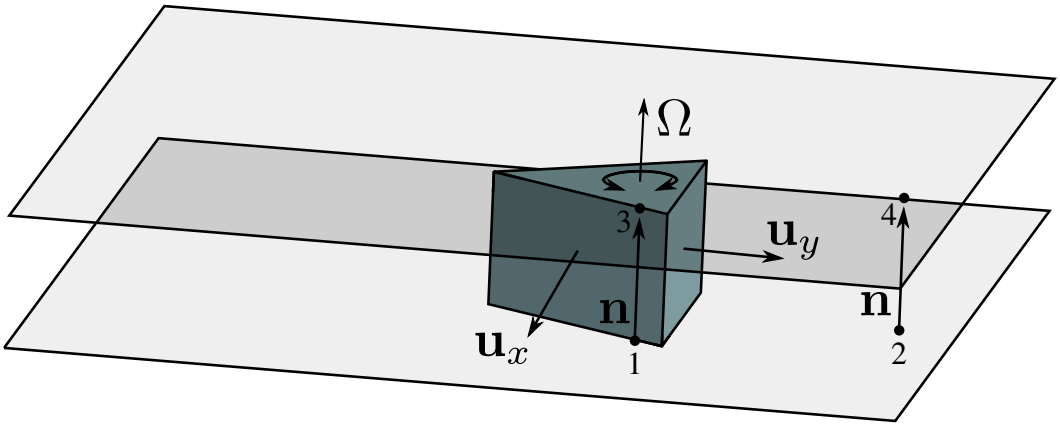


Figure 10-20. *Planar joint*. This joint is derived from the rotational joint by relaxing the constraints normal to the centerline. Relative displacements along the direction of the centerline are excluded.

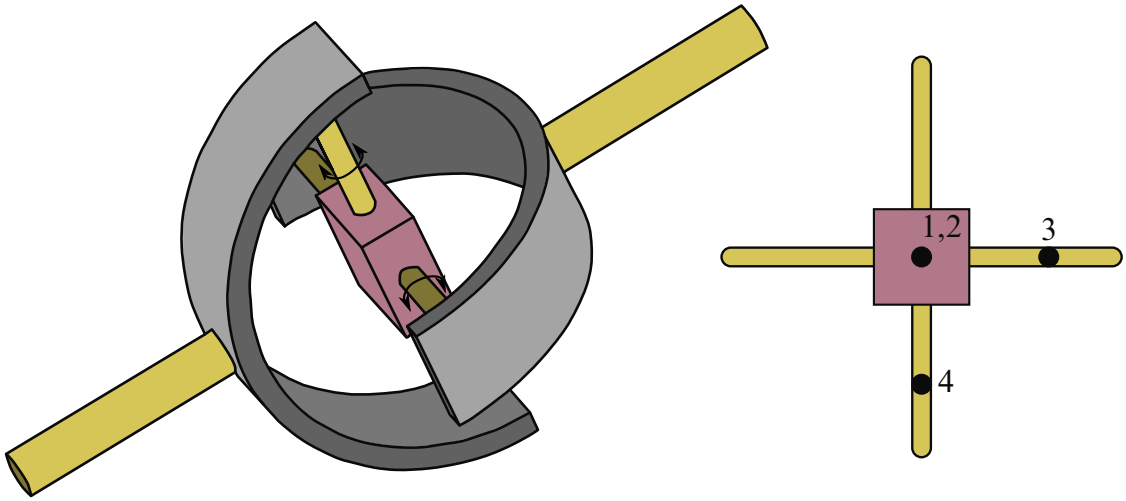


Figure 10-21. *Universal Joint*. Nodes 1 and 2 are initially coincident. The segments formed by nodal pairs (1, 3) and (2, 4) must be orthogonal; they serve as axes about which the two bodies may undergo relative rotation. The universal joint excludes all other relative motion and the axes remain orthogonal at all time.

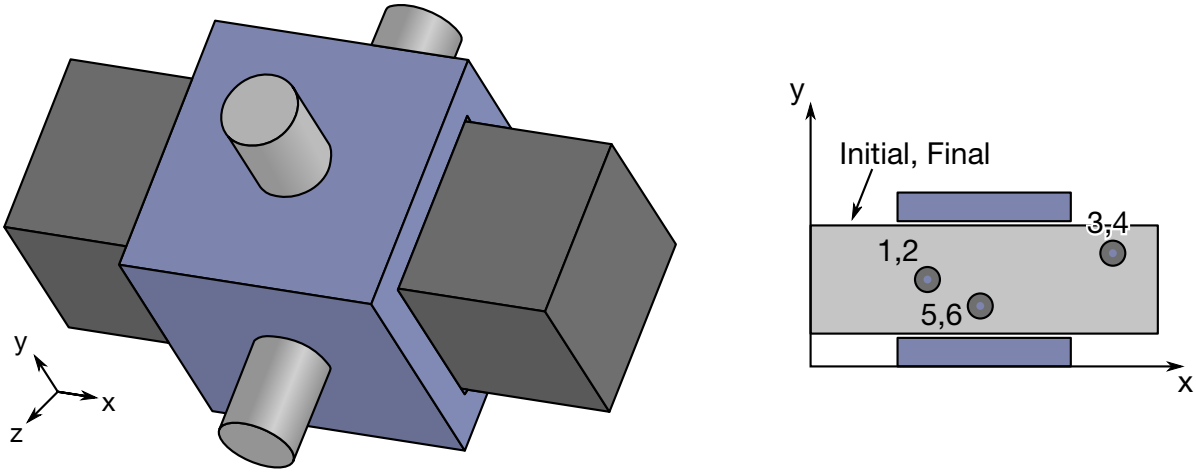


Figure 10-22. *Locking Joint*. A locking joint couples two rigid bodies in all six degrees-of-freedom. The forces and moments required to form this coupling are written to the jntforc file (*DATABASE_JNTFORC). As stated in the Remarks, forces and moments in jntforc are calculated halfway between N1 and N3. Nodal pairs (1, 2), (3, 4) and (5, 6) must be coincident. The three spatial points corresponding to three nodal pairs must be neither collocated nor collinear.

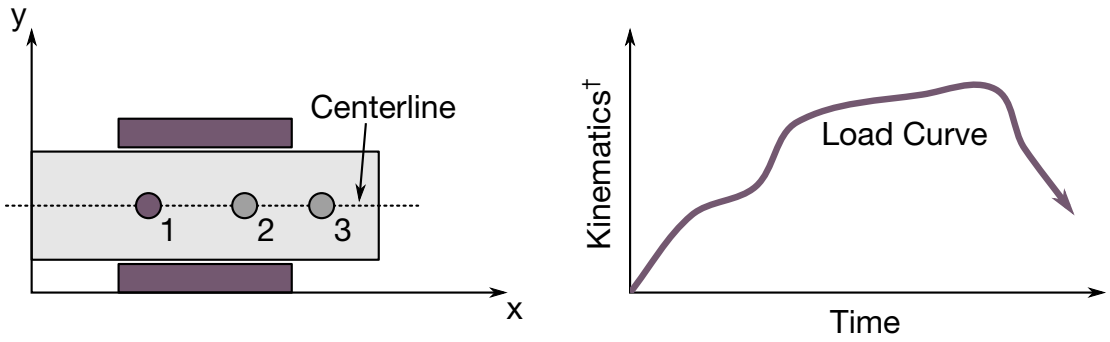


Figure 10-23. *Translational motor joint.* This joint is usually used in combination with a translational or a cylindrical joint. Node 1 and node 2 belong to the first rigid body and the second rigid body, resp. Furthermore, nodes 1 and 2 must be *coincident*. Node 3 may belong to either rigid body. Curve LCID defines the velocity/acceleration/displacement of N1 minus that of N2, where velocity/acceleration/displacement is measured in the direction of the vector N2-to-N3. Node 4 is not used and can be left blank. The value of the load curve may specify any of several kinematic measures; see TYPE.

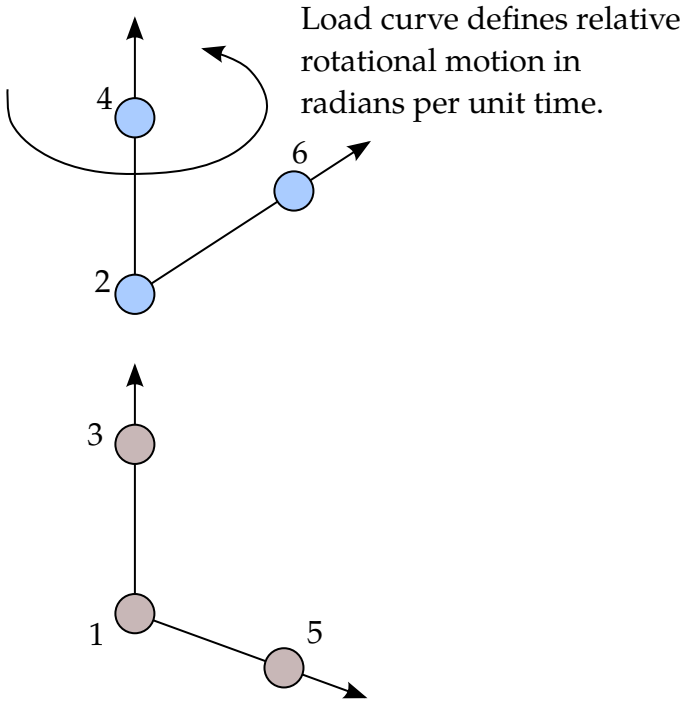


Figure 10-24. *Rotational motor joint.* This joint can be used in combination with other joints such as the revolute or cylindrical joints.

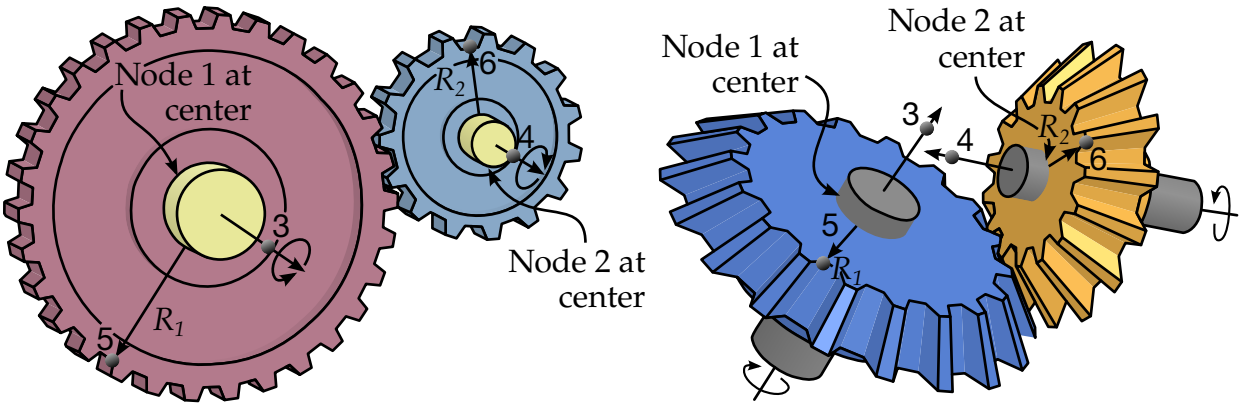


Figure 10-25. *Gear joints.* Nodal pairs (1, 3) and (2, 4) define axes that are orthogonal to the gears. Nodal pairs (1, 5) and (2, 6) define vectors in the plane of the gears. The ratio R_2/R_1 is specified but need not necessarily correspond to the geometry, if for instance the gear consists of spiral grooves. Note that the gear joint in itself does not maintain the contact point but this requires additional treatment, such as accompanying it with other joints. For implicit, nodes 5 and 6 must initially coincide for correct result in general.

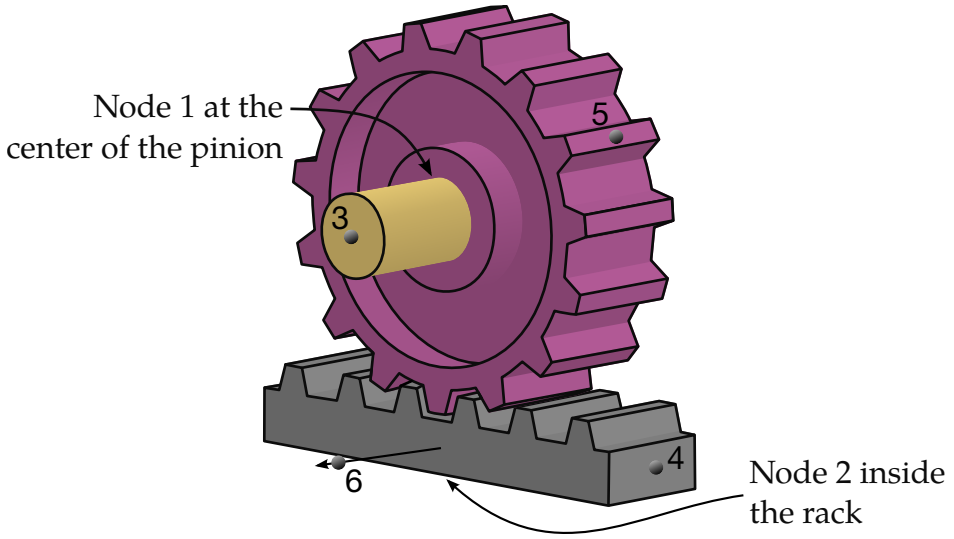


Figure 10-26. *Rack and pinion joint.* Nodal pair (1, 3) defines the axis of rotation of the first body (the pinion). Nodal pair (1, 5) is a vector in the plane of the pinion and is orthogonal to nodal pair (1, 3). Nodal pair (2, 4) defines the direction of travel for the second body (the rack). Nodal pair (2, 6) is parallel to the axis of the pinion and is thus parallel to nodal pair (1, 3). The value h is specified. The velocity of the rack is $h\omega_{\text{pinion}}$ in its direction of travel, where ω_{pinion} is positive according to the right-hand rule about the axis of the pinion.

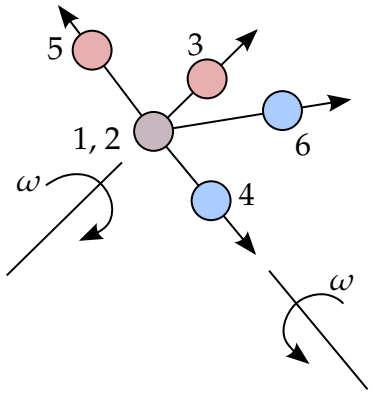


Figure 10-27. *Constant velocity joint.* Nodal pairs (1, 3) and (2, 4) define axes for the constant angular velocity. Nodal pairs (1, 5) and (2, 6) are orthogonal to nodal pairs (1, 3) and (2, 4), respectively. Here nodal points 1 and 2 must be coincident.

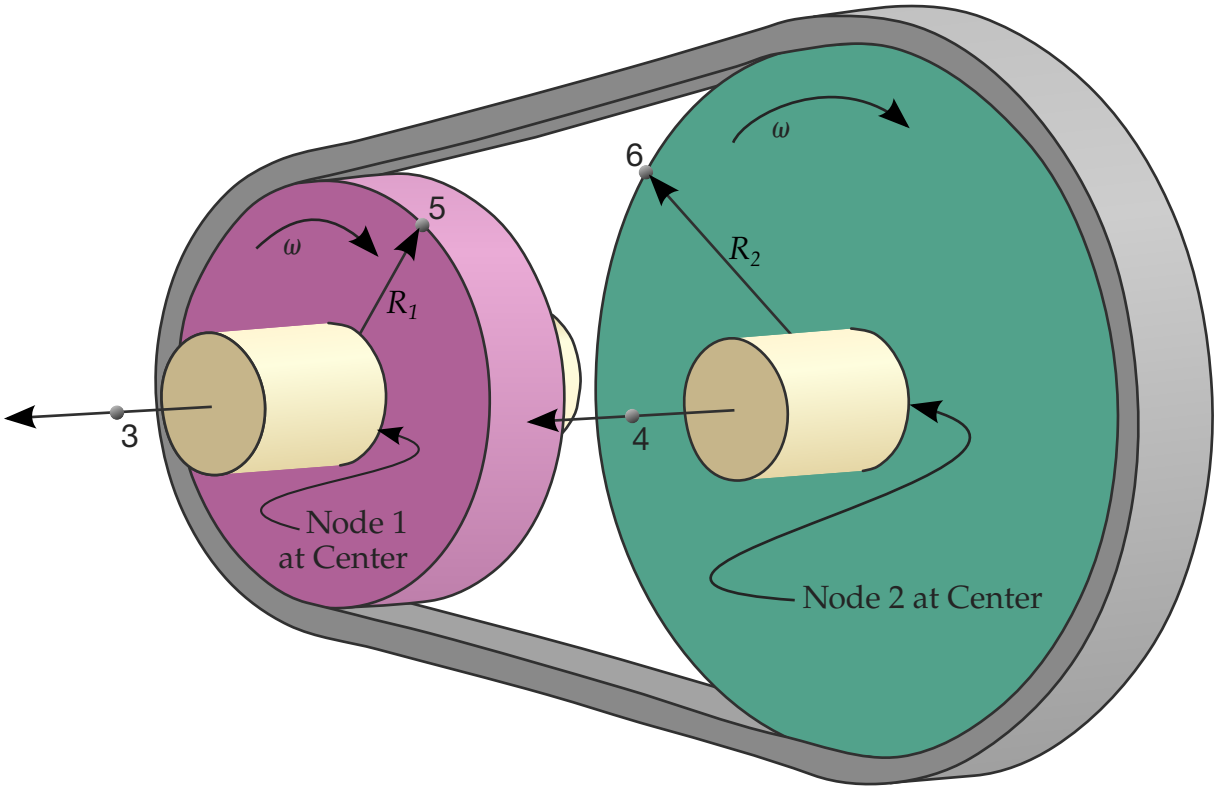


Figure 10-28. *Pulley joint.* Nodal pairs (1, 3) and (2, 4) define axes that are orthogonal to the pulleys. Nodal pairs (1, 5) and (2, 6) define vectors in the plane of the pulleys. The ratio R_2/R_1 is specified.

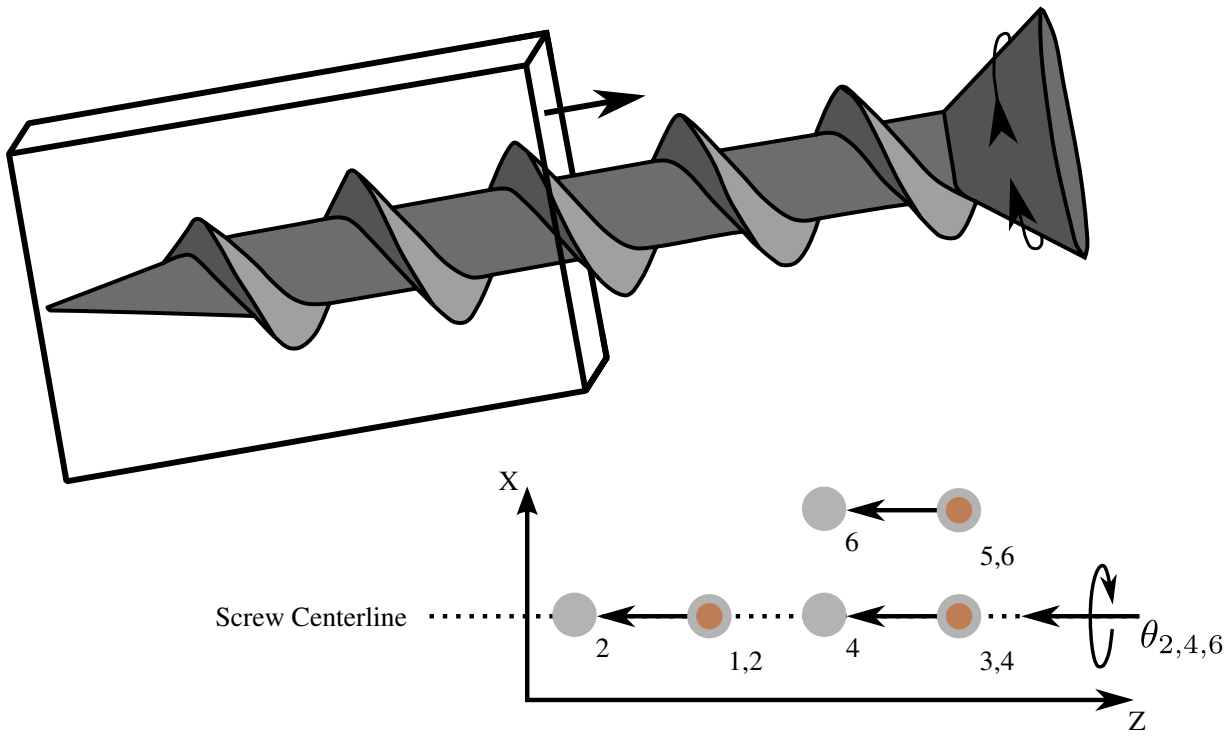


Figure 10-29. *Screw joint.* The second body (nodes 2, 4, and 6) translates in response to the spin of the first body (nodes 1, 3, and 5). Nodal pairs (1, 3) and (2, 4) lie along the same axis and nodal pairs (1, 5) and (2, 6) are orthogonal to the axis. PARM on Card 2 sets the helix ratio, $\dot{x}/\omega$. Note that, for a positive spin, the second body moves in the direction from node 1 to node 5.

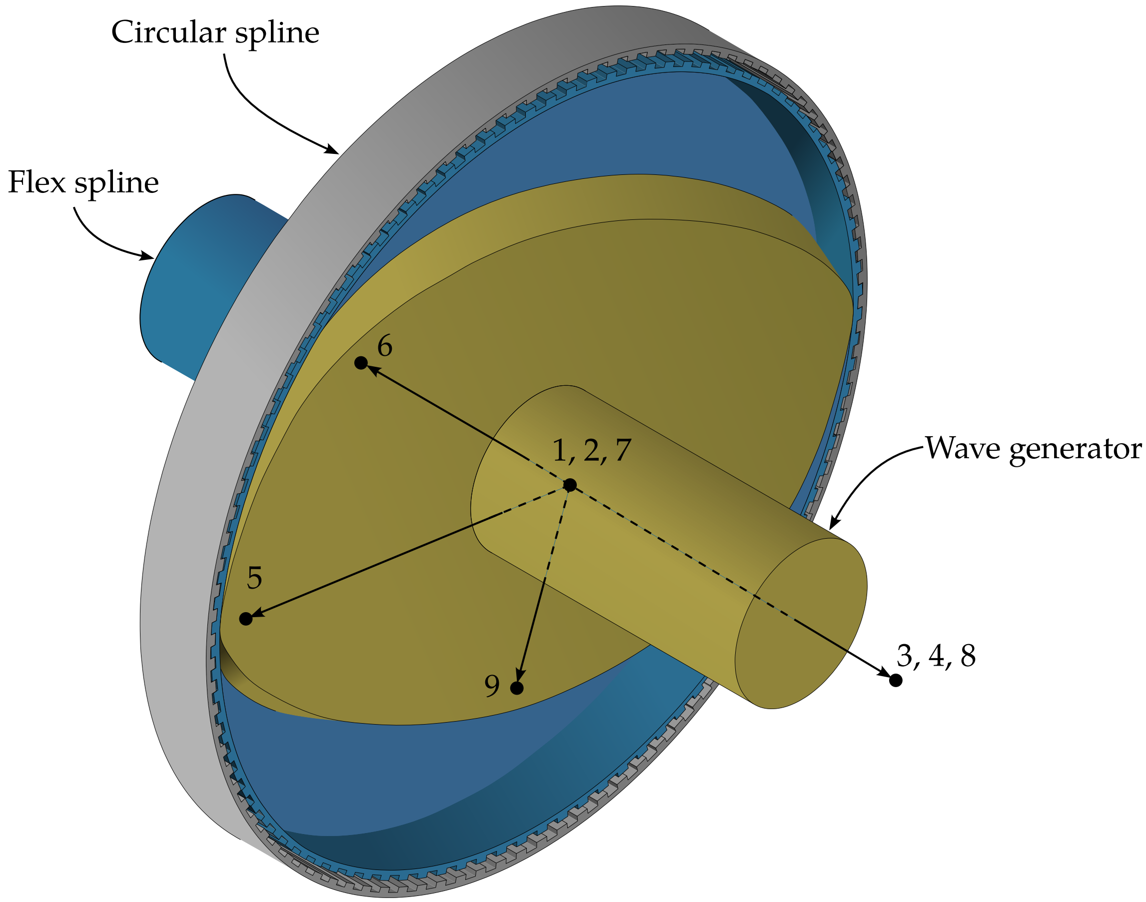


Figure 10-30. *Harmonic gear joint.* The flex spline (nodes 2, 4, and 6) translates in response to the spin of the wave generator (nodes 1, 3, and 5) relative to the spin of the circular spline (nodes 7, 8, and 9). Nodal pairs (1, 3),(2, 4), and (7, 8) lie along the same axis, and nodal pairs (1, 5),(2, 6), and (7,9) are orthogonal to the axis. The gear ratio is specified by the input parameter PARAM, and the relative motion of the three rigid bodies is restricted to rotations about the axis.

Example 1:

```
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_JOINT_PLANAR
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a planar joint between two rigid bodies.
$    - Nodes 91 and 94 are on rigid body 1.
$    - Nodes 21 and 150 are on rigid body 2.
$    - Nodes 91 and 21 must be coincident.
$        * These nodes define the origin of the joint plane.
$    - Nodes 94 and 150 must be coincident.
$        * To accomplish this, massless node 150 is artificially created at
$          the same coordinates as node 94 and then added to rigid body 2.
$        * These nodes define the normal of the joint plane (e.g., the
$          vector from node 91 to 94 defines the planes' normal).
```

```
$
*CONSTRAINED_JOINT_PLANAR
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       n1        n2        n3        n4        n5        n6       rps
        91        21        94       150                     0.000E+00
$
$
*NODE
$    nid               x               y               z      tc      rc
     150            0.00            3.00            0.00       0       0
$
*CONSTRAINED_EXTRA_NODES_SET
$      pid      nsid
         2         6
$*SET_NODE_LIST
$      sid
         6
$     nid1
       150
$
$$$  request output for joint force data
$
*DATABASE_JNTFORC
$  dt/cycl      lcdt
    0.0001
$
```

Example 2:

```
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_JOINT_REVOLUTE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Create a revolute joint between two rigid bodies.  The rigid bodies must
$  share a common edge to define the joint along.  This edge, however, must
$  not have the nodes merged together.  Rigid bodies A and B will rotate
$  relative to each other along the axis defined by the common edge.
$
$  Nodes 1 and 2 are on rigid body A and coincide with nodes 9 and 10
$  on rigid body B, respectively.  (This defines the axis of rotation.)
$
$  The relative penalty stiffness on the revolute joint is to be 1.0,
$
*CONSTRAINED_JOINT_REVOLUTE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       n1        n2        n3        n4        n5        n6       rps      damp
         1         9         2        10                            1.0
$
$  Note:  A joint stiffness is not mandatory for this joint to work.
$          However, to see how a joint stiffness can be defined for this
$          particular joint, see the corresponding example listed in:
$             *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

CONSTRAINED_JOINT_COOR_*TYPE_{OPTION}_{OPTION}_{OPTION}

CONSTRAINED_JOINT_COOR_*TYPE is a family of keywords all sharing a common set of data cards and option flags. The available joint variants are (one is mandatory):

*CONSTRAINED_JOINT_COOR_SPHERICAL

*CONSTRAINED_JOINT_COOR_REVOLUTE

*CONSTRAINED_JOINT_COOR_CYLINDRICAL

*CONSTRAINED_JOINT_COOR_PLANAR

*CONSTRAINED_JOINT_COOR_UNIVERSAL

*CONSTRAINED_JOINT_COOR_TRANSLATIONAL

*CONSTRAINED_JOINT_COOR_LOCKING

*CONSTRAINED_JOINT_COOR_TRANSLATIONAL_MOTOR

*CONSTRAINED_JOINT_COOR_ROTATIONAL_MOTOR

*CONSTRAINED_JOINT_COOR_GEARS

*CONSTRAINED_JOINT_COOR_RACK_AND_PINION

*CONSTRAINED_JOINT_COOR_CONSTANT_VELOCITY

*CONSTRAINED_JOINT_COOR_PULLEY

*CONSTRAINED_JOINT_COOR_SCREW

*CONSTRAINED_JOINT_COOR_HARMONIC

If the force output data is to be transformed into a local coordinate, use the option:

LOCAL

to define a joint ID and heading, the following option is available:

ID

and to define failure for penalty-based joints (LMF = 0 in *CONTROL_RIGID) use:

FAILURE

The order of the bracketed options is arbitrary.

Purpose: Define a joint between two rigid bodies; see Figures 10-16 through 10-30 of the *CONSTRAINED_JOINT_*TYPE* section. The connection coordinates are given instead of the nodal point IDs required in the previous section, *CONSTRAINED_JOINT_*TYPE*. Nodes are automatically generated for each coordinate and are constrained to the rigid body. When coincident nodes are expected in the initial configuration, only one connection coordinate is needed, since the second node's connection coordinate, if given, is ignored. The created nodal IDs are chosen to exceed the maximum user ID. The coordinates of the joint nodes are specified on Cards 2 - 8. The input that follows Card 8 is identical to that in the previous section.

In the first seven joint types above, excepting the Universal joint, the coordinate points within the nodal pairs (1, 2), (3, 4), and (5, 6) (see Figures 10-16 through 10-22) should coincide in the initial configuration, and the nodal pairs should be as far apart as possible to obtain the best behavior. For the Universal joint, the nodes within the coordinate pair (3, 4) do not coincide, but the lines drawn between nodes (1, 3) and (2, 4) must be perpendicular.

For the gear joint, the nodes within the coordinate pair (1, 2) must not coincide.

For cylindrical joints, by setting node 3 to zero, it is possible to use a cylindrical joint to join a node that is not on a rigid body (node 1) to a rigid body (nodes 2 and 4).

Harmonic gear joints use a third reference rigid body defined by nodes 7, 8, and 9, However, since the nodal triplets (1, 2, 7) and (3, 4, 8) must coincide, only the nodal coordinates of node 9 are necessary on Card 8. Harmonic gear joints require the Lagrange multiplier formulation (LMF = 1 on *CONTROL_RIGID).

Card Summary:

Card ID. Include this card if the ID keyword option is used.

JID	HEADING

Card 1. This card is required.

RBID_A	RBID_B	RPS	DAMP	TMASS	RMASS	RBID_C	

Card 2. This card is required.

X1	Y1	Z1					

Card 3. This card is required.

X2	Y2	Z2					

Card 4. This card is required.

X3	Y3	Z3					

Card 5. This card is required.

X4	Y4	Z4					

Card 6. This card is required.

X5	Y5	Z5					

Card 7. This card is required.

X6	Y6	Z6					

Card 8. This card is only required for joint type HARMONIC.

X9	Y9	Z9					

Card 9. This card is required for joint types: MOTOR, GEARS, RACK_AND_PINION, PULLEY, SCREW, and HARMONIC.

PARM	LCID	TYPE	R1				

Card 10. Include this card if the LOCAL keyword option is used.

RAID	LST						

Card 11. Include this card if the FAILURE keyword option is used.

CID	TFAIL	COUPL					

Card 12. Include this card if the FAILURE keyword option is used.

NXX	NYY	NZZ	MXX	MYY	MZZ		

Data Card Definitions:

ID Card. Additional card for the ID keyword option. The heading is picked up by some of the peripheral LS-DYNA codes to aid in post-processing.

Card ID	1	2	3	4	5	6	7	8
Variable	JID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
JID	Joint ID. This must be a unique number.
HEADING	Joint descriptor. It is suggested that unique descriptions be used.

Card 1	1	2	3	4	5	6	7	8
Variable	RBID_A	RBID_B	RPS	DAMP	TMASS	RMASS	RBID_C	
Type	I	I	F	F	F	F	F	

VARIABLE	DESCRIPTION
RBID_A	Part ID of rigid body A. See Remark 2.
RBID_B	Part ID of rigid body B. See Remark 2.
RPS	Relative penalty stiffness (default = 1.0). See Remark 1.
DAMP	Damping scale factor on the default damping value. (Revolute and Spherical Joints): EQ.0.0: Set to 1.0 (the default) GT.0.0 and LE.0.01: No damping is used.
TMASS	Lumped translational mass. The mass is equally split between the first points defined for rigid bodies A and B. See Remark 3.
RMASS	Lumped rotational inertia. The inertia is equally split between the first points defined for rigid bodies A and B. See Remark 3.

VARIABLE	DESCRIPTION
RBID_C	Part ID of rigid body C for the harmonic gear joint. See Remark 2.

Card 2	1	2	3	4	5	6	7	8
Variable	X1	Y1	Z1					
Type	F	F	F					

Card 3	1	2	3	4	5	6	7	8
Variable	X2	Y2	Z2					
Type	F	F	F					

Card 4	1	2	3	4	5	6	7	8
Variable	X3	Y3	Z3					
Type	F	F	F					

Card 5	1	2	3	4	5	6	7	8
Variable	X4	Y4	Z4					
Type	F	F	F					

Card 6	1	2	3	4	5	6	7	8
Variable	X5	Y5	Z5					
Type	F	F	F					

Card 7	1	2	3	4	5	6	7	8
Variable	X6	Y6	Z6					
Type	F	F	F					

Card 8	1	2	3	4	5	6	7	8
Variable	X9	Y9	Z9					
Type	F	F	F					

VARIABLE	DESCRIPTION
X1, Y1, Z1	Coordinate of point 1 in rigid body A. Define for all joint types.
X2, Y2, Z2	Coordinate of point 2 in rigid body B. If points 1 and 2 are coincident in the specified joint type, the coordinate for point 1 is used.
X3, Y3, Z3	Coordinate of point 3 in rigid body A. Define for all joint types.
X4, Y4, Z4	Coordinate of point 4 in rigid body B. If points 3 and 4 are coincident in the specified joint type, the coordinate for point 3 is used.
X5, Y5, Z5	Coordinate of point 5 in rigid body A. Define for all joint types.
X6, Y6, Z6	Coordinate of point 6 in rigid body B. If points 5 and 6 are coincident in the specified joint type, the coordinate for point 5 is used.
X9, Y9, Z9	Coordinate of point 9 in rigid body C for the harmonic gear joint.

Rotational Properties Card. Additional card for joint types MOTOR, GEARS, RACK_-AND_PINION, PULLEY, SCREW, and HARMONIC.

Card 9	1	2	3	4	5	6	7	8
Variable	PARM	LCID	TYPE	R1				
Type	F	I	I	F				
Default	none	0	0	↓				

VARIABLE	DESCRIPTION
PARM	Parameter, which is a function of joint type: Gears: Define R_2/R_1 Rack and Pinion: Define h Pulley: Define R_2/R_1 Screw: Define $\dot{x}/\omega$ Motors: Leave blank Harmonic: Define gear ratio (see Remark 5 of *CONSTRAINED_JOINT_...)
LCID	Define load curve ID for MOTOR joints.
TYPE	Define an integer flag for MOTOR joints as follows: EQ.0: Translational/rotational velocity EQ.1: Translational/rotational acceleration EQ.2: Translational/rotational displacement
R1	Radius, R_1, for the gear and pulley joint types. If left undefined, nodal points 5 and 6 are assumed to be on the outer radius. R1 is the moment arm used to calculate the joint reaction forces. The ratio R_2/R_1 gives the transmitted moments, but not the forces. The force is the moment divided by distance R1.

Local Card. Additional card for LOCAL keyword option.

Card 10	1	2	3	4	5	6	7	8
Variable	RAID	LST						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
RAID	Rigid body or accelerometer ID. The force resultants are output in the local system of the rigid body or accelerometer.
LST	Flag for local system type: EQ.0: Rigid body EQ.1: Accelerometer

Failure Card 1. Additional card for the FAILURE keyword option.

Card 11	1	2	3	4	5	6	7	8
Variable	CID	TFAIL	COUPL					
Type	I	F	F					
Default	global	0.0	0.0					

Failure Card 2. Additional card for the FAILURE keyword option.

Card 12	1	2	3	4	5	6	7	8
Variable	NXX	NYY	NZZ	MXX	MYY	MZZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
CID	Coordinate ID for resultants in the failure criteria. EQ.0: Global coordinate system
TFAIL	Time for joint failure. EQ.0.0: Joint never fails.
COUPL	Coupling between the force and moment failure criteria: LE.0.0: Failure criteria are identical to those of spotwelds. GT.0.0: The force and moment results are considered independently. See the remarks in *CONSTRAINED_JOINT_*TYPE*.
NXX	Axial force resultant N_{XX_F} at failure. EQ.0.0: Failure due to this component is not considered.
NYY	Force resultant N_{YY_F} at failure. EQ.0.0: Failure due to this component is not considered.
NZZ	Force resultant N_{ZZ_F} at failure. EQ.0.0: Failure due to this component is not considered.
MXX	Torsional moment resultant M_{XX_F} at failure. EQ.0.0: Failure due to this component is not considered.
MYY	Moment resultant M_{YY_F} at failure. EQ.0.0: Failure due to this component is not considered.

VARIABLE	DESCRIPTION
MZZ	Moment resultant M_{ZZ_F} at failure. EQ.0.0: Failure due to this component is not considered.

Remarks:

1. **Penalty method.** When the penalty method is used (see *CONTROL_RIGID), at each time step, the relative penalty stiffness is multiplied by a function dependent on the step size to give the maximum stiffness that will not destroy the stability of the solution. The explicit time integrator can become unstable when the penalty stiffness is too large. If instabilities occur, we recommend decreasing the time step or reducing the penalty scale factor to eliminate these problems.

2. **Joint nodes.** The coordinates specified in this formulation are used to generate up to six nodes and assign them to the appropriate rigid body for updating. RBID_A updates coordinates 1, 3, and 5, while RBID_B updates coordinates 2, 4, and 6. There is no method to detect when the wrong bodies are assigned, and therefore, you must ensure that the ordering of RBID_A and RBID_B is correct. For some joints, such as the spherical joint, the coordinate points are the same, and the ordering does not matter. However, for others, such as the rack and pinion, the ordering is critical. For the harmonic joint, three extra nodes, 7, 8, and 9, are generated, belonging to RBID_C.

3. **Mass and inertia.** Unless the rigid bodies are massless or nearly massless, there is usually no need to add mass or inertia to the rigid bodies, and TMASS and RMASS may be safely set to zero. Specifying values that are larger than necessary may perturb the motion of the bodies and result in significant errors. Since the masses and inertias are included in the mass calculations in the output, the values should be set to zero if the masses and inertia tensors are to be used outside of LS-DYNA.

CONSTRAINED_JOINT_STIFFNESS_*OPTION_{OPTION}

Available options include:

FLEXION-TORSION

GENERALIZED

TRANSLATIONAL

CYLINDRICAL

If desired a description of the joint stiffness can be provided with the option:

TITLE

which is written into the d3hsp and jntforc files.

Purpose: Define optional rotational, translational or cylindrical joint stiffness for joints defined by *CONSTRAINED_JOINT_*OPTION*. These definitions apply to all joints even though degrees of freedom that are considered in the joint stiffness capability may be constrained out in some joint types. The energy that is dissipated with the joint stiffness option is written for each joint in joint force file with the default name, jntforc. In the global energy balance this energy is included with the energy of the discrete elements, that is, the springs and dampers.

Card Summary:

Card Title. This card is included if and only if the TITLE keyword option is used.

TITLE

Card 1. This card is required.

JSID	PIDA	PIDB	CIDA	CIDB	JID	RPS	

Card 2a.1. This card is included if the FLEXION-TORSION keyword option is used.

LCIDAL	LCIDG	LCIDBT	DLCIDAL	DLCIDG	DLCIDBT		

Card 2a.2. This card is included if the FLEXION-TORSION keyword option is used.

ESAL	FMAL	ESBT	FMBT				

Card 2a.3. This card is included if the FLEXION-TORSION keyword option is used.

SAAL	NSABT	PSABT					

Card 2b.1. This card is included if the GENERALIZED keyword option is used.

LCIDPH	LCIDT	LCIDPS	DLCIDPH	DLCIDT	DLCIDPS		

Card 2b.2. This card is included if the GENERALIZED keyword option is used.

ESPH	FMPH	EST	FMT	ESPS	FMPS		

Card 2b.3. This card is included if the GENERALIZED keyword option is used.

NSAPH	PSAPH	NSAT	PSAT	NSAPS	PSAPS		

Card 2c.1. This card is included if the TRANSLATIONAL keyword is used.

LCIDX	LCIDY	LCIDZ	DLCIDX	DLCIDY	DLCIDZ		

Card 2c.2. This card is included if the TRANSLATIONAL keyword is used.

ESX	FFX	ESY	FFY	ESZ	FFZ		

Card 2c.3. This card is included if the TRANSLATIONAL keyword is used.

NSDX	PSDX	NSDY	PSDY	NSDZ	PSDZ	FS	FD

Card 2d.1. This card is included if the CYLINDRICAL keyword is used.

LCIDR		LCIDZ	DLCIDR	DLCIDP	DLCIDZ	LCIDT	DLCIDT

Card 2d.2. This card is included if the CYLINDRICAL keyword is used.

ESR	FFR			ESZ	FFZ	RAD1	RAD2

Card 2d.3. This card is included if the CYLINDRICAL keyword is used.

	PSDR			NSDZ	PSDZ	FS	FD

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

Card 1	1	2	3	4	5	6	7	8
Variable	JSID	PIDA	PIDB	CIDA	CIDB	JID	RPS	
Type	I	I	I	I	I	I	F	
Default	none	none	none	none	CIDA	0	1.0	

VARIABLE	DESCRIPTION
TITLE	Description of joint stiffness for output files jntforc and d3hsp
JSID	Joint stiffness ID
PIDA	Part ID for rigid body A; see *PART.
PIDB	Part ID for rigid body B; see *PART.
CIDA	Coordinate ID for rigid body A; see *DEFINE_COORDINATE_OPTION. For the translational and cylindrical stiffnesses, the local coordinate system must be defined by nodal points, *DEFINE_COORDINATE_NODES, since the first nodal point in each coordinate system is used to track the motion.
CIDB	Coordinate ID for rigid body B. If zero, the coordinate ID for rigid body A is used; see *DEFINE_COORDINATE_OPTION. For the translational and cylindrical stiffnesses, the local coordinate system must be defined by nodal points, *DEFINE_COORDINATE_NODES, since the first nodal point in each coordinate system is used to track the motion.
JID	Joint ID for the joint reaction forces. If zero, tables can not be used in place of load curves for defining the frictional moments.

VARIABLE	DESCRIPTION
RPS	Relative penalty stiffness used for joint friction calculation. It is the same parameter as RPS in *CONSTRAINED_JOINT_*TYPE*. It only applies for keyword options TRANSLATIONAL and CYLINDRICAL. FS and FD must be defined in either Card 2c. 3 or Card 2d. 3. It can be used to calculate the joint force, so we can define it here instead of in *CONSTRAINED_JOINT_*TYPE*.

Flexion-Torsion Joint Stiffness Cards:

Card 2 for FLEXION-TORSION option.

Card 2a.1	1	2	3	4	5	6	7	8
Variable	LCIDAL	LCIDG	LCIDBT	DLCIDAL	DLCIDG	DLCIDBT		
Type	I	I	I	I	I	I		
Default	0.0	1.0	0.0	0	1.0	0		

VARIABLE	DESCRIPTION
LCIDAL	Load curve ID for α-moment as a function of rotation in radians. See Figure 10-31 where it should be noted that $0 \le \alpha \le \pi$. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.
LCIDG	Load curve ID for γ as a function of a scale factor which scales the bending moment due to the α rotation. This load curve should be defined in the interval $-\pi \le \gamma \le \pi$. See *DEFINE_CURVE. EQ.0: The scale factor defaults to 1.0.
LCIDBT	Load curve ID for β-torsion moment as a function of twist in radians. See *DEFINE_CURVE. EQ.0: The applied twist is set to 0.0.
DLCIDAL	Load curve ID for α-damping momentas a function of rate of rotation in radians per unit time. See *DEFINE_CURVE. EQ.0: damping is not considered.
DLCIDG	Load curve ID for γ-damping scale factor as a function of rate of rotation in radians per unit time. This scale factor scales the α-damping moment. See *DEFINE_CURVE. EQ.0: The scale factor defaults to 1.0.
DLCIDBT	Load curve ID for β-damping torque as a function of rate of twist. See *DEFINE_CURVE. EQ.0: Damping is not considered.

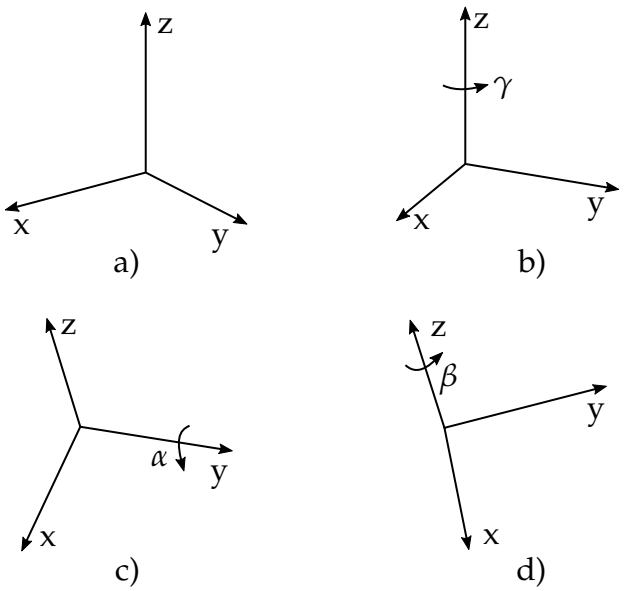


Figure 10-31. The angles γ, α, and β align rigid body one with rigid body two for the FLEXION-TORSION option, using z-y-z Euler angles (in the order shown by the figure labels) to calculate the moments. An illustrative version of this figure is shown in Figure 10-32.

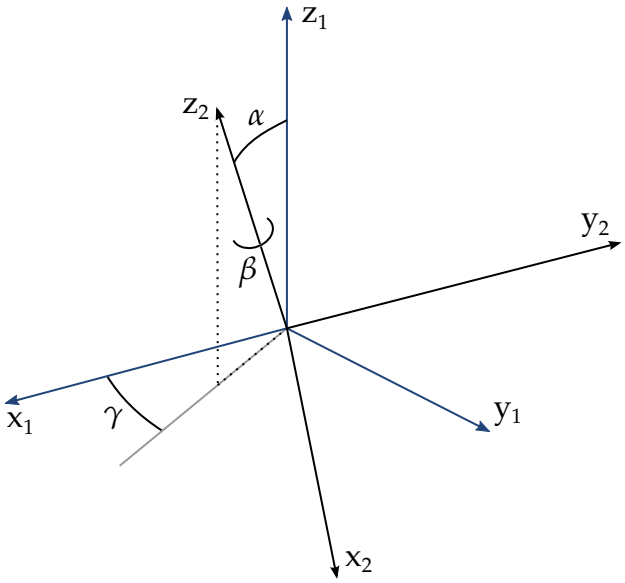


Figure 10-32. Flexion-torsion joint angles. If the initial positions of the local coordinate axes of the two rigid bodies connected by the joint do not coincide, the angles, α and γ, are initialized and torques will develop instantaneously based on the defined load curves. The angle β is also initialized but no torque will develop about the local axis on which β is measured. Rather, β will be measured relative to the computed offset.

Card 3 for FLEXION-TORSION option.

Card 2a.2	1	2	3	4	5	6	7	8
Variable	ESAL	FMAL	ESBT	FMBT				
Type	F	F/I	F	F/I				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
ESAL	Elastic stiffness per unit radian for friction and stop angles for α rotation. See Figure 10-33. EQ.0.0: Friction and stop angles are inactive for α rotation.
FMAL	Frictional moment limiting value for α rotation. This option may also be thought of as an elastic-plastic spring. See Figure 10-33. EQ.0.0: Friction is inactive for α rotation. LT.0: -FMAL is the load curve or table ID defining the yield moment as a function of α rotation. A table permits the moment to also be a function of the joint reaction force and requires the specification of JID on Card 1.
ESBT	Elastic stiffness per unit radian for friction and stop angles for β twist. EQ.0.0: Friction and stop angles are inactive for β twist.
FMBT	Frictional moment limiting value for β twist. This option may also be thought of as an elastic-plastic spring. EQ.0.0: The friction is inactive for β twist. LT.0: -FMBT is the load curve or table ID defining the yield moment versus β rotation. A table permits the moment to also be a function of the joint reaction force and requires the specification of JID on Card 1.

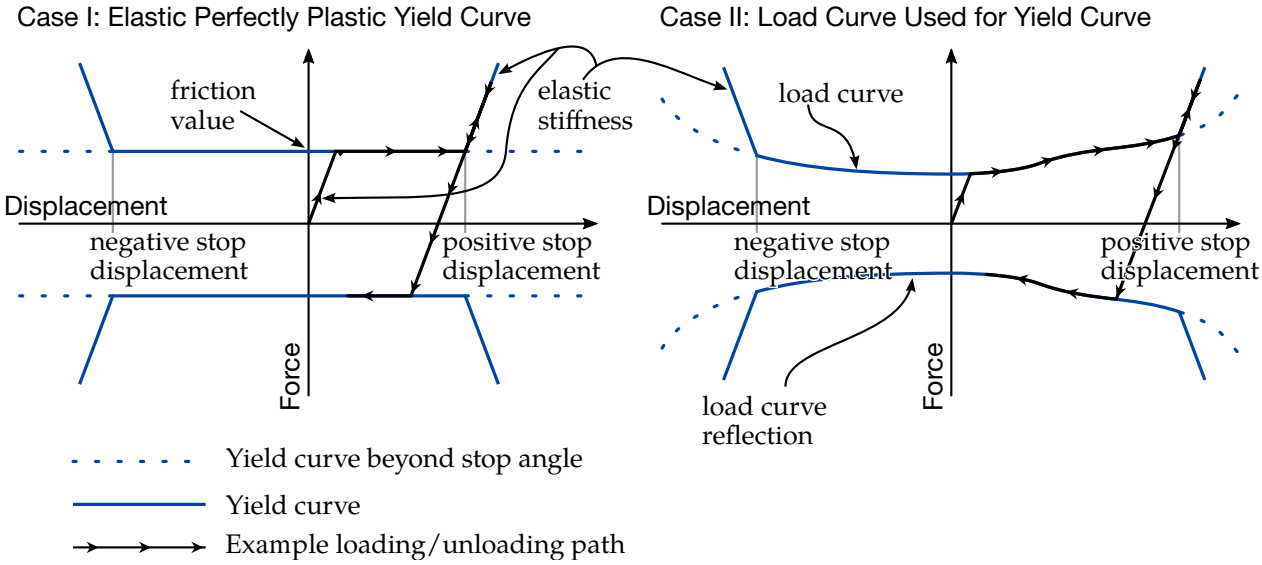


Figure 10-33. *Friction model.* The friction model is motivated by plasticity, and it is implemented for both rotational and translational joints. In the context of a rotational joint, the y-axis is to be interpreted as moment (rotational force) and the x-axis is to be interpreted as rotation. *Case I* (left) is activated by a positive friction value. *Case II* (right) is activated by a negative integer friction value, the absolute value of which specifies a load curve. *See the friction, elastic, and stop angle/displacement parameters from the input cards* (FM[var], ES[var], NSA[var], PSA[var]).

Card 4 for FLEXION-TORSION option.

Card 2a.3	1	2	3	4	5	6	7	8
Variable	SAAL	NSABT	PSABT					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
SAAL	Stop angle in degrees for α rotation where $0.0 \leq \alpha \leq \pi$. Ignored if zero. See Figure 10-33.
NSABT	Stop angle in degrees for negative β rotation. Ignored if zero.
PSABT	Stop angle in degrees for positive β rotation. Ignored if zero.

Remarks:

This option simulates the flexion-torsion behavior of a joint in a slightly different fashion than with the generalized joint option.

After the stop angles are reached, the torques increase linearly to resist further angular motion using the stiffness values on Card 2a.2. If the stiffness value is too low or zero, the stop will be violated.

The moment resultants generated from the moment as a function of rotation curve, damping moment as a function of rate-of-rotation curve, and friction are evaluated independently and are added together.

Generalized Joint Stiffness Cards:

Card 2 for GENERALIZED stiffness option.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	LCIDPH	LCIDT	LCIDPS	DLCIDPH	DLCIDT	DLCIDPS		
Type	I	I	I	I	I	I		
Default	0.0	0.0	0.0	0	0	0		

VARIABLE	DESCRIPTION
LCIDPH	Load curve ID for ϕ-moment as a function of rotation in radians. See Figure 10-34. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.
LCIDT	Load curve ID for θ-moment as a function of rotation in radians. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.
LCIDPS	Load curve ID for ψ-moment as a function of rotation in radians. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.
DLCIDPH	Load curve ID for ϕ-damping moment as a function of rate of rotation in radians per unit time. See *DEFINE_CURVE. EQ.0: Damping is not considered.
DLCIDT	Load curve ID for θ-damping moment as a function of rate of rotation in radians per unit time. See *DEFINE_CURVE. EQ.0: Damping is not considered.
DLCIDPS	Load curve ID for ψ-damping torque as a function of rate of rotation in radians per unit time. See *DEFINE_CURVE. EQ.0: Damping is not considered.

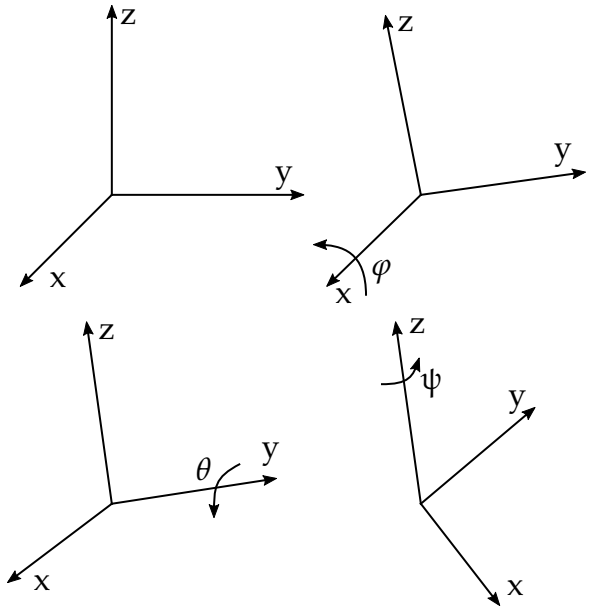


Figure 10-34. Definition of angles for the GENERALIZED joint stiffness.

Card 3 for GENERALIZED stiffness option.

Card 2b.2	1	2	3	4	5	6	7	8
Variable	ESPH	FMPH	EST	FMT	ESPS	FMPS		
Type	F	F/I	F	F/I	F	F/I		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ESPH	Elastic stiffness per unit radian for friction and stop angles for ϕ rotation. EQ.0.0: Friction and stop angles are inactive for ϕ rotation.
FMPH	Frictional moment limiting value for ϕ rotation. This option may also be thought of as an elastic-plastic spring. See Figure 10-33. EQ.0.0: Friction is inactive for ϕ rotation. LT.0: -FMPH is the load curve or table ID defining the yield moment as a function of ϕ rotation. A table permits the moment to also be a function of the joint reaction force and requires the specification of JID on Card 1.

VARIABLE	DESCRIPTION
EST	Elastic stiffness per unit radian for friction and stop angles for θ rotation. See Figure 10-33. EQ.0.0: Friction and stop angles are inactive for θ rotation.
FMT	Frictional moment limiting value for θ rotation. This option may also be thought of as an elastic-plastic spring. EQ.0.0: Friction is inactive for θ rotation. LT.0: -FMT is the load curve or table ID defining the yield moment as a function of θ rotation. A table permits the moment to also be a function of the joint reaction force and requires the specification of JID on Card 1.
ESPS	Elastic stiffness per unit radian for friction and stop angles for ψ rotation. EQ.0.0: Friction and stop angles are inactive for ψ rotation.
FMPS	Frictional moment limiting value for ψ rotation. This option may also be thought of as an elastic-plastic spring. EQ.0.0: Friction is inactive for ψ rotation. LT.0: -FMPS is the load curve or table ID defining the yield moment as a function of ψ rotation. A table permits the moment to also be a function of the joint reaction force and requires the specification of JID on Card 1.

Card 4 for GENERALIZED stiffness option.

Card 2b.3	1	2	3	4	5	6	7	8
Variable	NSAPH	PSAPH	NSAT	PSAT	NSAPS	PSAPS		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
NSAPH	Stop angle in degrees for negative ϕ rotation. See Figure 10-33. Ignored if zero.

VARIABLE	DESCRIPTION
PSAPH	Stop angle in degrees for positive ϕ rotation. Ignored if zero.
NSAT	Stop angle in degrees for negative θ rotation. Ignored if zero.
PSAT	Stop angle in degrees for positive θ rotation. Ignored if zero.
NSAPS	Stop angle in degrees for negative ψ rotation. Ignored if zero.
PSAPS	Stop angle in degrees for positive ψ rotation. Ignored if zero.

Remarks:

After the stop angles are reached, the torques increase linearly to resist further angular motion using the stiffness values on Card 2b.2. Reasonable stiffness values must be chosen. If the stiffness values are too low or zero, the stop will be violated.

If the initial local coordinate axes do not coincide, the angles, ϕ, θ, and ψ, will be initialized and torques will develop instantaneously based on the defined moment as a function of rotation curves.

There are two methods available to calculate the rotation angles between the coordinate systems. For more information, see the JNTF parameter on *CONTROL_RIGID.

Example:

```
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a joint stiffness for the revolute joint described in
$    *CONSTRAINED_JOINT_REVOLUTE
$
$  Attributes of the joint stiffness:
$    - Used for defining a stop angle of 30 degrees rotation
$        (i.e., the joint allows a positive rotation of 30 degrees and
$           then imparts an elastic stiffness to prevent further rotation)
$    - Define between rigid body A (part 1) and rigid body B (part 2)
$    - Define a local coordinate system such that local x corresponds
$         to the joint’s axis of revolution and the angle phi is the angle
$         of rotation about that axis.
$    - The elastic stiffness per unit radian for the stop angle is 100.
$    - Variables left blank are not used during the simulation.
$
*CONSTRAINED_JOINT_STIFFNESS_GENERALIZED
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     jsid      pida      pidb      cida      cidb
         1         1         2         5         5
```

```
$
$   lcidph     lcidt    lcidps   dlcidph    dlcidt   dlcidps

$
$     esph      fmps       est       fmt      esps      fmps
     100.0
$
$    nsaph     psaph      nsat      psat     nsaps     psaps
               30.0
$
$
*DEFINE_COORDINATE_NODES
$      cid        n1        n2        n3
         5         1         2         3
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

Translational Joint Stiffness Cards:

Card 2 for TRANSLATIONAL stiffness option.

Card 2c.1	1	2	3	4	5	6	7	8
Variable	LCIDX	LCIDY	LCIDZ	DLCIDX	DLCIDY	DLCIDZ		
Type	I	I	I	I	I	I		
Default	0.0	0.0	0.0	0	0	0		

VARIABLE	DESCRIPTION
LCIDX	Load curve ID for x-force as a function of x-distance between the origins of CIDA and CIDB based on the x-direction of CIDB. See *DEFINE_CURVE. EQ.0: The applied force is set to 0.0.
LCIDY	Load curve ID for y-force as a function of y-distance between the origins of CIDA and CIDB based on the y-direction of CIDB. See *DEFINE_CURVE. EQ.0: The applied force is set to 0.0.
LCIDZ	Load curve ID for z-force as a function of z-distance between the origins of CIDA and CIDB based on the z-direction of CIDB. See *DEFINE_CURVE. EQ.0: The applied force is set to 0.0.
DLCIDX	Load curve ID for x-damping force as a function of rate of x-translational displacement per unit time between the origins of CIDA and CIDB based on the x-direction of CIDB. See *DEFINE_CURVE. EQ.0: Damping is not considered.
DLCIDY	Load curve ID for y-damping force as a function of rate of y-translational displacement per unit time between the origins of CIDA and CIDB based on the y-direction of CIDB. See *DEFINE_CURVE. EQ.0: Damping is not considered.

VARIABLE	DESCRIPTION
DLCIDZ	Load curve ID for z-damping force as a function of rate of z-translational displacement per unit time between the origins of CIDA and CIDB based on the z-direction of CIDB. See *DEFINE_CURVE. EQ.0: Damping is not considered.

Card 3 TRANSLATIONAL stiffness option.

Card 2c.2	1	2	3	4	5	6	7	8
Variable	ESX	FFX	ESY	FFY	ESZ	FFZ		
Type	F	F/I	F	F/I	F	F/I		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ESX	Elastic stiffness for friction and stop displacement for x-translation. See Figure 10-33. EQ.0.0: Friction and stop angles are inactive for x-translation.
FFX	Frictional force limiting value for x-translation. This option may also be thought of as an elastic-plastic spring. See Figure 10-33. EQ.0.0: Friction is inactive for x-translation. LT.0: -FFX is the load curve ID defining the yield force as a function x-translation.
ESY	Elastic stiffness for friction and stop displacement for y-translation. EQ.0.0: Friction and stop angles are inactive for y-translation.
FFY	Frictional force limiting value for y-translation. This option may also be thought of as an elastic-plastic spring. EQ.0.0: Friction is inactive for y-translation. LT.0: -FFY is the load curve ID defining the yield force as a function of y-translation.

VARIABLE	DESCRIPTION
ESZ	Elastic stiffness for friction and stop displacement for z-translation. EQ.0.0: Friction and stop angles are inactive for z-translation.
FFZ	Frictional force limiting value for z-translation. This option may also be thought of as an elastic-plastic spring. EQ.0.0: Friction is inactive for z-translation. LT.0: -FFZ is the load curve ID defining the yield force as a function of z-translation.

Card 4 for TRANSLATIONAL stiffness option.

Card 2c.3	1	2	3	4	5	6	7	8
Variable	NSDX	PSDX	NSDY	PSDY	NSDZ	PSDZ	FS	FD
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
NSDX	Stop displacement for negative x-translation. Ignored if zero. See Figure 10-33.
PSDX	Stop displacement for positive x-translation. Ignored if zero.
NSDY	Stop displacement for negative y-translation. Ignored if zero.
PSDY	Stop displacement for positive y-translation. Ignored if zero.
NSDZ	Stop displacement for negative z-translation. Ignored if zero.
PSDZ	Stop displacement for positive z-translation. Ignored if zero.
FS	Static friction coefficient
FD	Dynamic friction coefficient

Remarks:

After the stop displacements are reached, the force increases linearly to resist further translational motion using the stiffness values on Card 2c.2. Reasonable stiffness values must be chosen. If the stiffness values are too low or zero, the stop will be violated.

Cylindrical Joint Stiffness Cards:

Card 2 for CYLINDRICAL stiffness option.

Card 2d.1	1	2	3	4	5	6	7	8
Variable	LCIDR		LCIDZ	DLCIDR	DLCIDP	DLCIDZ	LCIDT	DLCIDT
Type	I		I	I	I	I	I	I
Default	0		0	0	0	0	0	0

VARIABLE	DESCRIPTION
LCIDR	Load curve ID for *r*-force as a function of *r*-distance between the origins of CIDA and CIDB. See *DEFINE_CURVE. EQ.0: The applied force is set to 0.0.
LCIDZ	Load curve ID for *z*-force as a function of *z*-distance between the origins of CIDA and CIDB. See *DEFINE_CURVE. EQ.0: The applied force is set to 0.0.
DLCIDR	Load curve or table ID for *r*-damping force as a function of rate of *r*-distance per unit time and optionally *r*-distance (if table) between the origins of CIDA and CIDB. See *DEFINE_CURVE or *DEFINE_TABLE. EQ.0: Damping is not considered.
DLCIDP	Load curve or table ID for *p*-damping force as a function of rate of *p*-distance per unit time and optionally *r*-distance (if table) between the origins of CIDA and CIDB. See *DEFINE_CURVE or *DEFINE_TABLE. EQ.0: Damping is not considered.
DLCIDZ	Load curve or table ID for *z*-damping force as a function of rate of *z*-distance per unit time and optionally *r*-distance (if table) between the origins of CIDA and CIDB. See *DEFINE_CURVE or *DEFINE_TABLE. EQ.0: Damping is not considered.

VARIABLE	DESCRIPTION
LCIDT	Load curve ID for θ-moment as a function of angle θ between the z-directions of CIDA and CIDB. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.
DLCIDT	Load curve ID for θ-moment as a function of rate of angle θ between the z-directions of CIDA and CIDB. See *DEFINE_CURVE. EQ.0: The applied moment is set to 0.0.

Card 3 CYLINDRICAL stiffness option.

Card 2d.2	1	2	3	4	5	6	7	8
Variable	ESR	FFR			ESZ	FFZ	RAD1	RAD2
Type	F	F/I			F	F/I	F	F
Default	0.0	0.0			0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
ESR	Elastic stiffness for friction and stop displacement for r-translation. See Figure 10-33. EQ.0.0: Friction and stop angles are inactive for r-translation.
FFR	Frictional force limiting value for r-translation. This option may also be thought of as an elastic-plastic spring. See Figure 10-33. EQ.0.0: Friction is inactive for r-translation. LT.0: -FFR is the load curve ID defining the yield force as a function r-translation.
ESZ	Elastic stiffness for friction and stop displacement for z-translation. EQ.0.0: Friction and stop angles are inactive for z-translation.
FFZ	Frictional force limiting value for z-translation. This option may also be thought of as an elastic-plastic spring. EQ.0.0: Friction is inactive for z-translation.

VARIABLE	DESCRIPTION
	LT.0: -FFZ is the load curve ID defining the yield force as a function of z-translation.
RAD1	Radius of pin, must be strictly positive.
RAD2	Radius of hole, must be strictly larger than RAD1 to model play in the connection.

Card 4 for CYLINDRICAL stiffness option.

Card 2d.3	1	2	3	4	5	6	7	8
Variable		PSDR			NSDZ	PSDZ	FS	FD
Type		F			F	F	F	F
Default		0.0			0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PSDR	Stop displacement for r-translation. Ignored if zero.
NSDZ	Stop displacement for negative z-translation. Ignored if zero.
PSDZ	Stop displacement for positive z-translation. Ignored if zero.
FS	Static friction coefficient
FD	Dynamic friction coefficient

Remarks:

As shown in Figure 10-35, the cylindrical joint stiffness models a pin in a hole with a gap. The pin has a radius r_1 (RAD1) while the hole has a radius r_2 (RAD2), with $0 < r_1 < r_2$, so the gap is at most $g = r_2 - r_1$. This joint stiffness needs no associated joint, but all directions can be constrained by appropriate usage of this keyword.

Let $\mathbf{x}_i$ be the origin of coordinate system associated with body i, $i = 1, 2$, defined by *DEFINE_COORDINATE_NODES. Then the radial distance to be used in the curves/tables is defined as

$$r = \left| \left(\mathbf{I} - \mathbf{z}\mathbf{z}^T \right) (\mathbf{x}_2 - \mathbf{x}_1) \right| \ ,$$

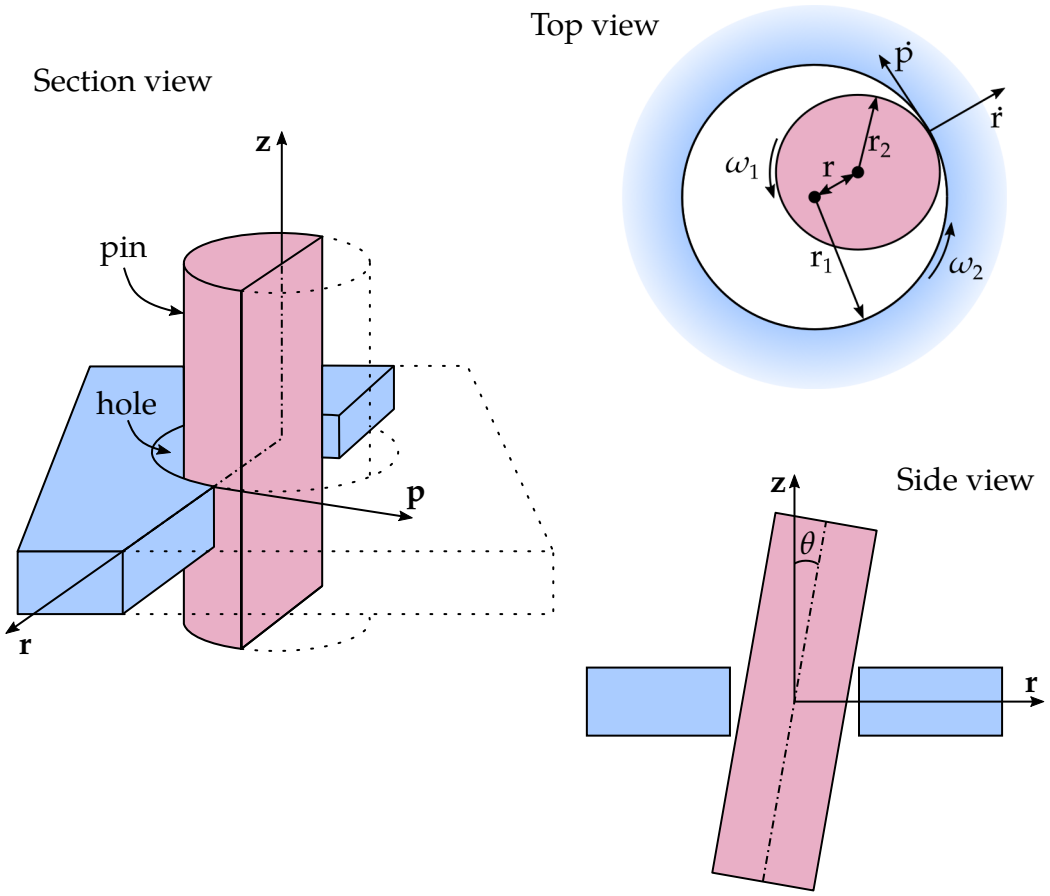


Figure 10-35. Cylindrical joint stiffness models a pin in a hole.

and the radial direction is defined as

$$\mathbf{r} = \frac{\left(\mathbf{I} - \mathbf{z}\mathbf{z}^T\right)(\mathbf{x}_2 - \mathbf{x}_1)}{r} \ .$$

See LCIDR, ESR, FFR and PSDR. Here $\mathbf{I}$ is the unit matrix, and $\mathbf{z}$ is the local z-direction of the coordinate system in body 2. We basically assign a cylindrical coordinate system $\{\mathbf{r}, \mathbf{p}, \mathbf{z}\}$ with the z-direction co-linear with the cartesian z-direction in body 2. The radial distance is thus the (non-directional) distance between body 1 and body 2 in the local xy-plane of body 2.

Likewise, we define the z-distance to be used in the curves/tables as

$$z = \left|\mathbf{z}^T(\mathbf{x}_2 - \mathbf{x}_1)\right| \ ,$$

which is the *directional* distance from body 1 to body 2 in the z-direction. See LCIDZ, ESZ, FFZ, NSDZ and PSDZ.

From this, the rate of r and z are obtained by taking the time derivative quantities, $\dot{r}$ and $\dot{z}$, that are also used in curves/tables for damping purposes; see DLCIDR and DLCIDZ. If these are defined by tables, the second argument is the radial distance, r, which can be used to not apply any damping when the gap is positive. For numerical purposes, and

for the sake of introducing a hysteresis, $\dot{r}$ is always positive, so DLCIDR only needs to be defined for positive $\dot{r}$.

Last, we define $\mathbf{p}$ as the cross production of $\mathbf{z}$ and $\mathbf{r}$, that is,

$$\mathbf{p} = \mathbf{z} \times \mathbf{r}$$

This is the tangential direction along the circumference of the pin or the hole, depending on how you observe it. The tangential velocity is defined as

$$\dot{p} = \mathbf{p}^T(\dot{\mathbf{x}}_2 - (r_1 + r)\boldsymbol{\omega}_2 \times \mathbf{r} - \dot{\mathbf{x}}_1 + r_1\boldsymbol{\omega}_1 \times \mathbf{r}) \ ,$$

where $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ are the rotational velocities of bodies 1 and 2, respectively. The interpretation of this is the relative velocity of the contact point between the hole (body 2) and the pin (body 1) in direction $\mathbf{p}$ when the gap is zero. This can be used to define damping in the p-direction (see DLCIDP) as a type of simplified friction model. Also, here a table can be used with r as the second argument.

To maintain co-linearity of the z-directions of the pin and hole, the angle θ is defined as illustrated in the picture. This quantity, and its time derivative $\dot{\theta}$, can be used to enforce this co-linearity; see LCIDT and DLCIDT. For numerical purposes, and for the sake of introducing hysteresis, $\dot{\theta}$ is always positive so DLCIDT only needs to be defined for positive $\dot{\theta}$.

*CONSTRAINED_JOINT_USER_FORCE

Purpose: Define input data for a user subroutine to generate force resultants as a function of time and joint motion.

Card 1	1	2	3	4	5	6	7	8
Variable	FID	JID	NHISV					
Type	I	I	I					
Default	none	none	0					

User Subroutine Constants Cards. Define up to 48 optional user constants (6 cards total) for the user subroutine. This input is terminated after 48 constants are defined or when the next keyword ("*") card is encountered.

Card 2	1	2	3	4	5	6	7	8
Variable	CONST1	CONST2	CONST3	CONST4	CONST5	CONST6	CONST7	CONST8
Type	F	F	F	F	F	I	I	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
FID	Joint user force ID.
JID	Joint ID for which this user force input applies.
NHISV	Number of history variables required for this definition. An array NHISV long is allocated and passed into the user subroutine. This array is updated in the user subroutine.
CONST*n*	A constant which is passed into the user subroutine.

*CONSTRAINED_LAGRANGE_IN_SOLID_{OPTION1}_{OPTION2}

Purpose: This command provides the coupling mechanism for modeling Fluid-Structure Interaction (FSI). The structure can be constructed from Lagrangian shell and/or solid entities. The multi-material fluids are modeled by ALE formulation.

Available options for *OPTION1* include:

<BLANK>

EDGE

This option may be used to allow the coupling between the edge of a shell part or part set and one or more ALE multi-material groups (AMMG). It accounts for the shell thickness in the coupling calculation. The edge thickness is the same as the shell thickness. This option only works when the Lagrangian set is defined as a part or a part set ID. It will not work for a segment set. One application of this option is a simulation of a Lagrangian blade (a shell part) cutting through some ALE material.

Available options for *OPTION2* include:

<BLANK>

TITLE

To define a coupling (card) ID number and title for each coupling card. If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition. The ID number can be used to delete coupling action in a restart input deck using the *DELETE_FSI card.

Card Summary:

Card ID. This card is included if and only if the TITLE keyword option is used.

COUPID	TITLE

Card 1. This card is required.

LSTRSID	ALESID	LSTRSTYP	ALESTYP	NQUAD	CTYPE	DIREC	MCOUP

Card 2. This card is required.

START	END	PFAC	FRIC	FRCMIN	NORM	NORMTYP	DAMP

Card 3. This card is required.

K	HMIN	HMAX	ILEAK	PLEAK	LCIDPOR	NVENT	IBLOCK

Card 4. This card is required for CTYPE = 11 and 12. Otherwise it is optional. If Card 5 is included, then this card must be included.

IBOXID	IPENCHK	INTFORC	IALESOF	LAGMUL	PFACMM	THKF	

Card 5. This card is required for CTYPE = 11 and 12. Otherwise it is optional. If this card is included, then Card 4 must be included.

A1	B1	A2	B2	A3	B3		POREINI

Card 6. NVENT of this card must be defined (one card for defining each vent hole). The last NVENT cards for *CONSTRAINED_LAGRANGE_IN_SOLID are assumed to be Card(s) 6; therefore, Cards 4 and 5 are not mandatory when Card(s) 6 are defined.

VENTSID	VENTYP	VTCOEF	POPPRES	COEFLC			

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number. If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70).

Card 1	1	2	3	4	5	6	7	8
Variable	LSTRSID	ALESID	LSTRSTYP	ALESTYP	NQUAD	CTYPE	DIREC	MCOUP
Type	I	I	I	I	I	I	I	I
Default	none	none	0	0	0	2	1	0

VARIABLE	DESCRIPTION
LSTRSID	Set ID defining a part, part set, or segment set ID of the Lagrangian structure (see *PART, *SET_PART or *SET_SEGMENT). See Remark 1.
ALESID	Set ID defining a part or part set ID of the ALE solid elements (see *PART or *SET_PART). See Remark 1.
LSTRSTYP	LSTRSID set type (see Remark 1): EQ.0: Part set ID (PSID) EQ.1: Part ID (PID) EQ.2: Segment set ID (SSID)
ALESTYP	ALESID set type (see Remark 1): EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
NQUAD	Number of coupling points distributed over each coupled Lagrangian surface segment. See Remark 2. EQ.0: NQUAD will be set by default to 2, GT.0: An NQUAD × NQUAD coupling points distribution over each Lagrangian segment is defined, LT.0: NQUAD is reset to a positive value. Coupling at nodes is obsolete.
CTYPE	Fluid-Structure coupling method. CTYPEs 1 and 2 are not supported in MPP. EQ.1: Constrained acceleration EQ.2: Constrained acceleration and velocity (default, see Remark 3) EQ.3: Constrained acceleration and velocity, normal direction only EQ.4: Penalty coupling for shell and solid elements (without erosion)

NOTE: For RIGID Lagrangian structure PARTS a penalty coupling method (CTYPE = 4) must be used.

VARIABLE	DESCRIPTION
	EQ.5: Penalty coupling allowing erosion in the Lagrangian entities EQ.6: Penalty coupling designed for airbag modeling which automatically controls the DIREC parameter internally. It is equivalent to setting {CTYPE = 4; DIREC = 1} for the unfolded region; and {CTYPE = 4; DIREC = 2} for the folded region. For both cases: {ILEAK = 2; FRCMIN = 0.3}. EQ.11: Coupling designed to couple Lagrangian porous shell to ALE material. When this option is used, THKF, the 7th column parameter of optional Card 4a and the first 2 parameters of optional Card 4b must be defined. See *LOAD_BODY_POROUS and Remark 14 below. EQ.12: Coupling designed to couple Lagrangian porous solid to ALE material. When this option is used, Ai & Bi parameters of optional Card 4b must be defined (Card 4a must be defined but can be blank). See *LOAD_BODY_POROUS and Remark 15 below.
DIREC	For CTYPE = 4 or 5 Coupling direction (see Remark 4): EQ.1: Normal direction, compression, and tension (default) EQ.2: Normal direction, compression only EQ.3: All directions For CTYPE = 12 Flag to activate an element coordinate system: EQ.0: The forces are applied in the global directions. EQ.1: The forces are applied in a local system attached to the Lagrangian solid. The system is consistent with AOPT = 1 in *LOAD_BODY_POROUS. (See Remark 15).
MCOUP	For CTYPE = 4, 5, 6, 11, or 12 Multi-material option (see Remark 5): EQ.0: Couple with all multi-material groups EQ.1: Couple with material with highest density

VARIABLE	DESCRIPTION
	LT.0: MCOUP must be an integer. -MCOUP refers to a set ID of an ALE multi-material group. See *SET_MULTI-MATERIAL_GROUP.

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	PFAC	FRIC	FRCMIN	NORM	NORMTYP	DAMP
Type	F	F	F	F	F	I	I	F
Default	0.0	10^{10}	0.1	0.0	0.5	0	0	0.0

VARIABLE	DESCRIPTION
START	Start time for coupling.
END	End time for coupling. If less than zero, coupling will be turned off during dynamic relaxation. After dynamic relaxation phase is finished, the absolute value will be taken as end time.
PFAC	For CTYPE = 4,5 or 6 Penalty factor. PFAC is a scale factor for scaling the estimated stiffness of the interacting (coupling) system. It is used to compute the coupling forces to be distributed on the Lagrangian and ALE parts GT.0: Fraction of estimated critical stiffness. LT.0: PFAC must be an integer, and -PFAC is a load curve ID. The curve defines the coupling pressure on the *y*-axis as a function of the penetration along the *x*-axis. See Remark 6. See "How to correct leakage." For CTYPE = 11 or 12 Time step factor
FRIC	Coefficient of friction (used with DIREC = 1 and 2 only).
FRCMIN	Minimum volume fraction of a coupled ALE multi-material group (AMMG) or fluid in a multi-material ALE element to activate coupling. Default value is 0.5. Reducing FRCMIN (typically, between 0.1 and 0.3) would turn on coupling earlier to prevent leakage in high velocity impact cases.

VARIABLE	DESCRIPTION
NORM	A Lagrangian segment will couple to fluid on only one side of the segment. NORM determines which side. See Remark 7. EQ.0: Couple to fluid (AMMG) on head-side of Lagrangian segment normal vector. EQ.1: Couple to fluid (AMMG) on tail-side of Lagrangian segment normal vector.
NORMTYP	Penalty coupling spring (or force) direction (DIREC = 1 or 2): EQ.0: Normal vectors are interpolated from nodal normals. (default). EQ.1: Normal vectors are interpolated from segment normals. This is sometimes a little more robust for sharp Lagrangian corners, and folds.
DAMP	Damping factor for penalty coupling. This is a coupling-damping scaling factor. Typically, it may be between 0.0 and 1.0 (see Remark 8).

Card 3	1	2	3	4	5	6	7	8
Variable	K	HMIN	HMAX	ILEAK	PLEAK	LCIDPOR	NVENT	IBLOCK
Type	F	F	F	I	F	I	I	I
Default	0.0	none	none	0	0.1	0	0	0

VARIABLE	DESCRIPTION
K	Thermal conductivity of a virtual fluid between the Lagrangian structure surface and the ALE material. See Remark 9.
HMIN	The absolute value is minimum air gap in heat transfer, $h_{\min}$ (See Remark 9). LT.0.0: Turn on constraint based thermal nodal coupling between LAG structure and ALE fluids. GE.0.0: Minimum air gap. If zero, default to 10^{-6}.

VARIABLE	DESCRIPTION
HMAX	Maximum air gap in heat transfer, h_{max}. There is no heat transfer above this value. See Remark 9.
ILEAK	Coupling leakage control flag (Remark 10): EQ.0: None (default), EQ.1: Weak, leakage control is turned off if penetrating volume fraction > FRCMIN + 0.2 EQ.2: Strong, with improved energy consideration. Leakage control is turned off if penetrating volume fraction > FRCMIN + 0.4
PLEAK	Leakage control penalty factor, 0 < PLEAK < 0.2 is recommended. This factor influences the additional coupling force magnitude to prevent leakage. It is conceptually similar to PFAC. Almost always, the default value (0.1) is adequate.
LCIDPOR	If this is a positive integer, a load curve ID (LCID) defining porous flow through coupling segment: Abscissa = x = ($P_{up} - P_{down}$) Ordinate = y = relative porous fluid velocity Where P_{up} and P_{down} are, respectively, the upstream and downstream pressures across of the porous coupling segment. The relative porous velocity is the ALE fluid velocity relative to the moving Lagrangian segment. This experimental data curve must be provided by the user. If LCIDPOR is a negative integer: The porous flow is controlled by the parameters FLC, FAC, ELA under *MAT_FABRIC card. **CAUTION:** The pressure under the FAC load curve is "absolute upstream pressure" (see Remark 11). Abscissa = x = absolute upstream pressure Ordinate = y = relative porous fluid velocity For CTYPE = 11 or CTYPE = 12 and POREINI = 0.0: LT.0: The load curve |LCIDPOR| is a factor versus time of the porous force computed by the Ergun equation. See Remark 14.

VARIABLE	DESCRIPTION
	GT.0: The load curve LCIDPOR is a porous force versus velocity, which replaces the force computed by the Ergun equation. See Remark 14.
	For CTYPE = 11 or CTYPE = 12 and POREINI > 0.0:
	NE.0: The load curve |LCIDPOR| is a factor versus time of the porous force computed by the Ergun equation. See Remark 14.
NVENT	The number of vent surface areas to be defined. Each venting flow surface is represented by one or more Lagrangian segments (or surfaces). For airbag applications, this may be referred to as "isentropic" venting where the isentropic flow equation is used to compute the mass flow rate based on the ratio of the upstream and downstream pressures P_{up}/P_{down}. For each of the NVENT vent surfaces, an additional card of format 6 defining the geometrical and flow properties for each vent surface will be read in. The vented mass will simply be deleted from the system and cannot be visualized as in the case of physical venting. See Remark 12.
IBLOCK	Flag to control the venting (or porous) flow blockage due to Lagrangian contact during ALE computation. EQ.0: Off EQ.1: On The venting definition is defined in this command. However, the venting flow may be defined through either the LCIDPOR parameter in this command or via the *MAT_FABRIC parameters (FLC, FAC, ELA). However, note that FVOPT (blocking) parameter under *MAT_FABRIC applies only to CV computation.

Card 4. This card is required for CTYPE 11 & 12 but is otherwise optional.

Card 4	1	2	3	4	5	6	7	8
Variable	IBOXID	IPENCHK	INTFORC	IALESOF	LAGMUL	PFACMM	THKF	
Type	I	I	I	I	F	I	F	
Default	0	0	0	0	0.0	0	0.0	

VARIABLE	DESCRIPTION
IBOXID	A box ID defining a box region in space in which ALE coupling is activated. GT.0: At time = 0.0, the Lagrangian segments inside this box are remembered. In subsequent coupling computation steps, there is no need to search for the Lagrangian segments again. LT.0: At each FSI bucket sort, the Lagrangian segments inside this box are marked as active coupling segments. This makes the coupling operate more efficiently when a structured mesh is approaching ALE domain, such as hydroplaning and bird strike.
IPENCHK	Only for CTYPE = 4 Initial penetration check flag (See Remark 13): EQ.0: Do not check for initial penetration. EQ.1: Check and save initial ALE material penetration across a Lagrangian surface (d_0), but do not activate coupling at $t = 0$. In subsequent steps ($t > 0$) the actual penetration is computed as follows: $\underbrace{\text{Actual Penetration}}_{d_a} = \underbrace{\text{Total Penetration}}_{d_T} - \underbrace{\text{Initial Penetration}}_{d_0}$
INTFORC	A flag to turn on or off the output of ALE coupling pressure and forces on the Lagrangian segments (or surfaces). EQ.0: Off EQ.1: On

VARIABLE	DESCRIPTION
	Note that the coupling pressures and forces are computed based on the coupling stiffness response to the ALE fluid penetration. When INFORC = 1 and a *DATABASE_BINARY_FSIFOR (DBF) card is defined, LS-DYNA writes out the segment coupling pressure and forces to the binary interface force file for contour plotting. This interface force file must be given a name on the execution line, for example: ls-dyna i=**inputfilename.k** … h=**interfaceforcefilename** The time interval between output is defined by "dt" in the DBF card. To plot the binary data in this file: ls-prepost **interfaceforcefilename**
IALESOF	An integer flag to turn ON/OFF a supplemental Lagrange multiplier FSI constraint which provides a coupling force in addition to the basic penalty coupling contribution. This is a hybrid coupling method. EQ.0: OFF (default). EQ.1: Turn ON the hybrid Lagrange-multiplier method. LAGMUL multiplier factor is read.
LAGMUL	A Lagrange multiplier factor with a range between 0.0 and 0.05 may be defined. A typical value may be 0.01. This should never be greater than 0.1. EQ.0: OFF (default). GT.0: Turn ON the Lagrange-multiplier method and use LAGMUL as a coefficient for scaling the penalty factor.
PFACMM	Mass-based penalty stiffness factor computational options. This works in conjunction with PFAC = constant (not a load curve). The coupling penalty stiffness (CPS) is computed based on an estimated effective coupling mass. EQ.0: CPS $\propto$ PFAC $\times \min(m_{\text{Lagrangian}}, m_{\text{ALE}})$, default. EQ.1: CPS $\propto$ PFAC $\times \max(m_{\text{Lagrangian}}, m_{\text{ALE}})$. EQ.2: CPS $\propto$ PFAC $\times \sqrt{m_{\text{Lagrangian}} m_{\text{ALE}}}$, geometric-mean of the masses.

VARIABLE	DESCRIPTION
	EQ.3: CPS $\propto$ PFAC $\times K_{\text{Lagrangian}}$ where K is the bulk modulus of the Lagrangian part
THKF	For all CTYPE choices except 11: A flag to account for the coupling thickness of the Lagrangian shell part. LT.0: Use positive value of |THKF| for coupling segment thickness. EQ.0: Do not consider coupling segment thickness. GT.0: Coupling segment thickness scale factor. For CTYPE = 11: This thickness is required for volume calculation. GT.0: (Fabric) Thickness scale factor. The base shell thickness is taken from the *PART definition. LT.0: User-defined (Fabric) thickness. The fabric thickness is set to |THKF|.

Porous Coupling. This card applies only to CTYPE 11 & 12. If this card is defined, Card 4 must be defined before this card.

Card 5	1	2	3	4	5	6	7	8
Variable	A1	B1	A2	B2	A3	B3		PORFINI
Type	F	F	F	F	F	F		F
Default	0.0	0.0	0.0	0.0	0.0	0.0		0.0

VARIABLE	DESCRIPTION
A1	Viscous coefficient for the porous flow Ergun equation (see Remark 14). GT.0: For CTYPE = 11 $A1 = A_n$ which is the coefficient for normal-to-segment direction.

VARIABLE	DESCRIPTION
	For CTYPE = 12 $A1 = A_x$ which is the coefficient for the x-direction in the coordinate system specified by DIREC. LT.0: If POREINI = 0.0, the coefficient is time dependent through a load curve ID defined by |A1|. If POREINI > 0.0, the coefficient is porosity dependent through a load curve ID defined by |A1|. The porosity is defined by PORE (see POREINI).
B1	Inertial coefficient for the porous flow Ergun equation (see Remark 14). GT.0: For CTYPE = 11 $B1 = B_n$ which is the coefficient for normal-to-segment direction. For CTYPE = 12 $B1 = B_x$ which is the coefficient for the x-direction of a coordinate system specified by DIREC. LT.0: If POREINI = 0.0, the coefficient is time dependent through a load curve id defined by |B1|. If POREINI > 0.0, the coefficient is porosity dependent through a load curve ID defined by |B1|. The porosity is defined by PORE (see POREINI).
A2	For CTYPE = 12 Viscous coefficient for the porous flow Ergun equation (see Remark 15). GT.0: Coefficient for the y-direction of a coordinate systems specified by DIREC. $A2 = A_y$ LT.0: If POREINI = 0.0, the coefficient is time dependent through a load curve ID defined by |A1|. If POREINI > 0.0, the coefficient is porosity dependent through a load curve ID defined by |A2|. The porosity is defined by PORE (see POREINI).

VARIABLE	DESCRIPTION
B2	For CTYPE = 12 Inertial coefficient for the porous flow Ergun equation (see Remark 15). GT.0: Coefficient for the y-direction of a coordinate system specified by DIREC. $$B2 = B_y$$ LT.0: If POREINI = 0.0 and B2 < 0, the coefficient is time dependent through a load curve ID defined by |B2|. If POREINI > 0.0 and B2 < 0, the coefficient is porosity dependent through a load curve ID defined by |B2|. The porosity is defined by PORE (see POREINI).
A3	For CTYPE = 12 Viscous coefficient for the porous flow Ergun equation (see Remark 15). GT.0: Coefficient for the z-direction of a coordinate system specified by DIREC. $$A3 = A_z$$ LT.0: If POREINI = 0.0 and A3 < 0, the coefficient is time dependent through a load curve ID defined by |A3|. If POREINI > 0.0 and A3 < 0, the coefficient is porosity dependent through a load curve ID defined by |A3|. The porosity is defined by PORE (see POREINI).
B3	For CTYPE = 12 Inertial coefficient for the porous flow Ergun equation (see Remark 15). GT.0: Coefficient for the z-direction of a coordinate system specified by DIREC. $$B3 = B_z$$ LT.0: If POREINI = 0.0 and B3 < 0, the coefficient is time dependent through a load curve ID defined by |B3|. If POREINI > 0.0 and B3 < 0, the coefficient is porosity dependent through a load curve ID defined by |B3|. The porosity is defined by PORE (see POREINI).

VARIABLE	DESCRIPTION
POREINI	For CTYPE = 11 or CTYPE = 12 POREINI is the initial volume of pores in an element. The current volume is $$\text{PORE} = \text{POREINI} \times \frac{v(t)}{v(t_0)}$$ where $v(t)$ and $v(t_0)$ are the current and initial element volumes respectively.

Venting Geometry Card(s). These card(s) set venting geometry. It is repeated NVENT times (one card for defining each vent hole). It is defined only if NVENT > 0 on Card 3. The last NVENT cards for *CONSTRAINED_LAGRANGE_IN_SOLID are taken to be Card(s) 6; therefore, Cards 4 and 5 are not mandatory when Card(s) 6 are defined.

Card 6	1	2	3	4	5	6	7	8
Variable	VENTSID	VENTYP	VTCOEF	POPPRES	COEFLC			
Type	I	I	I	F	I			
Default	0	0	0	0.0	0			

VARIABLE	DESCRIPTION
VENTSID	Set ID of the vent hole shape.
VENTYP	Vent surface area set ID type: EQ.0: Part set ID (PSID). EQ.1: Part ID (PID). EQ.2: Segment set ID (SGSID).
VTCOEF	Flow coefficient for each vent surface area.
POPPRES	Venting pop pressure limit. If the pressure inside the airbag is lower than this pressure, then nothing is vented. Only when the pressure inside the airbag is greater than POPPRES that venting can begin.
COEFLC	A time-dependent multiplier load curve for correcting the vent flow coefficient, with values ranging from 0.0 to 1.0.

Guidelines:

Due to the complexity of this card, some comments on simple, efficient and robust coupling approach are given here. These are not rigid guidelines, but simply some experience-based observations.

1. **Definition (fluid and structure).** The term *fluid*, in the Fluid-Structure Interaction (FSI), refers to materials with ALE element formulation, not indicating the phase (solid, liquid or gas) of those materials. In fact, solid, liquid, and gas can all be modeled by the ALE formulation. The term *structure* refers to materials with Lagrangian element formulation.

2. **Default values (CTYPE and MCOUP).** In general, penalty coupling (CTYPE 4 & 5) is recommended, and MCOUP = negative integer is a better choice to define a specific ALE multi-material group (AMMG) to be coupled to the Lagrangian surface. At the minimum, all parameters on Card 1 are to be specified, and the default values for most are good starting choices (except MCOUP).

3. **How to correct leakage.** If there is leakage, PFAC, FRCMIN, NORMTYPE, and ILEAK are the 4 parameters that can be adjusted.

   a) For hard structure (steel) and very compressible fluid (air), PFAC may be set to 0.1 (or higher). PFAC = constant value.

   b) Next, keeping PFAC = constant, set PFACMM = 3 (optional Card 4a). This option scales the penalty factor by the bulk modulus of the Lagrangian structure. This new approach has also shown to be effective for some airbag applications.

   c) The next approach may be switching from constant PFAC to a load curve approach, that is, PFAC = load curve and PFACMM = 0. By looking at the pressure in the system near leakage original location, we can get a feel for the pressure required to stop it.

   d) If leakage persists after some iterations on the coupling force controls, one can subsequently try to set ILEAK = 2 in combination with the other controls to prevent leakage.

   e) If the modifications fail to stop the leakage, maybe the meshes have to be redesigned to allow better interactions between the Lagrangian and Ale materials.

   In the example below, the underlined parameters are usually defined parameters. A full card definition is shown for reference.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*CONSTRAINED_LAGRANGE_IN_SOLID
$  LSTRSID    ALESID  LSTRSTYP   ALESTYP     NQUAD     CTYPE     DIREC     MCOUP
         1        11         0         0         4         4         2      -123
```

```
$     START       END      PFAC      FRIC     FRCMIN      NORM  NORMTYPE      DAMP
        0.0       0.0       0.1      0.00        0.3         0         0       0.0
$        CQ      HMIN      HMAX     ILEAK     PLEAK   LCIDPOR     NVENT    IBLOCK
         0         0         0         0       0.0         0         0         0
$4A IBOXID   IPENCHK   INTFORC   IALESOF    LAGMUL    PFACMM      THKF
$        0         0         0         0         0         0         0
$4B     A1        B1        A2        B2        A3        B3
$      0.0       0.0       0.0       0.0       0.0       0.0
$4C VNTSID   VENTYPE  VENTCOEF   POPPRES  COEFLCID  (STYPE:0=PSID;1=PID;2=SGSID)
$        0         0         0       0.0         0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

Remarks:

1. **Meshing.** In order for a fluid-structure interaction (FSI) to occur, a Lagrangian (structure) mesh must spatially overlap with an ALE (fluid) mesh. Each mesh should be defined with independent node IDs. LS-DYNA searches for the spatial intersection of between the Lagrangian and ALE meshes. Where the meshes overlap, there is a possibility that interaction may occur. In general, LSTRSID, ALESID, LSTRSTYP and ALESTYP are required definitions for specifying overlapping-domains coupling search.

2. **Number of coupling points.** The number of coupling points, NQUAD × NQUAD, is distributed over the surface of each Lagrangian segment. Generally, 2 or 3 coupling points per each Eulerian/ALE element width is adequate. Consequently, the appropriate NQUAD values must be estimated based on the *relative resolutions* between the Lagrangian and ALE meshes.

   For example, if one Lagrangian shell element spans two ALE elements, then NQUAD for each Lagrangian segment should be 4 or 6. Alternatively, if two or three Lagrangian segments span one ALE element, then maybe NQUAD = 1 would be adequate.

   If either mesh compresses or expands during the interaction, the number of coupling points per ALE element will also change. The user must account for this and try to maintain at least two coupling points per each ALE element side length during the whole process to prevent leakage. Too many coupling points can result in instability, and not enough can result in leakage.

3. **The constraint method.** The constraint method violates kinetic energy balance. The penalty method is therefore recommended. Historically, CTYPE = 2 was sometimes used to couple Lagrangian beam nodes to ALE or Lagrangian solids, such as for modeling rebar in concrete or tire cords in rubber. For such constraint-based coupling of beams in solids, *CONSTRAINED_BEAM_IN_SOLID and *DEFINE_BEAM_SOLID_COUPLING are now preferred.

4. **Coupling direction.** DIREC = 2 (compression only) may be generally a more stable and robust choice for coupling direction. However, the physics of the problem should dictate the coupling direction. DIREC = 1 couples under both

tension and compression. This is sometimes useful; for example, in the case of a suddenly accelerating liquid in a container. DIREC = 3 is rarely appropriate because it models an extremely sticky fluid.

5. **Multi-material coupling option.** When MCOUP is a negative integer, such as MCOUP = -123, then an ALE multi-material set-ID (AMMSID) of 123 must exist. This is an ID defined by a *SET_MULTI-MATERIAL_GROUP_LIST card. This generally seems to be a better approach to couple to a specific set of AMMGs, and have a clearly defined fluid interface interacting with a Lagrangian surface. That way, any leakage may be visualized and the penalty force can be computed more precisely.

   The *Couple to all materials* option as activated by MCOUP = 0 is generally not recommended. LS-DYNA calculates the fluid coupling interface as the surface where the sum of coupled ALE materials occupies a volume fraction equal to 50%. Since MCOUP = 0 couples to all materials, the sum of all coupled ALE materials is, in this case, trivially 100%. Consequently, when MCOUP = 0, there will not be a fluid interface with which to track leakage.

6. **PFAC < 0 for CTYPE = 4, 5, or 6.** PFAC < 0 refers to a load curve. This curve may be constructed with the x-axis as the penetration distance (in absolute length unit) and the y-axis as the maximum estimated coupling pressure. This curve requires only 2 points. The first point is (0.0,0.0) because no FSI pressure applies when no penetration occurs. The x-value for the second point is 10% of the minimum ALE element width, and the y-value is the to be determined maximum observable pressure in the ALE domain. To determine this y-value, run the model once with PFAC = 0.1 (the default). Then, select only the ALE domain and plot its pressure fringes. Next, step through time to identify the maximum pressure in the ALE domain, selecting the pressure peaks at or near the FSI surfaces. This maximum pressure becomes the y-value of the second point.

7. **Normal vector direction.** The normal vectors (NV) of a Lagrangian shell part are defined by the order of the nodes in *ELEMENT definitions, using the right hand rule, and for a segment set, the order of nodes defined in *SET_SEGMENT. Let the side pointed to by NV be "positive". The penalty method measure penetration as the distance the ALE fluid penetrates from the positive side to the negative side of the Lagrangian segment. Only fluid on the positive side will be "seen" and coupled to.

   Therefore, all normal vectors of the Lagrangian segments should point uniformly toward the ALE fluid(s), AMMGs, to be coupled to. If NV point uniformly away from the fluid, coupling is not activated. In this case, coupling can be activated by setting NORM = 1. Sometimes a shell part or mesh is generated such that its normal vectors do not point uniformly in a consistent direction (all toward the inside or outside of a container, etc.) You should always check for

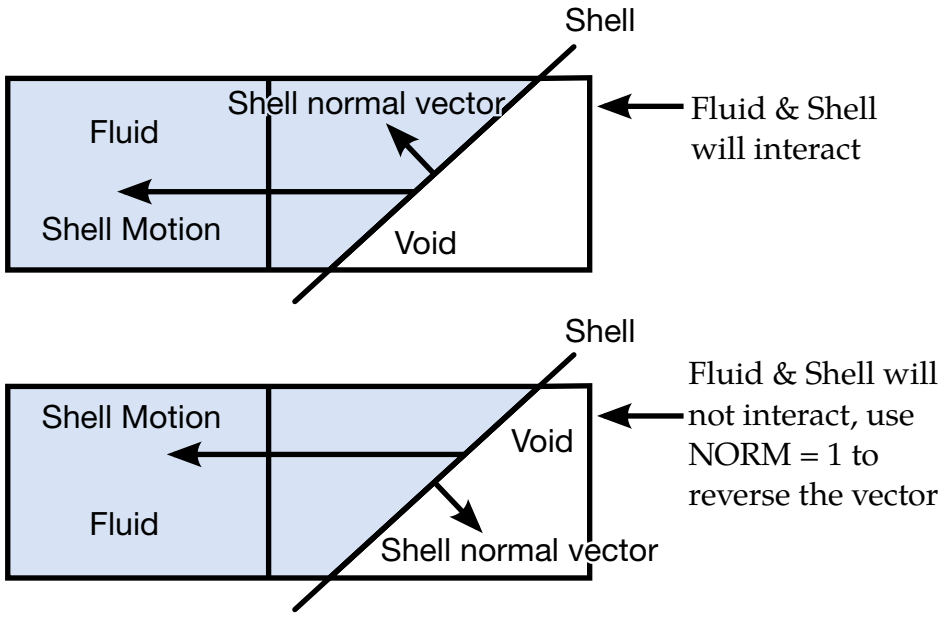


Figure 10-36. Shell Motion

the normal vectors of any Lagrangian shell part interacting with any fluid. The NORM parameter may be used to flip the normal direction of all the segments included in the Lagrangian structure set. See Figure 10-36.

8. **Coupling-damping factor.** The user-input coupling-damping factor (DAMP) is used to scale down the critical-damping force (~ damper constant × velocity). For a mass-to-rigid-wall system connected by a parallel-spring-damper connector, we can obtain solution for a critically-damped case. DAMP is a factor for scaling down the amount of damping, with DAMP = 1 being a critically-damped case.

9. **Heat transfer.** The method used is similar to that done by *CONTACT_…_THERMAL_… card, except radiation heat transfer is not considered. A gap, l, is assumed to exist between the two materials undergoing heat transfer (one is Lagrangian and the other ALE). The convection heat transfer in the gap is assumed to approach simple conduction across the medium in the gap.

$$q = K\frac{dT}{dx} \sim h\Delta T \Rightarrow h \sim \frac{K}{l}$$

The heat flux is typically defined as an energy transfer rate per unit area, $q \sim \frac{[\,J/s\,]}{m^2}$. The constant K is the thermal conductivity of the material in the gap; h, is the equivalent convection heat transfer coefficient; and ΔT is the temperature difference between the ALE and Lagrangian structure sides. There are 3 possible scenarios:

$$h \sim \begin{cases} 0 & \text{HMAX} < l \\ K/l & \text{HMIN} \le l \le \text{HMAX} \\ K/\text{HMIN} & 0 < l < \text{HMIN} \end{cases}$$

The ALE fluid must be modelled using the ALE single material with void element formulation (ELFORM = 12) because the LS-DYNA thermal solver supports only one temperature per node. However, a workaround enables partial support for ELFORM = 11. Rather than using the thermal solver's nodal temperature field, the ALE temperature is derived from element's internal energy using the heat capacity. The heat is then extracted from or added to the internal energy of ALE elements. This feature was implemented to calculate the heat exchange between a gas mixture, modeled with *MAT_GAS_MIXTURE and ALE multi-material formulation ELFORM = 11, and a Lagrangian container.

HMIN < 0 turns on constraint-based thermal nodal coupling between the Lagrangian surface nodes and ALE fluid nodes. This option only works with ALE single material with void element formulation (ELFORM = 12). Once a Lagrangian surface node is in contact with ALE fluid (gap = 0), the heat transfer described above is turned off. Instead the Lagrangian surface node temperature is constrained to the ALE fluid temperature field.

10. **Leakage control.** The dominate force preventing leakage across a coupled Lagrangian surface should be the penalty associated with the coupling. Forces from the leakage control algorithm feature should be secondary. The *DATABASE_FSI keyword controls the "dbfsi" file, which reports both the coupling forces and the leakage control force contribution. It is useful for debugging and fine-tuning.

    ILEAK = 2 conserves energy; thus, it is better for airbag applications. Leakage control should only be enabled when (1) coupling to a specific AMMG (MCOUP as a negative integer) is activated, and (2) the fluid interface is clearly defined and tracked through the *ALE_MULTI-MATERIAL_GROUP card.

11. **Pressure definition in porous flow.** There are currently two methods to model porous flow across a Lagrangian shell structure. Both methods involve defining an empirical data curve of relative porous gas velocity as a function of system pressure. The pressure definitions, however, are slightly different depending on the choice of parameter defined:

    a) When porous flow is modelled using the LCIDPOR parameter (part of *this* keyword), the velocity response curve expected to be given in terms of the pressure difference: $P_{\text{upstream}} - P_{\text{downstream}}$.

b) When LCIDPOR is negative, porous flow is modelled using the *MAT_-FABRIC material model. The FAC field in *MAT_FABRIC contains a load curve ID given in terms of absolute upstream pressure, rather than in terms of the pressure difference.

The *AIRBAG_ALE keyword assumes that the curve referenced by FAC in *MAT_FABRIC is given in terms of absolute upstream pressure. These absolute pressure data are *required* for the CV phase. During the ALE phase, LS-DYNA automatically shifts the FAC curve left (negative) by 1 atmospheric pressure for the porous coupling calculation, which uses gauge pressure, rather than absolute pressure.

The mass flowing across a porous Lagrangian surface can be tracked by the "mout" parameter of the optional "dbfsi" ASCII output file, which may be enabled with the *DATABASE_FSI keyword.

12. **Venting.** There are 2 methods to model (airbag) venting. The accumulated mass output of both may be tracked using the *DATABASE_FSI card ("mout" parameter in the "dbfsi" ASCII output file).

    a) **Isentropic venting.** In isentropic venting, (define NVENT on Card 3) the flow crossing the vent hole surface is estimated from the isentropic equation. All airbag shell normal vectors should point uniformly in the same direction: typically, inward. The shell elements for the vent holes, included in the Lagrangian coupling set, should also point in the same direction as the airbag meaning usually inward. For more details on isentropic venting, see *AIRBAG_WANG_NEFSKE mass flow rate equation for field OPT = 1 and 2.

    b) **Physical venting.** Physical venting models involve holes in the Lagrangian structure (usually airbags). The shell parts representing the vent holes may be either excluded from the Lagrangian coupling set, or, if included, have normal vectors reversed from the rest of the airbag. Typically, this means the holes having outward facing normal vectors, since the rest of the airbag has inward pointing normal vectors. With either approach the holes produce no coupling force to stop fluid leakage.

    When a particular AMMG is present on both sides of the same Lagrangian shell surface, penalty coupling can break down. Therefore, it is recommended that *ALE_FSI_SWITCH_MMG_ID be used to switch the AMMG ID of the vented gas so that the vented gas outside the bag does not lead to leakage.

13. **Initial penetration check.** Typically, penetration check (IPENCHK) should only be used if there is high coupling force applied at $t = 0$. For example, consider a Lagrangian container, filled with non-gaseous fluid, such as ALE liquid

or solid, using the *INITIAL_VOLUME_FRACTON_GEOMETRY command. Sometimes due to mesh resolution or complex container geometry, there is initial penetration of the fluid across the container surface. This can give rise to a sharp and immediate coupling force on the fluid at $t = 0$. Turning on IPENCHK may help eliminate this spike in coupling force.

14. **Porous flow for shell elements.** For shell elements and CTYPE = 11, the Ergun-type empirical porous flow equation is applied to the normal flow direction across the porous surface. The pressure gradient along the segment normal direction is

$$\frac{dP}{dx_n} = A_n(\varepsilon, \mu)V_n + B_n(\varepsilon, \rho)|V_n|V_n$$

where the subscript n refers to the direction normal to the porous Lagrangian shell surface and where,

a) V_n is the relative normal-to-porous-shell-surface fluid velocity component.

b) $A_n(\varepsilon, \mu) = A_1(\varepsilon, \mu)$ is a viscous coefficient of the Ergun-type porous flow equation. As applied here, it should contain the fluid dynamic viscosity, μ , and shell porosity, ε , information.

c) $B_n(\varepsilon, \rho) = B_1(\varepsilon, \rho)$ is an inertial coefficient of the Ergun-type porous flow equation. As applied here, it should contain the fluid density, ρ, and shell porosity, ε, information.

The force increment applied per segment is

$$F_n = \frac{d\rho}{dx_n} \times \text{THKF} \times S,$$

where S is the segment surface area.

If *DEFINE_POROUS_LAGRANGIAN defines the porous properties of a Lagrangian structure element, the porous forces are computed with an equation similar to the one used in *LOAD_BODY_POROUS

> **NOTE:** $A_i(\varepsilon, \mu)$, $B_i(\varepsilon, \rho)$, and THKF are required input for porous shell coupling.

15. **Porous flow for solid elements.** For porous solid, CTYPE = 12, the pressure gradient along each global direction (i) can be computed similarly.

$$\frac{dP}{dx_i} = A_i(\varepsilon, \mu)V_i + B_i(\varepsilon, \rho)|V_i|V_i \text{ for } i = 1,2,3$$

where,

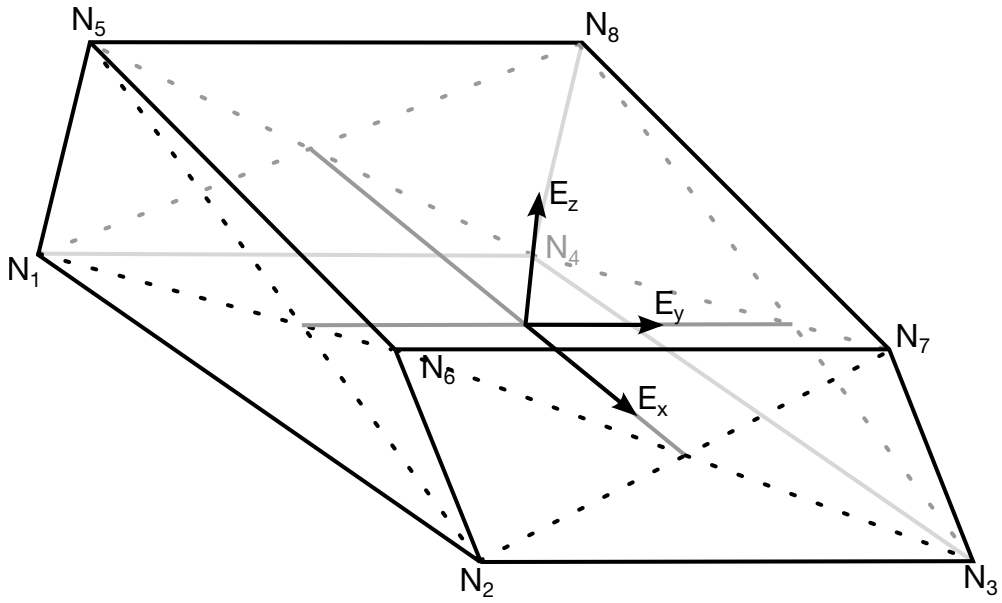


Figure 10-37. The E_x direction is aligned along the line segment connecting the centers of the 2-3-6-7 and the 1-4-8-5 faces. The E_y direction is orthogonal to the E_x direction and in the plane containing both E_x and the segment connecting the centers of the 1-2-6-5 and 3-4-8-7 faces. The E_z is normal to this plane.

a) V_i is the relative fluid velocity component through the porous solid in the 3 global directions.

b) $A_i(\varepsilon, \mu)$ is a viscous coefficient of the Ergun-type porous flow equation in the i[th] direction. As applied here, it should contain the fluid dynamic viscosity, μ, and shell porosity, ε, information.

c) $B_i(\varepsilon, \rho)$ is an inertial coefficient of the Ergun-type porous flow equation in the i[th] direction. As applied here, it should contain the fluid density (ρ) and solid porosity (ε) information.

> **NOTE:** $A_i(\varepsilon, \mu)$, and $B_i(\varepsilon, \rho)$ are required input for porous solid coupling.

If DIREC = 1, the pressure gradient in a solid is applied in a local reference coordinate system defined in Figure 10-37. If *DEFINE_POROUS_LAGRANGIAN defines the porous properties of a Lagrangian structure element, the local system can be adapted and the porous forces are computed with an equation similar to the one used in *LOAD_BODY_POROUS.

16. **Output.** To obtain the sliding energy output for each FSI, specify a *DATABASE_SLEOUT in the input deck. Including this keyword causes a file called slefsi to be output. The format of slefsi follows sleout exactly.

*CONSTRAINED_LINEAR_GLOBAL

Purpose: Define linear constraint equations between displacements and rotations, which can be defined in global coordinate systems.

> **WARNING:** For the explicit solver, nodes of a nodal constraint equation cannot be members of another constraint equation, a constraint set that contains the same degrees-of-freedom, tied interface, or rigid bodies. Nodes must not be subject to multiple, independent, and possibly conflicting constraints. Furthermore, care must be taken to ensure that single point constraints applied to nodes in a constraint equation do not conflict with the degrees-of-freedom constrained by the constraint equation.
>
> The same applies for the implicit solver; however multiple nodal constraint equations may be applied to the various degrees-of-freedom of the same node, provided that there are no conflicting constraints.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							
Default	none							

DOF Card. Define one card for each constrained degree-of-freedom. Input is terminated at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	NID	DOF	COEF					
Type	I	I	F					
Default	none	0	0					

VARIABLE	DESCRIPTION
LCID	Linear constraint definition ID. This ID can be used to identify a set to which this constraint is a member.
NID	Node ID
DOF	Degree of freedom in the global coordinate system: EQ.1: Displacement along global x-direction EQ.2: Displacement along global y-direction EQ.3: Displacement along global z-direction EQ.4: Global rotation about global x-axis EQ.5: Global rotation about global y-axis EQ.6: Global rotation about global z-axis EQ.7: Nodal electric voltage of piezoelectric material; see *MAT_ADD_PZELECTRIC. The voltage of the 1st node can only be defined as a linear combination of the voltage of other nodes, meaning all DOFs must be 7 for such an application.
COEF	Nonzero coefficient, C_k

Remarks:

In this section, linear constraint equations of the form:

$$\sum_{k=1}^{n} C_k u_k = C_0$$

can be defined, where u_k are the displacements and C_k are user defined coefficients. Unless LS-DYNA is initialized by linking to an implicit code to satisfy this equation at the beginning of the calculation, the constant C_0 is assumed to be zero. The first constrained degree-of-freedom is eliminated from the equations-of-motion:

$$u_1 = C_0 - \sum_{k=2}^{n} \frac{C_k}{C_1} u_k \ .$$

Its velocities and accelerations are given by

$$\dot{u}_1 = -\sum_{k=2}^{n} \frac{C_k}{C_1} \dot{u}_k$$

$$\ddot{u}_1 = -\sum_{k=2}^{n} \frac{C_k}{C_1} \ddot{u}_k,$$

respectively. A transformation matrix, **L**, is constructed relating the unconstrained, **u**, and constrained, $\mathbf{u}_c$, degrees-of-freedom. The constrained accelerations used in the above equation are given by:

$$\ddot{\mathbf{u}}_c = \left[\mathbf{L}^{\mathrm{T}}\mathbf{M}\mathbf{L}\right]^{-1}\mathbf{L}^{\mathrm{T}}\mathbf{F}$$

where **M** is the diagonal lumped mass matrix and **F** is the right hand side force vector. This requires the inversion of the condensed mass matrix which is equal in size to the number of constrained degrees-of-freedom minus one.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_LINEAR_GLOBAL
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Constrain nodes 40 and 42 to move identically in the z-direction.
$
$  When the linear constraint equation is applied, it goes like this:
$
$     0 = C40uz40 + C42uz42
$
$       = uz40 - uz42
$
$     uz40 = uz42
$
$   where,
$     C40  =  1.00  coefficient for node 40
$     C42  = -1.00  coefficient for node 42
$     uz40 = displacement of node 40 in z-direction
$     uz42 = displacement of node 42 in z-direction
$
$
*CONSTRAINED_LINEAR
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$        i
$       id
         2
$
$      nid       dof     coef
        40         3     1.00
        42         3    -1.00
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
```

*CONSTRAINED_LINEAR_LOCAL

Purpose: Define linear constraint equations between displacements and rotations which can be defined in a local coordinate system. Each node may have a unique coordinate ID.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							
Default	none							

DOF Cards. Define one card for each constrained degree-of-freedom. Input is terminated at next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	NID	DOF	CID	COEF				
Type	I	I	I	F				
Default	none	0	global	0				
Remarks	1							

VARIABLE	DESCRIPTION
LCID	LCID for linear constraint definition. This ID can be used to identify a set to which this constraint is a member.
NID	Node ID
DOF	Degree-of-freedom in the local coordinate system; EQ.1: Displacement along local x-direction EQ.2: Displacement along local y-direction EQ.3: Displacement along local z-direction EQ.4: Local rotation about local x-axis

VARIABLE	DESCRIPTION
	EQ.5: Local rotation about local y-axis
	EQ.6: Local rotation about local z-axis
CID	Local coordinate system ID number. If the number is zero, the global coordinate system is used.
COEF	Nonzero coefficient, C_k

Remarks:

> **WARNING:** For the explicit solver, nodes of a nodal constraint equation cannot be members of another constraint equation, a constraint set that contains the same degrees-of-freedom, tied interface, or rigid bodies. Nodes must not be subject to multiple, independent, and possibly conflicting constraints. Furthermore, care must be taken to ensure that single point constraints applied to nodes in a constraint equation do not conflict with the degrees-of-freedom constrained by the constraint equation.
>
> The same applies for the implicit solver; however multiple nodal constraint equations may be applied to the various degrees-of-freedom of the same node, provided that there are no conflicting constraints.

With this keyword, you can define linear constraint equations of the following form:

$$\sum_{k=1}^{n} C_k \, u_k^L = C_0 \ ,$$

where u_k^L are the displacements in the local coordinate systems and C_k are user-defined coefficients. Unless you initialize LS-DYNA by linking to an implicit code to satisfy this equation at the beginning of the calculation, LS-DYNA assumes the constant C_0 is zero. The first constrained degree-of-freedom is eliminated from the equations-of-motion:

$$u_1^L = C_0 - \sum_{k=2}^{n} \frac{C_k}{C_1} u_k^L \ .$$

Its velocities and accelerations are given by

$$\dot{u}_1^L = -\sum_{k=2}^{n} \frac{C_k}{C_1} \dot{u}_k^L$$
$$\ddot{u}_1^L = -\sum_{k=2}^{n} \frac{C_k}{C_1} \ddot{u}_k^L$$

respectively. The local displacements are calculated every time step using the local coordinate systems defined by the user. More than one degree-of-freedom for a node can be constrained by specifying a card for each degree-of-freedom.

*CONSTRAINED_LOCAL_{OPTION}

OPTION allows an optional ID to be given:

ID

Purpose: Define a local boundary constraint plane.

ID Card. Additional card for the ID keyword option.

Optional	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	TC	RC	DIR	X	Y	Z	CID	TOL
Type	1	1	1	F	F	F	1	F
Default	0	0	none	0.0	0.0	0.0	none	0.0

VARIABLE	DESCRIPTION
ID	Optional ID which can be referred to by *SENSOR_CONTROL. This ID must be unique and cannot be shared with *BOUNDARY_SPC.
HEADING	An optional descriptor that will be written into the d3hsp file and the spcforc file.
TC	Translational constraint in local system: EQ.0: no translational constraints EQ.1: constrained x translation EQ.2: constrained y translation EQ.3: constrained z translation EQ.4: constrained x and y translations

VARIABLE	DESCRIPTION
	EQ.5: constrained y and z translations
	EQ.6: constrained x and z translations
	EQ.7: constrained x, y, and z translations
RC	Rotational constraint in local system:
	EQ.0: no rotational constraints
	EQ.1: constrained x-rotation
	EQ.2: constrained y-rotation
	EQ.3: constrained z-rotation
	EQ.4: constrained x and y rotations
	EQ.5: constrained y and z rotations
	EQ.6: constrained z and x rotations
	EQ.7: constrained x, y, and z rotations
DIR	Direction of normal for local constraint plane:
	EQ.1: local x,
	EQ.2: local y,
	EQ.3: local z.
X	Local x-coordinate of a point on the local constraint plane
Y	Local y-coordinate of a point on the local constraint plane
Z	Local z-coordinate of a point on the local constraint plane
CID	Coordinate system ID for orientation of the local coordinate system
TOL	User-defined tolerance in length units. If non-zero, the internal mesh-size dependent tolerance gets replaced by this value.

Remarks:

Nodes within a mesh-size-dependent tolerance are constrained on a local plane. This option is recommended for use with *r*-method adaptive remeshing where nodal constraints are lost during the remeshing phase.

*CONSTRAINED_MULTIPLE_GLOBAL

Purpose: Define global multi-point constraints for imposing periodic boundary condition in displacement field.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default								

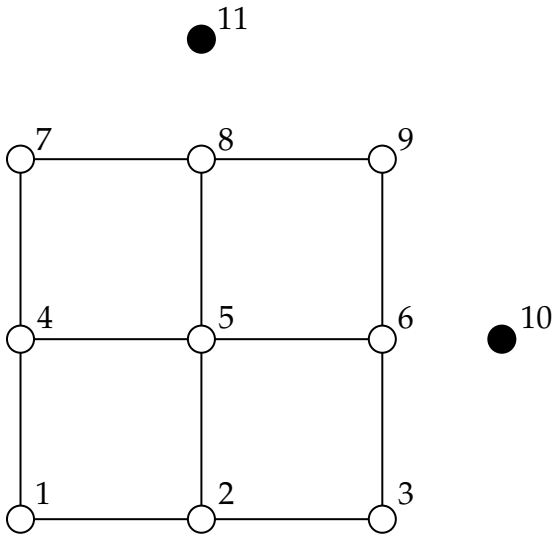
NOTE: For each constraint equation include a set of cards consisting of (1) a C*onstraint Equation Definition Card* and (2) NMP *Coefficient Cards*.

Constraint Equation Definition Card.

Card 2	1	2	3	4	5	6	7	8
Variable	NMP							
Type	I							
Default								

Coefficient Cards. The next NMP cards adhere to this format. Each card sets a single coefficient in the constraint equation.

Card 3	1	2	3	4	5	6	7	8
Variable	NID	DIR	COEF					
Type	I	I	F					
Default								




```
*CONSTRAINED_MULTIPLE_GLOBAL
1
3
 3, 1, 1.0
 1, 1,-1.0
10, 1,-1.0
3
 8, 1, 1.0
 2, 1,-1.0
11, 1,-1.0

*CONSTRAINED_MULTIPLE_GLOBAL
2
3
 3, 2, 1.0
 1, 2,-1.0
10, 2,-1.0
...
```

$$u_1^{(3)} - u_1^{(1)} - u_1^{(10)} = 0$$

$$u_1^{(8)} - u_1^{(2)} - u_1^{(11)} = 0$$

$$u_2^{(3)} - u_2^{(1)} - u_2^{(10)} = 0$$

Figure 10-38. Simple example.

VARIABLE	DESCRIPTION
ID	Constraint set identification. All constraint sets should have a unique set ID.
NMP	Number of nodes to be constrained mutually.
NID	Nodal ID
DIR	Direction in three-dimensional space to be constrained EQ.1: x direction EQ.2: y direction EQ.3: z direction

VARIABLE	DESCRIPTION
	LT.0: Extra DOFs for user defined element formulation (e.g. -1: the 1st extra DOF; -2: the 2nd extra DOF; …)
COEF	Coefficient α_{NID} in constraint equation: $\sum_{\text{NID}} \alpha_{\text{NID}} u_{\text{DIR}}^{(\text{NID})} = 0.$

Remarks:

1. Defining multi-point constraints by this keyword can be demonstrated by the following example: a two-dimensional unit square with four quadrilateral elements and 11 nodes as shown in Figure 10-38, where the nodes #10 and #11 are two dummy nodes serving as control points.

***CONSTRAINED_NODAL_RIGID_BODY**_*{OPTION}_{OPTION}_{OPTION}_{OPTION}*

Purpose: Define a nodal rigid body which is a rigid body that consists of defined nodes. Unlike *CONSTRAINED_NODE_SET, which permits only constraints on translational motion, here we use the equations of rigid body dynamics to update the motion of the nodes, and therefore, rotations of the nodal sets are admissible.

Include any combination of the following available options:

<BLANK>

INERTIA

OVERRIDE

SPC

THERMAL

TITLE

The order of the options in the keyword name is arbitrary.

If *not* using the INERTIA option, LS-DYNA computes the inertia tensor from the nodal masses. The nodal masses and coordinates determine the mass properties. This rigid body may have arbitrary motion. When using the INERTIA option, constant translational and rotational velocities can be defined in a global or local coordinate system.

When using the OVERRIDE option, LS-DYNA automatically turns off all conflicting rigid body or constraint definitions until *SENSOR_CONTROL turns off the overriding nodal rigid body constraint defined with this keyword. When the overriding rigid body constraint is turned off, all conflicting rigid bodies and constraints are turned back on automatically. For example, you can use this feature to rigidize the whole model when the entire model moves like a rigid body involving no deformation.

If the center of mass is constrained, use the SPC option.

The THERMAL option allows imposing a constraint on the temperature distribution for the nodal rigid body. With this constraint, LS-DYNA sets the temperature of a particular node to the average temperature of the other nodes in the nodal rigid body.

The TITLE option enables providing a description for the nodal rigid body.

Card Summary:

Card Title. Include this keyword for the TITLE keyword option.

TITLE

Card 1. This card is required.

PID	CID	NSID	PNODE	IPRT	DRFLAG	RRFLAG	

Card 2. Include this card if using the SPC keyword option.

CMO	CON1	CON2	SPCNID	XSPC	YSPC	ZSPC	

Card 3. Include this card for the INERTIA keyword option.

XC	YC	ZC	TM	IRCS	NODEID		

Card 4. Include this card for the INERTIA keyword option.

IXX	IXY	IXZ	IYY	IYZ	IZZ		

Card 5. Include this card for the INERTIA keyword option.

VTX	VTY	VTZ	VRX	VRY	VRZ		

Card 6. Include this card if IRCS = 1. See Card 3.

XL	YL	ZL	XLIP	YLIP	ZLIP	CID2	

Card 7. This card is read when using the OVERRIDE keyword option. It is optional.

ICNT	IBAG	IPSM					

Card 8. Include this card for the THERMAL keyword option.

IDTHRM							

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

VARIABLE	DESCRIPTION
TITLE	Description for the nodal rigid body

Card 1	1	2	3	4	5	6	7	8
Variable	PID	CID	NSID	PNODE	IPRT	DRFLAG	RRFLAG	
Type	I	I	I	I	I	I	I	
Default	none	0	PID	0	0	0	0	

VARIABLE	DESCRIPTION
PID	Part ID of the nodal rigid body
CID	Optional coordinate system ID for the rigid body local system; see *DEFINE_COORDINATE_OPTION. The rigid body data and the degree-of-freedom releases are output in this local system. This local system rotates with the rigid body.
NSID	Nodal set ID, see *SET_NODE_OPTION. This nodal set defines the rigid body. EQ.0: NSID = PID; that is, the node set ID and the part ID are assumed to be identical.
PNODE	An optional node (a massless node is allowed) used for post-processing rigid body data. If PNODE is not located at the rigid body's center of mass, LS-DYNA resets the initial coordinates of PNODE to the center of mass. If CID is defined, LS-DYNA outputs the velocities and accelerations of PNODE in the local system to the d3plot and d3thdt files unless you specify PNODE as a negative

VARIABLE	DESCRIPTION
	number, in which case the global system is used.
IPRT	Print flag. For nodal rigid bodies, the following values apply: EQ.-2: Do not write data into rbdout. Output the forces and moments in bndout with respect to the nodal rigid body's local coordinate system. EQ.-1: Write data into rbdout. Output the forces and moments in bndout with respect to the nodal rigid body's local coordinate system. EQ.0: Defaults to the value of IPRTF in *CONTROL_OUTPUT for all nodal rigid bodies except two-noded rigid bodies. For two-noded rigid bodies, printing to rbdout is suppressed. This behavior exists to avoid excessively large rbdout files when the model contains many two-noded welds. Note that IPRTF does not change the coordinate system for bndout. Thus, for all nodal rigid bodies, the forces and moments in the bndout file are expressed in the global coordinate system. EQ.1: Write data into rbdout. Output the forces and moments in bndout with respect to the global coordinate system. EQ.2: Do not write data into rbdout. Output the forces and moments in bndout with respect to the global coordinate system.
DRFLAG	Displacement release flag for all nodes except the first node in the definition (see Remark 3). EQ.-7: Release the x, y, and z displacements in the global system EQ.-6: Release the z and x displacements in the global system EQ.-5: Release the y and z displacements in the global system EQ.-4: Release the x and y displacements in the global system EQ.-3: Release the z displacement in the global system EQ.-2: Release the y displacement in the global system EQ.-1: Release the x displacement in the global system EQ.0: Off for rigid body behavior EQ.1: Release the x displacement in the rigid body's local system EQ.2: Release the y displacement in the rigid body's local

VARIABLE	DESCRIPTION
	system
	EQ.3: Release the z displacement in the rigid body's local system
	EQ.4: Release the x and y displacements in the rigid body's local system
	EQ.5: Release the y and z displacements in the rigid body's local system
	EQ.6: Release the z and x displacements in the rigid body's local system
	EQ.7: Release the x, y, and z displacements in the rigid body's local system
RRFLAG	Rotation release flag for all nodes except the first node in the definition (see Remark 3).
	EQ.-7: Release the x, y, and z rotations in the global system
	EQ.-6: Release the z and x rotations in the global system
	EQ.-5: Release the y and z rotations in the global system
	EQ.-4: Release the x and y rotations in the global system
	EQ.-3: Release the z rotation in the global system
	EQ.-2: Release the y rotation in the global system
	EQ.-1: Release the x rotation in the global system
	EQ.0: Off for rigid body behavior
	EQ.1: Release the x rotation in the rigid body's local system
	EQ.2: Release the y rotation in the rigid body's local system
	EQ.3: Release z rotation in the rigid body's local system
	EQ.4: Release the x and y rotations in the rigid body's local system
	EQ.5: Release the y and z rotations in the rigid body's local system
	EQ.6: Release the z and x rotations in the rigid body's local system
	EQ.7: Release the x, y, and z rotations in the rigid body's local system

Constraint Card. Additional card for the SPC keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	CMO	CON1	CON2	SPCNID	XSPC	YSPC	ZSPC	
Type	F	I	I	I	F	F	F	
Default	0.0	0	0	0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
CMO	Constraint option, CMO (see Remark 4): EQ.+2.0: Constraints applied in global directions at the coordinates given by XSPC, YSPC, and ZSPC or the initial coordinates of node SPCNID. Unless prescribed motion is applied to the rigid body, the constraint coordinates are fixed in time. EQ.+1.0: Constraints applied in global directions, EQ.0.0: No constraints, EQ.-1.0: Constraints applied in local directions (SPC constraint). EQ.-2.0: Constraints applied in local directions (SPC constraint) at the coordinates given by XSPC, YSPC, and ZSPC or the initial coordinates of node SPCNID. Unless prescribed motion is applied to the rigid body, the constraint coordinates are fixed in time.
CON1	First constraint parameter. If CMO > 0.0, then specify the global translational constraint: EQ.0: No constraints, EQ.1: Constrained x displacement, EQ.2: Constrained y displacement, EQ.3: Constrained z displacement, EQ.4: Constrained x and y displacements, EQ.5: Constrained y and z displacements, EQ.6: Constrained z and x displacements,

VARIABLE	DESCRIPTION
	EQ.7: Constrained x, y, and z displacements. If CMO < 0.0, then specify the local coordinate system ID. See *DEFINE_COORDINATE_OPTION. This coordinate system is fixed in time.
CON2	Second constraint parameter: If CMO > 0.0, then specify the global rotational constraint: EQ.0: No constraints, EQ.1: Constrained x rotation, EQ.2: Constrained y rotation, EQ.3: Constrained z rotation, EQ.4: Constrained x and y rotations, EQ.5: Constrained y and z rotations, EQ.6: Constrained z and x rotations, EQ.7: Constrained x, y, and z rotations. If CMO < 0.0, then specify the local (SPC) constraint: EQ.000000: No constraint, EQ.100000: Constrained x translation, EQ.010000: Constrained y translation, EQ.001000: Constrained z translation, EQ.000100: Constrained x rotation, EQ.000010: Constrained y rotation, EQ.000001: Constrained z rotation. To specify a combination of local constraints, input the sum of the desired constraints.
SPCNID	For |CMO| = 2.0, the constraint coordinates (see below) are the (initial) coordinates of the node with this ID.
XSPC,YSPC,ZSPC	Coordinates where the constraints act. Superseded by SPC-NID.

Inertia Card 1. Additional card for the INERTIA keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	TM	IRCS	NODEID		
Type	F	F	F	F	I	I		
Default	0.	0.	0.	0.	0	0		

VARIABLE	DESCRIPTION
XC	*x*-coordinate of center of mass. If nodal point, NODEID, is defined, XC, YC, and ZC are ignored and the coordinates of the nodal point, NODEID, are taken as the center of mass.
YC	*y*-coordinate of center of mass
ZC	*z*-coordinate of center of mass
TM	Translational mass
IRCS	Flag for inertia tensor reference coordinate system: EQ.0: Global inertia tensor, EQ.1: Local inertia tensor is given in a system defined by the orientation vectors given in Card 6.
NODEID	Optional nodal point defining the CG of the rigid body. If this node is not a member of the set NSID above, its motion will not be updated to correspond with the nodal rigid body after the calculation begins. PNODE and NODEID can be identical if and only if PNODE physically lies at the mass center at time zero.

Inertia Card 2. Second additional card for the INERTIA keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	IXX	IXY	IXZ	IYY	IYZ	IZZ		
Type	F	F	F	F	F	F		
Default	none	0	0	none	0	0		

VARIABLE	DESCRIPTION
IXX	I_{xx}, xx component of inertia tensor
IXY	I_{xy}, xy component of inertia tensor
IXZ	I_{xz}, xz component of inertia tensor
IYY	I_{yy}, yy component of inertia tensor
IYZ	I_{yz}, yz component of inertia tensor
IZZ	I_{zz}, zz component of inertia tensor

Inertia Card 3. Third additional card for the INERTIA keyword option. *INITIAL_VELOCITY can overwrite the velocities defined below.

Card 5	1	2	3	4	5	6	7	8
Variable	VTX	VTY	VTZ	VRX	VRY	VRZ		
Type	F	F	F	F	F	F		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
VTX	x-rigid body initial translational velocity in global coordinate system.
VTY	y-rigid body initial translational velocity in global coordinate system.

VARIABLE	DESCRIPTION
VTZ	z-rigid body initial translational velocity in global coordinate system.
VRX	x-rigid body initial rotational velocity in global coordinate system.
VRY	y-rigid body initial rotational velocity in global coordinate system.
VRZ	z-rigid body initial rotational velocity in global coordinate system.

Local Inertia Tensor Card. Additional card required for IRCS = 1 (see Inertia Card 1). Define two local vectors or a local coordinate system ID.

Card 6	1	2	3	4	5	6	7	8
Variable	XL	YL	ZL	XLIP	YLIP	ZLIP	CID2	
Type	F	F	F	F	F	F	I	
Default	CID2	CID2	CID2	CID2	CID2	CID2	0	

VARIABLE	DESCRIPTION
XL	x-coordinate of local x-axis. Origin lies at (0,0,0).
YL	y-coordinate of local x-axis
ZL	z-coordinate of local x-axis
XLIP	x-coordinate of local in-plane vector
YLIP	y-coordinate of local in-plane vector
ZLIP	z-coordinate of local in-plane vector
CID2	Local coordinate system ID, see *DEFINE_COORDINATE_… With this option, leave fields 1-6 blank.

Optional synchronization card for OVERRIDE. Optional card for OVERRIDE option.

Card 7	1	2	3	4	5	6	7	8
Variable	ICNT	IBAG	IPSM					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
ICNT	Flag for contact synchronization: EQ.0: No synchronization EQ.1: Since no contact exists when both the tracked and reference sides belong to the same rigid body, setting ICNT = 1 will turn off/on all contact definitions of which the tracked and reference sides belong to the same nodal rigid body PID when PID is turned on/off by *SENSOR_CONTROL. With ICNT = 1, all related contact definitions need the optional ID; see *CONTACT.
IBAG	Flag for control volume airbag synchronization: EQ.0: No synchronization EQ.1: Since airbag pressure will not change when all segments constituting the airbag belong to the same rigid body, setting IBAG = 1 will skip the control volume airbag calculations for airbags that belong to PID when PID is on. The airbag calculation will resume when *SENSOR_CONTROL turns off PID. The related airbag time-dependent curves will be offset to not include the time that PID is on. With IBAG = 1, all associated airbags need the optional ID; see *AIRBAG.
IPSM	Flag for prescribed-motion synchronization: EQ.0: No synchronization EQ.1: Prescribed boundary conditions (*BOUNDARY_PRESCRIBED_MOTION) for PID are turned off automatically when *SENSOR_CONTROL turns off PID. A prescribed boundary condition not for PID and not for all nodes belonging to PID, that is, boundary conditions that only

VARIABLE	DESCRIPTION
	apply to some of the nodes belonging to PID, are turned off when PID is active to avoid boundary condition conflict. Those boundary conditions are turned on when *SENSOR_CONTROL turns off PID. The related time-dependent curves will be offset to not include the time that PID is on. With IPSM > 0, all related boundary prescribed motion cards need the optional ID; see *BOUNDARY_PRESCRIBED.
	EQ.2: Same as IPSM = 1, however, without time offset when those boundary conditions not for PID are turned on.

Thermal Constraint Card. Required card for the THERMAL keyword option.

Card 8	1	2	3	4	5	6	7	8
Variable	IDTHRM							
Type	I							
Default	1							

VARIABLE	DESCRIPTION
IDTHRM	Flag for the thermal constraint option:
	EQ.1: Set the temperature of the first node defined in node set NSID as the average of the other nodal temperatures in the node set.
	LT.0: Set the temperature of node |IDTHRM| as the average of the nodal temperatures for the nodes in node set NSID. The average excludes node |IDTHERM| if it is a member of this node set.

Remarks:

1. **Local coordinate system.** The local coordinate system is set up in the following way. After the local *x*-axis is defined, the local *z*-axis is computed from the cross-product of the local *x*-axis vector with the given in-plane vector. Finally, the local *y*-axis is determined from the cross-product of the local *z*-axis with the local *x*-axis. The local coordinate system defined by CID has the advantage

that the local system can be defined by nodes in the rigid body which makes repositioning of the rigid body in a preprocessor much easier since the local system moves with the nodal points.

2. **Nodal rigid body visualization.** Use PLOTEL (*CONTROL_RIGID) to visualize the nodal rigid bodies.

3. **Release conditions.** The first node in the nodal rigid body definition is treated as the lead for the case where DRFLAG and RRFLAG are nonzero. The first node always has six degrees-of-freedom. The release conditions in the global system are sometimes convenient in small displacement linear analysis. Otherwise, they are not recommended. We strongly recommend that release conditions are only used for a two-noded nodal rigid body, especially for implicit calculations.

4. **Constraints.** Constraint directions for rigid bodies (CMO ≠ 0) are fixed, that is, not updated, with time. To impose a constraint on a rigid body such that the constraint direction is updated as the rigid body rotates, use *BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL. The constraint defined therein refers to the local system CID, which is updated with time.

   We strongly advise not applying nodal constraints, such as by *BOUNDARY_SPC_*OPTION*, to nodes of a rigid body as doing so may compromise the intended constraints in the case of an explicit simulation. In an implicit simulation, constraints can, however, be applied to the nodes of a rigid body.

   If the intended constraints are not with respect to the calculated center of mass of the rigid body in explicit simulations, the following alternative approaches can be used:

   a) Set |CMO| = 2 and choose the point at which the constraints shall act. These coordinates are referenced both by the constraints given on this card and those on *BOUNDARY_PRESCRIBED_MOTION_*OPTION*. Note, |CMO| = 2 is not available for implicit simulations.

   b) *CONSTRAINED_JOINT_*OPTION* may often be used to obtain the desired effect. This approach typically entails defining a second rigid body that is fully constrained and then defining a joint between the two rigid bodies.

   c) Another alternative for defining rigid body constraints that are not with respect to the calculated center of mass of the rigid body is to manually specify the initial center of mass location using *PART_INERTIA. When using *PART_INERTIA, a full set of mass properties must be specified. Note that changing its mass properties will affect the rigid body's dynamic behavior.

Setting |CMO| = 2 in not only allows for a constraint point other than the center of mass. The motion prescribed by *BOUNDARY_PRESCRIBED_MOTION also acts on this point. In addition, setting |CMO| = 2 treats the constraints (including those from *BOUNDARY_PRESCRIBED_MOTION) differently from |CMO| = 1. To allow for an arbitrary constraint point, the constraints are applied and solved during the kinematic update of the rigid body. Since the inertia does not need to be modified and no joints are involved, setting |CMO| = 2 is more accurate compared to options b and c above. No time penalty is to be expected.

To obtain reaction forces from constraints, see the SPC2BND flag of *CONTROL_OUTPUT.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_NODAL_RIGID_BODY
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a rigid body consisting of the nodes in nodal set 61.
$
$  This particular example was used to connect three separate deformable
$  parts.  Physically, these parts were welded together.  Modeling-wise,
$  however, this joint is quite messy and is most conveniently modeled
$  by making a rigid body using several of the nodes in the area.  Physically,
$  this joint was so strong that weld failure was never of concern.
$
*CONSTRAINED_NODAL_RIGID_BODY
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid       cid      nsid
        45                  61
$
$     nsid = 61   nodal set ID number, requires a *SET_NODE_option
$      cid        not used in this example; output will be in global coordinates
$
$
*SET_NODE_LIST
$      sid
        61
$     nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
       823      1057      1174      1931      2124      1961      2101
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*CONSTRAINED_NODE_INTERPOLATION

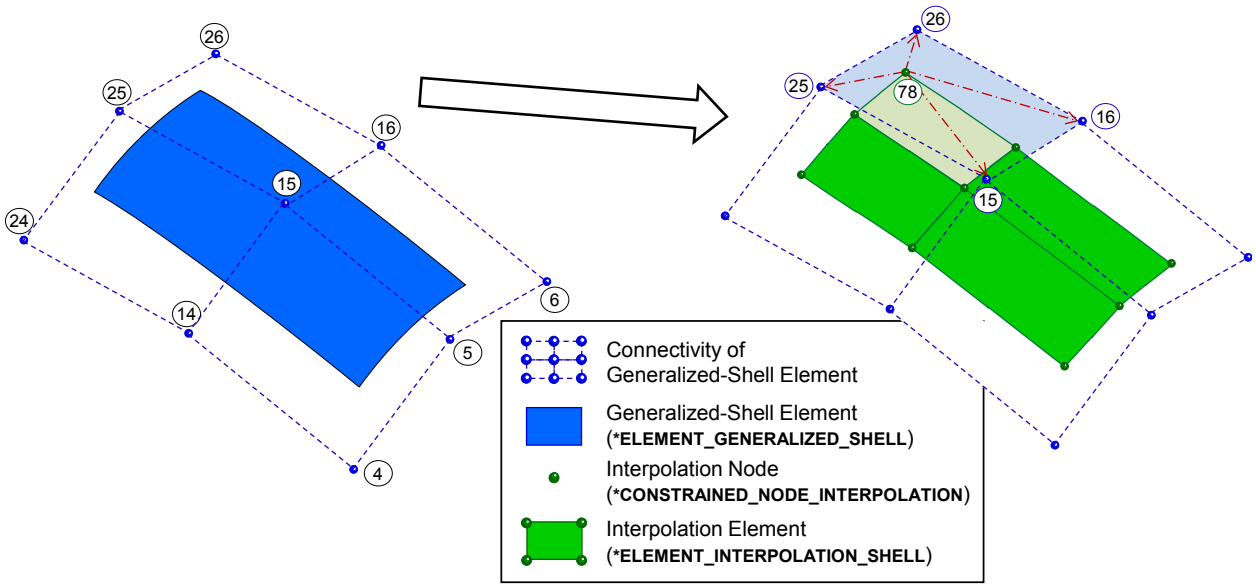
Purpose: Define constrained nodes for the use of *ELEMENT_INTERPOLATION_-SHELL and *ELEMENT_INTERPOLATION_SOLID to model contact and to visualize the results of generalized elements (see *ELEMENT_GENERALIZED_SHELL/SOLID). The displacements of these nodes are dependent on their corresponding controlling nodes.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	NUMCN						
Type	I	I						
Default	none	none						

Weighting Factor Cards. Define NUMCN controlling nodes and weights (one controlling node per CN*i* and W*i* pair) to constrain NID. See Remark 2. See *ELEMENT_-GENERALIZED_SHELL/SOLID. Each Weighting Factor Card can accommodate four master nodes. Include as many Weighting Factor Cards as needed.

Card 2	1	2	3	4	5	6	7	8
Variable	CN1	W1	CN2	W2	CN3	W3	CN4	W4
Type	I	F	I	F	I	F	I	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
NID	Node ID of the interpolation node as defined in *NODE (see Remark 1)
NUMCN	Number of nodes controlling the interpolation node
CN*i*	Node ID of controlling node *i*
W*i*	Weighting factor of controlling node *i*




```
*CONSTRAINED_NODE_INTERPOLATION
$---+--NID----+NUMCN----+----3----+----4----+----5----+----6----+----7----+----8
        78          4
$---+--CN1----+---W1----+--CN2----+---W2----+--CN3----+---W3----+--CN4----+---W4
        26       0.35         25       0.32         15       0.18         16       0.15
```

Figure 10-39. Example of a *CONSTRAINED_NODE_INTERPOLATION card

Remarks:

1. **Nodal Definition.** The coordinates of an interpolation node must be defined in *NODE. The fields TC and RC must be both be set to 7 for the interpolation node in *NODE.

2. **Interpolation Model.** The displacements of the interpolation node, $\mathbf{d}_{\text{IN}}$, are interpolated based on the displacements of the corresponding controlling nodes, $\mathbf{d}_i$, and the appropriate weighting factors, w_i. The interpolation is computed as follows:

$$\mathbf{d}_{\text{IN}} = \sum_{i=1}^{\text{NUMCN}} w_i \mathbf{d}_i.$$

*CONSTRAINED_NODE_SET_{OPTION}

To define an ID for the constrained node set the following option is available:

<BLANK>

ID

If the ID keyword option is used, an additional card is required.

Purpose: Define nodal constraint sets for translational motion in global coordinates. No rotational coupling. See Figure 10-40. Nodal points included in the sets should not be subjected to any other constraints including prescribed motion, such as motions prescribed with the *BOUNDARY_PRESCRIBED_MOTION options.

ID Card. Additional card for ID keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	CNSID							
Type	I							

Card 2	1	2	3	4	5	6	7	8
Variable	NSID	DOF	TF					
Type	I	I	F					
Default	none	none	10^{20}					
Remarks	1, 2		3					

VARIABLE	DESCRIPTION
CNSID	Optional constrained node set ID
NSID	Nodal set ID; see *SET_NODE_*OPTION*.

***CONTRAINED_NODE_SET**

Since no rotation is permitted, this option should *not* be used to model rigid body behavior involving rotations

***CONSTRAINED_NODAL_RIGID_BODY**
***CONSTRAINED_SPOTWELD**

Behavior is like a rigid beam. These options *may* be used to model spotwelds.

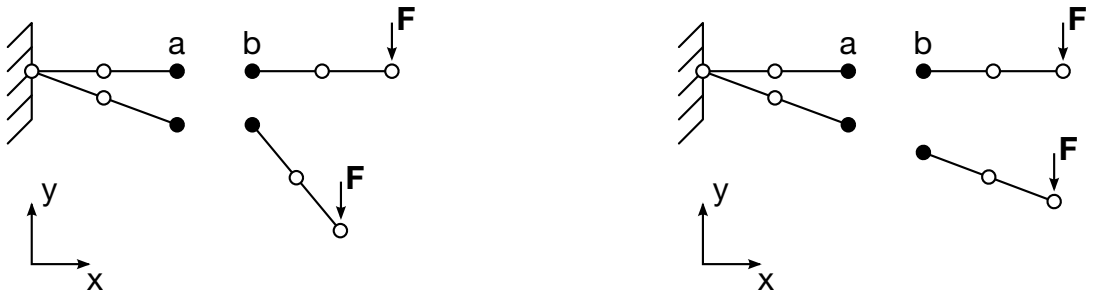


Figure 10-40. Two different ways to constrain node *a* and *b*. For rigid-body type situations this card, *CONSTRAINED_NODE_SET, may lead to un-physical results.

VARIABLE	DESCRIPTION
DOF	Applicable degrees-of-freedom: EQ.1: *x*-translational degree-of-freedom EQ.2: *y*-translational degree-of-freedom EQ.3: *z*-translational degree-of-freedom EQ.4: *x* and *y*-translational degrees-of-freedom EQ.5: *y* and *z*-translational degrees-of-freedom EQ.6: *z* and *x*-translational degrees-of-freedom EQ.7: *x*, *y*, and *z*-translational degrees-of-freedom EQ.8: Electric potential of piezoelectric material
TF	Failure time for nodal constraint set

Remarks:

1. **Mass.** The masses of the nodes are summed up to determine the total mass of the constrained set.

2. **Nodal Rigid Body.** Note that the definition of a nodal rigid body is not possible with this input. For nodal rigid bodies the keyword input *CONSTRAINED_NODAL_RIGID_BODY_*OPTION* must be used.

3. **Failure Time.** When the failure time, TF, is reached the nodal constraint becomes inactive, and the constrained nodes may move freely.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_NODE_SET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Constrain all the nodes in a nodal set to move equivalently
$  in the z-direction.
$
*CONSTRAINED_NODE_SET
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
      nsid       dof        tf
         7         3      10.0
$
$     nsid = 7    nodal set ID number, requires a *SET_NODE_option
$      dof = 3    nodal motions are equivalent in z-translation
$      tf  = 3    at time=10. the nodal constraint is removed
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*CONSTRAINED_NODE_TO_NURBS_PATCH_{OPTION}

Purpose: To add additional massless nodes to the surface of a NURBS patch. The motion of the nodes is governed by the NURBS patch. Forces applied to the nodes are distributed to the NURBS patch. Penalty method is used to handle the displacement boundary conditions CON (see below) on the specified nodes.

To specify node sets instead of individual nodes use the option:

SET

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NSID	CON	CID	SF	DBFLG		
Type	I	I	I	I	F	I		
Default	none	none	0	none	1.0	0		

VARIABLE	DESCRIPTION
PATCHID	Patch ID.
NSID	Nodal set ID or node ID depending on the *OPTION*.
CON	Constraint parameter for extra node(s) of NSID. EQ.000000: no constraint EQ.100000: constrained x translation EQ.010000: constrained y translation EQ.001000: constrained z translation EQ.000100: constrained x rotation EQ.000010: constrained y rotation EQ.000001: constrained z rotation Any combination of local constraints can be specified by adding the number 1 into the corresponding column. For example, "1110" means constrained z-translation, x-rotation and y-rotation.
CID	Coordinate system ID for constraint
SF	Penalty force scale factor for the penalty-based constraint

VARIABLE	DESCRIPTION
DBFLG	Discrete beam flag. If CON = 0 and displacement boundary conditions are applied to nodes specified in NSID, then this flag must be set to 1. When DBFLG = 1, discrete beam elements are created to connect nodes in NSID to the patch.

*CONSTRAINED_POINTS

Purpose: Constrain two points with the specified coordinates connecting two shell elements at locations other than nodal points. In this option, the penalty method is used to constrain the translational and rotational degrees-of-freedom of the points. Force resultants are written into the swforc ASCII file for post-processing.

Card 1	1	2	3	4	5	6	7	8
Variable	CID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	EID1	X1		Y1		Z1				
Type	I	F		F		F				
Default	none	0.		0.		0.				

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	EID2	X2		Y2		Z2				
Type	I	F		F		F				
Default	none	0.		0.		0.				

Card 4	1	2	3	4	5	6	7	8
Variable	PSF	FAILA	FAILS	FAILM				
Type	F	F	F	F				
Default	1.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
CID	Constrained points ID.
EID*i*	Shell element ID, *i* = 1, 2.
X*i*, Y*i*, Z*i*	Coordinates of the constrained points, *i* = 1, 2.
PSF	Penalty scale factor (Default = 1.0).
FAILA	Axial force resultant failure value, no failure if zero.
FAILS	Shear force resultant failure value, no failure if zero.
FAILM	Moment resultant failure value, no failure if zero.

*CONSTRAINED_RIGID_BODIES_{OPTION}

Available options include:

<BLANK>

SET

For SET option, PIDC refers to a part set.

Purpose: Merge two rigid bodies. One rigid body, called the constrained rigid body, is merged into another one, called the lead rigid body. This command applies to parts comprised of *MAT_RIGID but not to nodal rigid bodies (*CONSTRAINED_NODAL_-RIGID_BODY).

Card 1	1	2	3	4	5	6	7	8
Variable	PIDL	PIDC	IFLAG					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
PIDL	Lead rigid body part ID; see *PART.
PIDC	Constrained rigid body part ID (see *PART) or constrained rigid body part set ID for the SET keyword option (see *SET_PART)
IFLAG	This flag is meaningful *if and only if* the inertia properties of the lead part, PIDL, are defined in *PART_INERTIA. See Remark 1. EQ.1: Update the center-of-gravity, the translational mass, and the inertia matrix of PIDL to reflect its merging with the constrained rigid body (PIDC). EQ.0: The merged PIDC will not affect the properties defined in *PART_INERTIA for PIDL since the properties are assumed to already account for merged parts. If the properties are not defined in a *PART_INERTIA definition, the inertia properties of PIDC will be computed from its nodal masses.

Remarks:

1. **Inertial Properties.** The constrained rigid body is merged to the lead rigid body. Unless the inertial properties of the lead rigid body are defined with *PART_INERTIA, the inertial properties computed by LS-DYNA are based on the combination of the lead rigid body plus all the rigid bodies which are constrained to it. If the inertial properties of the lead are specified with *PART_INERTIA, then the treatment of those properties depends on IFLAG. Note that a lead rigid body may have many rigid bodies constrained to it, but it may not be constrained to another rigid body.

2. **Common Nodes.** Independent rigid bodies must not share common nodes since each rigid body updates the motion of its nodes independently of the other rigid bodies. If common nodes exist between rigid bodies, the rigid bodies sharing the nodes must be merged.

3. **Completely Separated Rigid Bodies.** Completely separated rigid bodies that do not share any common nodal points or boundaries may also be merged. All actions valid for the lead rigid body, such as constraints and given velocity, are now also valid for the newly-created rigid body.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_RIGID_BODIES
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Rigidly connect parts 35, 70, 71, and 72 to part 12.
$  All parts must be defined as rigid.
$
$  This example is used to make a single rigid body out of the five parts
$  that compose the back end of a vehicle.  This was done to save cpu time
$  and was determined to be valid because the application was a frontal
$  impact with insignificant rear end deformations.  (The cpu time saved
$  was from making the parts rigid, not from merging them - merging was
$  more of a convenience in this case for post processing, for checking
$  inertial properties, and for joining the parts.)
$
*CONSTRAINED_RIGID_BODIES
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     pidl      pidc
        12        35
        12        70
        12        71
        12        72
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*CONSTRAINED_RIGID_BODY_INSERT

Purpose: This keyword is for modeling die inserts. One rigid body, called the constrained rigid body, is constrained to move with another rigid body, called the lead rigid body, in all directions except for one.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	PIDL	PIDC	COORDID	IDIR			
Type	I	I	I	I	I			
Default	none	none	none	none	3			

Card 2	1	2	3	4	5	6	7	8
Variable	MFLAG	MCID	DEATHM					
Type	I	I	F					
Default	0	0	↓					

Card 3	1	2	3	4	5	6	7	8
Variable	PARTB	DEATHB						
Type	I	F						
Default	0	0.0						

VARIABLE	DESCRIPTION
ID	Insert ID
PIDL	Lead (die) rigid body part ID (see *PART)
PIDC	Constrained (die insert) rigid body part ID (see *PART)

VARIABLE	DESCRIPTION
COORDID	Coordinate system ID
IDIR	Direction in which the insert moves independently of the die: EQ.1: Local *x*-direction EQ.2: Local *y*-direction EQ.3: Local *z*-direction (default)
MFLAG	Motion flag: EQ.0: Relative motion is unconstrained. EQ.1: The displacement of the insert relative to the die is imposed. EQ.2: The velocity of the insert relative to the die is imposed. EQ.3: The acceleration of the insert relative to the die is imposed.
MCID	Curve defining the motion of the die insert relative to the die
DEATHM	Death time of the imposed motion. If it is equal to 0.0, the motion is imposed for the entire analysis.
PARTB	Part ID for a discrete beam connected between the insert and die
DEATHB	Death time for the discrete beam specified by PARTB

Remarks:

1. **Time Integrators.** This capability is supported by both the implicit and explicit time integrators; however, the joint death time, DEATHM, feature works only for explicit integration with the penalty method.

2. **Joint Reaction Forces.** The translational joint constraining the die and the die insert are automatically generated. The joint reaction forces will appear in the jntforc output file.

3. **Translational Degree of Freedom.** The translational motor constraining the remaining translational degree of freedom is also automatically generated, and its reaction forces also appear in the jntforc output file.

4. **Discrete Beam.** The automatically generated beam has its data written to the d3plot file and all of the optional appropriate output files.

*CONSTRAINED_RIGID_BODY_STOPPERS

Purpose: Rigid body stoppers provide a convenient way of controlling the motion of rigid tooling in metal forming applications. The motion of a "lead" rigid body is limited by load curves. This option will stop the motion based on a time dependent constraint. The stopper overrides prescribed motion boundary conditions (except relative displacement) operating in the same direction for both the lead and constrained rigid bodies. See Figure 10-41.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCMAX	LCMIN	PSIDMX	PSIDMN	LCVMNX	DIR	VID
Type	I	I	I	I	I	I	I	I
Default	none	0	0	0	0	0	none	0

Card 2	1	2	3	4	5	6	7	8
Variable	TB	TD		STIFF				
Type	F	F		F				
Default	0	1021		0.0				

VARIABLE	DESCRIPTION
PID	Part ID of lead rigid body (see *PART)
LCMAX	Load curve ID defining the maximum coordinate or displacement as a function of time (see *DEFINE_CURVE): LT.0: Load curve ID |LCMAX| provides an upper bound for the displacement of the rigid body. EQ.0: No limitation of the maximum displacement. GT.0: Load curve ID LCMAX provides an upper bound for the position of the rigid body center of mass.
LCMIN	Load curve ID defining the minimum coordinate or displacement as a function of time (see *DEFINE_CURVE):

VARIABLE	DESCRIPTION
	LT.0: Load curve ID |LCMIN| defines a lower bound for the displacement of the rigid body. EQ.0: No limitation of the minimum displacement. GT.0: Load curve ID LCMIN defines a lower bound for the position of the rigid body center of mass.
PSIDMX	Optional part set ID of rigid bodies that are constrained in the maximum coordinate direction to the lead rigid body. The part set definition (see *SET_PART_COLUMN) may be used to define the closure distance (D_1 and D_2in Figure 10-41) which activates the constraint. The constraint does not begin to act until the lead rigid body stops. If the distance between the lead rigid body is greater than or equal to the closure distance, the constrained rigid body motion away from the lead rigid body also stops. However, the constrained rigid body is free to move towards the lead rigid body. If the closure distance is input as zero (0.0), then the constrained rigid body stops when the lead stops.
PSIDMN	Optional part set ID of rigid bodies that are constrained in the minimum coordinate direction to the lead rigid body. The part set definition, (see *SET_PART_COLUMN) may be used to define the closure distance (D_1 and D_2 in Figure 10-41) which activates the constraint. The constraint does not begin to act until the lead rigid body stops. If the distance between the lead rigid body is less than or equal to the closure distance, the constrained rigid body motion towards the lead rigid body also stops. However, the constrained rigid body is free to move away from the lead rigid part. If the closure distance is input as zero (0.0), then the constrained rigid body stops when the lead stops.
LCVMX	Load curve ID which defines the maximum absolute value of the velocity as a function of time that is allowed for the lead rigid body. See *DEFINE_CURVE: EQ.0: no limitation on the velocity.
DIR	Direction stopper acts in: EQ.1: x-translation, EQ.2: y-translation, EQ.3: z-translation, EQ.4: Arbitrary, defined by vector VID (see below),

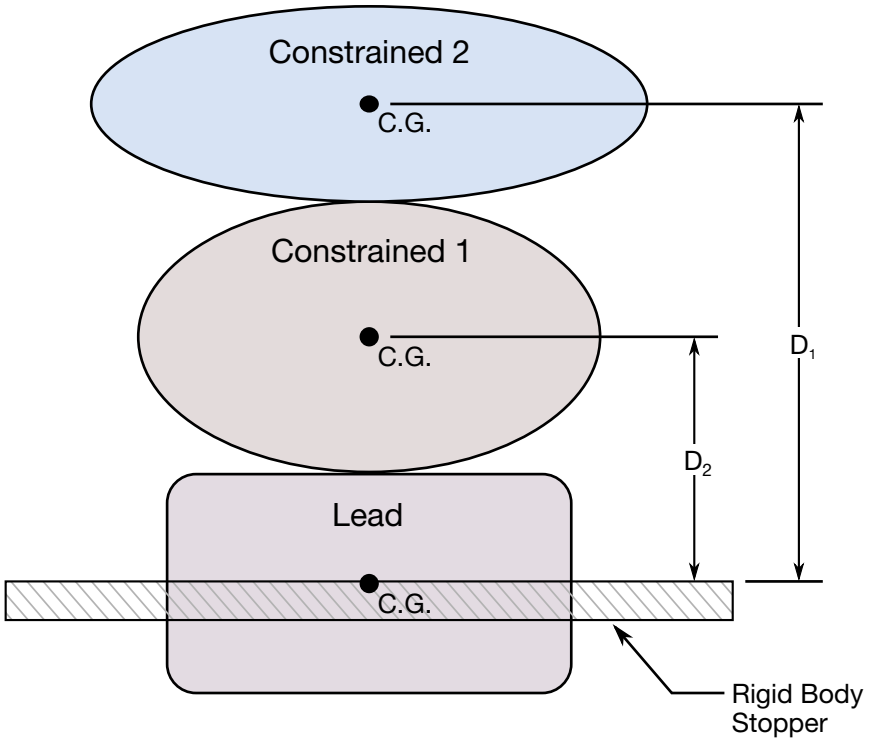


Figure 10-41. When the lead rigid body reaches the rigid body stopper, the velocity component into the stopper is set to zero. Constrained rigid bodies 1 and 2 also stop if the distance between their mass centers and the lead rigid body is less than or equal to the input values D_1 and D_2, respectively.

VARIABLE	DESCRIPTION
	EQ.5: x-axis rotation, EQ.6: y-axis rotation, EQ.7: z-axis rotation, EQ.8: Arbitrary, defined by vector VID (see below).
VID	Vector for arbitrary orientation of stopper, see *DEFINE_VECTOR.
TB	Time at which stopper is activated.
TD	Time at which stopper is deactivated.
STIFF	Augmentation stiffness for implicit; see Remark 2.

Remarks:

1. **Controlling the motion in an arbitrary direction.** By defining the optional part sets in the minimum or maximum coordinate direction (PSIDMX or PSIDMN), the motion can be controlled in an arbitrary direction.

2. **Augmentation stiffness.** For implicit, inequality constraints are imposed by using an augmented Lagrangian multiplier technique, where the augmentation stiffness can be specified with field STIFF. If STIFF is zero, then it is computed internally based on the loads and displacements associated with the stoppers. If this results in bad behavior, the stiffness may be selected so that STIFF = FORCE/DISP, where FORCE is a characteristic load and DISP is a characteristic length. A too large value may lead to inaccuracies and a too small value may lead to convergence problems. It is worth mentioning that this approach renders an indefinite matrix, and the number of negative eigenvalues seen by the linear solver is 2 (one constraint in positive and one in negative direction) per rigid body stopper.

*CONSTRAINED_RIVET_{OPTION}

To define an ID for the rivet, the following option is available:

<BLANK>

ID

If the ID is defined, an additional card is required.

Purpose: Define massless rivets between non-contiguous nodal pairs. The nodes must not have the same coordinates. The action is such that the distance between the two nodes is kept constant throughout any motion. No failure can be specified.

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	RID							
Type	I							
Default	0							

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	TF					
Type	I	I	F					
Default	none	none	10^{20}					
Remarks	1		2					

VARIABLE	DESCRIPTION
RID	Optional rivet ID
N1	Node ID
N2	Node ID

VARIABLE	DESCRIPTION
TF	Failure time for the nodal constraint set

Remarks:

1. **Multiple constraints.** Nodes connected by a rivet cannot be members of another constraint set that constrain the same degrees of freedom, a tied interface, or a rigid body, that is, nodes cannot be subjected to multiple, independent, and possibly conflicting constraints. Also, care must be taken to ensure that single point constraints applied to nodes in a constraint set do not conflict with the constraint sets constrained degrees of freedom.

2. **Failure time.** When the failure time, TF, is reached, the rivet becomes inactive, and the constrained nodes may move freely.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_RIVET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Connect node 382 to node 88471 with a massless rivet.
$
*CONSTRAINED_RIVET
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       n1         n2         tf
       382     88471        0.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*CONSTRAINED_SHELL_IN_SOLID_{OPTION1}_{OPTION2}

This keyword can take the following two forms:

*CONSTRAINED_SHELL_IN_SOLID

*CONSTRAINED_SHELL_IN_SOLID_PENALTY

To define a coupling ID and heading for this keyword, the following options are available:

ID

TITLE

Purpose: Provide either constraint-based or penalty-based coupling between shells and the solids/thick shells in which the shells are embedded.

To use the constraint-based coupling, this keyword takes the form of *CONSTRAINED_SHELL_IN_SOLID. It constrains shell structures to move with Lagrangian solids/thick shells. This keyword constrains both acceleration and velocity. This keyword, together with *CONSTRAINED_BEAM_IN_SOLID, intends to sidestep certain limitations of the CTYPE = 2 implementation in *CONSTRAINED_LAGRANGE_IN_SOLID. Notable features of this keyword include:

1. **Tetrahedral and pentahedral solid elements are supported.** They are no longer treated as degenerated hexahedra as in the CTYPE = 2 implementation.

2. **Velocity/Fixed boundary condition.** The CTYPE = 2 implementation failed to constrain shell nodes that were buried inside elements whose nodes had velocity/fixed boundary conditions prescribed.

3. **Optimized Sorting.** Sorting subroutine is optimized for larger problems to achieve better performance and less memory usage.

To use the penalty-based coupling, this keyword takes the form of *CONSTRAINED_SHELL_IN_SOLID_PENALTY. A penalty spring is attached between coupling points on the shell and in the solid/thick shell element. Penalty spring stiffness is calculated based on the geometric mean of shell and solid's bulk modulus. The magnitude of this coupling force can be controlled using PSSF (penalty spring stiffness scale factor). This penalty coupling conserves kinetic energy much better in transient problems such as blast loading.

If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SHSID	SSID	SHSTYP	SSTYP				
Type	I	I	I	I				
Default	none	none	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	START	END				PSSF		
Type	F	F				F		
Default	0.	10^{20}				0.1		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition.
SHSID	Set ID defining a part or part set ID of the Lagrangian shell structure (see *PART, *SET_PART).
SSID	Set ID defining a part or part set ID of the Lagrangian solid elements or thick shell elements (see *PART or *SET_PART).

VARIABLE	DESCRIPTION
SHSTYP	Set type of SHSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
SSTYP	Set type of SSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
START	Start time to activate the coupling: LT.0: Start time is set to |START|. When negative, start time is followed during the dynamic relaxation phase of the calculation. After the completion of dynamic relaxation, coupling is activated regardless of the value of END. EQ.0: Start time is inactive, meaning coupling is always active GT.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, if END > 0, start time applies both during and after dynamic relaxation.
END	End time to deactivate the coupling: LT.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, negative END indicates that coupling is inactive during dynamic relaxation. After dynamic relaxation the start and end times are followed and set to |START| and |END|, respectively. EQ.0: END defaults to 10^{20}. GT.0: END sets the time at which the coupling is deactivated.
PSSF	Penalty spring stiffness scale factor. Only available in penalty form.

*CONSTRAINED_SHELL_TO_SOLID

Purpose: Define a tie between a shell edge and solid elements. Nodal rigid bodies can perform the same function and may also be used.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	NSID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
NID	Shell node ID
NSID	Solid nodal set ID, see *SET_NODE_*OPTION*.

Remarks:

The shell-brick interface, an extension of the tied surface capability, ties regions of hexahedron elements to regions of shell elements. A shell node may be tied to up to nine brick nodes lying along the tangent vector to the nodal fiber. See Figure 10-42. During the

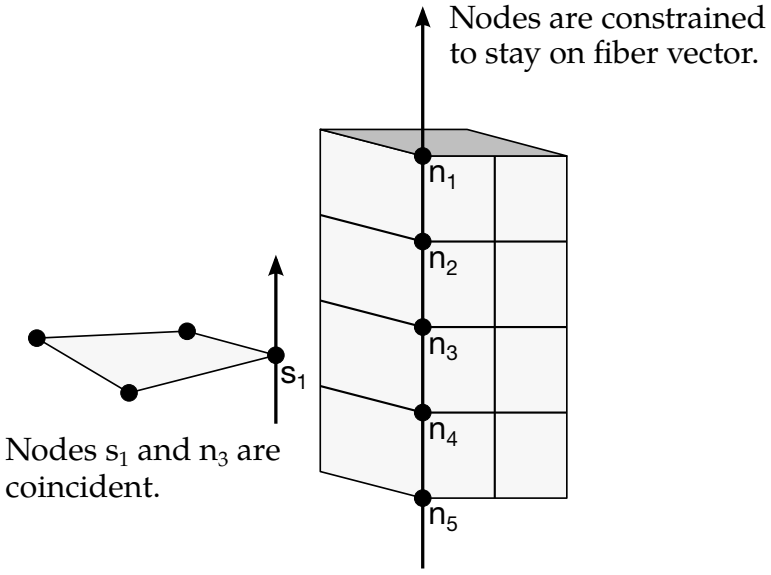


Figure 10-42. The interface between shell elements and solids ties a shell node, s_1, to a line of nodes on the solid elements, n_1 to n_5. It is very important for the nodes to be aligned.

calculation, the brick nodes thus constrained, must lie along the fiber but can move relative to each other in the fiber direction. The shell node stays on the fiber at the same relative spacing between the first and last brick node. The brick nodes must be input in the order in which they occur, in either the plus or minus direction, as one moves along the shell node fiber.

This feature is intended to tie four node shells to eight node shells or solids; it is not intended for tying eight node shells to eight node solids.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_SHELL_TO_SOLID
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Tie shell element, at node 329, to a solid element at node 203.
$    - nodes 329 and 203 are coincident
$
$  Additionally, define a line of nodes on the solids elements, containing
$  node 203, that must remain in the same direction as the fiber of the shell
$  containing node 329.  In other words:
$
$    - Nodes 119, 161, 203, 245 and 287 are nodes on a solid part that
$       define a line on that solid part.
$    - This line of nodes will be constrained to remain linear throughout
$       the simulation.
$    - The direction of this line will be kept the same as the fiber of the
$       of the shell containing node 329.
$
*CONSTRAINED_SHELL_TO_SOLID
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      nid      nsid
       329         4
$
*SET_NODE_LIST
$      sid
         4
$     nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
       119       161       203       245       287
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

***CONSTRAINED_SOIL_PILE**_{OPTION1}_{OPTION2}

Purpose: Define penalty-based coupling between 1D beam elements and 3D solid elements. The beam elements must not share nodes with the solid elements, and the beam element mesh does not have to align with the solid element mesh. The functionality is similar to *CONSTRAINED_BEAM_IN_SOLID, except that *CONSTRAINED_SOIL_-PILE is specific to geotechnical applications, such as piles in soil or anchors in rock. For general applications in which beam elements must be coupled with solids (such as reinforcement bars in concrete), *CONSTRAINED_BEAM_IN_SOLID is recommended due to its simpler input. See the general modeling requirements for *CONSTRAINED_SOIL_-PILE in Remarks 1 through 4.

The data cards described below specify the nonlinear coupling characteristics, expressing the resisting stress on the outer surface of the pile as a function of displacement of the pile relative to the soil. Movement in the axial (sliding) direction has a different coupling characteristic from movement perpendicular to the axis of the pile. In geotechnical terminology, the perpendicular coupling characteristics are sometimes described as "P-y springs" while the axial characteristics are described as "P-z springs". Additionally, a coupling characteristic is required for axial movement of the base of the pile, sometimes referred to as the "end-bearing" or the "toe" of the pile. Thus, the overall coupling depends on a set of three nonlinear coupling characteristics (base, axial, and perpendicular).

Where piles pass through layers of soils with different material properties, different coupling properties may be appropriate for each soil layer. To model this, include one *CONSTRAINED_SOIL_PILE keyword containing multiple sets of coupling properties. Each set of coupling properties is associated with a particular soil part ID or part set ID. LS-DYNA will then select the coupling properties for each node of the pile according to the part ID of the soil element within which that pile node is situated. LS-DYNA will automatically find the node at the free end of each pile for allocation of base coupling properties (see description of LOCAL below for which free end will be taken as the base).

OPTION1 specifies the format in which the coupling properties are defined on Cards 3 through 6. Available options are:

<BLANK>

CONSTANTS

CURVES

Selecting CURVES allows you to specify the properties with load curves while CONSTANTS allows you to specify the coupling with constant values. The input for the CURVES option is simpler than that of the CONSTANTS option, but the CONSTANTS option reduces the need to create load curves. If *OPTION1* is unset (<BLANK>), then the input is the same as for CONSTANTS, except that Card 2 is omitted. Card 2 allows you to add damping to the axial coupling shear stress and to pick which side of the pile is the

base. We recommend using CONSTANTS or CURVES for *OPTION1* instead of not setting it. <BLANK> is included solely for backwards compatibility.

OPTION2 specifies whether the soil parts to which the coupling properties apply are defined by part ID (see *PART) or by part set ID (see *SET_PART). Available options are:

<BLANK>

SET

If unset (<BLANK>), part IDs are assumed.

Card Summary:

Card Sets. Include one of Cards 1 and 2 (unless *OPTION1* is unset) total. Then for each soil part/part set include one set of either Cards 3a through 6a (or possibly 7a) or Cards 3b through 6b depending on *OPTION1*. Include as many sets as required to define the coupling properties for all the relevant soil layers. This input ends with the next keyword ("*") card.

Card 1. This card is required.

PBSID	DIAM		PIDNS	PIDNB	ERROR	NRING	NRINGB

Card 2. Include this card unless OPTION1 is unset (<BLANK>).

DAMP	LOCAL	INSTRF	TIMSTR	ICD7A	DAMPH		

Card 3a. Include this card when OPTION1 is set to CONSTANTS or not used (<BLANK>). Include as many sets of Card 3a through Card 6a (or possibly 7a) as needed.

PID/PSID	ACU	BCU	LCCU	ASTIFFS	BSTIFFS	ASTIFFB	ZREF

Card 4a. Include this card when OPTION1 is set to CONSTANTS or not used.

KBCON	KBCU	KBSX	KBSY	KBSZ	BSTFAC	BHYPER	BLC

Card 5a. Include this card when OPTION1 is set to CONSTANTS or not used.

KVCON	KVCU	KVSX	KVSY	KVSZ	VSTFAC	VHYPER	VLC

Card 6a. Include this card when OPTION1 is set to CONSTANTS or not used.

KHCON	KHCU	KHSX	KHSY	KHSZ	HSTFAC	HHYPER	HLC

Card 7a. Include this card only when OPTION1 is set to CONSTANTS and ICD7A on Card 2 is nonzero.

KHFCON	KHFCU	KHFSX	KHFSY	KHFSZ	FRATIO		

Card 3b. Include this card when OPTION1 is set to CURVES. Include as many sets of Card 3b through Card 6b as needed.

PID/PSID	ZREF						

Card 4b. Include this card when OPTION1 is set to CURVES.

BLCZ	BLC	BLCSH	BLCSV				

Card 5b. Include this card when OPTION1 is set to CURVES.

VLCZ	VLC	VLCSH	VLCSV				

Card 6b. Include this card when OPTION1 is set to CURVES.

HLCZ	HLC	HLCSH	HLCSV	HLCFD			FRATIO

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PBSID	DIAM		PIDNS	PIDNB	ERROR	NRING	NRINGB
Type	I	F		I	I	I	I	I
Default	none	0.0		auto	auto	0	1	↓

VARIABLE	DESCRIPTION
PBSID	Part set ID containing beam elements for coupling (the piles). See Remarks 2 and 3.
DIAM	Pile diameter (optional). If zero or blank, the pile diameter will be taken automatically from the section properties of the beam element. See Remarks 3 and 13.
PIDNS	ID for automatically generated part containing visualization elements for perpendicular and axial coupling. If not specified, LS-

VARIABLE	DESCRIPTION
	DYNA will assign a part ID. See Remarks 14 and 15.
PIDNB	ID for automatically generated part containing visualization elements for base coupling. If not specified, LS-DYNA will assign a part ID. See Remarks 14 and 15.
ERROR	Action taken if any coupling point is not constrained within a soil element: EQ.0: Stop with an error message. EQ.1: Warn and continue.
NRING	Number of coupling points around circumference at each pile node (see Remarks 11, 12 and 13): EQ.1: One coupling point coincident with pile node GT.1: NRING coupling points equally spaced around the circumference of the pile
NRINGB	Number of extra rings of coupling points on base, in addition to those around the pile circumference. By default, NRINGB is chosen automatically to distribute the base stress as uniformly as possible (see Remarks 11 and 12).

This card is included unless *OPTION1* is unset (<BLANK>).

Card 2	1	2	3	4	5	6	7	8
Variable	DAMP	LOCAL	INSTRF	TIMSTR	ICD7A	DAMPH		
Type	F	I	I	F	I	F		
Default	0.0	1	0	0.0	0	0.0		

VARIABLE	DESCRIPTION
DAMP	Optional damping coefficient for axial coupling (stress/velocity units). An additional axial coupling shear stress equal to DAMP times the axial velocity of the pile relative to the soil will be generated. See Remark 17.
LOCAL	Flag to identify which free end of a pile is treated as the base:

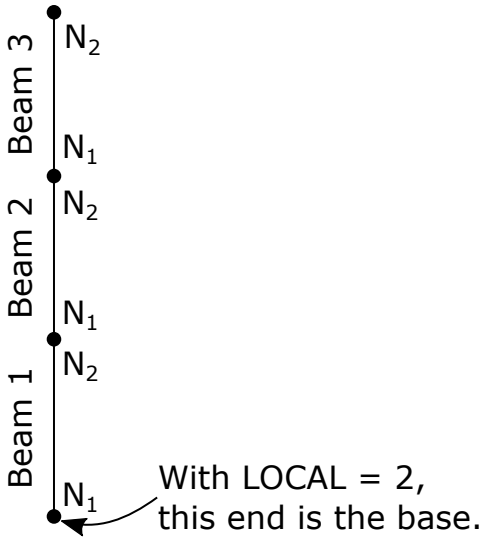


Figure 10-43. Example of a beam element topology in a pile. We recommend each beam having the same order for N_1 and N_2, so that the base is not ambiguous for LOCAL = 2.

VARIABLE	DESCRIPTION
	EQ.1: End with the most negative global Z-coordinate EQ.2: End which is Node 1 of the attached beam element topology. See Figure 10-43.
INSTRF	Flag to control definition of soil stress for Cards 4a through 6a and Cards 4b through 6b: EQ.0: Use time-varying soil stress. TIMSTR is ignored. EQ.1: Use time-varying soil stress until time TIMSTR. Then use the soil stress that existed at time TIMSTR for the remainder of the analysis.
TIMSTR	Time at which the effect of soil stress is frozen
ICD7A	If nonzero and *OPTION1* is CONSTANTS, Card 7a will be read.
DAMPH	Optional damping coefficient for perpendicular coupling (stress/velocity units). An additional perpendicular coupling shear stress equal to DAMPH times the perpendicular velocity of the pile relative to the soil will be generated. See Remark 17.

CONSTANTS General Coupling Properties Card. Include a set of Cards 3a through 6a for each soil part/part set when *OPTION1* is CONSTANTS or unset. Include as many sets as needed. See Remark 6.

Card 3a	1	2	3	4	5	6	7	8
Variable	PID/PSID	ACU	BCU	LCCU	ASTIFFS	BSTIFFS	ASTIFFB	ZREF
Type	I	F	F	I	F	F	F	F
Default	none	0.0	0.0	optional	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PID/PSID	Part ID or part set ID (depending on *OPTION2*) containing solid elements for coupling (the soil). See Remarks 4 and 16.
ACU	Constant term in depth-dependence formula. Units of stress.
BCU	Coefficient on relative Z-coordinate in depth-dependence formula. Units of stress/length. Note that soil strengths (and therefore coupling properties) generally increase with depth, meaning they increase with an increasingly negative Z-coordinate. Therefore, this term is usually negative.
LCCU	Optional load curve ID giving stress (stress units) as a function of relative Z-coordinate (length units). If defined, LCCU overrides ACU and BCU. Note that "increasing depth" corresponds to "increasingly negative relative Z-coordinate".
ASTIFFS	Generic stiffness term. Units of stress / length.
BSTIFFS	Generic Z-coordinate-dependent stiffness term. Units of stress / length^2
ASTIFFB	Base stiffness. Units of stress / length.
ZREF	Reference Z-coordinate to calculate "relative Z-coordinate". See Remark 16.

CONSTANTS Base Coupling Properties Card. Include a set of Cards 3a through 6a for each soil part/part set when *OPTION1* is CONSTANTS or unset. See Remarks 6 and 8.

Card 4a	1	2	3	4	5	6	7	8
Variable	KBCON	KBCU	KBSX	KBSY	KBSZ	BSTFAC	BHYPER	BLC
Type	F	F	F	F	F	F	F	I
Default	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0

VARIABLE	DESCRIPTION
KBCON	Base coupling, constant term (stress units)
KBCU	Base coupling, coefficient for Cu (dimensionless)
KBSX	Base coupling, coefficient for effective global *X*-stress (dimensionless)
KBSY	Base coupling, coefficient for effective global *Y*-stress (dimensionless)
KBSZ	Base coupling, coefficient for effective global *Z*-stress (dimensionless)
BSTFAC	Base coupling, factor on elastic stiffness (dimensionless)
BHYPER	Base coupling, hyperbolic curve limit (dimensionless)
BLC	Base coupling, load curve ID for dimensionless factor on stress as a function of displacement

CONSTANTS Axial Coupling Properties Card. Include a set of Cards 3a through 6a for each soil part/part set when *OPTION1* is CONSTANTS or unset. See Remarks 6 and 9.

Card 5a	1	2	3	4	5	6	7	8
Variable	KVCON	KVCU	KVSX	KVSY	KVSZ	VSTFAC	VHYPER	VLC
Type	F	F	F	F	F	F	F	I
Default	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0

VARIABLE	DESCRIPTION
KVCON	Axial coupling, constant term (stress units)
KVCU	Axial coupling, coefficient for Cu (dimensionless)
KVSX	Axial coupling, coefficient for effective global *X*-stress (dimensionless)
KVSY	Axial coupling, coefficient for effective global *Y*-stress (dimensionless)
KVSZ	Axial coupling, coefficient for effective global *Z*-stress (dimensionless)
VSTFAC	Axial coupling, factor on elastic stiffness (dimensionless)
VHYPER	Axial coupling, hyperbolic curve limit (dimensionless)
VLC	Axial coupling, load curve ID for dimensionless factor on stress as a function of displacement

CONSTANTS Perpendicular Coupling Properties Card. Include a set of Cards 3a through 6a for each soil part/part set when *OPTION1* is CONSTANTS or unset. See Remarks 6 and 10.

Card 6a	1	2	3	4	5	6	7	8
Variable	KHCON	KHCU	KHSX	KHSY	KHSZ	HSTFAC	HHYPER	HLC
Type	F	F	F	F	F	F	F	I
Default	0.0	0.0	0.0	0.0	0.0	1.0	0.0	none

VARIABLE	DESCRIPTION
KHCON	Perpendicular coupling, constant term (stress units)
KHCU	Perpendicular coupling, coefficient for Cu (dimensionless)
KHSX	Perpendicular coupling, coefficient for effective global *X*-stress (dimensionless)
KHSY	Perpendicular coupling, coefficient for effective global *Y*-stress (dimensionless)
KHSZ	Perpendicular coupling, coefficient for effective global *Z*-stress (dimensionless)
HSTFAC	Perpendicular coupling, factor on elastic stiffness (dimensionless)
HHYPER	Perpendicular coupling, hyperbolic curve limit (dimensionless)
HLC	Perpendicular coupling, load curve ID for dimensionless factor on stress as a function of displacement

CONSTANTS Perpendicular Coupling Friction Properties Card. This card defines resistance to perpendicular motion during unload/reload ("post-hole" behavior), described as "friction stress" below. Include a Card 7a after each set of Cards 3a through 6a when *OPTION1* is CONSTANTS and ICD7A on Card 2 is nonzero. See Remarks 6 and 10.

Card 7a	1	2	3	4	5	6	7	8
Variable	KHFCON	KHFCU	KHFSX	KHFSY	KHFSZ	FRATIO		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
KHFCON	Perpendicular coupling friction stress, constant term (stress units)
KHFCU	Perpendicular coupling friction stress, coefficient for Cu (dimensionless)
KHFSX	Perpendicular coupling friction stress, coefficient for effective global *X*-stress (dimensionless)
KHFSY	Perpendicular coupling friction stress, coefficient for effective global *Y*-stress (dimensionless)
KHFSZ	Perpendicular coupling friction stress, coefficient for effective global *Z*-stress (dimensionless)
FRATIO	Perpendicular coupling friction stress as a ratio of the maximum perpendicular coupling stress derived from KHCON through KHSZ on Card 6a. If defined, FRATIO overrides KHFCON through KHFSZ above. FRATIO provides an alternative way to define the friction stress which is simpler than KHFCON through KHFSZ

CURVES General Coupling Properties Card. Include a set of Cards 3b through 6b for each soil part/part set when *OPTION1* is CURVES. Include as many sets as needed.

Card 3b	1	2	3	4	5	6	7	8
Variable	PID/PSID	ZREF						
Type	I	F						
Default	none	0.0						

VARIABLE	DESCRIPTION
PID/PSID	Part ID or part set ID (depending on *OPTION2*) containing solid elements for coupling (the soil). See Remarks 4 and 16.
ZREF	Reference *Z*-coordinate, used in calculation of "relative *z*-coordinate". For example, ZREF may be located at the soil surface. See Remarks 7 and 16.

CURVES Base Coupling Properties Card. Include a set of Cards 3b through 6b for each soil part/part set when *OPTION1* is CURVES. See Remarks 7 and 8.

Card 4b	1	2	3	4	5	6	7	8
Variable	BLCZ	BLC	BLCSH	BLCSV				
Type	I	I	I	I				
Default	0	0	optional	optional				

VARIABLE	DESCRIPTION
BLCZ	For base coupling, load curve ID defining ultimate strength (stress units) as a function of relative *Z*-coordinate (length units)
BLC	For base coupling, load curve ID containing normalized mobilization curve: dimensionless factor on stress as a function of displacement
BLCSH	For base coupling, optional load curve ID containing coefficient for initial effective horizontal soil stress (dimensionless) as a function

VARIABLE	DESCRIPTION
	of relative Z-coordinate
BLCSV	For base coupling, optional load curve ID containing coefficient for initial effective vertical soil stress (dimensionless) as a function of relative Z-coordinate

CURVES Axial Coupling Properties Card. Include a set of Cards 3b through 6b for each soil part/part set when *OPTION1* is CURVES. See Remarks 7 and 9.

Card 5b	1	2	3	4	5	6	7	8
Variable	VLCZ	VLC	VLCSH	VLCSV				
Type	I	I	I	I				
Default	0	0	optional	optional				

VARIABLE	DESCRIPTION
VLCZ	For axial coupling, load curve ID defining ultimate strength (stress units) as a function of relative Z-coordinate (length units)
VLC	For axial coupling, load curve ID containing normalized mobilization curve: dimensionless factor on stress as a function of displacement
VLCSH	For axial coupling, optional load curve ID containing coefficient for initial effective horizontal soil stress (dimensionless) as a function of relative Z-coordinate
VLCSV	For axial coupling, optional load curve ID containing coefficient for initial effective vertical soil stress (dimensionless) as a function of relative Z-coordinate

CURVES Perpendicular Coupling Properties Card. Include a set of Cards 3b through 6b for each soil part/part set when *OPTION1* is CURVES. See Remarks 7 and 10.

Card 6b	1	2	3	4	5	6	7	8
Variable	HLCZ	HLC	HLCSH	HLCSV	HLCFD			FRATIO
Type	I	I	I	I	I			F
Default	0	0	optional	optional	optional			0.0

VARIABLE	DESCRIPTION
HLCZ	For perpendicular coupling, load curve ID defining ultimate strength (stress units) as a function of relative Z-coordinate (length units)
HLC	For perpendicular coupling, load curve ID containing normalized mobilization curve: dimensionless factor on stress as a function of displacement
HLCSH	For perpendicular coupling, optional load curve ID containing coefficient for initial effective horizontal soil stress (dimensionless) as a function of relative Z-coordinate
HSCSV	For perpendicular coupling, optional load curve ID containing coefficient for initial effective vertical soil stress (dimensionless) as a function of relative Z-coordinate
HLCFD	For perpendicular coupling, optional load curve ID containing frictional resistance (stress units) during "post-hole" unload/reload as a function of relative Z-coordinate. See Remark 10. With HLCFD there is no way to make the frictional resistance depend on soil stress.
FRATIO	Alternative method of defining frictional resistance for perpendicular coupling during "post-hole" unload/reload. FRATIO is the stress as a ratio of the maximum perpendicular coupling stress derived from HLCZ, HLCSH and HLCSV. If defined, FRATIO overrides HLCFD.

Remarks:

1. **General requirements.** This keyword is available only for 3D models with the explicit solver. The model's global Z-axis should be vertically upwards. If this is not the case, then it will not be possible to define depth-dependent coupling characteristics or to use the default of the LOCAL field for identifying the base of piles.

2. **Mesh requirements.** The beam elements must not share nodes with the solid elements. The beam element mesh does not have to align with the solid element mesh. Each pile must consist of contiguously meshed beam elements. One *CONSTRAINED_SOIL_PILE definition can connect multiple piles to multiple soil layers. Although piles are generally aligned vertically, there is no requirement for this to be the case in the model. We recommend having consistent beam element local axes along a pile. Thus, neighboring elements should have Node 1 and Node 2 the same way around, such as Node 1 always being below Node 2.

3. **Beam element properties.** Beam elements must be ELFORM 1 or 2. *CONSTRAINED_SOIL_PILE treats the beams as if they have a solid circular cross-section with diameter DIAM. If DIAM is nonzero, the same value is used for all the beam elements included in the coupling. If DIAM is zero or blank, the diameter of the assumed circular cross-section is calculated for each beam element separately from the area of the cross-section given on the *SECTION card. The actual section shape may be non-circular and/or non-solid, but then the surface area of the pile assumed by *CONSTRAINED_SOIL_PILE will be different from the true surface area. This difference will result in the coupling stress being multiplied by an incorrect area to obtain the force. In these cases, DIAM should be defined such that $\pi \times$ DIAM equals the perimeter of the pile cross-section. For the beam elements, rigid material is not permitted and constraint-based features such as Nodal Rigid Bodies or SPC boundary conditions must not be present on their nodes.

4. **Solid element properties.** The solid elements must not have more than 8 nodes. There is no other restriction on element type or material type. Rigid material type is permitted, and there is no restriction regarding constraints or other features present on the nodes of the solids.

5. **Coupling characteristics: general.** The coupling characteristics are defined in terms of stress on the outer surface of the pile as nonlinear functions of displacement of the pile relative to the soil. Coupling stress is calculated separately at each coupling point (coupling point locations are discussed in Remark 11). The response under monotonically increasing displacement is defined by a "backbone curve", which is defined differently according to whether *OPTION1* is CONSTANTS (Remark 6) or CURVES (Remark 7). Backbone curves for all three

coupling characteristics (base, axial, and perpendicular) follow the same pattern but with different input parameters. Unload/reload behavior is described in Remarks 8 through 10.

6. **Backbone coupling curves when *OPTION1* is CONSTANTS or unset.** The coupling follows an elastic/plastic relation with an initial elastic stiffness and a yield stress. The elastic stiffnesses (stress/displacement) for base, axial, and perpendicular coupling (S_{Base}, S_{Ax} and S_{Perp}, respectively) are defined as follows:

$$S_{\text{Base}} = \text{ASTIFFB} \times \text{BSTFAC}$$
$$S_{\text{Ax}} = (\text{ASTIFFB} + \text{BSTIFF} \times z_r) \times \text{VSTFAC}$$
$$S_{\text{Perp}} = (\text{ASTIFF} + \text{BSTIFF} \times z_r) \times \text{HSTFAC}$$

Here z_r is the relative Z-coordinate defined as $Z_0 - \text{ZREF}$. Z_0 is the initial Z-coordinate of the pile node.

The yield stresses for base, axial, and perpendicular coupling ($\sigma_{Y,\,\text{Base}}$, $\tau_{Y,\,\text{Ax}}$ and $\sigma_{Y,\,\text{Perp}}$, respectively) are defined as follows:

$$\sigma_{Y,\,\text{Base}} = f(d) \times \left\{\text{KBCON} + \text{KBCU} \times \text{Cu}(z_r) + \text{KBSX} \times \sigma'_x + \text{KBSY} \times \sigma'_y + \text{KBSZ} \times \sigma'_z\right\}$$
$$\tau_{Y,\,\text{Ax}} = f(d) \times \left\{\text{KVCON} + \text{KVCU} \times \text{Cu}(z_r) + \text{KVSX} \times \sigma'_x + +\text{KVSY} \times \sigma'_y + \text{KVSZ} \times \sigma'_z\right\}$$
$$\sigma_{Y,\text{Perp}} = f(d) \times \left\{\text{KHCON} + \text{KHCU} \times \text{Cu}(z_r) + \text{KHSX} \times \sigma'_x + \text{KHSY} \times \sigma'_y + \text{KHSZ} \times \sigma'_z\right\}$$

Here d is relative displacement (see Remark 12), z_r is the relative Z-coordinate as defined above, and σ'_x, σ'_y and σ'_z are the soil stresses in the global X, Y and Z directions, respectively. Depending on INSTR and TIMSTR (see Card 2), either time-varying soil stresses or the soil stresses at time TIMSTR are used in this calculation. Note that INSTR and TIMSTR were introduced in R14. In R13, the CONSTANTS keyword option always used current (time-varying) soil stress. If pore water is modeled (see *CONTROL_PORE_FLUID), then these are "effective stresses", meaning the stress generated by the material model excluding any pore pressure. Note that *CONSTRAINED_SOIL_PILE does not require pore pressure to be modelled.

The depth-dependence function, $\text{Cu}(z_r)$, is defined as follows: If LCCU is zero or blank, $\text{Cu}(z_r) = \text{ACU} + \text{BCU} \times z_r$. Otherwise, $\text{Cu}(z_r) = \text{LCCU}(z_r)$.

The function $f(d)$ has three possible forms, illustrated here for base coupling (input parameters BLC and BHYPER), but the same rules apply for axial coupling (VLC and VHYPER) and perpendicular coupling (HLC and HHYPER).

a) If BHYPER and BLC are both zero, $f(d)$ is unity, resulting in a bilinear elastic-perfectly-plastic coupling characteristic.

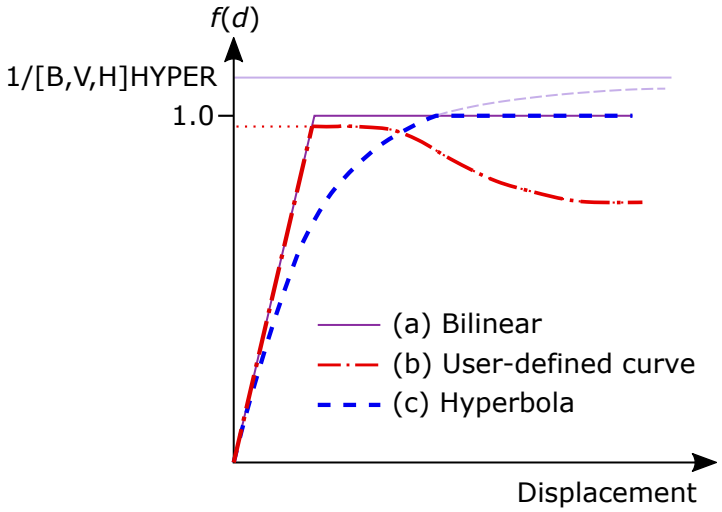


Figure 10-44. Forms of $f(d)$ available when *OPTION1* is CONSTANTS or unset.

b) If the load curve BLC is nonzero, $f(d)$ is the value of the load curve. BLC is an optional load curve (see *DEFINE_CURVE) giving a non-dimensional factor on yield stress as a function of displacement.

c) If BHYPER is nonzero (typical value: 0.95), the stress as a function of displacement follows a hyperbolic curve:

$$f(d) = \min\left[1.0, \quad {}^{d_{\max}}\Big/_{\text{BHYPER} \times (d_{\text{elastic}} + d_{\max})}\right] .$$

Here $d_{\max}$ is the maximum displacement that has occurred so far, and d_{elastic} is the elastic displacement calculated (for the example of base coupling) from $\sigma_{\max}/S_{\text{Base}}$. $\sigma_{\max}$ is the stress term in curly brackets in the equation for $\sigma_{Y,Base}$ above.

The three options for $f(d)$ are illustrated in Figure 10-44.

7. **Backbone coupling curves when *OPTION1* is CURVES.** The coupling stresses for base, axial, and perpendicular coupling (σ_{Base}, τ_{Ax}, and σ_{Perp}, respectively) are defined as follows:

$$\sigma_{\text{Base}} = \left\{\text{BLCZ}(z_r) + 0.5\left(\sigma'_x + \sigma'_y\right) \times \text{BLCSH}(z_r) + \sigma'_z \times \text{BLCSV}(z_r)\right\} \times \text{BLC}(d)$$
$$\tau_{\text{Ax}} = \left\{\text{VLCZ}(z_r) + 0.5\left(\sigma'_x + \sigma'_y\right) \times \text{VLCSH}(z_r) + \sigma'_z \times \text{VLCSV}(z_r)\right\} \times \text{VLC}(d)$$
$$\sigma_{\text{Perp}} = \left\{\text{HLCZ}(z_r) + 0.5\left(\sigma'_x + \sigma'_y\right) \times \text{HLCSH}(z_r) + \sigma'_z \times \text{HLCSV}(z_r)\right\} \times \text{HLC}(d)$$

Here d is relative displacement (see Remark 12). z_r is the relative Z-coordinate defined as $Z_0 - \text{ZREF}$. Z_0 is the initial Z-coordinate of the pile node). σ'_x, σ'_y and σ'_z are the *initial* stresses in the global X, Y and Z directions. Depending on INSTRF and TIMSTR (see Card 2), either time-varying soil stresses or the soil stresses at time TIMSTR are used in this calculation. Note that INSTR and TIMSTR were introduced in R14. In R13, the CURVES option always used initial soil

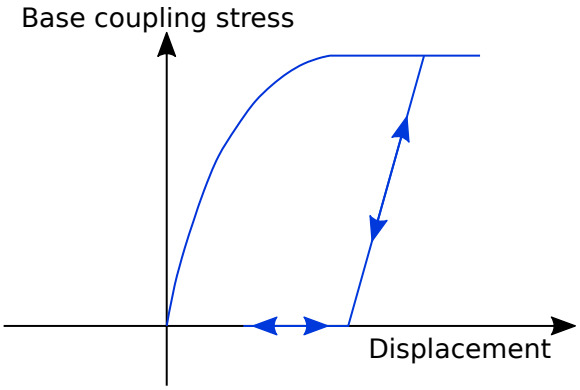


Figure 10-45. Base coupling unload/reload behavior

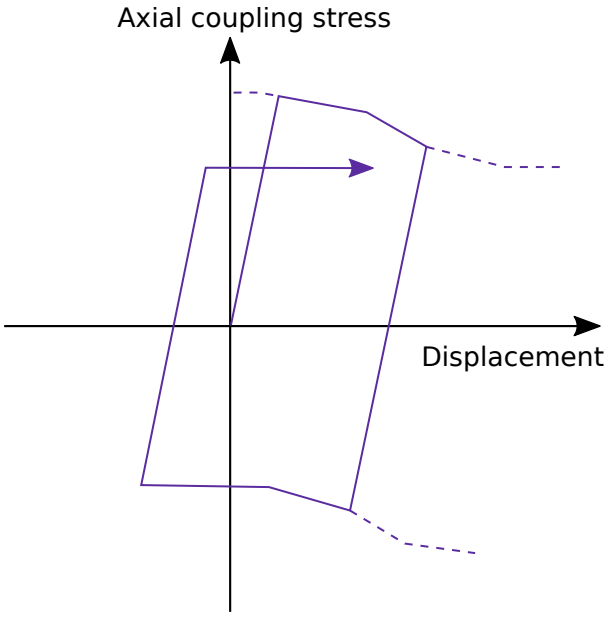


Figure 10-46. Axial coupling unload/reload behavior

stress (that is, the soil stress at time zero). If pore water is modeled (see *CONTROL_PORE_FLUID), then these are "effective stress," meaning the stress generated by the material model, excluding any pore pressure. Note that *CONSTRAINED_SOIL_PILE does not require pore pressure to be modelled.

The "normalized mobilization curves" BLC, VLC, HLC, if defined, must have a first point at (0,0). The initial elastic coupling stiffness is determined from the above equations using the gradient of the line between the first and second points of the normalized mobilization curve. If any of the curves (BLCZ, BLC, VLCZ, HLCSH, etc.) are left blank, the corresponding term in the above equations is set to zero. For example, if BLC is left blank, then the base coupling stress will always be zero, meaning no base coupling.

8. **Directionality of loading and unload/reload behavior: base coupling.** Base coupling resists relative movement only in the axial direction corresponding to

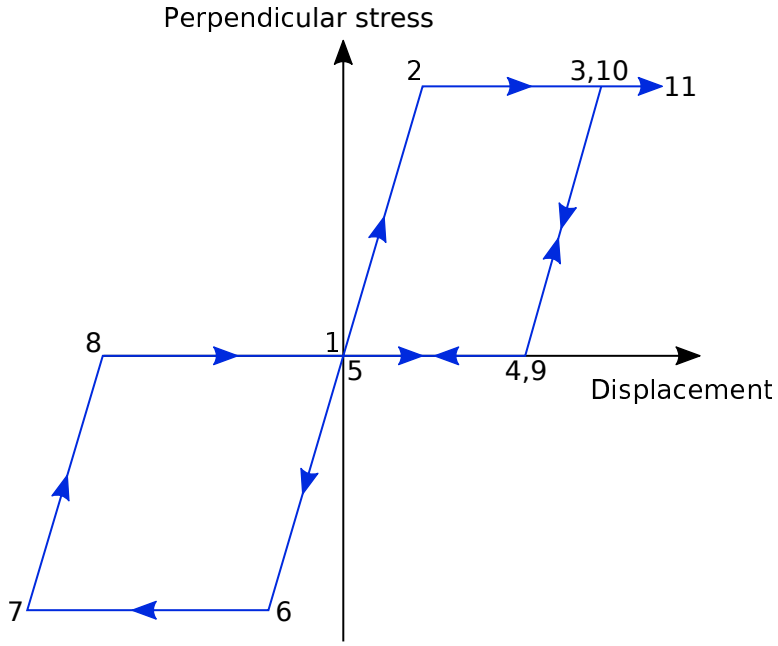


Figure 10-47. Perpendicular coupling unload/reload behavior

a pile being pressed downwards into the soil. This coupling results in a compressive axial load on the pile. Uplift is not resisted and results in a gap between the base of the pile and the soil. On reloading, any gap must close before compressive loading of the pile can resume. Figure 10-45 illustrates this unloading/reloading behavior for when BHYPER on Card 4a specifies the shape of the coupling.

9. **Directionality of loading and unload/reload behavior: axial coupling.** Axial coupling shear stresses resist sliding of the pile in both axial directions. An elasto-plastic approach is adopted. Thus, if the direction of axial movement is reversed, stresses also reverse. Figure 10-46 illustrates a possible response when a user-defined mobilization curve VLC is specified.

10. **Directionality of loading and unload/reload behavior: perpendicular coupling.** Perpendicular coupling stresses are calculated such that, if a cyclical back-and-forth motion is applied to a pile, the effect is that of an elongating hole (sometimes called "post-hole" behavior named for the observed behavior of the hole in the ground when a fence-post is repeatedly bent in one direction and then the other). As an illustration, in Figure 10-47, HLC and HHYPER are left blank, and a cycling motion is applied. If additional frictional resistance is defined (Card 7a for the CONSTANTS option; input parameters HLCFD or FRATIO for CURVES option), the friction stress is applied as an offset in the y-axis direction to the sections of the curve between points 4 and 5 and between points 8 and 9.

Movement could potentially occur at any angle in the plane perpendicular to the pile axis. LS-DYNA tracks the movement and gaps in 8 directions at 45-degree

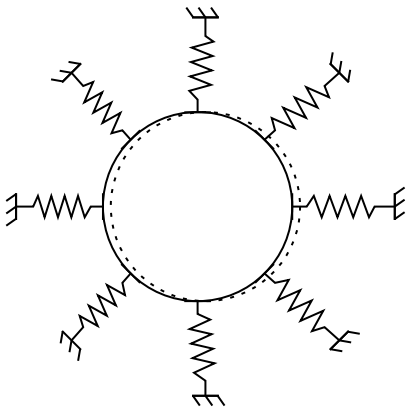


Figure 10-48. Example of how perpendicular motion in a pile is tracked.

intervals around the pile circumference. For these directions, LS-DYNA applies the input coupling characteristics. For intermediate angles of movement, an interpolation method is used resulting in small differences between the input stress-displacement coupling characteristic and the one achieved by the model. For instance, the resistance to movement at 20 degrees would be a combination of that at 0-degrees and 45-degrees. See Figure 10-48.

11. **Location of coupling points.** LS-DYNA calculates the coupling stresses at each coupling point using the input coupling characteristics. By default (NRING = 0 or 1), one coupling point is coincident with each pile node. We recommend the default when the solid element size is greater than or equal to the pile diameter. Where the soil elements are smaller than the pile diameter, we recommend distributing the coupling stresses onto all the soil elements through which the outer surface of the pile passes. To do this, specify a value of NRING > 1 to define a ring of coupling points level with each pile node around the circumference of the pile. Typically, you should select a value of NRING such that about 1 or 2 coupling points are present in each soil element on the circumference of the pile. Irrespective of the setting of NRING, we recommend NRINGB = 0, meaning that the coupling points on the base of the pile are automatically chosen to obtain an approximately uniform distribution. NRING and NRINGB influence the coupling point mesh only in the plane perpendicular to the pile axis. There is no option to refine the coupling point mesh in the axial direction. See Figure 10-49 for illustration.

12. **Definition of displacement of the pile relative to the soil.** Relative displacement at a given coupling point is the difference of displacement between a pile point and a soil point. The pile point displacement is calculated as if a rigid link were present between the pile node and the coupling point. Therefore, when NRING > 1, the rotation of the pile node times the radial offset between pile node and coupling point contributes to the displacement. The soil point retains a constant relative position (iso-parametric coordinates) within the soil element.

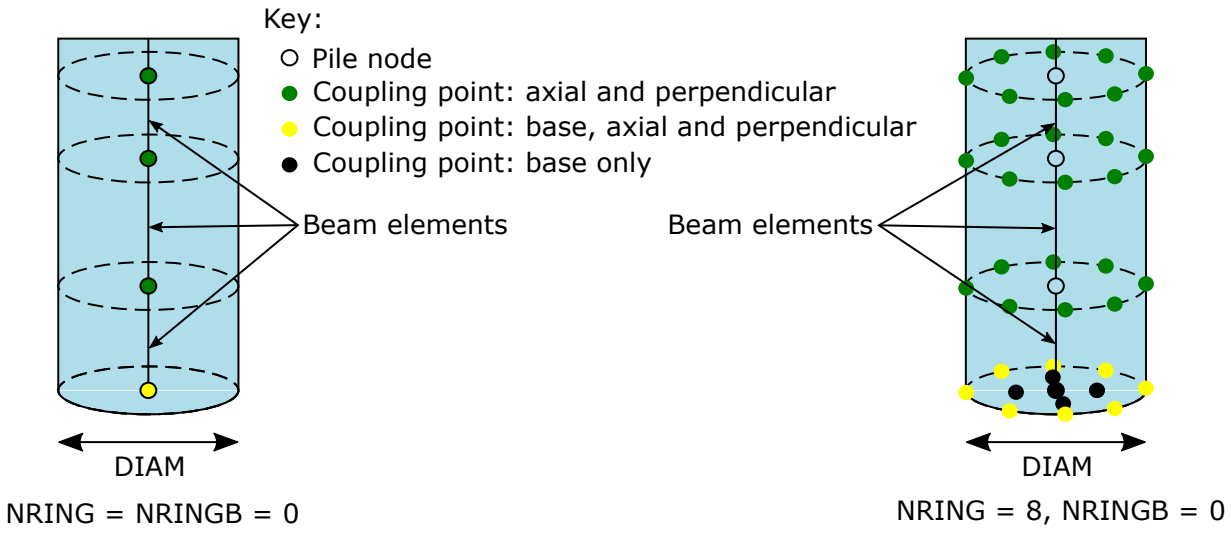


Figure 10-49. Location of coupling points

These calculations do not create any additional constraints on the pile node or soil nodes.

13. **Relation between coupling stress and force on the pile.** The base, axial, and perpendicular coupling forces are calculated at each coupling point from the coupling stresses as follows:

$$F_{\text{Base}} = \sigma_{\text{Base}} A_{\text{Bp}}$$
$$F_{\text{Ax}} = \tau_{\text{Ax}} \times \pi D \times 0.5(L_1 + L_2)/\text{NRING}$$
$$F_{\text{Perp}} = \sigma_{\text{Perp}} \times D \times 0.5(L_1 + L_2)/\text{NRING}$$

Here A_{Bp} is the area associated with a base coupling point, D is the pile diameter (DIAM), and L_1 and L_2 are the lengths of the two pile beam elements that meet at the pile node in question. Note that the axial force is based on the whole perimeter being loaded by the shear stress, while the perpendicular force is based on the stress times the area of the pile projected in the perpendicular direction.

The coupling forces are applied to the pile node and reacted on the nodes of the soil element containing the coupling point. If NRING > 1, moments arising from the coupling force times the radial offset between the pile node and the coupling point are applied to the pile node.

14. **Coupling visualization elements.** LS-DYNA automatically creates a beam element at each coupling point. These are for visualizing the coupling stresses only; they do not create additional forces in the solution. LS-DYNA automatically creates *PART, *SECTION_BEAM and *MAT_NULL cards for the visualization elements. If desired, the IDs of the automatically generated *PART cards may be specified with input fields PIDNS and PIDNB.

It is permitted, but not necessary, to define the *PART cards with IDs PIDNS and PIDNB in the keyword file. In this case, the two *PARTs should reference a *SECTION_BEAM with ELFORM = 6 and a *MAT_NULL. The density in the material keyword input, the volume in the section input, and the inertia in the section input should have small, nonzero values.

15. **Viewing the coupling stresses.** In the d3plot output file, the coupling visualization beam elements contain fake forces which are actually the coupling stresses. For the elements in part PIDNS, axial force (or X-Force) is really the perpendicular coupling stress in the local x direction; Y-Shear (or Y-Force) is really the perpendicular coupling stress in the local y direction; and, Z-Shear (or Z-Force) is really the axial coupling stress.

    For the elements in part PIDNB, the axial force (or X-Force) and Y-Shear (or Y-Force) are zero, and the Z-Shear (or Z-Force) is really the base coupling stress.

    The nodes of the visualization beam elements stay coincident with the pile and soil coupling points mentioned in Remark 11 throughout the analysis, enabling visualization of the pile/soil relative displacements (see Remark 12).

16. **Depth-dependent properties.** Soil stiffness and strength generally increase with depth. Soil-pile coupling properties are related to soil properties, and therefore, they also tend to increase with depth. *CONSTRAINED_SOIL_PILE offers options for defining coupling properties that vary with depth (expressed via functions of Z-coordinate relative to ZREF), but depth-dependent coupling could also be achieved by defining multiple layers of soil, each with a different part ID, and then defining unique but non-depth-dependent properties for each layer.

17. **Damping.** Damping of the axial and perpendicular responses may be defined using DAMP and DAMPH, respectively. These fields are not usually required for quasi-static simulations for which other damping methods are available, so DAMP and DAMPH are usually set to zero. However, damping may sometimes be desirable in some dynamic simulations.

18. **Mass-scaling.** LS-DYNA checks the numerical stability of the coupling, based on the coupling stiffness and the masses of the pile and soil nodes. If the model uses mass-scaling (negative DT2MS on *CONTROL_TIMESTEP), and if a coupling point would not otherwise be stable at the current timestep, LS-DYNA will automatically add mass to the pile and soil nodes to ensure stability. The mass added by *CONSTRAINED_SOIL_PILE during the first timestep is printed to the d3hsp file; search for "added mass for *CONSTRAINED_SOIL_PILE". If the amount of added mass seems excessive, consider reducing the initial elastic coupling stiffness (described in Remarks 6 and 7).

19. **Calibration of coupling properties.** The nonlinear coupling properties in *CONSTRAINED_SOIL_PILE account for the part of the deformation of the soil that is not captured explicitly by the soil mesh. The finer the soil mesh, the more localized deformation around the pile will be captured by the soil elements, and therefore, the less deformation needs to be accounted for by the coupling properties. Thus, realistic input properties for *CONSTRAINED_SOIL_PILE depend on soil mesh size. It is advisable to calibrate the input properties against a detailed model in which the pile (or part of a pile) is represented by solid elements embedded in finely-meshed soil and subjected to axial and perpendicular loading cases.

20. **Staged construction and the dynain file.** The dynain files written during a Staged Construction analysis (see *CONTROL_STAGED_CONSTRUCTION) contain a keyword called *INITIAL_SOIL_PILE_DATA, which enables the *CONSTRAINED_SOIL_PILE coupling stresses and history data to be carried forward into further analyses.

***CONSTRAINED_SOLID_IN_SOLID**_{OPTION1}_{OPTION2}

This keyword can take the following two forms:

*CONSTRAINED_SOLID_IN_SOLID

*CONSTRAINED_SOLID_IN_SOLID_PENALTY

To define an ID and heading, the following options are available:

ID

TITLE

Purpose: Provide either constraint-based or penalty-based coupling between solids and the solids/thick shells in which the solids are embedded. It is one of the keywords in the *CONSTRAINED_BEAM/SHELL/SOLID_IN_SOLID family.

To use the constraint-based coupling, this keyword takes the form of *CONSTRAINED_-SOLID_IN_SOLID. It constrains solid structures to move with Lagrangian solids/thick shells. This keyword constrains both acceleration and velocity. This keyword, together with *CONSTRAINED_BEAM_IN_SOLID and *CONSTRAINED_SHELL_IN_SOLID, intends to sidestep certain limitations of the CTYPE = 2 implementation in *CON-STRAINED_LAGRANGE_IN_SOLID. Notable features of this keyword include:

1. **Tetrahedral and pentahedral solid elements are supported.** They are no longer treated as degenerated hexahedra as in the CTYPE = 2 implementation.

2. **Velocity/Fixed Boundary Condition.** The CTYPE = 2 implementation fails to constrain solid nodes that are buried inside elements whose nodes have pre-scribed velocity/fixed boundary conditions.

3. **Optimized Sorting.** The sorting subroutine is optimized for larger problems to achieve better performance and less memory usage.

To use the penalty-based coupling, this keyword takes the form of *CONSTRAINED_-SOLID_IN_SOLID_PENALTY. A penalty spring is attached between coupling points on the solid and in the solid/thick shell element. Penalty spring stiffness is calculated based on the geometric mean of the two bulk moduli of the interacting elements. The magni-tude of this coupling force can be controlled using PSSF (penalty spring stiffness scale factor). This penalty coupling conserves kinetic energy much better in transient prob-lems, such as blast loading.

If a title is not defined, LS-DYNA will automatically create an internal title for this cou-pling definition.

Title Card. Additional card for TITLE and ID keyword options

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A						

Card 1	1	2	3	4	5	6	7	8
Variable	SSIDA	SSIDB	SSTYPA	SSTYPB				
Type	I	I	I	I				
Default	none	none	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	START	END				PSSF		
Type	F	F				F		
Default	0.	10^{20}				0.1		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number. If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition
SSIDA	Set ID defining a part or part set ID of the Lagrangian solid structure constrained to move with solid or thick shell elements specified with SSIDB (see *PART and *SET_PART).
SSIDB	Set ID defining a part or part set ID of the Lagrangian solid elements or thick shell elements which constrain SSIDA (see *PART and *SET_PART).

VARIABLE	DESCRIPTION
SSTYPA	Set type of SSIDA: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
SSTYPB	Set type of SSIDB: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
START	Start time to activate the coupling: LT.0.0: Start time is set to |START|. When negative, start time is followed during the dynamic relaxation phase of the calculation. After the completion of dynamic relaxation, coupling is active regardless of the value of END. EQ.0.0: Start time is inactive, meaning coupling is always active GT.0.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, if END > 0, start time applies both during and after dynamic relaxation.
END	End time to deactivate the coupling: LT.0.0: If END = -9999, START is interpreted as the curve or table ID defining multiple pairs of start-time and end-time. Otherwise, negative END indicates that coupling is inactive during dynamic relaxation. After dynamic relaxation, the start and end times are followed and set to |START| and |END|, respectively. EQ.0.0: END defaults to 10^{20}. GT.0.0: END sets the time at which the coupling is deactivated.
PSSF	Penalty spring stiffness scale factor. Only available in penalty form.

*CONSTRAINED_SPLINE

Purpose: Define an elastic cubic spline interpolation constraint. The displacements and slopes at the end points are continuous. The first and last nodes, which define the constraint, must be independent. The degrees-of-freedom of interior nodes may be either dependent or independent.

Card 1	1	2	3	4	5	6	7	8
Variable	SPLID	DLRATIO						
Type	I	I						
Default	0	0.10						

Node Cards. Include one card per independent/dependent node. The first and last nodes must be independent. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	NID	DOF						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
SPLID	Spline constraint ID
DLRATIO	Ratio of bending to torsional stiffness for an elastic tubular beam which connects the independent degrees-of-freedom. The default value is 0.10.
NID	Independent/dependent node ID. For explicit problems this node should not be a member of a rigid body or elsewhere constrained in the input.
DOF	Degrees-of-freedom. The list of dependent degrees-of-freedom consists of a number with up to six digits, with each digit representing a degree of freedom. For example, the value 1356 indicates

VARIABLE	DESCRIPTION
	that degrees of freedom 1, 3, 5, and 6 are controlled by the constraint. The default is 123456. Digit degree of freedom IDs: EQ.1: x EQ.2: y EQ.3: z EQ.4: Rotation about x-axis EQ.5: Rotation about y-axis EQ.6: Rotation about z-axis

*CONSTRAINED_SPOTWELD_{OPTION}_{OPTION}

To use a time-filtered force calculation for the forced-based failure criterion, the following option is available:

FILTERED_FORCE

To define an ID for the spot weld, the following option is available:

ID

The order of the options in the keyword name is arbitrary. Both options are optional.

Purpose: Define massless spot welds between non-contiguous nodal pairs.

The spot weld is a rigid beam that connects the nodal points of the nodal pairs; thus, nodal rotations and displacements are coupled. The spot welds must be connected to nodes having rotary inertias, that is, beams or shells. If this is not the case, for example, if the nodes belong to solid elements, use the option: *CONSTRAINED_RIVET. During implicit calculations this case is treated like a rivet, constraining only the displacements. Note that shell elements do not have rotary stiffness in the normal direction and, therefore, this component cannot be transmitted.

Spot welded nodes must not have the same coordinates. Coincident nodes in a spot weld can be handled by *CONSTRAINED_NODAL_RIGID_BODY. Brittle and ductile failures are supported by this model. Brittle failure is based on the resultant forces acting on the weld, and ductile failure is based on the average plastic strain value of the shell elements that include the spot welded node. Spot welds that are connected to massless nodes are automatically deleted in the initialization phase, and a warning message is printed in the messag file and the d3hsp file.

WARNING: The accelerations of spot welded nodes are output as zero into the various databases, but if the accelerations of spot welded nodes are required, use either the *CONSTRAINED_GENERALIZED_WELD or the *CONSTRAINED_NODAL_RIGID_BODY input. However, if the output interval is frequent enough, accurate acceleration time histories can be obtained from the velocity time history by differentiation in the post-processing phase.

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	WID							
Type	I							
Default	0							

Card 1	1	2	3	4	5	6	7	8
Variable	N1	N2	SN	SS	N	M	TF	EP
Type	I	I	F	F	F	F	F	F
Default	none	none	0.0	0.0	none	none	10^{20}	10^{20}

Filter Card. Additional card for the FILTERED_FORCE keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	NF	TW						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
WID	Optional weld ID.
N1	Node ID. See Remarks 1 and 5.
N2	Node ID. See Remarks 1 and 5.
SN	Normal force, S_n, at spot weld failure (see Remark 2 below). EQ.0.0: The failure criterion is disabled. GT.0.0: Normal force at spot weld failure

VARIABLE	DESCRIPTION
	LT.0.0: Curve ID which specifies the normal force at spot weld failure as a function of the nodal temperature
SS	Shear force, S_s, at spot weld failure (see Remark 2 below). EQ.0.0: The failure criterion is disabled. GT.0.0: Shear force at spot weld failure LT.0.0: Curve ID which specifies the shear force at spot weld failure as a function of the nodal temperature
N	Exponent for normal spot weld force (see Remark 2 below).
M	Exponent for shear spot weld force (see Remark 2 below).
TF	Failure time for nodal constraint set. See Remark 3.
EP	Effective plastic strain at failure. See Remark 4.
NF	Number of force vectors stored for filtering.
TW	Time window for filtering.

Remarks:

1. **Spot weld nodes and constraints.** Nodes connected by a spot weld cannot be members of another constraint set that constrains the same degrees of freedom, a tied interface, or a rigid body, meaning nodes cannot be subjected to multiple, independent, and possibly conflicting constraints. Also, care must be taken to ensure that single-point constraints applied to nodes in a constraint set do not conflict with the constraint sets constrained degrees of freedom.

2. **Spot weld failure criteria.** Failure of the spot welds occurs when:

$$\left(\frac{|f_n|}{S_n}\right)^n + \left(\frac{|f_s|}{S_s}\right)^m \geq 1 \ ,$$

where f_n and f_s are the normal and shear interface forces. Component f_n is non-zero for tensile values only. When SN and SM are specified by curve IDs, failure of the spot welds occurs when:

$$\left(\frac{|f_n|}{S_n(T)}\right)^n + \left(\frac{|f_s|}{S_s(T)}\right)^m \geq 1 \ ,$$

where T is the nodal temperature.

3. **Failure time.** When the failure time, TF, is reached, the spot weld becomes inactive, and the constrained nodes may move freely.

4. **Material plastic strain failure.** Spot weld failure due to plastic straining occurs when the effective nodal plastic strain exceeds the input value, ε^{p}_{fail}. This option can model the tearing out of a spotweld from the sheet metal since the plasticity is in the material that surrounds the spot weld, not the spot weld itself. A least squares algorithm is used to generate the nodal values of plastic strains at the nodes from the element integration point values. The plastic strain is integrated through the element and the average value is projected to the nodes via a least square fit. This option should only be used for the material models related to metallic plasticity and can result is slightly increased run times. Failures can include both the plastic and brittle failures.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_SPOTWELD
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Spotweld two nodes (34574 and 34383) with the approximate strength
$  of a 3/8" SAE Grade No 3 bolt.
$
*CONSTRAINED_SPOTWELD
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       n1        n2        sn        sf         n         m        tf        ps
     34574     34383      36.0      18.0       2.0       2.0       10.       1.0
$
$
$       sn = 36.0  normal failure force is 36 kN
$       sf = 18.0  shear failure force is 18 kN
$        n = 2.0   normal failure criteria is raised to the power of 2
$        m = 2.0   shear failure criteria is raised to the power of 2
$       tf = 10.0  failure occurs at time 10 unless strain failure occurs
$       ps = 2.0   plastic strain at failure
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

5. **Thermal problem.** The two nodes identified by this keyword are constrained to the same temperature in a thermal problem or in a coupled thermal-mechanical problem.

*CONSTRAINED_SPR2_{OPTION}

Available options include:

<BLANK>

TITLE

Purpose: Define a self-piercing rivet with failure. This self-piercing rivet (SPR2) model includes a plastic-like damage model that reduces the force and moment resultants to zero as the rivet fails. A diameter approximately equal to the rivet's diameter gives the domain of influence.

The location of the rivet is defined by a single node at the center of two riveted sheets. The algorithm performs a normal projection from the upper and lower sheets to the rivet node and locates all nodes within the user-defined diameter of influence. L. Olovsson of Impetus Afea developed the numerical implementation of this rivet model based on research on SPR point connector models originally carried out by SIMLab (NTNU) and SINTEF; see references by Porcaro, Hanssen, et al. [2006, 2006, 2007].

Initially, only two sheets (upper and lower) could be connected with one SPR2 node. But since release R9, up to 6 sheets can be connected with one SPR2 node by defining additional parts on optional Card 4. The following stacking sequence should be used: UPID – XPID1 – XPID2 – XPID3 – XPID4 – LPID. Omitted parts can be left blank. For instance, for a 3-sheet connection, the extra part lies between the upper and lower parts, and for a regular 2-sheet connection, Card 4 can be omitted entirely.

Starting with R13, not only shell element parts can be connected using this approach, but also solid elements. However, the new capability is still experimental and should be treated carefully. For instance, the solid element parts still having some kind of obvious planar structure is advantageous.

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

Card 1	1	2	3	4	5	6	7	8
Variable	UPID	LPID	NSID	THICK	D	FN	FT	DN
Type	I	I	I	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

Card 2	1	2	3	4	5	6	7	8
Variable	DT	XIN	XIT	ALPHA1	ALPHA2	ALPHA3	DENS	INTP
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	0.0

Card 3 is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	EXPN	EXPT	PIDVB					
Type	F	F	F					
Default	8.0	8.0	0					

Card 4 is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	XPID1	XPID2	XPID3	XPID4				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
TITLE	Name or description of the SPR2 defined in this keyword. This name is used as part title for the visualization beams, such as in the d3plot database. If undefined, the part title is "SPR2_NSID_...".
UPID	Upper sheet part ID
LPID	Lower sheet part ID
NSID	Node set ID of rivet location nodes
THICK	Total thickness of upper and lower sheets
D	Rivet diameter
FN	Rivet strength in tension (pull-out): GT.0: Constant value LT.0: Material data from an instantiation of *MAT_CONSTRAINED_SPR2 (*MAT_265) with MID of absolute value |FN|
FT	Rivet strength in pure shear
DN	Failure displacement in normal direction
DT	Failure displacement in tangential direction
XIN	Fraction of failure displacement at maximum normal force
XIT	Fraction of failure displacement at maximum tangential force
ALPHA1	Dimensionless parameter scaling the effective displacement
ALPHA2	Dimensionless parameter scaling the effective displacement
ALPHA3	Dimensionless parameter scaling the effective displacement. The sign of ALPHA3 can be used to choose the normal update procedure: GT.0: Incremental update (default) LT.0: Total update (recommended)
DENS	Rivet density (necessary for time step calculation)
INTP	Flag for interpolation. INTP is interpreted digit-wise, namely as, $\text{INTP} = [LK] = K + 10 \times L$

VARIABLE	DESCRIPTION
	The first digit, *K*, controls the interpolation method: K.EQ.0: Linear (default) K.EQ.1: Uniform (not recommended for an asymmetrical arrangement of the upper and lower nodes) K.EQ.2: Inverse distance weighting The second digit, *L*, controls if the nodal area is considered when computing the weight: L.EQ.0: Nodal area is not considered (default), L.EQ.1: Nodal area is considered.
EXPN	Exponent value for the load function in the normal direction
EXPT	Exponent value for the load function in the tangential direction
PIDVB	Part ID for visualization beams representing SPR2 in post-processing. EQ.0: Part ID automatically set (default). GT.0: PIDVB defines part ID.
XPID1	Extra part ID 1 for multi-sheet connection
XPID2	Extra part ID 2 for multi-sheet connection
XPID3	Extra part ID 3 for multi-sheet connection
XPID4	Extra part ID 4 for multi-sheet connection

Remarks:

Self-piercing rivets are a type of fastener that is sometimes used in place of spot welds to join sheet metal of similar or dissimilar materials. The rivet penetrates the upper sheet and expands to interlock with the lower sheet without penetration. The strength and fatigue characteristics of self-piercing rivets can meet or even exceed that of spot welds; consequently, their practical applications are expanding.

In the local description of the underlying model, all considerations are done in the plane-of-maximum opening defined by

$$\hat{\mathbf{n}}_o = \hat{\mathbf{n}}_l \times \hat{\mathbf{n}}_u \quad .$$

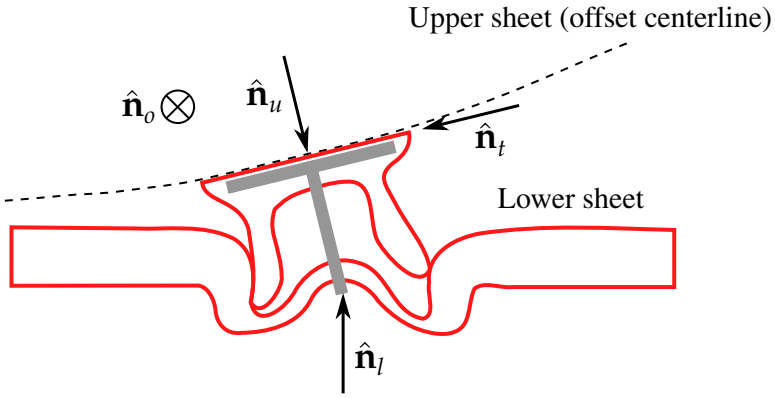


Figure 10-50. Plane of maximum opening.

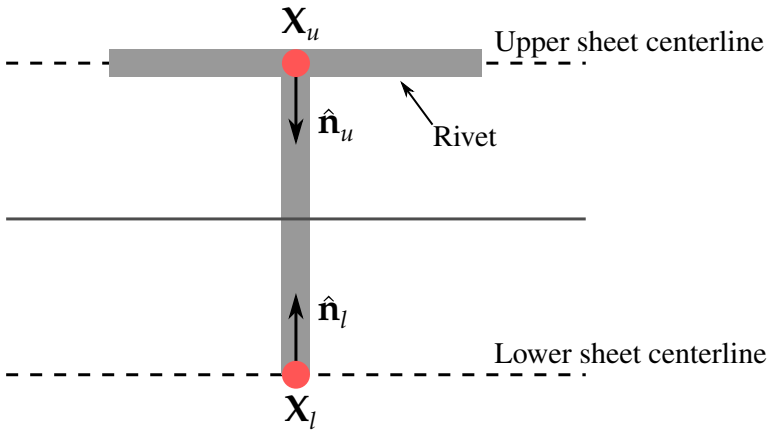


Figure 10-51. Single-sheet rivet system.

The unit normal vectors of the lower and upper sheets are $\hat{\mathbf{n}}_l$ and $\hat{\mathbf{n}}_u$, respectively (see Figure 10-50). The tangential unit normal vector of the rivet is

$$\hat{\mathbf{n}}_t = \hat{\mathbf{n}}_o \times \hat{\mathbf{n}}_u \quad .$$

A single-sheet rivet system is assumed, meaning the rivet translation and rotation follow the motion of the upper sheet. The opening appears on the lower sheet.

The local deformation is defined by normal stretch vector $\boldsymbol{\delta}_n$, tangential stretch δ_t and total stretch $\boldsymbol{\delta} = \boldsymbol{\delta}_n + \boldsymbol{\delta}_t$ (see Figure 10-52). At any given time the total stretch is computed from the position vectors: $\boldsymbol{\delta} = \mathbf{X}_l^r - \mathbf{X}_l^l$ so that the scalar measures of normal stretch and tangential stretch are $\delta_n = \boldsymbol{\delta} \cdot \hat{\mathbf{n}}_n$ and $\delta_t = \boldsymbol{\delta} \cdot \hat{\mathbf{n}}_t$. The normal and tangential forces f_n and f_t are then determined by the material model, which will be explained next. The moments on the rivet always satisfy,

$$M_u + M_l = \frac{(h_1 + h_2) f_t}{2} \quad .$$

The motion, the forces, and the moments are then distributed to the nodes within the radius of influence by a weighting function, which is, by default, linear (see parameter INTP).

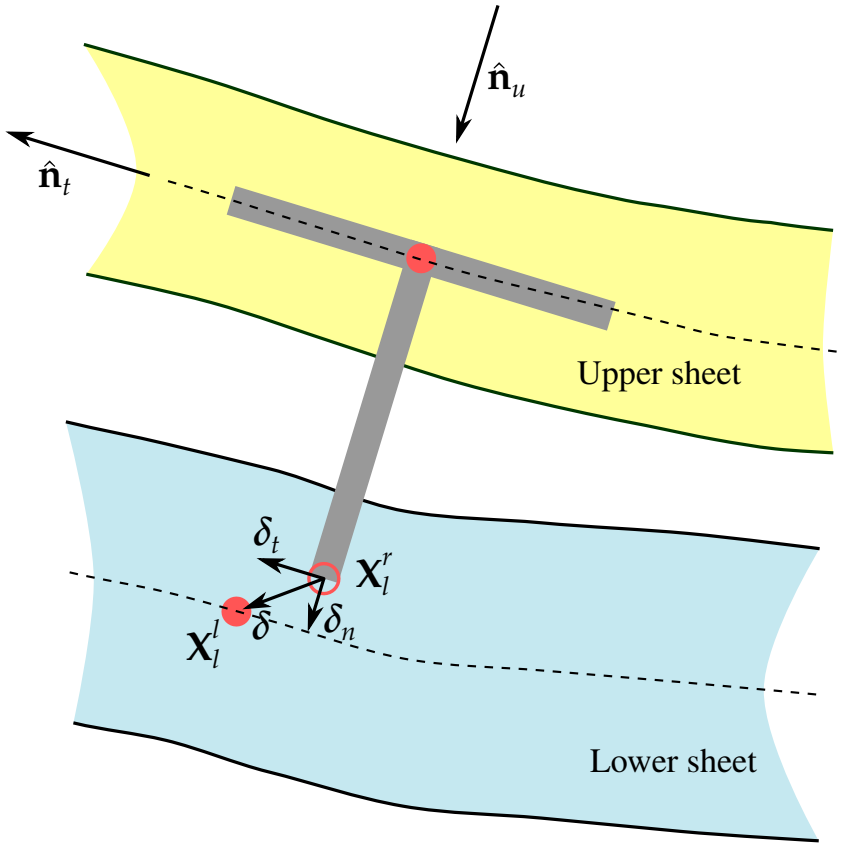


Figure 10-52. Local kinematics

The force-deformation relationship is defined by a nonlinear damage model for arbitrary mixed-mode loading conditions (combination of tension and shear). For pure tensile and pure shear loading, the behavior is given by,

$$f_n = \frac{f_n^{\max}\delta_n}{\eta_{\max}\delta_n^{\text{fail}}}\hat{f}_n(\eta_{\max}), \qquad f_t = \frac{f_t^{\max}\delta_t}{\eta_{\max}\delta_t^{\text{fail}}}\hat{f}_t(\eta_{\max}) \tag{1}$$

respectively where,

$$\hat{f}_n(\eta_{\max}) = \begin{cases} 1 - \left(\dfrac{\xi_n - \eta_{\max}}{\xi_n}\right)^{\text{EXPN}} & \eta_{\max} \le \xi_n \\ 1 - \dfrac{\eta_{\max} - \xi_n}{1 - \xi_n} & \eta_{max} > \xi_n \end{cases}$$
$$\hat{f}_t(\eta_{\max}) = \begin{cases} 1 - \left(\dfrac{\xi_t - \eta_{\max}}{\xi_t}\right)^{\text{EXPT}} & \eta_{\max} \le \xi_t \\ 1 - \dfrac{\eta_{\max} - \xi_t}{1 - \xi_t} & \eta_{max} > \xi_t \end{cases} \tag{2}$$

In pure tension and pure shear the damage measure, $\eta_{\max}(t)$, defined in (3), simplifies to coincide with strain as indicated in Figure 10-54.

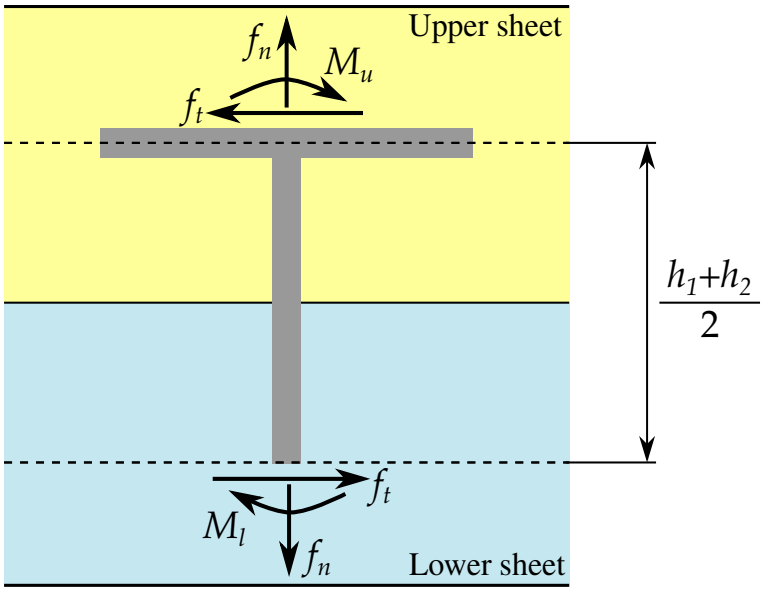


Figure 10-53. Local forces/moments

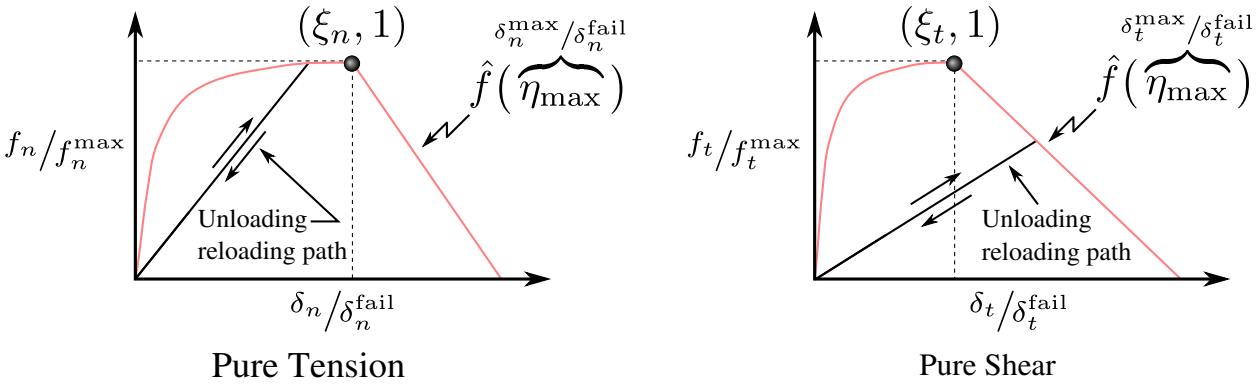


Figure 10-54. Force response of self penetrating rivet.

Usually, the material parameters $f_n^{\max}$, $f_t^{\max}$, δ_n^{fail}, and δ_t^{fail} can be determined directly from experiments, whereas material parameters ξ_n and ξ_t can be found by reverse engineering. For mixed-mode behavior, an effective displacement measure, $\eta(\theta)$, is given by

$$\eta(\theta, \eta_{\max}, t) = \left[\xi(\theta) + \frac{1 - \xi(\theta)}{\alpha(\eta_{\max})}\right] \sqrt{\left[\frac{\delta_n(t)}{\delta_n^{\text{fail}}}\right]^2 + \left[\frac{\delta_t(t)}{\delta_t^{\text{fail}}}\right]^2}, \tag{3}$$

where,

$$\theta = \arctan\left(\frac{\delta_n}{\delta_t}\right)$$
$$\eta_{\max}(t) = \max[\eta(t)].$$

The parameter $\xi(\theta)$ which ranges from 0 to 1 scales the effective displacement as a function of the direction of the displacement vector in the δ_n-δ_t-plane according to,

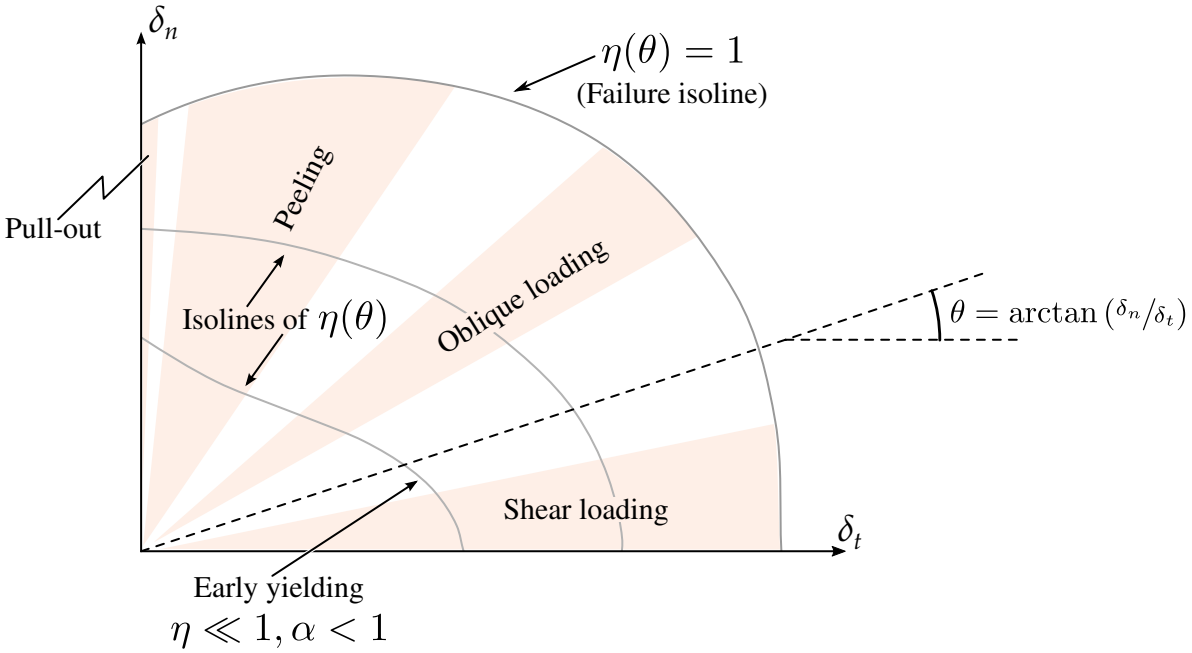


Figure 10-55. Isosurfaces of $\eta(\theta)$

$$\xi(\theta) = 1 - \frac{27}{4}\left(\frac{2\theta}{\pi}\right)^2 + \frac{27}{4}\left(\frac{2\theta}{\pi}\right)^3. \tag{4}$$

The directional scaling of the effective displacement is allowed to change as damage develops, which is characterized by the shape coefficient $\alpha(\eta_{\max})$ defined as

$$\alpha(\eta_{\max}) = \begin{cases} \dfrac{\xi_t - \eta_{\max}}{\xi_t}\alpha_1 + \dfrac{\eta_{\max}}{\xi_t}\alpha_2 & \eta_{\max} < \xi_t \\ \dfrac{1 - \eta_{\max}}{1 - \xi_t}\alpha_2 + \dfrac{\eta_{\max} - \xi_t}{1 - \xi_t}\alpha_3 & \eta_{\max} \geq \xi_t \end{cases}, \tag{5}$$

where α_1, α_2, and α_3 are material parameters.

The directional dependency of the effective displacement is necessary for an accurate force-displacement response in different loading directions. The coefficients α_1, and α_2 decrease the forces in the peeling and oblique loading cases to the correct levels. Both parameters are usually less than 1; whereas α_3 is typically larger than 1 as its main purpose is to moderate the failure displacement in oblique loading directions. Several qualitative features captured by this model are illustrated in Figure 10-55.

For the moment distribution, the difference between the upper sheet (stronger side where the rivet is entered) and the lower sheet (weaker side) is accounted for by a gradual transfer from the lower to the upper side as damage grows:

$$M_u = \frac{h_1 + h_2}{4}\left(1 + \frac{\eta_{\max} - \xi_1}{1 - \xi_1}\right)f_1, \qquad M_l = \frac{h_1 + h_2}{4}\left(1 - \frac{\eta_{\max} - \xi_1}{1 - \xi}\right)f_1 \tag{6}$$

Eventually the connection to the lower sheet becomes a moment free hinge.

We recommend using the drilling rotation constraint method for the connected components in explicit analysis. To use this method, field DRCPSID of *CONTROL_SHELL should refer to all shell parts involved in SPR2 connections.

*CONSTRAINED_TIE-BREAK

Purpose: Define a tied shell edge to shell edge interface that can release locally as a function of plastic strain of the shells surrounding the interface nodes. A rather ductile failure is achieved.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID1	NSID2	EPPF					
Type	I	I	F					
Default	none	none	0.					
Remarks		1, 2	3, 4					

VARIABLE	DESCRIPTION
NSID1	Node set ID for nodes on one side of the tied shell edge to shell edge interface; see *SET_NODE_*OPTION*.
NSID2	Node set ID for nodes on the other side of the tied shell edge to shell edge interface
EPPF	Plastic strain at failure

Remarks:

1. **Node ordering.** Nodes in NSID2 must be given in the order they appear as one moves along the edge of the surface.

2. **Restrictions.** Tie-breaks may not cross.

3. **Failure criterion.** Tie-breaks may be used to tie shell edges together with a failure criterion on the joint. If the average volume-weighted effective plastic strain in the shell elements adjacent to a node exceeds the specified plastic strain at failure, the node is released. The default plastic strain at failure is defined for the entire tie-break but can be overridden in node set NSID1 to define a unique failure plastic strain for each node.

4. **Model applications.** Tie-breaks may be used to simulate the effect of failure along a predetermined line, such as a seam or structural joint. When the failure

criterion is reached in the adjoining elements, nodes along the slideline will begin to separate. As this effect propagates, the tie-breaks will appear to "unzip," thus simulating failure of the connection.

*CONSTRAINED_TIED_NODES_FAILURE

Purpose: Define a tied node set with failure based on plastic strain. The nodes must be coincident.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	EPPF	ETYPE	PID				
Type	I	F	I	I				
Default	none	0.	0	0				
Remarks	1, 2, 4							

VARIABLE	DESCRIPTION
NSID	Nodal set ID; see *SET_NODE_*OPTION*
EPPF	Plastic strain, volumetric strain, or damage (*MAT_107, *MAT_-110, *MAT_224, or GISSMO) at failure.
ETYPE	Element type for nodal group: EQ.0: Shell EQ.1: Solid element
PID	Part ID. With PID > 0 and NSID = 0, the mesh of the corresponding part is automatically separated (creating new coincident nodes) and reassembled using this tied constraint with failure. This constraint applies to every node in the part. If some original nodes are part of node sets, such as for boundary conditions, these sets get automatically extended as well.

Remarks:

1. **Materials and failure.** This feature applies to solid and shell elements using plasticity material models and to solid elements using the honeycomb material *MAT_HONEYCOMB (EPPF = plastic volume strain). The failure variable is the volume strain for materials 26, 126, and 201. The failure variable is the damage for materials 107, 110, 224, or GISSMO, and the equivalent plastic strain is used for all other plasticity models. The specified nodes are tied together until the

average volume-weighted value of the failure variable exceeds the specified value. Entire regions of individual shell elements may be tied together, unlike the tie-breaking shell slidelines. The tied nodes are coincident until failure. When the volume-weighted average of the failure value is reached for a group of constrained nodes, the nodes of the elements that exceed this value are released to simulate the formation of a crack.

2. **Coincident nodes.** To use this feature to simulate failure, each element in the failure region should be generated with unique node numbers that are coincident in space with those of adjacent elements. Rather than merging these coincident nodes, the *CONSTRAINED_TIED_NODES_FAILURE option ties the nodal points together. A separate keyword definition must be used for each cluster of coincident nodes. As plastic strain develops and exceeds the failure strain, cracks will form and propagate through the mesh.

3. **Similar keywords.** Entire regions of individual elements may be tied together, unlike the *CONSTRAINED_TIE-BREAK option. This latter option is recommended when the location of failure is known, e.g., as in the plastic covers that hide airbags in automotive structures.

4. **Contact algorithms.** When using surfaces of shell elements defined using the *CONSTRAINED_TIED_NODES_FAILURE option in contact, it is best to define each node in the surface as a tracked node with the NODE_TO_SURFACE contact options. If this is not possible, the automatic contact algorithms beginning with *CONTACT_AUTOMATIC_... all of which include thickness offsets are recommended.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONSTRAINED_TIED_NODES_FAILURE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Tie shell elements together at the nodes specified in nodal set 101.  The
$  constraint will be broken when the plastic strain at the nodes exceeds 0.085.
$
$  In this example, four shell elements come together at a common point.
$  The four corners of the shells are tied together with failure as opposed
$  to the more common method of merging the nodes in the pre-processing stage.
$
*CONSTRAINED_TIED_NODES_FAILURE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid      eppf
       101     0.085
$
$
*SET_NODE_LIST
$      sid
       101
```

```
$      nid1       nid2       nid3       nid4       nid5       nid6       nid7       nid8
        775        778        896        897
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*CONTACT

The keyword *CONTACT provides a way of treating interaction between disjoint parts. Different types of contact may be defined:

CONTACT_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}_{OPTION5}

*CONTACT_ADD_WEAR

*CONTACT_AUTO_MOVE

*CONTACT_COUPLING

*CONTACT_ENTITY

*CONTACT_EXCLUDE_INTERACTION

CONTACT_FORCE_TRANSDUCER_*OPTION1_{OPTION2}

CONTACT_GEBOD_*OPTION

*CONTACT_GUIDED_CABLE

*CONTACT_INTERIOR

*CONTACT_RIGID_SURFACE

*CONTACT_SPG

*CONTACT_1D

CONTACT_2D_*OPTION1_{OPTION2}_{OPTION3}

CONTACT_*OPTION1*_... is the general form for defining a 3D contact algorithm. Wear models can be associated with a contact interface using *CONTACT_ADD_WEAR. *CONTACT_AUTO_MOVE is used for sheet metal forming applications and moves a surfb surface in a contact to close a gap between the surfa and surfb at a specified time. *CONTACT_COUPLING provides a means of coupling to deformable surfaces to MADYMO. *CONTACT_ENTITY treats contact using mathematical functions to describe the surface geometry for the reference surface. Portions of the 3D contact interface can be excluded from interacting with *CONTACT_EXCLUDE_INTERACTION. *CONTACT_FORCE_TRANSDUCER measures the contact forces for 3D contacts. It is needed for measuring the forces in self-contacts since LS-DYNA does not automatically extract this data in that case.

*CONTACT_GEBOD is a specialized form of the contact entity for use with the rigid body dummies (see *COMPONENT_GEBOD). *CONTACT_GUIDED_GABLE is a sliding contact for guiding 1D elements. *CONTACT_INTERIOR is used with soft foams where element inversion is sometimes a problem. Contact between layers of brick elements is treated to eliminate negative volumes. *CONTACT_RIGID_SURFACE is for modeling road surfaces for durability and NVH calculations. *CONTACT_SPG is for contact between SPG parts. *CONTACT_1D remains in LS-DYNA for historical reasons and is sometimes still used to model rebars which run along edges of brick elements. Lastly, *CONTACT_2D is the general 2D contact algorithm based on those used previously in LS-DYNA2D.

***CONTACT**_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}_{OPTION5}_{OPTION6}*

Purpose: Define a contact interface in a 3D model. For contact in 2D models, see *CONTACT_2D_*OPTION*.

Introduction:

The *CONTACT keyword creates a *contact definition*. A contact definition associates surfaces with a *contact algorithm*. A contact algorithm is built from characteristics specified over a small number of fields in the *CONTACT data set. The number of distinct contact algorithms in LS-DYNA equals the number of allowed combinations of characteristics. Not all data fields equally impact the nature of the contact algorithm. The most important characteristics are controlled by:

1. **OPTION1**. For some values of *OPTION1*, the contact algorithm is completely specified. For other values of *OPTION1*, it is not. *OPTION1* is often referred to as the *contact type* in this document.

2. **The SOFT field of Optional Card A.** This field does not lend itself to a concise summary, but in general, it is accurate to say that the SOFT field of Optional Card A can change the qualitative behavior of a contact algorithm. The most popular values of SOFT, namely, 0, 1, and 2, specify the kind of penalty algorithm invoked.

3. **OPTION4.** This option controls the behavior of tied contact algorithms.

The broadest categories of contact characteristics are whether a contact algorithm is penalty or constraint-based, how penetration is measured (nodes-to-segments or segments-to-segments), and whether the contact is nonsymmetric or symmetric:

1. **Penalty-based and constraint-based contact algorithms.** Penalty-based contact places normal interface springs between penetrating nodes/segments and the contact surface. These springs apply a contact force to reduce penetration. In constraint-based contact, nodes penetrating the contact surface are forced to the surface and often controlled with kinematic constraints.

2. **How penetration is measured.** Except for segment-to-segment contacts (SOFT = 2 on Optional Card A and Mortar contact), all LS-DYNA contact algorithms look for the nodes of one surface penetrating the segments of another surface; that is, they are node-to-segment contacts. In contrast, the segment-to-segment contact algorithms search for the segments of one surface penetrating the segments of another surface (see Note on segment-to-segment contact algorithms for a brief discussion).

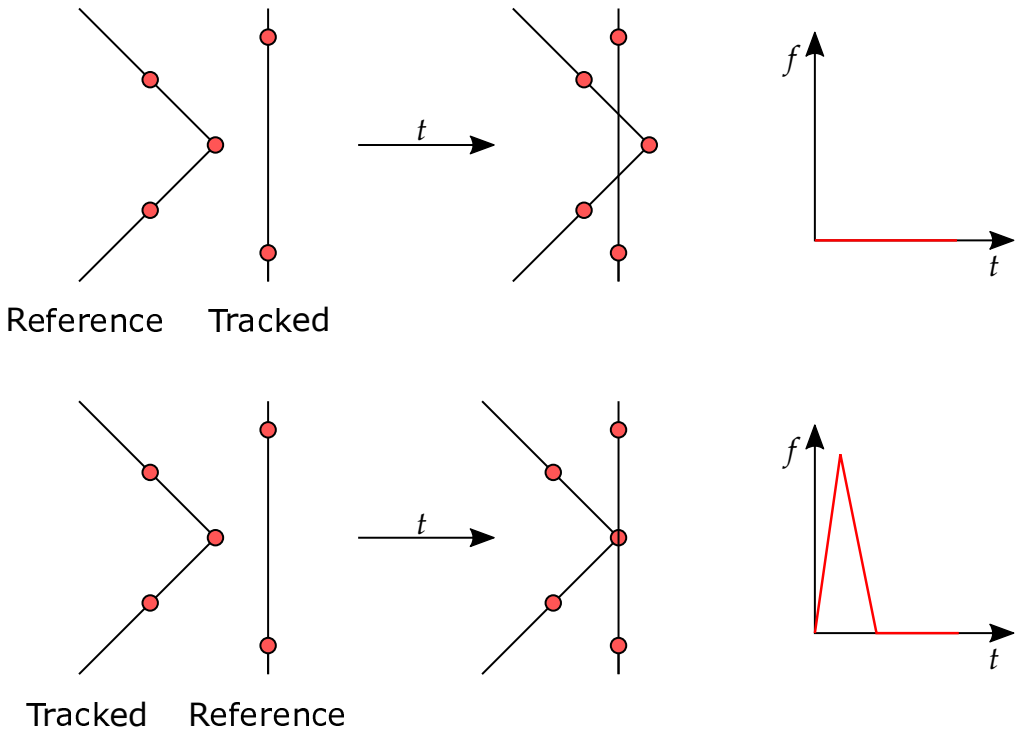


Figure 11-1. Illustration of the difference between the tracked and reference surfaces in nonsymmetric contact. In this example, nodes of the tracked surface are checked to see if they penetrate segments of the reference surface.

3. **Nonsymmetric and symmetric contact.** In nonsymmetric contact, penetration is only checked in one direction. The nodes/segments of one surface (the ***tracked surface***) are checked to see if they penetrate the segments of another surface (the ***reference surface***), but the nodes/segments of the other surface are not checked to see if they penetrate the first surface. See Figure 11-1. Symmetric contact checks both surfaces for penetration, usually by running the contact algorithm twice. Thus, both surfaces are the tracked and reference surfaces depending on the context. Previously, the tracked surface was known as the ***slave***, and the reference surface was known as the ***master***.

Setting the Contact Interface

Another essential consideration influenced by the contact characteristics is specifying the two sides of the contact interface. SURFA and SURFB set the two sides of the contact interface on Mandatory Card 1. SURFA is equivalent to ***SSID***, and SURFB is equivalent to ***MSID***. For nonsymmetric contact, SURFA specifies the ***tracked*** surface, while SURFB specifies the ***reference*** surface, meaning that nodes or segments of SURFA are checked to see if they penetrate SURFB. The nodes/segments of SURFB can penetrate the segments of SURFA for these kinds of contact. See Figure 11-1.

For symmetric contacts, both sides of the contact interface are checked for penetration, and contact forces are applied accordingly. The choice of SURFA and SURFB is arbitrary for symmetric contact. Depending on the context, each surface acts as the reference and tracked surfaces. For single-surface contact, only SURFA is input, and penetration is checked on both sides of the self-contact interface.

For tied contacts, SURFA nodes are tied to SURFB segments, usually at the beginning of the simulation. Note that there are some single-surface tied contacts where only SURFA is input, and the surface is tied to itself.

Note on segment-to-segment contact algorithms
We have implemented two kinds of segment-to-segment contact algorithms: SOFT = 2 and Mortar. Note that in this manual, we will use segment-to-segment to refer to SOFT = 2 and will always mention Mortar when referring to Mortar segment-to-segment to reduce confusion. Major differences between these algorithms include that Mortar is devised to work well for the implicit solver and is nonsymmetric. In contrast, segment-to-segment is generally symmetric (with some exceptions) and was designed to improve contact behavior compared to nodes-to-segment contacts in certain situations. About SOFT = 2 under Optional Card A briefly discusses segment-to-segment contact. See Remark 14 in the General Remarks section for a discussion of Mortar contact.

Note that previously, SOFT = 2 was called segment-based contact. We have changed this to segment-to-segment contact because it is more descriptive and matches the literature.

DATA CARDS FOR *CONTACT KEYWORD

Cards *must* appear in the *exact* order listed below.

CARD	DESCRIPTION
ID	Card required when *OPTION3* is set to ID; otherwise, this card is omitted.
MPP 1	Card required when *OPTION5* is set to MPP.
MPP 2	Optional card that can *only* be included if *OPTION5* is set to MPP.
Card 1	Always required.
Card 2	Always required.
Card 3	Always required.
Card 4	Required for the following permutations of *CONTACT.

NOTE: The format of Card 4 (which can include multiple cards) is **different** for each option listed.

AUTOMATIC_..._TIEBREAK

AUTOMATIC_..._TIEBREAK_USER

AUTOMATIC_SURFACE_..._COMPOSITE/LUBRICATION

AUTOMATIC_SINGLE_SURFACE_TIED

AUTOMATIC_..._TIED_WELD

CONSTRAINT_*type*

DRAWBEAD

ERODING_*type*

*type*_INTERFERENCE

RIGID_*type*

TIEBREAK_NODES_...

TIEBREAK_SURFACE_...

CARD	DESCRIPTION
	SURFACE_TO_SURFACE_CONTRACTION_JOINT
THRM 1	Required if *OPTION2* is set to either THERMAL or THERMAL_-FRICTION. Otherwise, omit.
THRM 2	Required if *OPTION2* is set to THERMAL_FRICTION. Otherwise, omit.
THRM 2.1	Inclusion of this card depends on the input on THRM 2.
ORFR 1, ORFR 2, ORFR 3, ORFR 4	Required if *OPTION6* is set. Otherwise, omit. Contains friction coefficients.
Optional Card A	Optional parameters.

NOTE: Default values are highly optimized.

NOTE: *Required* if including Optional Card B. If Optional Card A is a blank line, then values are set to their defaults, and Optional Card B may follow.

Optional Card B	Optional parameters. *Required* if including Optional Card C. (See Optional Card A note; similar logic applies)
Optional Card C	Optional parameters. *Required* if including Optional Card D. (See Optional Card A note; similar logic applies.)
Optional Card D	Optional parameters. *Required* if including Optional Card E. (See Optional Card A note; similar logic applies.)
Optional Card E	Optional parameters. *Required* if including Optional Card F. (See Optional Card A note; similar logic applies.)
Optional Card F	Optional parameters. *Required* if including Optional Card G. (See Optional Card A note; similar logic applies.)
Optional Card G	Optional parameters

OPTIONS FOR *CONTACT KEYWORD

OPTION	REQUIRED	DESCRIPTION
OPTION1	Yes	Specifies contact type
OPTION2	No	Flag for thermal
OPTION3	No	Flag indicating ID cards follow
OPTION4	No	Offset options
OPTION5	No	Flag for MPP
OPTION6	No	Flag for orthotropic friction

Allowed values for *OPTION1*:

The following list covers the unique terminology used to describe LS-DYNA's contact types. Additional notes on contact types and a few examples are provided at the end of this manual page in "General Remarks: *CONTACT." The LS-DYNA Theoretical Manual covers the algorithms at a higher level.

Following this list are all the available values for *OPTION1*. All contact types are available for explicit and implicit calculations.

1. **Categories of contact types.** Generally, we can split the contact types specified with *OPTION1* into seven basic categories: one-way, two-way, single surface, tied, Mortar, constraint, and sliding only.

   a) One-way contact is nonsymmetric. The nodes (or segments) of SURFA cannot penetrate the segments of SURFB, but SURFB nodes and segments may penetrate SURFA. Thus, deciding on the surfaces for SURFA and SURFB is vital for this type of contact. Generally, SURFA should have a finer mesh. Compressive loads are transferred between the nodes (or segments) of SURFA and segments of SURFB when they are in contact.

   b) Two-way contact is like one-way contact, except it is symmetric. In this case, LS-DYNA checks for SURFA penetrating SURFB and then SURFB penetrating SURFA. As a result, the definitions of SURFA and SURFB are arbitrary. Using this type of contact costs roughly double that of one-way.

   c) Single surface contact requires only the input of SURFA. Unlike the other contact types, LS-DYNA checks the single surface for self-penetration. Its algorithms are based on contacts with SURFACE_TO_SURFACE in the name (see the theory manual for details).

d) Tied contact is for "gluing" two surfaces together, meaning no sliding or separation. It is particularly good for tying parts with incompatible meshes. For this contact, the nodes of SURFA (tracked) are tied to and, thus, constrained to move with the segments of SURFB (reference). No sliding is allowed, except for some TIEBREAK contacts. We also have some single-surface tied contacts for which only SURFA is input. In this case, the surface is tied to itself (AUTOMATIC_SINGLE_SURFACE_TIED, AUTOMATIC_-GENERAL_TIEBREAK, and AUTOMATIC_SINGLE_SURFACE_TIE-BREAK).

Tying only occurs at the beginning of the simulation unless the contact type is some of the TIEBREAK versions. Thus, if a tracked node enters a reference segment's vicinity during the simulation, the node will not be tied to the segment. For tying to occur, the tracked node must lie within the orthogonal projection of a reference segment, and the gap between the node and segment must be less than a certain tolerance (see Remark 4 in General Remarks). If *OPTION4* (offsets) is used, the tied tracked node can have an offset from the reference segment. Otherwise, during initialization, the node is moved to the surface. We recommend specifying tied contacts with node or segment sets instead of parts or part sets since they give you more control over what is tied. Identifying the surfaces this way prevents unintended tying.

Some tied contacts constrain only translational degrees of freedom, while others additionally constrain rotational degrees of freedom. Some types of this contact allow for failure. TIEBREAK is a special case of a tied contact allowing failure in which the contact usually becomes a regular one-way, two-way, or single surface version after failure. How a TIEBREAK contact behaves after failure depends on what the contact type would be if TIE-BREAK were excluded. Note that even for automatic tiebreak segment normals should be oriented properly for these types of contact (see Remark 6 under Card 4: AUTOMATIC_...TIEBREAK).

Tied contacts with TIEBREAK or PENALTY in the name or ones with OFF-SET and BEAM_OFFSET appended to the name using *OPTION4* are penalty-based. All others are constraint-based.

e) Mortar contact is a segment-to-segment-based contact with its own algorithms. It is designated with MORTAR in the name. It is especially well-suited for the implicit solver. It is not symmetric since segments on SURFA are not treated the same as SURFB. Mortar contact includes a single surface version (AUTOMATIC_SINGLE_SURFACE_MORTAR) where only SUR-FA is input to specify the contact interface. For more details about Mortar contact, see Remark 14 in the General Remarks section and the theory manual.

f) For this discussion, constraint category contacts have CONSTRAINT in the name. These contacts are constraint-based. They can be symmetric or non-symmetric depending on the setting of KPF on Card 4. See Remark 11 in General Remarks.

g) Sliding-only contact is one of the oldest types. LS-DYNA contains two versions: SLIDING_ONLY and SLIDING_ONLY_PENALTY. The first is constraint-based, while the second is penalty-based. Sliding-only contact restricts the nodes of SURFA to slide along SURFB. No separation is allowed with these methods. These contact types help treat interfaces where the gaseous detonation products of a high explosive act on a solid material. See the theory manual for more details. This type of contact is only available for the explicit solver.

2. **NODES_TO_SURFACE.** Contacts with NODES_TO_SURFACE in the name are one-way or tied contacts. SURFA can be defined with a node set for these contacts, and contact is only nodes to segments (neither segment-to-segment nor Mortar contact is available).

3. **SURFACE_TO_SURFACE.** Contacts with SURFACE_TO_SURFACE (also called surface-to-surface contact) in the name can have SURFA defined with segment sets and shell sets. SURFA cannot be represented with node sets. Note that being able to input as segments does not mean that the contact formulation is segment-to-segment (SOFT = 2 or Mortar). The contact formulation for most penalty contacts depends on the value of SOFT on Optional Card A. Unless the contact formulation is segment-to-segment (SOFT = 2) or the contact type includes MORTAR, the formulation looks at nodal penetrations into segments, not segment penetration into segments.

   SURFACE_TO_SURFACE contacts have special, advanced features compared to NODES_TO_SURFACE contact, partly due to how they are defined. For instance, SURFACE_TO_SURFACE contacts support thermal transfer, while NODES_TO_SURFACE generally contacts do not. Note that the algorithms for SINGLE_SURFACE are based on SURFACE_TO_SURFACE. Thus, parameters that apply to surface-to-surface contact apply to SINGLE_SURFACE.

4. **AUTOMATIC**. Automatic contacts are designated with AUTOMATIC in the type. The algorithms for this type are better than those for non-automatic contacts for disjoint meshes. Automatic contacts are two-sided in that the algorithms detect penetration on either side of a shell element. In contrast, non-automatic contacts are one-sided. Thus, segment orientation does not matter for automatic contact but is crucial for non-automatic contact. Therefore, automatic contacts are advantageous for crash analysis.

   To detect penetration on both sides, automatic contacts include a contact thickness of half the thickness on each side of the shell midplane (see Remark 13 in

the General Remarks). They also include a contact surface on the exterior edges of a shell surface with a radius of half the contact thickness. Because of how these projections are performed, the shell surface has a continuous contact surface. Similarly, for beam elements where beam contact is considered, the contact surface is offset from the centerline of the beam element by an equivalent radius of the beam cross-section (see the theory manual). Because of the contact surfaces, modeling appropriate gaps between parts is critical for automatic contact with beam and shell parts. See also Remark 3 in the General Remarks.

5. **ERODING.** Contact types with ERODING in the name are suitable for when elements in the contact interface fail and are deleted. These types allow for updating the contact surface due to element deletion.

6. **SMOOTH.** For SMOOTH contact, a smooth curve-fitted surface represents the reference surface segments to provide a more accurate representation of the actual surface, reduce the contact noise, and produce smoother results with coarser meshes.

   SMOOTH contact is unavailable in SMP for FORMING_SURFACE_TO_SURFACE and SURFACE_TO_SURFACE contacts.

   All contact options that include SMOOTH are available for MPP. Furthermore, for SURFACE_TO_SURFACE and SINGLE_SURFACE contacts with the SMOOTH option, both sides of the contact interface are smoothed every cycle, thereby slowing the contact treatment considerably.

   The SMOOTH option does not apply to segment-to-segment (SOFT = 2) contacts.

7. **DRAWBEAD.** DRAWBEAD contacts are a particular type of one-way contact for simulating draw beads. SURFA should be the nodes defining the draw bead, while SURFB is the blank. See Card 4: DRAWBEAD and the theory manual for a complete description.

8. **FORMING.** Contact types that include FORMING in the name are mainly used for metal forming applications. A connected mesh is not required for the SURFB (tooling) side, but the orientation of the mesh *must* be in the same direction. These contact types are based on AUTOMATIC-type contacts, and consequently, the performance is better than the original two surface contacts.

9. **INTERFERENCE.** Contact types with INTERFERENCE are intended for modeling parts with an interference fit. Therefore, LS-DYNA does not check for initial penetrations for this contact. The overlap is instead removed by contact forces, causing stress and deformation in the interference fit parts.

Single Surface or Self-Contact

AIRBAG_SINGLE_SURFACE

AUTOMATIC_GENERAL (see Remark 2 in General Remarks)

AUTOMATIC_GENERAL_EDGEONLY (see Remark 2 in General Remarks)

AUTOMATIC_GENERAL_INTERIOR (see Remark 2 in General Remarks)

AUTOMATIC_SINGLE_SURFACE (see Remark 2 in General Remarks)

AUTOMATIC_SINGLE_SURFACE_SMOOTH

ERODING_SINGLE_SURFACE

SINGLE_EDGE (see Remark 3 in General Remarks)

SINGLE_SURFACE

One-Way Contact

AUTOMATIC_BEAMS_TO_SURFACE

AUTOMATIC_NODES_TO_SURFACE

AUTOMATIC_NODES_TO_SURFACE_SMOOTH

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_SMOOTH

DRAWBEAD

DRAWBEAD_BENDING

DRAWBEAD_INITIALIZE

ERODING_NODES_TO_SURFACE

FORMING_NODES_TO_SURFACE

FORMING_NODES_TO_SURFACE_SMOOTH

FORMING_ONE_WAY_SURFACE_TO_SURFACE

FORMING_ONE_WAY_SURFACE_TO_SURFACE_SMOOTH

NODES_TO_SURFACE

NODES_TO_SURFACE_INTERFERENCE

NODES_TO_SURFACE_SMOOTH

ONE_WAY_SURFACE_TO_SURFACE

ONE_WAY_SURFACE_TO_SURFACE_INTERFERENCE

ONE_WAY_SURFACE_TO_SURFACE_SMOOTH

RIGID_NODES_TO_RIGID_BODY

RIGID_BODY_ONE_WAY_TO_RIGID_BODY

Two-Way Contact

AUTOMATIC_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE_SMOOTH

ERODING_SURFACE_TO_SURFACE

FORMING_SURFACE_TO_SURFACE

FORMING_SURFACE_TO_SURFACE_SMOOTH

RIGID_BODY_TWO_WAY_TO_RIGID_BODY

SURFACE_TO_SURFACE

SURFACE_TO_SURFACE_INTERFERENCE

SURFACE_TO_SURFACE_SMOOTH

SURFACE_TO_SURFACE_CONTRACTION_JOINT

Tied Contact

The following types are tied contacts (not tiebreak). Remark 7 in the General Remarks section discusses which tied contacts can be used with rigid bodies. Remark 4 in General Remarks concerns the tying criterion. For a discussion of which TIED contact to use in which situation, see Remark 5 in General Remarks. For using TIED contacts with the implicit solver, see Remark 6 in General Remarks.

AUTOMATIC_SINGLE_SURFACE_TIED

AUTOMATIC_SURFACE_TO_SURFACE_TIED_WELD

SPOTWELD

SPOTWELD_WITH_TORSION (see Remark 8 in General Remarks)

SPOTWELD_WITH_TORSION_PENALTY

TIED_NODES_TO_SURFACE

TIED_SHELL_EDGE_TO_SURFACE

TIED_SHELL_EDGE_TO_SOLID

TIED_SURFACE_TO_SURFACE (see Remark 3 under Mandatory Card 1)

TIED_SURFACE_TO_SURFACE_FAILURE (SMP only)

The following contacts are tiebreak contacts, which means that these contacts generally turn into regular sliding contacts after failure. See the Card 4 remarks for each type for the failure criteria and the behavior after failure. They use penalty-based contact formulations.

AUTOMATIC_GENERAL_TIEBREAK

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK_DAMPING

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK_USER

AUTOMATIC_SINGLE_SURFACE_TIEBREAK

AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK

AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_USER

TIEBREAK_NODES_TO_SURFACE

TIEBREAK_NODES_ONLY

TIEBREAK_SURFACE_TO_SURFACE

TIEBREAK_SURFACE_TO_SURFACE_ONLY

Mortar Contact

AUTOMATIC_SINGLE_SURFACE_MORTAR

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED_WELD

AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_MORTAR

AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_USER_MORTAR

FORMING_SURFACE_TO_SURFACE_MORTAR

Constraint Contacts

CONSTRAINT_NODES_TO_SURFACE

CONSTRAINT_SURFACE_TO_SURFACE

Sliding Only Contacts

SLIDING_ONLY

SLIDING_ONLY_PENALTY

Allowed values for *OPTION2*:

THERMAL

THERMAL_FRICTION

> **NOTE:** For sliding contacts, the *THERMAL* and *THERMAL_-FRICTION* options are restricted to contact types having *"SURFACE_TO_SURFACE"* or *"AUTOMATIC_SINGLE_SURFACE"* as part of the *OPTION1* name. For tied contacts, both *"TIED_NODES_TO_-SURFACE"* and *"TIED_SHELL_EDGE_TO_SURFACE"* are supported only if SURFA is *not* a node set.

Allowed value for *OPTION3*:

ID

Allowed values for *OPTION4*:

OPTION4 specifies that offsets may be used with the tied contact types. If one of these three offset options is set, offsets are permitted for these contact types. If not, the nodes are projected back to the contact surface during the initialization phase, and a constraint formulation is used. Note that in a constraint formulation, the nodes of rigid bodies are not permitted in the definition.

OFFSET

BEAM_OFFSET

CONSTRAINED_OFFSET

OFFSET keyword option

The OFFSET option switches the formulation from a constraint-type formulation to one that is penalty-based, where discrete spring elements between the tracked nodes and reference segments transfer the force and moment (if applicable) resultants. Rigid bodies can be used with the OFFSET option.

OFFSET is available when *OPTION1* is:

TIED_NODES_TO_SURFACE

TIED_SHELL_EDGE_TO_SURFACE

TIED_SURFACE_TO_SURFACE

With OFFSET, no coupling occurs between the transmitted forces and moments; thus, equilibrium is not enforced. For TIED_NODES_TO_SURFACE, the moment equilibrium is enforced by setting FTORQ = 2 on Card E. For TIED_SHELL_EDGE_TO_SURFACE contact, the BEAM_OFFSET option may be preferred since corresponding moments accompany transmitted forces.

BEAM_OFFSET keyword option

The BEAM_OFFSET option switches the formulation from a constraint-type formulation to a penalty-based one. Beam-like springs transfer force and moment resultants between the tracked nodes and the reference segments. Rigid bodies can be used with this option.

BEAM_OFFSET is available when *OPTION1* is:

TIED_SHELL_EDGE_TO_SURFACE

SPOTWELD

BEAM_OFFSET is also available when *OPTION1* is:

AUTOMATIC_SINGLE_SURFACE_TIEBREAK

AUTOMATIC_GENERAL_TIEBREAK

but moments are transferred only when the variable FTORQ is invoked (see Optional Card E). Those moments are transmitted to the reference surface segments through nodal forces applied to the nodes of these segments.

CONSTRAINED_OFFSET keyword option

The CONSTRAINED_OFFSET option is a constraint-type formulation. CONSTRAINED_OFFSET is available when *OPTION1* is:

TIED_NODES_TO_SURFACE

TIED_SHELL_EDGE_TO_SOLID (MPP only)

TIED_SHELL_EDGE_TO_SURFACE

TIED_SURFACE_TO_SURFACE

SPOTWELD

Allowed value *OPTION5*:

MPP

Allowed value for *OPTION6*:

ORTHO_FRICTION

ORTHO_FRICTION is available when *OPTION1* is:

AUTOMATIC_SURFACE_TO_SURFACE(_SMOOTH /_MORTAR)

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE(_SMOOTH)

FORMING_ONE_WAY_SURFACE_TO_SURFACE(_SMOOTH)

ID Card:

Additional card for ID keyword option

Card ID	1	2	3	4	5	6	7	8
Variable	CID	HEADING						
Type	I	A70						

The contact ID is needed during full deck restarts for contact initialization. If the contact ID is undefined, the default ID is determined by the sequence of the contact definitions; that is, the first contact definition has an ID of 1, the second, 2, and so forth. For a successful run in a full deck restart without contact IDs, no contact interfaces can be deleted, and added contact interfaces must be placed after the last definition in the previous run. Some of the peripheral LS-DYNA codes pick up the ID and headings to aid in post-processing.

VARIABLE	DESCRIPTION
CID	Contact interface ID. This must be a unique number.
HEADING	Interface descriptor. We suggest using unique descriptions.

MPP Cards:

Variables set with these cards are only active when using MPP LS-DYNA.

MPP Card 1. Additional card for the MPP option. When SOFT = 2 on Optional Card A, this card is ignored but still read in.

MPP 1	1	2	3	4	5	6	7	8
Variable	IGNORE	BCKT	LCBCKT	NS2TRK	INITITR	PARMAX		CPARM8
Type	I	I	I	I	I	F		I
Default	0	200	none	3	2	↓		0

MPP Card 2. The keyword reader interprets the card following MPP Card 1 as MPP Card 2 if an ampersand occupies the first column of the card. Otherwise, it is interpreted as Card 1. When SOFT = 2 on Optional Card A, GRPABLE is the *only* field on this card that is not ignored.

MPP 2	1	2	3	4	5	6	7	8
Variable	&	CHKSEGS	PENSF	GRPABLE		IGTOL		
Type		I	F	I		F		
Default		0	1.0	0		0.0		

VARIABLE	DESCRIPTION
IGNORE	For automatic and non-automatic contacts, setting this variable to 1 turns on the "ignore initial penetrations" for this contact. Alternatively, for automatic contacts only, setting IGNORE = 1 on Card 4 of *CONTROL_CONTACT or on Optional Card C of *CONTACT turns on this option. In other words, for automatic contacts, if IGNORE is set to 1 in any of three places, initial penetrations are tracked. For non-automatic contacts, setting IGNORE to 1 here causes initial penetrations to be tracked.
BCKT	Bucket sort frequency. This parameter does not apply when SOFT = 2 on Optional Card A or to Mortar contacts. For these two exceptions, the BSORT option on Optional Card A applies instead.

VARIABLE	DESCRIPTION
	Note that NSBCS on *CONTROL_CONTACT does not apply to SOFT = 0 and SOFT = 1 for MPP.
LCBCKT	Load curve for bucket sort frequency. This parameter does not apply when SOFT = 2 on Optional Card A or to Mortar contacts. For the two exceptions, the negative BSORT option on Optional Card A applies instead.
NS2TRK	Number of potential contacts to track for each tracked node. The normal input for this (DEPTH on Optional Card A) is ignored.
INITITR	Number of iterations to perform when trying to eliminate initial penetrations. Note that an input of 0 means 0, not the default value, 2. Leaving this field blank will set INITITR to 2.
PARMAX	The parametric extension distance for contact segments. The MAXPAR parameter on Optional Card A is not used for MPP. For non-tied contacts, the default is 1.0005. For *CONTACT_TIEBREAK_..., PARMAX is hardcoded as 1.1. For all other tied contacts, the default is 1.035, and the actual extension used is computed as follows:

$$\text{PARMAX}_{\text{computed}} = \begin{cases} 1.0 + \text{PARMAX} & 0.0 < \text{PARMAX} < 0.5 \\ \text{PARMAX} & 1.0 \le \text{PARMAX} \le 1.0004 \\ \max(\text{PARMAX}, 1.035) & \text{otherwise} \end{cases}$$

CPARM8 — Flag for the behavior of AUTOMATIC_GENERAL contacts. CPARM8's value is interpreted as two separate flags: OPT1 and OPT2, according to the rule,

$$\text{CPARM8} = \text{OPT1} + \text{OPT2}.$$

When OPT1 and OPT2 are *both* set, *both* options are active.

OPT1. Flag to exclude beam-to-beam contact from the same PID.

- EQ.0: The flag is not set (default).
- EQ.1: The flag is set.
- EQ.2: The flag is set. CPARM8 = 2 additionally permits contact treatment of spot weld (type 9) beams in AUTOMATIC_GENERAL contacts. See Remark 2 for details.

OPT2. Flag to shift a generated beam, affecting only shell-edge-to-shell-edge treatment. See also SRNDE in Optional Card E.

- EQ.10: A beam generated on an exterior shell edge will be

VARIABLE	DESCRIPTION
	shifted into the shell by half the shell thickness. Therefore, the shell-edge-to-shell-edge contact starts right at the shell edge and not at an extension of the shell edge.
CHKSEGS	If this value is nonzero, then LS-DYNA performs a special check at time 0 for inverted (or nearly inverted) elements for node-to-surface and surface-to-surface contacts. These elements are removed from contact. These poorly formed elements have been known to occur on the tooling in metal forming problems, which allows these problems to run. It should not typically be needed for reasonable meshes.
PENSF	This option is used together with IGNORE for 3D forging problems. If nonzero, the IGNORE penetration distance is multiplied by this value each cycle, effectively pushing the tracked node back to the surface. This is useful for nodes that might get generated below the reference surface during 3D remeshing. Care should be exercised, as energy may be generated, and stability may be affected for values lower than 0.95. A value in the range of 0.98 to 0.99 or higher (but < 1.0) is recommended.
GRPABLE	Set to 1 to invoke an alternate MPP communication algorithm for various SINGLE_SURFACE (including AUTOMATIC_GENERAL), NODES_TO_SURFACE, SURFACE_TO_SURFACE, ERODING, and SOFT = 2 contacts. This groupable algorithm does not support all contact options, including MORTAR. It is still under development. It can be significantly faster and scale better than the standard algorithm when more than two or three applicable contact types are defined in the model. It is intended to speed up contact processing without changing the contact's behavior. See also *CONTROL_MPP_CONTACT_GROUPABLE. EQ.-1: Do not use the groupable algorithm for this contact. EQ.0: Do not use the groupable algorithm for the contact. However, GRP on *CONTROL_MPP_CONTACT_GROUPABLE overrides this setting. EQ.1: Use the groupable algorithm for this contact if supported. However, GRP on *CONTROL_MPP_CONTACT_GROUPABLE overrides this setting.
IGTOL	Scale factor for determining the contact exclusion distance. This option is available for MPP only. It applies to *CONTACT_AUTOMATIC_GENERAL and *CONTACT_AUTOMATIC_SIN-

VARIABLE	DESCRIPTION
	GLE_SURFACE with SOFT = 0 or 1. See Remark 4 for details.

Remarks:

1. **SOFT = 2.** Except for the GRPABLE field, the MPP cards are ignored by segment-to-segment (SOFT = 2 on Optional Card A) contact. When SOFT = 2, the BSORT parameter on Optional Card A can override the default bucket sort frequency. See About SOFT = 2 for more details about segment-to-segment contact.

2. **Spot welds in contacts.** In general, the contacts ignore parts with spot-weld material. Setting OPT1 on CPARM8 can change this behavior. Depending on the contact type and type of ID in SURFA / SURFB, spot-weld parts can be included in the contact.

   a) For SURFATYP / SURFBTYP = 2 (part set), we have:

      i) For AUTOMATIC_GENERAL contacts, exclude all spot-weld parts. However, if OPT1 = 2, the spot-weld beams in the set are included.

      ii) For all other contacts, include those spot-weld parts contained in the set.

   b) For SURFATYP / SURFBTYP = 5 (include all), all spot-weld parts are excluded. However, if OPT1 = 2, spot-weld beams are included (regardless of contact type).

   c) For SURFATYP / SURFBTYP = 6 (exempted part set), all spot-weld parts are excluded (regardless of contact type or OPT1 value).

3. **Shell reference surface for tied contacts in MPP.** Unless the MPP contact is groupable (see GRPABLE here and GRP on *CONTROL_MPP_CONTACT_GROUPABLE), MPP tied contacts ignore the influence of NLOC on *SECTION_SHELL, OFFSET on *ELEMENT_SHELL_OFFSET, CNTCO on *CONTROL_SHELL, and SHLOFF on Optional Card G on the shell reference surface. Non-groupable tied contacts always use the plane determined by the nodes for the shell reference surface. Thus, these fields for non-groupable tied contacts have no effect on determining what is tied or not. Groupable tied contacts honor the settings of these fields.

4. **IGTOL.** Setting IGTOL > 0.0 causes the generation of a list of nodes excluded from contact for each segment. Each list contains nodes within IGTOL*(segment thickness + node thickness) as measured by traveling along the edges of the contact segments (see Figures 11-2 and 11-3). Note that LS-DYNA builds the

exclusion list for each segment once during the first cycle. A value of 1.0 for IGTOL should be good for most reasonable meshes. Setting IGTOL > 0.0 enables the behavior of SSTHK = 1 on *CONTROL_CONTACT automatically for this contact.

This option will improve the behavior for cases when SOFT = 0 or 1, SSTHK = 1 on *CONTROL_CONTACT, and the mesh size is equal to or smaller than the thickness. Without it, spurious stresses/deformation might be observed depending on the geometry (see Figure 11-2).

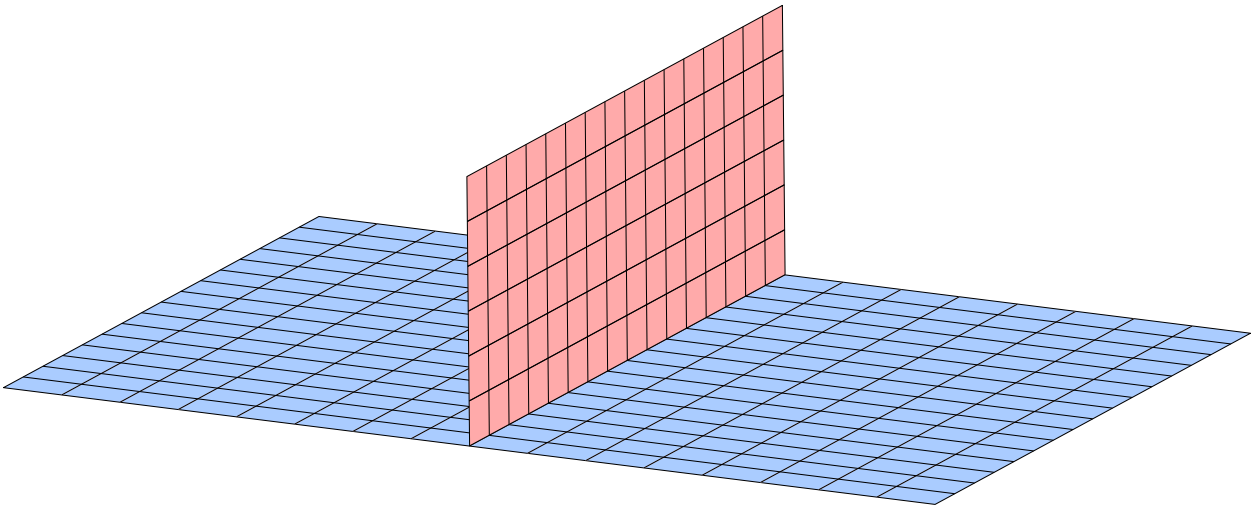


Figure 11-2. Example of a geometry that may require setting IGTOL > 0.0 when the mesh size is equal to or smaller than the thickness.

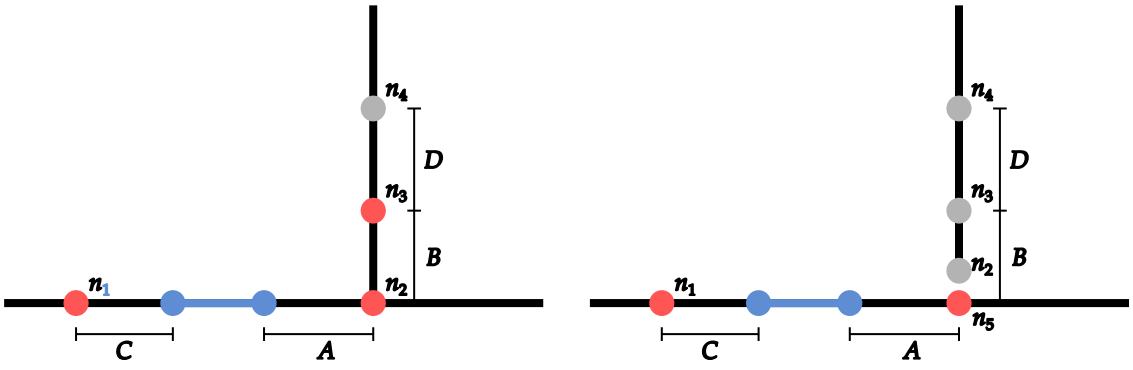


Figure 11-3. This figure illustrates two different scenarios for excluding nodes from contact with the segment in blue. In both scenarios red indicates nodes excluded from contact and grey nodes not excluded. We measure the distance by moving along the edges of the contact segments. In the left figure, n_3 is at a distance of $A + B$ from the segment and n_1 is at a distance C from the segment. In the right figure, n_2 and n_3 are not excluded because they do not connect to the segment through the mesh.

Mandatory Card 1:

> **NOTE:** SURFA is equivalent to the side of contact specified previously SSID, and SURFB is equivalent to the side of the contact specified previously with MSID.

Card 1	1	2	3	4	5	6	7	8
Variable	SURFA	SURFB	SURFATYP	SURFBTYP	SABOXID	SBBOXID	SAPR	SBPR
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none			0	0
Remarks	1	2			optional	optional		

VARIABLE	DESCRIPTION
SURFA	Segment set ID, node set ID, part set ID, part ID, shell element set ID, or branch ID for specifying the SURFA side of the contact interface (see Setting the Contact Interface). See *SET_SEGMENT, *SET_NODE_*OPTION*, *PART, *SET_PART or *SET_SHELL_*OPTION*. For ERODING_SINGLE_SURFACE and ERODING_SURFACE_TO_SURFACE contact types, use either a part ID or a part set ID. For ERODING_NODES_TO_SURFACE contact, use a node set that includes all nodes that may be exposed to contact as element erosion occurs. EQ.0: Includes all parts in the case of single surface contact types.
SURFB	Segment set ID, part set ID, part ID, shell element set ID, or branch ID for the SURFB side of the contact (see Setting the Contact Interface). EQ.0: SURFB side is not applicable for single surface contact types.
SURFATYP	ID type of SURFA: EQ.0: Segment set ID for surface-to-surface contact EQ.1: Shell element set ID for surface-to-surface contact

VARIABLE	DESCRIPTION
	EQ.2: Part set ID EQ.3: Part ID EQ.4: Node set ID for nodes-to-surface contact EQ.5: Include all non-spot-weld parts (the SURFA field is ignored). See also CPARM8. EQ.6: Part set ID for exempted parts. All non-exempted parts (except spot-weld parts) are included in the contact. EQ.7: Branch ID; see *SET_PART_TREE. For AUTOMATIC_BEAMS_TO_SURFACE contact, either a part set ID or a part ID can be specified.
SURFBTYP	ID type of SURFB: EQ.0: Segment set ID EQ.1: Shell element set ID EQ.2: Part set ID EQ.3: Part ID EQ.5: Include all non-spot-weld parts (the SURFB field is ignored). See also CPARM8. EQ.6: Part set ID for exempted parts. All non-exempted parts (except spot-weld parts) are included in the contact. EQ.7: Branch ID; see *SET_PART_TREE.
SABOXID	Include in the contact definition only those SURFA nodes/segments within box SABOXID (corresponding to BOXID in *DEFINE_BOX), or if SABOXID is negative, only those SURFA nodes/segments within contact volume |SABOXID| (corresponding to CVID in *DEFINE_CONTACT_VOLUME). SABOXID can be used only if SURFATYP is set to 2, 3, 5, or 6, meaning SURFA is a part ID or part set ID. SABOXID is not available for ERODING contact types.
SBBOXID	Include in the contact definition only those SURFB segments within box SBBOXID (corresponding to BOXID in *DEFINE_BOX), or if SBBOXID is negative, only those SURFB segments within contact volume |SBBOXID| (corresponding to CVID in *DEFINE_CONTACT_VOLUME). SBBOXID can be used only if SURFBTYP is set to 2, 3, 5, or 6, meaning SURFB is a part ID or part set ID. SBBOXID is not available for ERODING contact types.

VARIABLE	DESCRIPTION
SAPR	Include the SURFA side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files, and optionally in the dynain file for wear: EQ.0: Do not include. EQ.1: SURFA side forces included. EQ.2: Same as 1 but also allows SURFA nodes to be written as *INITIAL_CONTACT_WEAR to dynain; see NCYC on *INTERFACE_SPRINGBACK_LSDYNA.
SBPR	Include the SURFB side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files, and optionally in the dynain file for wear: EQ.0: Do not include. EQ.1: SURFB side forces included. EQ.2: Same as 1, but also allow SURFB nodes to be written as *INITIAL_CONTACT_WEAR to dynain; see NCYC on *INTERFACE_SPRINGBACK_LSDYNA.

Remarks:

1. **SURFA Set and Single Surface Contacts.** Setting the SURFA set ID equal to zero is valid only for the single surface contact algorithms, meaning the options:

   SINGLE_SURFACE

   AUTOMATIC_GENERAL_...

   AUTOMATIC_SINGLE_SURFACE_...

   AIRBAG_…

   ERODING_SINGLE_SURFACE

2. **SURFB Set with Single Surface Contact** A SURFB set ID is not defined for the single surface contact algorithms (including AUTOMATIC_GENERAL).

3. **Selecting Sides of Contact Interface for TIED_SURFACE_TO_SURFACE.** For *CONTACT_TIED_SURFACE_TO_SURFACE, we generally recommend making SURFA the more finely meshed part. This recommendation has exceptions. For instance, if the coarsely meshed SURFB side deforms, the finer SURFA side will deform with high-stress localizations in the folding lines.

Mandatory Card 2:

Card 2	1	2	3	4	5	6	7	8
Variable	FS	FD	DC	VC	VDC	PENCHK	BT	DT
Type	F	F	F	F	F	I	F	F
Default	0.	0.	0.	0.	0.	0	0.	10^{20}

VARIABLE **DESCRIPTION**

If *OPTION1* is TIED_SURFACE_TO_SURFACE_FAILURE, then

FS Normal tensile stress at failure. Failure occurs if

$$\left[\frac{\max(0.0, \sigma_{\text{normal}})}{FS}\right]^2 + \left[\frac{\sigma_{\text{shear}}}{FD}\right]^2 > 1$$

where σ_{normal} and σ_{shear} are the interface normal and shear stresses.

FD Shear stress at failure. See FS.

Else

FS Static coefficient of friction. If FS is > 0 and not equal to 2. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact according to,

$$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}|v_{\text{rel}}|} \ .$$

The three other possibilities are:

EQ.2: For a subset of SURFACE_TO_SURFACE type contacts (see Remark 1 below), FD is a table ID (see *DEFINE_TABLE). That table specifies two or more values of contact pressure, with each pressure value in the table corresponding to a curve of friction coefficient as a function of relative velocity. Thus, the friction coefficient becomes a function of pressure and relative velocity. See Figure 11-4.

EQ.-2: If only one friction table is defined using *DEFINE_FRICTION, it will be used, and there is no need to specify parameter FD. If more than one friction table is defined, the friction table ID is defined by FD

VARIABLE	DESCRIPTION

below.

EQ.-1: If the frictional coefficients defined in the *PART section are to be used, set FS to -1.0.

WARNING: The FS = -1.0 and F = -2.0 options apply only to contact types:

SINGLE_SURFACE
AIRBAG_SINGLE_SURFACE
AUTOMATIC_GENERAL
AUTOMATIC_SINGLE_SURFACE
AUTOMATIC_SINGLE_SURFACE_MORTAR
AUTOMATIC_NODES_TO_SURFACE
AUTOMATIC_SURFACE_TO_SURFACE
AUTOMATIC_SURFACE_TO_SURFACE_MORTAR
AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE
ERODING_SINGLE_SURFACE

FD Dynamic coefficient of friction. If FS > 0 and not equal to 2, the frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact according to,

$$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-DC|v_{\text{rel}}|}$$

Otherwise:

FS.EQ.-2: Friction table ID if more than one friction table is defined

FS.EQ.2: Table ID for a table that specifies two or more values of contact pressure, with each pressure value in the table corresponding to a curve of friction coefficient as a function of relative velocity.

End If

DC Exponential decay coefficient. The frictional coefficient is assumed to depend on the relative velocity, v_{rel}, of the surfaces in contact

VARIABLE	DESCRIPTION

$$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}|v_{\text{rel}}|}.$$

VC — Coefficient for viscous friction. This is necessary to limit the friction force to a maximum. A limiting force is computed as $F_{\text{lim}} = \text{VC} \times A_{\text{cont}}$ with A_{cont} being the area of the segment contacted by the node in contact. The suggested value for VC is the yield stress in shear VC $= \sigma_o/\sqrt{3}$ where σ_0 is the yield stress of the contacted material.

VDC — Viscous damping coefficient in percent of critical or the coefficient of restitution expressed as a percentage (see ICOR on Optional Card E). To avoid undesirable oscillation in contact, such as for sheet forming simulation, a contact damping perpendicular to the contacting surfaces is applied. When ICOR, the 6th column of Optional Card E, is not defined or 0, the applied damping coefficient is given by

$$\xi = \frac{\text{VDC}}{100}\xi_{\text{crit}} \ ,$$

where VDC is an integer (in units of percent) between 0 and 100. The formula for critical damping is

$$\xi_{\text{crit}} = 2m\omega,$$

where m is determined by nodal masses as

$$m = \min(m_{\text{tracked}}, m_{\text{reference}}) \ ,$$

and ω is determined from k, the interface stiffness, according to

$$\omega = \sqrt{k\frac{m_{\text{tracked}} + m_{\text{reference}}}{m_{\text{tracked}}m_{\text{reference}}}} \ .$$

PENCHK — Small penetration in the contact search option. If the tracked node penetrates more than the segment thickness times the factor XPENE (see *CONTROL_CONTACT), the penetration is ignored, and the tracked node is set free. The thickness is taken as the shell thickness if the segment belongs to a shell element, or it is taken as 1/20 of its shortest diagonal if the segment belongs to a solid element. This option applies to the surface-to-surface contact algorithms. See Table 11-2 for contact types and more details.

BT — Birth time (contact surface becomes active at this time):

LT.0: Birth time is set to |BT|. When negative, birth time is followed during the dynamic relaxation phase of the calculation. After the completion of dynamic relaxation,

VARIABLE	DESCRIPTION
	contact is activated regardless of the value of BT.
	EQ.0: Birth time is inactive, meaning contact is always active
	GT.0: If DT = -9999, BT is interpreted as the curve or table ID defining multiple pairs of birth-time/death-time; see Remark 2 below. Otherwise, if DT > 0, birth time applies both during and after dynamic relaxation.
DT	Death time (contact surface is deactivated at this time):
	LT.0: If DT = -9999, BT is interpreted as the curve or table ID defining multiple pairs of birth-time/death-time. Otherwise, negative DT indicates that contact is inactive during dynamic relaxation. After dynamic relaxation, the birth and death times are followed and set to |BT| and |DT|, respectively.
	EQ.0: DT defaults to 1020.
	GT.0: DT sets the time at which the contact is deactivated.

Remarks:

1. **Implemented contacts for FS = 2.** The FS = 2 method of specifying the friction coefficient as a function of pressure and relative velocity is implemented for all contact types for which SOFT = 2 (see When segment-to-segment contact is available in the Optional Card A section) and for all Mortar contacts. When FS = 2 and SOFT = 2, we recommend setting FNLSCL to a value in the range of 0.5 to 1.0 and setting DNLSCL to 0 (refer to Remark 5 under the description of Optional Card D for *CONTACT). These recommendations can be ignored for Mortar contact as they do not apply.

   The following ONE_WAY contact types can safely be used with FS = 2 when SOFT = 0 or 1.

ONE_WAY_SURFACE_TO_SURFACE	(SMP and MPP)
AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE	(MPP only)
FORMING_ONE_WAY_SURFACE_TO_SURFACE	(MPP only)

   FS = 2 with SOFT = 0 or 1 is implemented but not advised for the following contact types:

SURFACE_TO_SURFACE	(SMP and MPP)

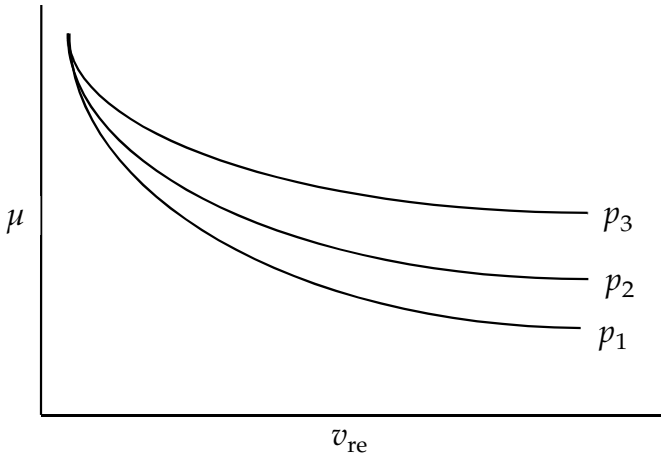


Figure 11-4. Friction coefficient, μ, can be a function of relative velocity and pressure. See Remark 1 for FS = 2.0.

AUTOMATIC_SURFACE_TO_SURFACE	(MPP only)
FORMING_SURFACE_TO_SURFACE	(MPP only)

A caveat pertaining to the MPP contacts listed above is that the "groupable" option must *not* be invoked. See *CONTROL_MPP_CONTACT_GROUPABLE.

For SOFT = 0 or 1, FS = 2 is not implemented in SMP for non-Mortar AUTOMATIC and FORMING contact types. The static friction coefficient will literally be taken as 2.0 if FS is set to 2 for these SMP contacts.

2. **Contact birth and death times.** If DT = -9999, BT is taken to be the ID of a curve (*DEFINE_CURVE) or the ID of a table (*DEFINE_TABLE(_2D)). The curve(s) define multiple birth times and death times as ordered (x, y) pairs, that is, each data point in the curve defines a time window during which the contact is active. The curve(s) must have at least two data points to satisfy general curve input requirements. For example, a curve with two data points $(20, 30)$ and $(50, 70)$ activates the contact when $20 \leq$ time ≤ 30 and when $50 \leq$ time ≤ 70. To define two separate curves that apply to the dynamic relaxation phase and the subsequent normal phase of the analysis, respectively, BT should point to a table ID. That table should have only two VALUEs, 0 and 1. The curve corresponding to the table VALUE = 0 defines the time windows of active contact for the normal phase of the analysis, and the curve corresponding to the table VALUE = 1 defines the time windows of active contact for the dynamic relaxation phase of the analysis. An example follows.

```
*CONTACT_TIED_SURFACE_TO_SURFACE
3,3,0,3
$ DT = -9999 has a very special meaning (see User's Manual).
,,,,,, 600, -9999
```

```
*DEFINE_TABLE_2D
600
$$ value of "0"  indicates that the corresponding curve is for the normal phase
$$ value of "1"  indicates that the corresponding curve is for the DR phase
0,701
1,702
$$ Curves must have at least 2 points to satisfy general input requirements.
*DEFINE_CURVE
$ this curve gives a range of time for which contact is active during the normal phase
701
$ active from t=0 to t=1.1e-3
0,1.1e-3
$ active from t=10 to t=20
10,20
*DEFINE_CURVE
$ this curve gives a range of time for which contact is active during DR
702
$ birth contact at time=2.e-3, death at time = 1.0
2.e-3,1
$ active from t=1 to t=2\
1,2
```

Mandatory Card 3:

Card 3	1	2	3	4	5	6	7	8
Variable	SFSA	SFSB	SAST	SBST	SFSAT	SFSBT	FSF	VSF
Type	F	F	F	F	F	F	F	F
Default	1.	1.	element thickness	element thickness	1.	1.	1.	1.

VARIABLE	DESCRIPTION
SFSA	Scale factor on default SURFA penalty stiffness when SOFT = 0 or SOFT = 2; see also *CONTROL_CONTACT. For MORTAR *frictional* contact, this is the stiffness scale factor for the entire contact, and SFSB does not apply. LT.0: -SFSA gives the load curve ID for contact pressure as a function of penetration depth. This is supported for Mortar contact only.
SFSB	Scale factor on default SURFB penalty stiffness when SOFT = 0 or SOFT = 2; see also *CONTROL_CONTACT. For MORTAR *tied* contact, this is an additional stiffness scale factor, resulting in a total stiffness scale of SFSA × SFSB.
SAST	Optional contact thickness for SURFA surface (overrides default contact thickness). This option applies to contact with shell and beam elements. SAST has no bearing on the actual thickness of the elements; it only affects the location of the contact surface. For the *CONTACT_TIED_… options, SAST and SBST (below) can be defined as negative values, which will cause the determination of whether or not a node is tied to depend only on the separation distance relative to the absolute value of these thicknesses (see Remark 4 in General Remarks). More information is given under General Remarks: *CONTACT.
SBST	Optional contact thickness for SURFB surface (overrides default contact thickness). This option applies only to contact with shell elements. For the TIED options, see SAST above.
SFSAT	Scale factor applied to the contact thickness of the SURFA surface. This option applies to contact with shell and beam elements. SF-

VARIABLE	DESCRIPTION
	SAT has no bearing on the actual thickness of the elements; it only affects the location of the contact surface. SFSAT is ignored if SAST is nonzero except in the case of MORTAR contact (see Remark 14 in the General Remarks: *Contact section).
SFSBT	Scale factor applied to the contact thickness of the SURFB surface. This option applies only to contact with shell elements. SFSBT has no bearing on the actual thickness of the elements; it only affects the location of the contact surface. SFSBT is ignored if SBST is nonzero except in the case of MORTAR contact (see Remark 14 in the General Remarks: *Contact section).
FSF	Coulomb friction scale factor. The Coulomb friction value is scaled as μ_{sc} = FSF × μ_c; see Mandatory Card 2.
VSF	Viscous friction scale factor. If this factor is defined, the limiting force becomes: F_{lim} = VSF × VC × A_{cont}; see Mandatory Card 2.

Remarks:

The fields FSF and VSF above can be overridden segment by segment on the *SET_SEGMENT or *SET_SHELL_*OPTION* cards for the **SURFA surface only** as A3 and A4 and for the **SURFB surface only** as A1 and A2. See *SET_SEGMENT and *SET_SHELL_*OPTION*.

Card 4: AUTOMATIC_…_TIEBREAK

This card 4 is mandatory for:

CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_*{OPTION}

CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK_*{OPTION}

*CONTACT_AUTOMATIC_SINGLE_SURFACE_TIEBREAK

*CONTACT_AUTOMATIC_GENERAL_TIEBREAK

If the response parameter, OPTION, below is set to 9 or 11, three damping constants can be defined for the various failure modes. To do this, set the keyword option to

DAMPING

For OPTION = -11, -9, 2, 4, 6, 7, 8, 9, and 11 of *CONTACT_AUTOMATIC_SURFACE_-TO_SURFACE_TIEBREAK, the Mortar treatment may be activated. This is primarily intended for implicit analysis. The keyword option for this is

MORTAR

The DAMPING option cannot be combined with the MORTAR option. With the MORTAR option, the adhesive strength, which is essentially the inverse of the damage, can be output to the intfor file by specifying NTIED on *DATABASE_EXTENT_INTFOR.

Card 4	1	2	3	4	5	6	7	8
Variable	OPTION	NFLS	SFLS	PARAM	ERATEN	ERATES	CT2CN	CN
Type	I	F	F	F	F	F	F	F
Default	required	required	required	↓	0.0	0.0	1.0	↓

Damping Card. Additional card when OPTION = 9 or 11 with the DAMPING keyword option active.

Card 4.1a	1	2	3	4	5	6	7	8
Variable	DMP_1	DMP_2	DMP_3					
Type	F	F	F					
Default	0.0	0.0	0.0					

OPTION = 13/14 Cards. Two additional cards for the case of OPTION = 13 or 14.

Card 4.1b	1	2	3	4	5	6	7	8
Variable	G1C_0	G1C_INF	EDOT_G1	T0	T1	EDOT_T	FG1	LCG1C
Type	F	F	F	F	F	F	F	F

OPTION = 13/14 Cards. Two additional cards for the case of OPTION = 13 or 14.

Card 4.2b	1	2	3	4	5	6	7	8
Variable	G2C_0	G2C_INF	EDOT_G2	S0	S1	EDOT_S	FG2	LCG2C
Type	F	F	F	F	F	F	F	F

OPTION = 13/14 Cards. One additional card for the case of OPTION = 13 or 14 and PARAM < 0 (negative thickness value).

Card 4.3b	1	2	3	4	5	6	7	8
Variable	RFILTF							
Type	F							

VARIABLE	DESCRIPTION
OPTION	Response: EQ.-11: See 11. NFLS/SFLS/ERATEN/ERATES are functions of

VARIABLE	DESCRIPTION
	temperature. MORTAR option only.
EQ.-9:	See 9. NFLS/SFLS/ERATEN/ERATES are functions of temperature. MORTAR option only.
EQ.-3:	See 3. Moments are transferred. We recommend using contact type AUTOMATIC_ONE_WAY_SURFACE_-TO_SURFACE_TIEBREAK with this option.
EQ.-2:	See 2. Moments are transferred. We recommend using contact type AUTOMATIC_ONE_WAY_SURFACE_-TO_SURFACE_TIEBREAK with this option.
EQ.-1:	See 1. Moments are transferred. We recommend using contact type AUTOMATIC_ONE_WAY_SURFACE_-TO_SURFACE_TIEBREAK with this option.
EQ.1:	Tracked nodes in contact and those that come into contact will permanently stick. Tangential motion is inhibited.
EQ.2:	Tiebreak is active for nodes that are initially in contact. Until failure, tangential motion is inhibited. If PARAM is set to unity (1.0), shell thickness offsets are ignored, and the orientation of the shell surfaces is required such that the outward normals point to the opposing contact surface.
EQ.3:	Same as 1 above but with failure after sticking.
EQ.4:	Tiebreak is active for nodes initially in contact, but tangential motion with frictional sliding is permitted.
EQ.5:	Tiebreak is active for nodes that are initially in contact. Stress is limited by the yield condition described in Remark 3 below. Damage behavior is modeled by a curve that defines normal stress as a function of the gap (crack opening). This option can be used to represent deformable glue bonds.
EQ.6:	This option is only for solids and thick shells. Tiebreak is active for nodes that are initially in contact. Failure stress must be defined for tiebreak to occur. After the failure stress tiebreak criterion is met, damage is a linear function of the distance between points initially in contact. When the distance equals PARAM, damage is fully developed, and interface failure occurs. After failure, this option behaves as a surface-to-surface contact. We recommend using type AUTOMATIC_ONE_WAY_-

VARIABLE	DESCRIPTION
	SURFACE_TO_SURFACE_TIEBREAK with this option.
	EQ.7: Dycoss Discrete Crack Model. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option. See Remark 3.
	EQ.8: Similar to OPTION = 6, but it works with offset shell elements. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option
	EQ.9: Discrete Crack Model with power law and B-K damage models. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option. See Remark 3.
	EQ.10: Similar to OPTION = 7, but it works with offset shell elements. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option.
	EQ.11: Similar to OPTION = 9, but it works with offset shell elements. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option
	EQ.13: Elastoplastic, rate-dependent damage model based on *MAT_240. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option. See Remark 7.
	EQ.14: Similar to OPTION = 13, but it works with offset shell elements. We recommend using type AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with this option
NFLS	Normal failure stress for OPTION = 2, 3, 4, 6, 7, 8, ±9, 10, or ±11. For OPTION = 5, NFLS becomes the plastic yield stress as defined in Remark 5. For OPTION = 9 or 11 and NFLS < 0, a load curve with ID |NFLS| is referenced, defining normal failure stress as a function of element size. See Remark 3. For OPTION = -9 or -11 and NFLS < 0, |NFLS| is the ID of a load curve giving normal failure stress as function of temperature; it applies to the Mortar option only.
SFLS	Shear failure stress for OPTION = 2, 3, 6, 7, 8, ±9, 10, or ±11. For OPTION = 4, SFLS is a frictional stress limit if PARAM = 1. This frictional stress limit is independent of the normal force at the tie.

VARIABLE	DESCRIPTION
	For OPTION = 5, SFLS becomes a curve ID that defines normal stress as a function of the gap. For OPTION = 9 or 11 and SFLS < 0, |SFLS| references a load curve ID, defining shear failure stress as a function of element size. See Remark 3. For OPTION = -9 or -11 and SFLS < 0, |SFLS| is the ID of a load curve giving shear failure stress as function of temperature; it applies to the Mortar option only.
PARAM	For OPTION = 2, setting PARAM = 1 causes the shell thickness offsets to be ignored. For OPTION = 4, setting PARAM =1 causes SFLS to be a frictional stress limit. For OPTION = 6 or 8, PARAM is the critical distance, CCRIT, at which the interface failure is complete. For OPTION = 7 or 10, PARAM is the friction angle in degrees. For OPTION = 9 or 11, it is the exponent in the damage model. A positive value invokes the power law, while a negative one invokes the B-K model. See *MAT_138 for additional details. For OPTION = 13 or 14, it is the thickness of the tiebreak layer; a value greater than zero is recommended. A negative sign in front of the thickness value invokes the additional reading of Card 4.3b for OPTION = 13 or 14, with the absolute value taken as the thickness. The default value is 1.0 for OPTIONs 9 and 11 and 0.0 otherwise.
ERATEN	For OPTION = 7, ±9, 10, ±11 only. Normal energy release rate (stress × length) used in the damage calculation; see Lemmen and Meijer [2001]. For OPTION = -9 or -11, this is the ID of a load curve giving normal energy release rate as a function of temperature; it applies to the Mortar option only.
ERATES	For OPTION = 7, ±9, 10, ±11 only. Shear energy release rate (stress × length) used in the damage calculation; see Lemmen and Meijer [2001]. For OPTION = -9 or -11, this is the ID of a load curve giving shear energy release rate as a function of temperature; it applies to the Mortar option only.
CT2CN	The ratio of the tangential stiffness to the normal stiffness for OPTION = 9, 11, 13, and 14. The default is 1.0.
CN	Normal stiffness (stress/length) for OPTION = 9, 11, 13, and 14 and for OPTION = 2, 4, 6, 7, and 8 for the MORTAR option only. If CN is not given explicitly, penalty stiffness divided by segment area is used (default). This optional stiffness should be used with care since contact stability can be affected. A warning message with a recommended time step is given initially.
DMP_1	Mode I damping force per unit velocity per unit area

VARIABLE	DESCRIPTION
DMP_2	Mode II damping force per unit velocity per unit area
DMP_3	Mode III damping force per unit velocity per unit area
G1C_0, …	All variables on Cards 4.1b, 4.2b, and 4.3b are the same as in *MAT_-240.

Remarks:

1. **Contact behavior after failure.** After failure, this contact option behaves as a surface-to-surface contact with thickness offsets. After failure, no interface tension is possible.

2. **Restrictions.** Segment-to-segment contact (SOFT = 2) is *not* implemented for the tiebreak option. *CONTACT_AUTOMATIC_SINGLE_SURFACE_TIEBREAK and *CONTACT_AUTOMATIC_GENERAL_TIEBREAK are supported only for OPTIONs 1 through 5 and only for MPP.

3. **Tiebreak, failure, damage, and crack initiation.** The following list overviews the failure criteria for the various OPTIONs.

   a) For OPTION = 2, 3, and 6, the tiebreak failure criterion has normal and shear components:

   $$\left(\frac{|\sigma_n|}{\text{NFLS}}\right)^2 + \left(\frac{|\sigma_s|}{\text{SFLS}}\right)^2 \geq 1 \ .$$

   σ_n is the tensile normal stress and is taken as zero if the normal stress is compressive. σ_s is the shear stress.

   b) For OPTION = 4, the tiebreak failure criterion has only a tensile normal stress component:

   $$\frac{|\sigma_n|}{\text{NFLS}} \geq 1 \ .$$

   c) For OPTION = 5, the stress is limited by a perfectly plastic yield condition. For ties in tension, the yield condition is

   $$\frac{\sqrt{\sigma_n^2 + 3|\sigma_s|^2}}{\text{NLFS}} \leq 1 \ .$$

   For ties in compression, the yield condition is

   $$\frac{\sqrt{3|\sigma_s|^2}}{\text{NLFS}} \leq 1 \ .$$

The stress is also scaled by the damage function, which is obtained from the load curve. For ties in tension, both normal and shear stress are scaled. For ties in compression, only shear stress is scaled.

d) For OPTION = 6 or 8, damage initiates when the stress meets the failure criterion. The damage function then scales the stress. Assuming no load reversals, the energy released due to the failure of the interface is approximately $0.5 \times S \times$ CCRIT, where

$$S = \sqrt{\max(\sigma_n, 0)^2 + |\sigma_s|^2}$$

at the initiation of damage. This interface may be used to simulate crack propagation. For the energy release to be correct, the contact penalty stiffness must be much larger than

$$\frac{\min(\text{NFLF}, \text{SFLS})}{\text{CCRIT}} \ .$$

e) OPTION = 7 and 10 are implementations of the Dycoss Discrete Crack Model described in Lemmen and Meijer [2001]. The relation for the crack initiation is given as

$$\left[\frac{\max(\sigma_n, 0)}{\text{NFLS}}\right]^2 + \left[\frac{\sigma_s}{\text{SFLS} - \sin(\text{PARAM})\min(0, \sigma_n)}\right]^2 = 1 \ .$$

f) OPTION = 9 and 11 are based on the fracture model in the cohesive material model *MAT_COHESIVE_MIXED_MODE, which describes the model in detail. Failure stresses/peak tractions NFLS and/or SFLS can be defined as functions of characteristic element length (square root of the reference segment area) using load curves. With these options, nearly the same global responses (for instance, load-displacement curve) can be obtained with coarse meshes compared to fine meshes. In general, lower peak tractions are needed for coarser meshes. See also *MAT_138.

4. **Determining the state of the tiebreak surface.** For OPTIONs 6 thru 11 of *CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK, the component labeled "contact gap" in the intfor database (*DATABASE_BINARY_INTFOR) indicates the condition of the tiebreak surface. The "contact gap" represents a damage value ranging from 0 (tied, no damage) to 1 (released, fully damaged).

5. **Automatic tiebreak tying tolerance.** Tying in the AUTOMATIC_..._TIEBREAK contacts occurs if the tracked node is within a small tolerance of the reference surface *after* accounting for contact thicknesses. For MPP, the tolerance is given by

$$\text{tol} = 0.01\sqrt{2 \times \text{reference segment area}} \ .$$

For SMP, the tolerance is

$$0.4(\text{tracked contact thickness} + \text{reference contact thickness}) .$$

6. **Defining SURFB and SURFA sets.** We recommend defining the SURFA and SURFB sides of tiebreak contact using segment sets rather than part or part set IDs. Doing this enables being more selective when choosing which segments to tie and ensures that contact stresses calculated from nodal contact forces are not diluted by segments not actually on the actual contact surface. It also gives more direct control over the contact segment normal vectors when segment sets are used.

   Segment normal vectors should point toward the opposing contact surface to distinguish tension from compression properly. Also, unlike regular automatic contacts, segment normal vectors need to be properly oriented in MPP because they are not automatically oriented. This issue can lead to incorrect results upon failure.

7. **Variables applicable to OPTIONs 13 and 14.** OPTIONs 13 and 14 are based on material model 240; see LS-DYNA Keyword User's Manual, Volume II. Applicable variables are PARAM, CT2CN, CN, and the variables on optional Cards 4.1b and 4.2b. NFLS, SFLS, ERATEN, and ERATES are not used.

Card 4: AUTOMATIC_…_SURFACE_TO_SURFACE_TIEBREAK_USER_...

These cards, 4.1, 4.2, and 4.3, are mandatory for:

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_USER

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_USER_MORTAR

*CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK_USER

Card 4.1	1	2	3	4	5	6	7	8
Variable	OPTION	NHV	CT2CN	CN	OFFSET	NHMAT	NHWLD	
Type	I	I	F	F	I	I	I	
Default	required	0	1.0	0.0	0	0	0	

Card 4.2	1	2	3	4	5	6	7	8
Variable	UP1	UP2	UP3	UP4	UP5	UP6	UP7	UP8
Type	F	F	F	F	F	F	F	F

Card 4.3	1	2	3	4	5	6	7	8
Variable	UP9	UP10	UP11	UP12	UP13	UP14	UP15	UP16
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
OPTION	User tiebreak type (101 - 105 inclusive). A number between 101 and 105 must be chosen. Corresponding subroutine `utb<OPTION>` in dyn21cnt.f will be called for the non-Mortar option, while subroutine `mortar_usrtbrk` will be called for the Mortar option.
NHV	Number of history variables (maximum of 3 for non-Mortar, arbitrary for Mortar)

VARIABLE	DESCRIPTION
CT2CN	Ratio of the tangential stiffness to the normal stiffness
CN	Normal stiffness (stress/length). If CN is not given explicitly, penalty stiffness divided by segment area is used (default). This optional stiffness should be used with care since contact stability can be affected. A warning message with a recommended time step is given initially.
OFFSET	Flag for offset treatment (only for non-Mortar option): EQ.0: No offset EQ.1: With offset for shell elements
NHMAT	Number of material history variables to be read in the user tiebreak routine (Mortar option only)
NHWLD	Number of tied weld history variables to be read in the user tiebreak routine, assuming they have been carried over from a previous simulation (Mortar option only). See Remark 4.
UP1…UP16	User parameters

Remarks:

1. **User-defined tiebreak model.** This option allows the implementation of a user-defined tiebreak model. For more information, please check the comments in subroutine `utb101` or `mortar_usrtbrk` (dyn21cnt.f).

2. **Contact behavior after failure.** After failure, this contact option behaves as a surface-to-surface contact with thickness offsets. After failure, no interface tension is possible.

3. **Preferred contact type.** We recommend using the …_ONE_WAY_... definition for this option.

4. **Mortar option.** Essentially, the same behavior can be expected for the Mortar contact as for the non-Mortar contact. The Mortar capability, however, is accompanied by some extra features. For instance, this Mortar contact may be used to assess the adhesive properties of a preceding lamination process by reading history variables from a previous simulation that used a corresponding user Mortar tied weld model (see Card 4: AUTOMATIC_SURFACE_TO_SURFACE_..._TIED_WELD). To do this, the ID of the present tiebreak contact should be the same as the ID of the earlier tied weld contact leading up to the start of the current delamination simulation, and NHWLD should state how many

history variables to read. Furthermore, *INTERFACE_SPRINGBACK_LSDYNA with FTYPE = 3 and CFLAG = 1 should be used to create a dynain.lsda in the previous simulation which will be read with *INCLUDE in the current simulation, all in accordance with standard continuation procedures.

Card 4: AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE/LUBRICATION

This contact model is designed to simulate the processing of laminated composite materials or certain types of lubrication. Surfaces in contact may support shear up to the limit defined by MODEL and be in compression or in tension up to the tensile limit, σ_f, defined by TFAIL. After TFAIL is reached, the contact fails in both tension and shear. If the surfaces come back into contact, the bonding heals, and the contacting surfaces may support shear and tension.

This Card 4 is mandatory for:

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_LUBRICATION

Card 4	1	2	3	4	5	6	7	8
Variable	TFAIL	MODEL	CIDMU	CIDETA	D			
Type	F	I	I	I	F			
Default	required	required	required	required	0.0			

VARIABLE	DESCRIPTION
TFAIL	Tensile traction, σ_f, required for failure
MODEL	Model for shear response (see Remark 2): EQ.1: Limiting shear stress depends on CIDMU in both tension and compression. EQ.2: Limiting shear stress depends on CIDETA in tension and CIDMU in compression. EQ.3: Limiting shear stress depends on CIDETA in both tension and compression.
CIDMU	Curve ID for the coefficient of friction, $\mu(H)$, as a function of the Hershey number, H
CIDETA	Curve ID for the viscosity, $\eta(T)$, as a function of temperature, T (see Remark 1)
D	Composite film thickness

Remarks:

1. **Viscosity.** The viscosity, $\eta(T)$, is defined as a function of temperature by CIDETA. The viscosity value is not extrapolated if the temperature falls outside the temperature range defined by the curve.

2. **Shear stress limits.** The following lists the shear stress limits for the different values of MODEL:

   a) *MODEL = 1.* The coefficient of friction, μ, for MODEL = 1 is defined in terms of the Hershey number, $H = \eta(T)V/(p + \sigma_f)$, where p is the contact pressure (positive in compression and negative in tension) and V the relative velocity between the surfaces. The shear stress in tension and compression is limited according to

   $$\tau \le \mu(H)\left(p + \sigma_f\right) \ .$$

   b) *MODEL = 2.* The coefficient of friction, μ, for MODEL = 2 is defined in terms of the Hershey number, $H = \eta(T)V/p$. Note that the definition of the Hershey number for this model differs from MODEL = 1. In compression, the shear stress is limited by

   $$\tau \le \mu(H)p \ .$$

   and in tension, the shear stress is limited according to

   $$\tau \le \frac{\eta(T)V}{d} \ .$$

   c) *MODEL = 3.* The shear stress for MODEL = 3 in tension and compression is limited according to

   $$\tau \le \frac{\eta(T)V}{d} \ .$$

3. **Temperature initialization.** As of R15, not initializing the temperature of the model leads to an error termination with error KEY+1490 ("*CONTACT_xxx_COMPOSITE/LUBRICATION: Temperature not set!"). In previous versions, the solver did not throw this error. However, not setting the temperature could lead to unpredictable behavior or the run to eventually error terminate because obtaining the viscosity requires the temperature (see Remark 1). To resolve this issue, set the temperature field for the model, such as with *LOAD_THERMAL_CONSTANT.

Card 4: AUTOMATIC_SINGLE_SURFACE_TIED

This special feature was initially implemented to allow for the calculation of eigenvalues and eigenvectors on geometries connected by a contact interface using the AUTOMATIC_SINGLE_SURFACE options. It also can be used more generally to tie a set of parts based only on their proximity to each other in the initial geometry. This contact type does not revert to non-tied AUTOMATIC_SINGLE_SURFACE behavior at any time during the simulation.

This Card 4 is mandatory for:

*CONTACT_AUTOMATIC_SINGLE_SURFACE_TIED

Card 4	1	2	3	4	5	6	7	8
Variable	CLOSE							
Type	F							
Default	Rem 1							

VARIABLE	DESCRIPTION
CLOSE	Tolerance for tying surfaces using a penalty formulation. Surfaces closer than CLOSE are tied. Surfaces farther apart than CLOSE are not tied.

Remarks:

1. **CLOSE default.** If a value of CLOSE is not input, the default value of CLOSE is calculated for each tracked node and is equal to 0.0106 times the square root of the area of the nearest reference segment.

2. **Rigid body modes.** If there is significant separation between the tied surfaces, the rigid body modes will be opposed by the contact stiffness, and the calculated eigenvalues for rigid body rotations will not be zero.

3. **Recommended settings.** We recommend setting TIEDID on Optional Card D to 1 when using this contact type.

Card 4: AUTOMATIC_SURFACE_TO_SURFACE_..._TIED_WELD

This special feature is implemented to allow for the simulation of welding. As regions of the surfaces are heated to the welding temperature and come into contact, the nodes are tied.

This Card 4 is mandatory for:

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIED_WELD

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED_WELD

Card 4	1	2	3	4	5	6	7	8
Variable	TEMP	CLOSE	HCLOSE	NTPRM	NMHIS	NSTWH	NMTWH	TIME
Type	F	F	F	F	F	F	F	F
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
TEMP	For the non-Mortar option, this is the minimum temperature required on both surfaces for tying. For the Mortar tied weld, surfaces tie when one of the surfaces reaches the specified temperature. Once the surfaces are tied, they remain tied even if the temperature drops. LT.0: |TEMP| represents the user tied weld ID passed to subroutine `mortar_usrtie` for defining an arbitrary condition to tie. This option is only available for MORTAR_TIED_WELD, and TEMP must be between -999 and -1.
CLOSE	Surfaces closer than CLOSE are tied. If CLOSE is left as 0.0, it defaults to one percent of the mesh characteristic length scale. Nodes that are above or below the surface will be tied if they are close enough to the surface.
HCLOSE	Thermal contact conductivity for a tied interface, in case of using the option THERMAL in the contact
NTPRM	If TEMP < 0, number of user-defined tied weld parameters; otherwise, ignored. NTPRM allocates the array `cprm(*)` in subroutine `mortar_usrtie`.

VARIABLE	DESCRIPTION
NMHIS	If TEMP < 0, number of material history variables accessible from the user-defined tied weld interface; otherwise ignored. NMHIS allocates the arrays `shis(*)` and `mhis(*)` in subroutine `mortar_usrtie`.
NSTWH	If TEMP < 0, number of SURFA tied weld history variables for the user-defined tied weld interface. NSTWH allocates the array `shst(*)` in subroutine `mortar_usrtie`.
NMTWH	If TEMP < 0, number of SURFB tied weld history variables for the user-defined tied weld interface. NMTWH allocates the array `mhst(*)` in subroutine `mortar_usrtie`.
TIME	This parameter applies to the Mortar tied weld contact only. The conditions above for tying/welding must be satisfied for at least TIME consecutive time units for tying/welding to occur. This option is intended to prevent premature tying/welding which may otherwise occur if only temperature is considered.

User Tied Welds Parameters Card. Insert as many cards needed to define NTPRM tied weld parameters.

Card 4.1	1	2	3	4	5	6	7	8
Variable	TPRM1	TPRM2	TPRM3	TPRM4	TPRM5	TPRM6	TPRM7	TPRM8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
TPRM*i*	User-defined tied welds parameter *i*

Remarks:

1. **Separation and rigid body modes.** If there is significant separation between the tied surfaces, the rigid body modes will not be opposed by the contact stiffness. In other words, the offset between the surfaces is handled like the contact with OFFSET.

2. **Contact behavior below welding temperature.** If the surfaces are below the welding temperature, the surfaces interact with the standard AUTOMATIC_-SURFACE_TO_SURFACE options.

3. **MORTAR contact.** The MORTAR option is primarily intended for implicit and is supported for SMP and MPP. With this option, the tied segments can be viewed in the intfor file by specifying NTIED on *DATABASE_EXTENT_INT-FOR. To implement a user-defined condition for tying segments, set TEMP < 0. See subroutine `mortar_usrtie` for comments and a sample code in the source code of an object version of LS-DYNA. The history variables can subsequently be used in a Mortar tiebreak routine for assessing the quality of the lamination (see Remark 4 in the Card 4: AUTOMATIC_..._SURFACE_TO_SURFACE_TIE-BREAK_USER_... section)._.

Card 4: CONSTRAINT_..._TO_SURFACE

This card 4 is mandatory for:

*CONTACT_CONSTRAINT_NODES_TO_SURFACE

*CONTACT_CONSTRAINT_SURFACE_TO_SURFACE

Card 4	1	2	3	4	5	6	7	8
Variable	KPF							
Type	F							
Default	0.0							

VARIABLE	DESCRIPTION
KPF	Kinematic partition factor for constraint: EQ.0.0: Fully symmetric treatment EQ.1.0: One-way treatment with SURFA nodes constrained to SURFB surface. Only the SURFA nodes are checked against contact. EQ.-1.0: One-way treatment with SURFB nodes constrained to SURFA surface. Only the SURFB nodes are checked against contact.

Card 4: DRAWBEAD

This card 4 is mandatory for:

*CONTACT_DRAWBEAD

*CONTACT_DRAWBEAD_BENDING

*CONTACT_DRAWBEAD_INITIALIZE

Note that variables related to the automatic multiple draw bead feature, meaning NBEAD, POINT1, POINT2, WIDTH, and EFFHGT, only work with draw beads defined with node sets, not beam elements. Also, NBEAD and the related variables on Card 4.4 can be used with the keyword option INITIALIZE or BENDING.

The BENDING keyword option is for weakening the blank underneath the draw bead. See Weakening Effect below.

Card 4.1	1	2	3	4	5	6	7	8
Variable	LCIDRF	LCIDNF	DBDTH	DFSCL	NUMINT	DBPID	ELOFF	NBEAD
Type	I	I	F	F	I	I	I	I
Default	required	none	0.0	1.0	0	0	0	optional

Bending Card. Additional card for BENDING keyword option. This card is for modeling the draw bead weakening effect.

Card 4.2	1	2	3	4	5	6	7	8
Variable	EPM	EPSCALE	ENDING					
Type	F	F	F					
Default	none	none	10.					

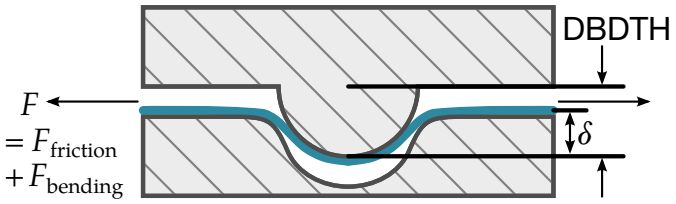


Figure 11-5. The draw bead contact model.

Initialization Card. Additional card for INITIALIZE keyword option. This card initializes the plastic strain and thickness of elements that pass under the draw bead.

Card 4.3	1	2	3	4	5	6	7	8
Variable	LCEPS	TSCALE	LCEPS2	OFFSET				
Type	I	F	I	F				
Default	required	1.0	optional	optional				

Additional card to be included if NBEAD in Card 4.1 is defined.

Card 4.4	1	2	3	4	5	6	7	8
Variable	POINT1	POINT2	WIDTH	EFFHGT				
Type	I	I	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
LCIDRF	GT.0: Load curve ID giving the bending component of the restraining force, $F_{bending}$, per unit draw bead length as a function of displacement, δ; see Figure 11-5. This force is due to the bending and unbending of the blank as it moves through the draw bead. The total restraining force is the sum of the bending and friction components. LT.0: |LCIDRF| gives the load curve ID defining maximum bead force as a function of normalized draw bead length. The abscissa values are between zero and one and are the normalized draw bead length. The ordinate gives the maximum allowed draw bead retaining force when the bead is

VARIABLE	DESCRIPTION
	in the fully closed position. If the draw bead is not fully closed, linear interpolation is used to compute the draw bead force.
LCIDNF	Load curve ID giving the normal force per unit draw bead length as a function of displacement, δ; see Figure 11-5. This force originates from bending the blank into the draw bead as the binder closes on the die. The normal force begins to develop when the distance between the die and binder is less than the draw bead depth. As the binder and die close on the blank, this force should diminish or reach a plateau.
DBDTH	Draw bead depth; see Figure 11-5. This depth is needed to determine the correct displacement, δ, from contact displacements.
DFSCL	Scale factor for load curve (default = 1.0). This factor scales the load curve ID, LCIDRF, above.
NUMINT	Number of equally spaced integration points along the draw bead: EQ.0: Internally calculated based on the element size of elements that interact with the draw bead. This is necessary for the correct calculation of the restraining forces. More integration points may increase the accuracy since the force is applied more evenly along the bead.
DBPID	Optional part ID for the automatically generated truss elements used to display the draw bead during post-processing. If undefined, LS-DYNA assigns a unique part ID.
ELOFF	Option to specify an element ID offset for the truss elements that are automatically generated to display the draw bead during post-processing. If undefined, LS-DYNA chooses a unique offset.
NBEAD	Number of line beads. It must be an odd integer.
EPM	Maximum strain the blank will experience when it passes the bead
EPSCALE	Scale factor to weaken the stress-strain curve
ENDING	Parameter to define the length of the bead ends where the weakening effect will phase out
LCEPS	Load curve ID defining the plastic strain as a function of the parametric coordinate through the shell thickness. The parametric

VARIABLE	DESCRIPTION
	coordinate must be defined in the interval between -1 and 1 inclusive. The value of plastic strain at the integration point is interpolated from this load curve. If the plastic strain at an integration point exceeds the value of the load curve at the time initialization occurs, the plastic strain at the point will remain unchanged.
TSCALE	Scale factor that multiplies the shell thickness as the shell element moves under the draw bead
LCEPS2	Optional load curve ID defining the plastic strain as a function of the parametric coordinate through the shell thickness, which is used after an element has traveled a distance equal to OFFSET. The parametric coordinate should be defined in the interval between -1 and 1 inclusive. The value of plastic strain at the integration point is interpolated from this load curve. If the plastic strain at an integration point exceeds the value of the load curve at the time initialization occurs, the plastic strain at the point will remain unchanged. Input parameters LCEPS2 and OFFSET provide a way to model the case where a material moves under two draw beads. In this latter case, the curve would be the sum of the plastic strains generated by moving under two consecutive beads.
OFFSET	If the center of an element has moved a distance equal to OFFSET, the load curve ID, LCEPS2, is used to reinitialize the plastic strain. The TSCALE scale factor is also applied.
POINT1	Node ID of the first node on a binder
POINT2	Node ID of a *matching* node on the opposing binder
WIDTH	Total bead width defining the distance between the innermost and outermost bead walls
EFFHGT	Effective bead height. The draw bead restraining force starts to take effect when the binder gap is less than EFFHGT.

Overview:

For this draw bead model, the blank is the SURFB part, and the male part of the draw bead is SURFA. The male part of the draw bead, which moves with the punch, is input as a curve defined using a list of nodes or a part consisting of beams, as discussed below. Associated with this curve is a *region of influence* characterized by the DBDTH field of Card 4.1.

As the punch comes down and the region of influence intersects the elements on the blank, forces are applied to the blank at the points of closest approach. These forces depend on the penetration distance, δ, which is geometrically defined in Figure 11-5. The draw bead force model consists of two terms:

1. a *resisting force* which is a function of δ, and is defined through the load curve specified in the LCIDRF field. This force is applied in a direction opposite to velocity.

2. a *normal force*, which pushes the male part of the draw bead away from the blank as specified by LCIDNF. This normal force, in turn, is used to model friction, which depends on the product of the friction coefficient and the normal force.

The curve representing the male part of the draw bead can be defined in three ways:

1. a *consecutive* list of *SURFA* nodes that lie along the bead

2. a part ID of a beam that lies along the draw bead

3. a part set ID of beams that lie along the draw bead

For straight draw beads, only two nodes or a single beam must be defined, that is, one at each end. For curved beads, many nodes or beams may be required to give the curvature of the bead geometry.

When beams are used to define the bead, each node, except for the first and last, must connect with two beam elements. Therefore, the number of SURFA nodes must equal the number of beam elements plus one.

The contact algorithm checks for penetration at the integration points. Integration points are equally spaced along the draw bead and do not depend on the nodal spacing used in the definition of the draw bead. By tying extra nodes to rigid bodies (see *CONSTRAINED_EXTRA_NODES or *CONSTRAINED_RIGID_BODIES), the draw bead nodal points do not need to belong to the element connectivities of the die and binder. The blank makes up the SURFB surface.

> **NOTE:** We recommended defining a box (see *DEFINE_BOX) around the draw bead to limit the size of the SURFB surface considered for the draw bead. This will substantially reduce the cost and memory requirements.

Multiple Draw Beads Model:

Developed in conjunction with the *Ford Motor Company Research & Advanced Engineering Laboratory*, the multiple draw bead feature provides a simple way (1) to model the

neglected effects of the draw bead width and (2) to attenuate the bead forces when the distance between the upper and lower binders is more than the draw bead height.

1. **Draw bead width correction.** As shown in Figure 11-6, a sheet blank's edge often does not cross the draw bead's definition curve but falls within its width. When the bead is modeled as a 1-dimensional (no width) curve, no forces may be applied to a major portion of the blank. The neglect of width effects leads to excessive blank edge draw-ins, resulting in either loose metal in the part or wrinkles on the draw wall or product surface.

   The multiple beads feature improves this shortcoming by replacing the single 1-dimensional bead with an equivalent set of beads distributed over the width of the physical bead. The bead force is distributed uniformly over the NBEAD sub-beads, such that the resultant force is equal to that of the original 1-dimensional bead. Note that NBEAD *must* be an odd integer.

   Figure 11-7 schematically represents the NBEAD = 3 case for which two additional line beads are automatically generated. The forces specified by the load curve, LCIDRF, is evenly distributed over the three beads. In Figure 11-8, bead forces are recovered from the ASCII rcforc files for both cases of NBEAD = 1 and 3, indicating the total force applied (shown on the left) on one single bead is distributed evenly among the three automatically generated beads for the case of NBEAD = 3.

   The stress distribution is also more realistic with multiple beads. In a channel draw (half model) as shown in Figure 11-12, no significant changes in mean stress values are found between NBEAD = 3 and one single line bead. In fact, the compressive stresses are more realistically and evenly distributed around the bead region, with stresses in NBEAD = 3 about 1/3 of those in a single line bead.

2. **Lower binder gap correction.** As originally implemented, the draw bead contact model applies the draw bead forces, as specified in the load curve, when the upper binder reaches the blank, regardless of the lower binder's position. If the lower binder is not in contact with the blank, LS-DYNA still applies draw bead forces, even though it is unphysical to do so. The EFFHGT, POINT1, and POINT2 fields together provide a simple model to avoid these unphysical forces. The POINT1 and POINT2 fields are taken as nodes on the opposing binders. The draw bead contact is disabled when the Euclidean distance between POINT1 and POINT2 is greater than EFFHGT; consequently, the two nodes *must* be chosen so they converge to a single point as the draw bead closes.

   As shown in Figure 11-9, a simple model was built to verify the effectiveness of the variable EFFHGT. The upper binder is pushed down to close with the lower binder while a strip of sheet blank is being pulled in the direction indicated. The distance between the binders is 12 mm initially, as shown in Figure 11-10. The

closing gap and pulling force in the x-direction were recovered throughout the simulation. With EFFHGT set to 8 mm, the pulling force history indicates the bead forces starting to take effect after the upper binder has traveled for 4 mm as expected (see Figure 11-11).

Weakening Effect:

Without BENDING, the contact algorithm ignores the bending/unbending effect and the huge normal stress when the blank goes through the draw beads. As a result, material stretching during forming simulations cannot be predicted. The algorithm invoked with BENDING weakens the blank as it travels through the bead and stretches it when it passes through the bead.

Example:

The following partial keyword example shows how to use NBEAD with the option INITIALIZE. A total of 3 beads will be generated, including the original bead. The total bead width between the inner and outer most beads is 10.0 mm (WIDTH). The effective bead height EFFHGT is 1.11 mm, which is the gap between the upper die and the lower binder upon closing. Note the feature NBEAD only works with draw beads that are defined using a node set (#1001, which is used as SURFB in *CONTACT…), and the node set is constrained together with the lower binder (PID 8).

```
*CONTACT_DRAWBEAD_INITIALIZE_ID
$#     cid                                                                title
      1001      DRAW DBEAD #1
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
$     Draw bead node set ID is 1001, blank PID is 11.
      1001        11         4         3         0      1001         0         0
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
     0.000     0.000     0.000     0.000 20.000000         0       0.0 1.000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
  0.200000  0.200000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
$#  lcidrf    lcidnf     dbdth     dfscl    numint     dbpid     eloff     NBEAD
     90905     90906    &dbdth  0.500000         0         0         0         3
$    LCEPS    TSCALE    LCEPS2    OFFSET
       212      0.95
$   POINT1    POINT2     WIDTH    EFFHGT
     49668     50595      10.0      1.11
*SET_NODE_LIST
$#     sid       da1       da2       da3       da4
      1001
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
     50820     50821     50822     50823     50824     50825     50826     50827
*CONSTRAINED_EXTRA_NODES_SET
$       PID      NSID
          8      1001
*DEFINE_CURVE
212
-1.0,0.05
1.0,0.05
```

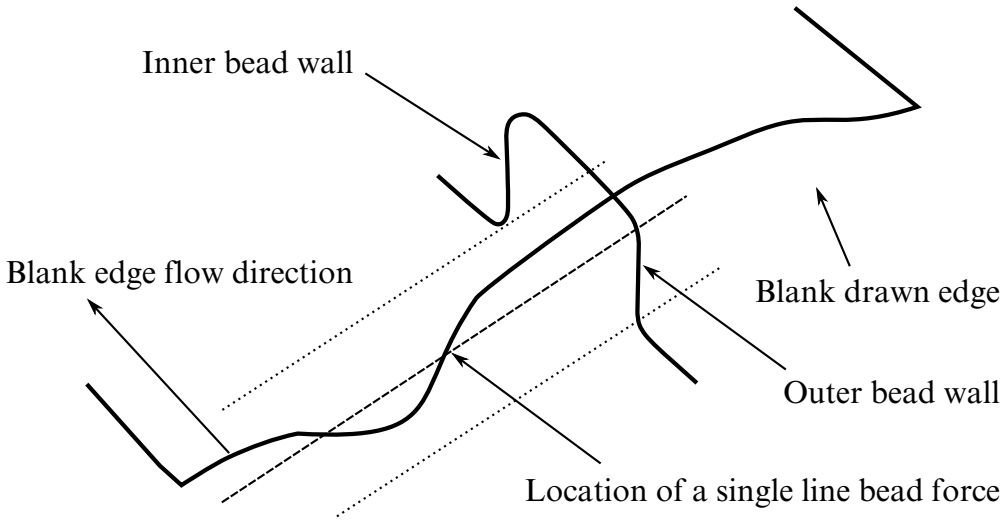


Figure 11-6. A possible scenario of sheet blank edge draw-in condition

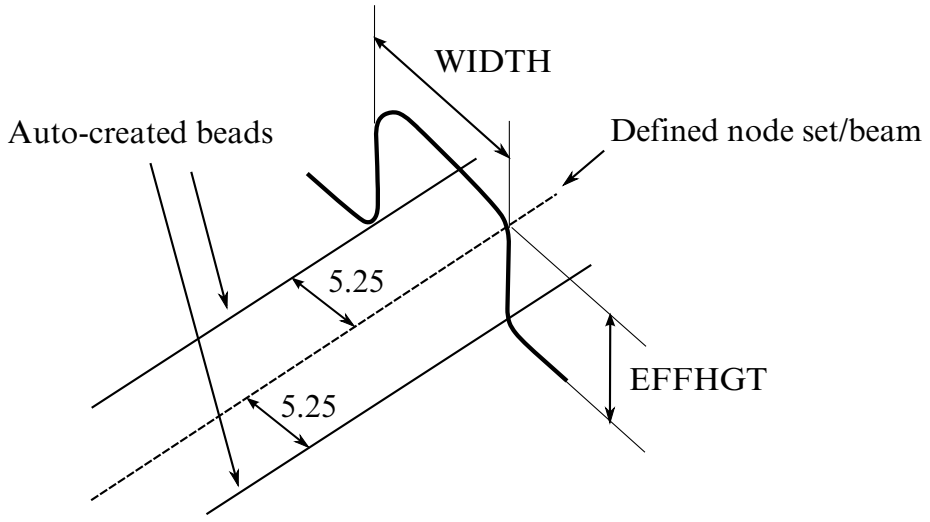


Figure 11-7. Definition of multiple draw beads

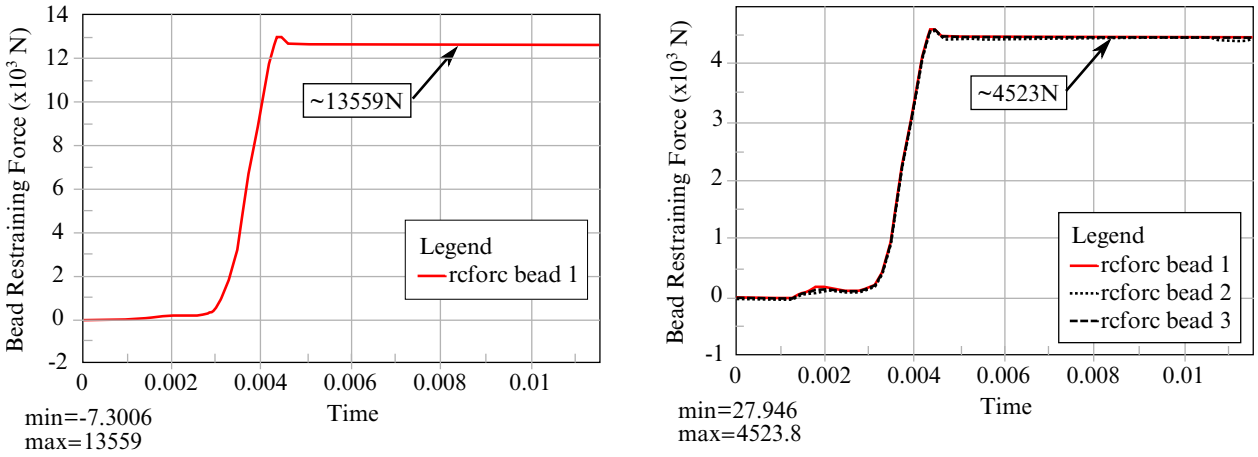


Figure 11-8. Bead force verification between NBEAD = 1 (left) and 3

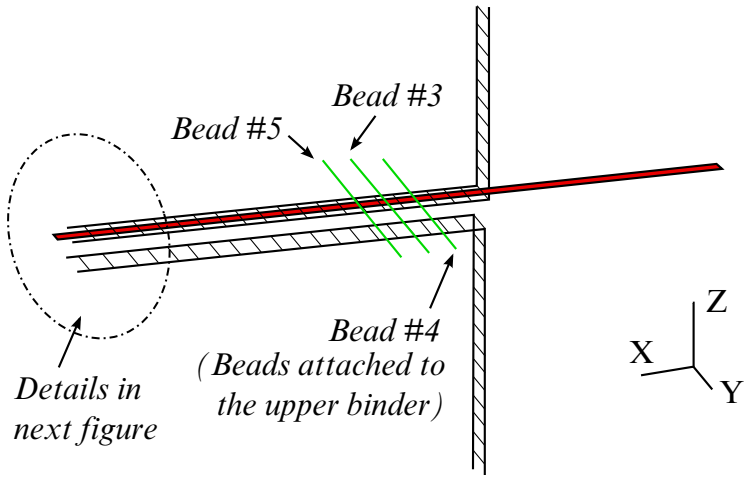


Figure 11-9. A verification model for the variable EFFHGT

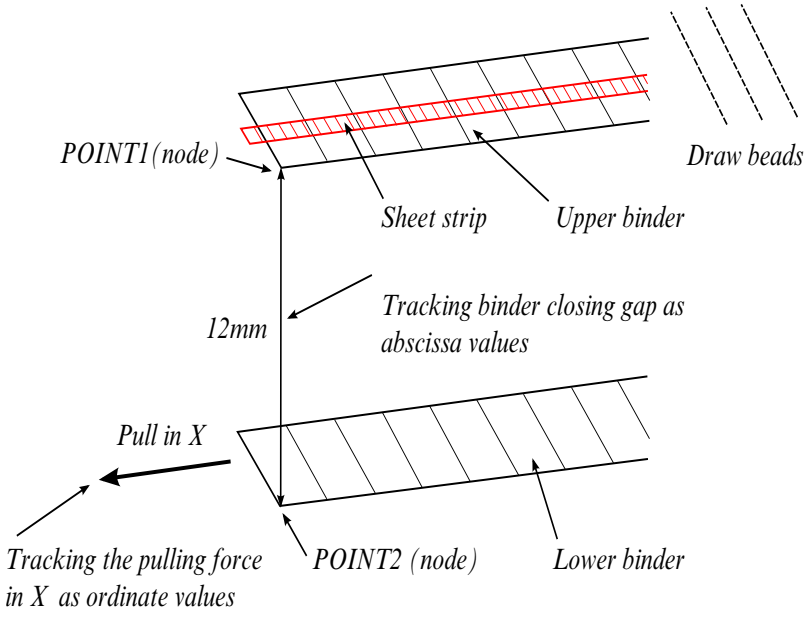


Figure 11-10. Tracking the closing gap and pulling distance

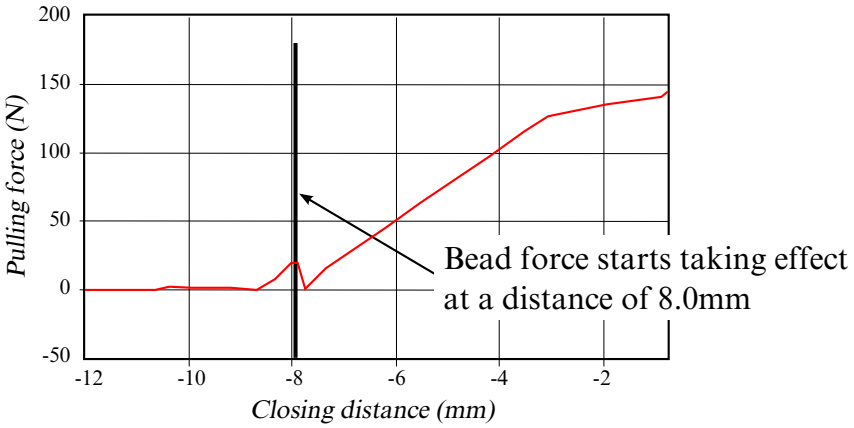
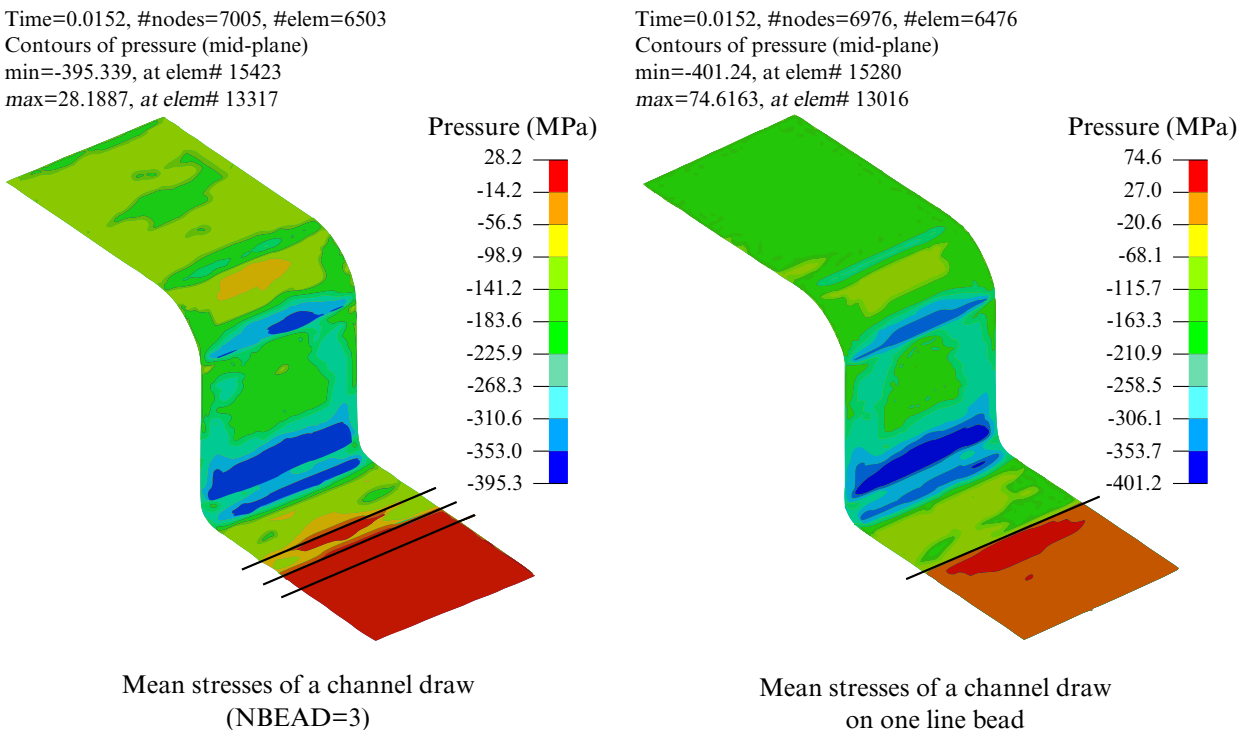


Figure 11-11. Pulling force (NODFOR) as a function of closure distance



Mean stresses of a channel draw (NBEAD=3)

Mean stresses of a channel draw on one line bead

Figure 11-12. Mean stress comparison between NBEAD = 3 and 1

Card 4: ERODING_..._SURFACE

ERODING contacts update the active contact segments to account for erosion (deletion) of elements on the contact surface. Segments that could potentially become exposed to contact during the simulation are determined during initialization, and that number of segments is reported in the contact description in d3hsp. However, at any given point in time during the simulation, only the subset consisting of currently exposed (external) segments is active in the contact evaluation.

This Card 4 is mandatory for:

*CONTACT_ERODING_NODES_TO_SURFACE

*CONTACT_ERODING_SINGLE_SURFACE

*CONTACT_ERODING_SURFACE_TO_SURFACE

Card 4	1	2	3	4	5	6	7	8
Variable	ISYM	EROSOP	IADJ					
Type	I	I	I					
Default	0	1	↓					

VARIABLE	DESCRIPTION
ISYM	Symmetry plane option: EQ.0: Off (default) EQ.1: Do not include faces with normal boundary constraints (such as brick element segments on a symmetry plane). This option is important for retaining the correct boundary conditions in the model with symmetry.
EROSOP	Erosion/interior node option (EROSOP is hardcoded to 1 for both SMP and MPP): EQ.0: Only exterior boundary information is saved. (This option is no longer supported.) EQ.1: Storage is allocated so that eroding contact can occur.
IADJ	Adjacent material treatment for solid elements (in the case of MPP, IADJ is hardcoded to 1):

VARIABLE	DESCRIPTION
	EQ.0: Solid element faces are included only for free boundaries (default for SMP).
	EQ.1: Solid element faces are included if they are on the boundary of the material subset. This option also allows for erosion within a body and the subsequent treatment of contact.

Remarks:

1. **Time step.** Eroding contact may control the time step (see ECDT in *CONTROL_CONTACT).

2. **SURFA type.** For ERODING_NODES_TO_SURFACE, define the SURFA side using a node set, not a part ID or part set ID.

3. **Negative volume failure criterion.** ERODING contact automatically invokes a negative volume failure criterion for all solid elements in the model, except as overridden by PSFAIL in *CONTROL_SOLID. PSFAIL will limit the negative volume failure criterion to a set of solid parts. A negative volume failure criterion circumvents an error termination due to negative volume by deleting solid elements that develop negative volume.

4. **Contact friction.** SMP LS-DYNA does not consider contact friction for *CONTACT_ERODING_NODES_TO_SURFACE and *CONTACT_ERODING_SURFACE_TO_SURFACE unless SOFT is set to 2 on Optional Card A. MPP LS-DYNA has no such exclusion for contact friction.

5. **SPH contact thickness.** The contact thickness (see SAST on Card 3 and ITHK = 1 on *CONTROL_SPH) of SPH nodes is not considered for *CONTACT_ERODING_NODES_TO_SURFACE in SMP. The thickness of the SPH node is set to zero for this type of contact. The thickness is considered for MPP.

Card 4: SURFACE_INTERFERENCE

This Card 4 is mandatory for:

*CONTACT_NODES_TO_SURFACE_INTERFERENCE

*CONTACT_ONE_WAY_SURFACE_TO_SURFACE_INTERFERENCE

*CONTACT_SURFACE_TO_SURFACE_INTERFERENCE

Purpose: This contact option provides a means of modeling parts that are shrink-fitted together and are, therefore, prestressed in the initial configuration. This option turns off the nodal interpenetration checks (which changes the geometry by moving the nodes to eliminate the interpenetration) at the start of the simulation and allows the contact forces to develop to remove the interpenetrations. The load curves defined in this section scale the interface stiffness constants such that the stiffness can increase slowly from zero to a final value with the effect that the interface forces also increase gradually to remove the overlaps.

Card 4	1	2	3	4	5	6	7	8
Variable	LCID1	LCID2						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
LCID1	Load curve ID for the curve scaling the interface stiffness during dynamic relaxation. This curve must originate at $(0,0)$ at time = 0 and gradually increase.
LCID2	Load curve ID for the curve scaling the interface stiffness during the transient calculation. This curve generally has a constant value of unity for the duration of the calculation if LCID1 is defined. If LCID1 = 0, this curve must originate at $(0,0)$ at time = 0 and gradually increase to a constant value.

Remarks:

1. **Shell thickness offsets.** In the case of MPP LS-DYNA, consideration of shell thickness offsets is controlled as in an "old" contact type, using SHLTHK, either

on *CONTROL_CONTACT or on Optional Card B of *CONTACT_..._INTERFERENCE.

In the case of SMP, shell thickness offsets are always considered with the lone exception being when THKOPT = 2 on Optional Card B of *CONTACT_..._INTERFERENCE.

2. **Penetrations and orientations.** The check to fix initial penetrations is skipped. Automatic orientation of shell elements is also skipped. Furthermore, segment orientation for shell elements and interpenetration checks *are skipped*. Therefore, it is necessary in the problem setup to ensure that all contact segments belonging to shell elements are properly oriented, meaning the outward normal vector of the segment, based on the right-hand rule relative to the segment numbering, must point to the opposing contact surface. Consequently, automatic contact generation should be avoided for parts composed of shell elements unless automatic generation is used on the SURFA side of a nodes-to-surface interface.

3. **Resolving an interference during a transient phase preceded by a dynamic relaxation phase.** If LCID1 = 0 and LS-DYNA performs dynamic relaxation (such as for initializing the stresses in other components), the INTEFERENCE contact treats initial penetrations as specified by the IGNORE parameters on *CONTACT and *CONTROL_CONTACT. To make the INTERFERENCE contact inactive during a dynamic relaxation phase and resolve the penetrations during the subsequent transient phase based on the curve LCID2, use either of the two following approaches:

   a) Set the contact birth time BT equal to a negative value (BT < 0) that is in magnitude anything larger than the expected completion time of the dynamic relaxation phase. After completing dynamic relaxation, LS-DYNA activates the contact regardless of the value of BT.

   b) Specify curve LCID1 with zero ordinates (meaning a zero contact stiffness) during the dynamic relaxation phase.

   For both approaches, originate curve LCID2 at $(0,0)$ at time = 0 and gradually increase the scale factor to a constant value.

Card 4: RIGID_TO_RIGID

This Card 4 is mandatory for:

*CONTACT_RIGID_NODES_TO_RIGID_BODY

*CONTACT_RIGID_BODY_ONE_WAY_TO_RIGID_BODY

*CONTACT_RIGID_BODY_TWO_WAY_TO_RIGID_BODY

Card 4	1	2	3	4	5	6	7	8
Variable	LCID	FCM	US		LCDC	DSF	UNLCID	
Type	I	I	F		I	F	I	
Default	required	required	LCID		optional	0.0	optional	

VARIABLE	DESCRIPTION
LCID	Load curve ID giving force as a function of penetration behavior for RIGID contact. See FCM below.
FCM	Force calculation method for RIGID contact: EQ.1: Load curve gives the total normal force on the surface as a function of maximum penetration of any node (RIGID_BODY_ONE_WAY only). EQ.2: Load curve gives the normal force on each node as a function of the penetration of a node through the surface (all RIGID contact types). EQ.3: Load curve gives normal pressure as a function of penetration of a node through the surface (RIGID_BODY_TWO_WAY and RIGID_BODY_ONE_WAY only). EQ.4: Load curve gives total normal force as a function of maximum soft penetration (RIGID_BODY_ONE_WAY only). In this case, the force will be followed based on the original penetration point.
US	Unloading stiffness for RIGID contact. The default is to unload along the loading curve. This should be equal to or greater than the maximum slope used in the loading curve.

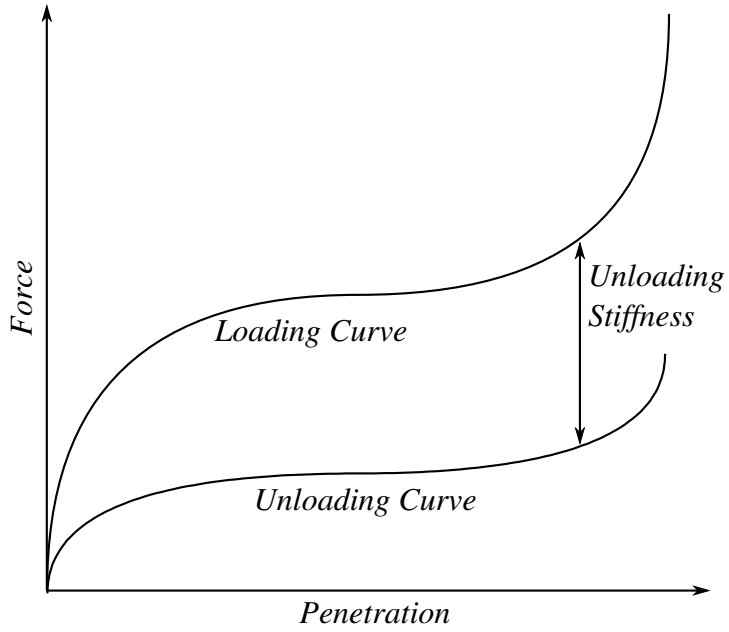


Figure 11-13. Behavior if an unloading curve is defined

VARIABLE	DESCRIPTION
LCDC	Load curve ID giving damping coefficient (DC) as a function of penetration velocity. The damping force FD is then: FD = DSF × DC × velocity.
DSF	Damping scaling factor
UNLCID	Optional load curve ID giving force as a function of penetration behavior for RIGID_BODY_ONE_WAY contact during unloading. This option requires the definition of the unloading stiffness, US. See Figure 11-13.

Card 4: TIEBREAK_NODES

This Card 4 is mandatory for:

*CONTACT_TIEBREAK_NODES_TO_SURFACE

*CONTACT_TIEBREAK_NODES_ONLY

Card 4	1	2	3	4	5	6	7	8
Variable	NFLF	SFLF	NEN	MES				
Type	F	F	F	F				
Default	required	required	2.	2.				

VARIABLE	DESCRIPTION
NFLF	Normal failure force. Only tensile failure, meaning tensile normal forces, will be considered in the failure criterion.
SFLF	Shear failure force
NEN	Exponent for normal force in the failure criterion
MES	Exponent for shear force in the failure criterion

Remarks:

1. **Node by node failure attributes.** These fields can be overridden node by node with the attributes on the *SET_NODE data cards.

2. **Failure criterion.** The tiebreak failure criterion is given as:

$$\left(\frac{|f_n|}{\text{NFLF}}\right)^{\text{NEN}} + \left(\frac{|f_s|}{\text{SFLF}}\right)^{\text{MES}} \geq 1 \ .$$

   Failure is assumed if the left side is larger than 1. f_n and f_s are the normal and shear interface forces. Both NFLF and SFLF must be defined. If failure in only tension or shear is required, then set the other failure force to a large value (10^{10}).

3. **Contact behavior after failure.** After failure, *CONTACT_TIEBREAK_-NODES_TO_SURFACE behaves as a nodes-to-surface contact with no thickness offsets (no interface tension possible) whereas the *CONTACT_TIEBREAK_-

NODES_ONLY stops acting as a contact. Prior to failure, the two contact types behave identically.

Card 4: TIEBREAK_SURFACE

This Card 4 is mandatory for:

*CONTACT_TIEBREAK_SURFACE_TO_SURFACE

*CONTACT_TIEBREAK_SURFACE_TO_SURFACE_ONLY

Card 4	1	2	3	4	5	6	7	8
Variable	NFLS	SFLS	TBLCID	THKOFF				
Type	F	F	I	I				
Default	required	required	0	0				

VARIABLE	DESCRIPTION
NFLS	Tensile failure stress
SFLS	Shear failure stress
TBLCID	Optional load curve ID defining the resisting tensile stress as a function of gap opening in the normal direction for the post-failure response. This option applies only to SMP and can be used to model adhesives.
THKOFF	Thickness offsets are considered if THKOFF = 1. If shell offsets are included in the meshed geometry, this option is highly recommended since segment orientations can be arbitrary and the contact surfaces can be disjoint. This option is *not* available in the MPP version of LS-DYNA. It works by substituting with *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK (OPTION = 2 if TBLCID is not specified; OPTION = 5 if TBLCID is specified).

Remarks:

1. **Segment-by-segment failure attributes.** The failure fields, NFLS and SFLS, can be overridden segment by segment on the *SET_SEGMENT or *SET_SHELL data cards for the SURFA surface with A1 and A2, respectively. These variables do not apply to the SURFB surface.

2. **Failure criterion.** The tiebreak contact fails when:

$$\left(\frac{|\sigma_n|}{\text{NFLS}}\right)^2 + \left(\frac{|\sigma_s|}{\text{SFLS}}\right)^2 \geq 1 \ .$$

Both NFLS and SFLS must be defined. If failure in only tension or shear is required, then set the other failure stress to a large value (10^{10}). When used with shells, contact segment normal vectors are used to establish the tension direction (as opposed to compression). Compressive stress does not contribute to the failure equation.

3. **Contact behavior after failure.** After failure, *CONTACT_TIEBREAK_SURFACE_TO_SURFACE behaves as a surface-to-surface contact with no thickness offsets while *CONTACT_TIEBREAK_SURFACE_TO_SURFACE_ONLY stops acting as a contact altogether. Until failure, it ties the SURFA nodes to the SURFB segments.

4. **Contact stiffness**. Contact stiffness is based on the SURFB side. The variables PENOPT and SOFT have no effect on contact stiffness.

Card 4: CONTRACTION_JOINT

This Card 4 is mandatory for:

*CONTACT_SURFACE_TO_SURFACE_CONTRACTION_JOINT

Purpose: This contact option turns on the contraction joint model designed to simulate the effects of sinusoidal joint surfaces (shear keys) in the contraction joints of arch dams and other concrete structures. The sinusoidal functions for the shear keys are defined according to the following three methods [Solberg and Noble 2002]:

a) *Method 1.*

$$\hat{g} = g - A\{1 - \cos[B(s_2 - s_1)]\}$$

b) *Method 2.*

$$\hat{g} = g - 2A\left|\sin\left[\frac{B(s_2 - s_1)}{2}\right]\right|$$

c) *Method 3 (default).*

$$\hat{g} = g - A\cos(Bs_2) + A\cos(Bs_1)$$

Here g is a gap function for the contact surface and $\hat{g}$ is a gap function for the joint surface. A is the key amplitude parameter, and B is the key frequency parameter. s_1 and s_2 are referential surfaces:

$$s_1 = \mathbf{X}_{\text{surface1}} \cdot \mathbf{T}_{\text{key}}$$
$$s_2 = \mathbf{X}_{\text{surface2}} \cdot \mathbf{T}_{\text{key}}$$
$$\mathbf{T}_{\text{key}} = \mathbf{T}_{\text{slide}} \times \mathbf{n}$$

$\mathbf{T}_{\text{slide}}$ is the free-sliding direction of the keys, and $\mathbf{n}$ is the surface normal in reference.

Card 4	1	2	3	4	5	6	7	8
Variable	MTCJ	ALPHA	BETA	TSVX	TSVY	TSVZ		
Type	I	F	F	F	F	F		
Default	0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
MTCJ	The method option for the gap function, $\hat{g}$
ALPHA	Key amplitude parameter, A

VARIABLE	DESCRIPTION
BETA	Key frequency parameter, B
TSVX	x component of the free sliding direction $\mathbf{T}_{\text{slide}}$
TSVY	y component of the free sliding direction $\mathbf{T}_{\text{slide}}$
TSVZ	z component of the free sliding direction $\mathbf{T}_{\text{slide}}$

THERMAL:

THRM 1 is mandatory for the THERMAL (and THERMAL_FRICTION) option, meaning:

*CONTACT_..._THERMAL_...

NOTE: If Card 4 is required, then it must go before THRM 1. (Card 4 is required for certain contact types.)

Thermal Card 1.

THRM 1	1	2	3	4	5	6	7	8
Variable	K	FRAD	H0	LMIN	LMAX	FTOSA	BC_FLG	ALGO
Type	F	F	F	F	F	F	I	I
Default	none	none	none	none	none	0.5	0	0

VARIABLE **DESCRIPTION**

K — Thermal conductivity of fluid between the contact surfaces. If a gap with a thickness l_{gap} exists between the contact surfaces, then the conductance due to thermal conductivity between the contact surfaces is

$$h_{cond} = \frac{K}{l_{gap}} .$$

Note that LS-DYNA calculates l_{gap} based on deformation (see Remark 2).

FRAD — Radiation factor, f_{rad}, between the contact surfaces:

$$f_{rad} = \frac{\sigma}{\frac{1}{\varepsilon_1} + \frac{1}{\varepsilon_2} - 1} ,$$

where

σ = Stefan-Boltman constant
ε_1 = emissivity of SURFA surface
ε_2 = emissivity of SURFB surface

LS-DYNA calculates the radiant heat transfer conductance as

$$h_{rad} = f_{rad}(T_{SURFA} + T_{SURFB})\left(T_{SURFA}^2 + T_{SURFB}^2\right) .$$

VARIABLE	DESCRIPTION
H0	Heat transfer conductance for closed gaps. Use this heat transfer conductance for gaps in the range $$0 \le l_{gap} \le l_{min} \ .$$
LMIN	Minimum gap, l_{min}. The heat transfer conductance defined above (H0) is used for gap thicknesses less than this value. If $l_{min} < 0$, then $-l_{min}$ is a load curve number defining l_{min} as a function time.
LMAX	No thermal contact if the gap is greater than this value (l_{max})
FTOSA	Fraction, f, of sliding friction energy partitioned to the SURFA surface (see Remark 3). Energy partitioned to the SURFB surface is $(1-f)$. EQ.0: Set to 0.5 (default). The sliding friction energy is partitioned 50% - 50% to the SURFA and SURFB surfaces in contact.
BC_FLAG	Thermal boundary condition flag: EQ.0: Thermal boundary conditions are on when parts are in contact. EQ.1: Thermal boundary conditions are off when parts are in contact.
ALGO	Thermal contact algorithm type: EQ.0: Two-way contact. Both surfaces change temperature due to contact EQ.1: One-way contact. The surface of SURFB does not change temperature due to contact. The surface of SURFA does change temperature. EQ.2: Two-way edge contact EQ.3: One-way edge contact (no heat transfer into the reference side)

NOTE: For ALGO = 2 and 3, shell edges must be on the SURFA side, and solid facets and shell surface need to be on the SURFB side.

Remarks:

1. **Contact types.** Thermal contact involves heat transfer across surfaces, so the contact definition must be of a SURFACE_TO_SURFACE or (as of R17) an AUTOMATIC_SINGLE_SURFACE type. Thus, in general, NODES_TO_SURFACE contacts cannot be used, and an error termination will result. The exception is TIED_NODES_TO_SURFACE and TIED_SHELL_EDGE_TO_SURFACE contacts with the SURFA side *not* defined with node sets. The TIED_NODES_TO_SURFACE and TIED_SHELL_EDGE_TO_SURFACE contacts also allow thermal contact with beam elements. These contacts can model, for instance, heat transfer through spot weld beams used for tying opposite surfaces. Thermal contact with beam elements is otherwise only supported for Mortar contacts by inheriting the Mortar algorithms from the mechanical side of the contact.

2. **Heat conductance.** The heat conductance is calculated as

$$h = \begin{cases} h_0 & 0 \le l_{\text{gap}} \le l_{\text{min}} \\ h_{\text{cond}} + h_{\text{rad}} & l_{\text{min}} < l_{\text{gap}} \le l_{\text{max}} \\ 0 & l_{\text{gap}} > l_{\text{max}} \end{cases}$$

   LS- DYNA calculates l_{gap} based on deformation.

3. **Sliding friction energy fraction.** f can be calculated using:

$$f = \frac{1}{1 + \dfrac{\sqrt{(\rho C_p k)_{\text{SURFB side material}}}}{\sqrt{(\rho C_p k)_{\text{SURFA side material}}}}}$$

THERMAL_FRICTION:

THRM 2 is required after THRM 1 if the FRICTION suffix is added to THERMAL.

*CONTACT_…_THERMAL_FRICTION_…

The blank (or workpiece) must be defined as SURFA in a metal forming model.

Purpose:

1. Define the mechanical static and dynamic friction coefficients as a function of temperature.
2. Define the thermal contact conductance as a function of temperature and pressure.

THERMAL Card 2.

THRM 2	1	2	3	4	5	6	7	8
Variable	LCFST	LCFDT	FORMULA	A	B	C	D	LCH
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

THERMAL Card 3 - User Subroutine Cards. Additional cards for FORMULA < 0. Use as many cards as needed to set |FORMULA| number of parameters.

THRM 2.1	1	2	3	4	5	6	7	8
Variable	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
LCFST	Load curve ID for static coefficient of friction as a function of temperature. The load curve value multiplies the coefficient value FS. For Mortar contact, the static coefficient of friction can be given as a load curve or table. In the case of a load curve, it is a function of

VARIABLE	DESCRIPTION
	the mean temperature, $(T_{SURFA} + T_{SURFB})/2$, in the contact interface. In the case of a table, the static coefficient of friction is an arbitrary function of T_{SURFA} and T_{SURFB}; each value of T_{SURFB} should be associated with a curve having T_{SURFA} as the abscissa.
LCFDT	Load curve ID for dynamic coefficient of friction as a function of temperature. The load curve value multiplies the coefficient value FD. For Mortar contact, the dynamic coefficient of friction can be given as a load curve or table. In the case of a load curve, it is a function of the mean temperature, $(T_{SURFA} + T_{SURFB})/2$, in the contact interface. In the case of a table, the dynamic coefficient of friction is an arbitrary function of T_{SURFA} and T_{SURFB}; each value of T_{SURFB} should be associated with a curve having T_{SURFA} as the abscissa.
FORMULA	Formula that defines the contact heat conductance as a function of temperature and pressure.

EQ.1: $h(P)$ is defined by load curve A, which contains data for contact conductance as a function of pressure.

EQ.2: $h(P)$ is given by the following where A, B, C and D although defined by load curves are typically constants for use in this formula. The load curves are functions of temperature.

$$h(P) = a + bP + cP^2 + dP^3$$

EQ.3: $h(P)$ is given by the following formula from [Shvets and Dyban 1964]:

$$h(P) = \frac{\pi k_{gas}}{4\lambda}\left[1. + 85\left(\frac{P}{\sigma}\right)^{0.8}\right] = \frac{a}{b}\left[1. + 85\left(\frac{P}{c}\right)^{0.8}\right]$$

where,

a: is evaluated from the load curve, A, for the thermal conductivity, k_{gas}, of the gas in the gap as a function of temperature.

b: is evaluated from the load curve, B, for the parameter grouping $\pi/4\lambda$. Therefore, this load curve should be set to a constant value. λ is the surface roughness.

c: is evaluated from the load curve, C, which specifies a stress metric for deformation (such as yield) as a function of temperature.

EQ.4: $h(P)$ is given by the following formula from [Li and

VARIABLE	DESCRIPTION

Sellars 1996]:

$$h(P) = a\left[1 - \exp\left(-b\frac{P}{c}\right)\right]^d$$

where,

a: is evaluated from the load curve, A, which defines a load curve as a function of temperature.

b: is evaluated from the load curve, B, which defines a load curve as a function of temperature.

c: is evaluated from the load curve, C, which defines a stress metric for deformation (such as yield) as a function of temperature.

d: is evaluated from the load curve D, which is a function of temperature.

EQ.5: $h(\text{gap})$ is defined by load curve A, which contains data for contact conductance as a function of the interface gap.

LT.0: This is equivalent to defining the keyword *USER_INTERFACE_CONDUCTIVITY. The user subroutine `usrhcon` will be called for this contact interface to define the contact heat transfer coefficient.

A — Load curve ID for the *a* coefficient used in the formula

B — Load curve ID for the *b* coefficient used in the formula

C — Load curve ID for the *c* coefficient used in the formula

D — Load curve ID for the *d* coefficient used in the formula

LCH — Load curve ID for *h*. If defined, this input takes precedence over any other definitions. This parameter can refer to a curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION). When LCH is a curve ID (and a function ID) it is interpreted as follows:

GT.0: The heat transfer coefficient is defined as a function of time, t, by a curve consisting of $(t, h(t))$ data pairs.

LT.0: The heat transfer coefficient is defined as a function of temperature, T, by a curve consisting of $(T, h(T))$ data pairs.

When the reference is to a function, it is prototyped as $h =$

VARIABLE	DESCRIPTION
	$h(t, T_{avg}, T_{SURFA}, T_{SURFB}, P, g)$ where: t = solution time T_{avg} = average interface temperature T_{SURFA} = SURFA segment temperature T_{SURFB} = SURFB segment temperature P = interface pressure g = gap distance between SURFA and SURFB segment
UC*i*	User parameters

ORTHO_FRICTION:

Additional cards for the ORTHO_FRICTION option:

*CONTACT_…_ORTHO_FRICTION_…

ORTHO_FRIC 1.

ORFR 1	1	2	3	4	5	6	7	8
Variable	FS1_SA	FD1_SA	DC1_SA	VC1_SA	LC1_SA	OACS_SA	LCFSA	LCPSA
Type	F	F	F	F	I	I	I	I
Default	0.	0.	0.	0.	0	0	0	0

ORTHO_FRIC 2.

ORFR 2	1	2	3	4	5	6	7	8
Variable	FS2_SA	FD2_SA	DC2_SA	VC2_SA	LC2_SA			
Type	F	F	F	F	I			
Default	0.	0.	0.	0.	0			

ORTHO_FRIC 3.

ORFR 3	1	2	3	4	5	6	7	8
Variable	FS1_SB	FD1_SB	DC1_SB	VC1_SB	LC1_SB	OACS_SB	LCFSB	LCPSB
Type	F	F	F	F	I	I	I	I
Default	0.	0.	0.	0.	0	0	0	0

ORTHO_FRIC 4.

ORFR 4	1	2	3	4	5	6	7	8
Variable	FS2_SB	FD2_SB	DC2_SB	VC2_SB	LC2_SB			
Type	F	F	F	F	I			
Default	0.	0.	0.	0.	0			

VARIABLE	DESCRIPTION
FS*n*_SA or SB	Static coefficient of friction in the local n orthotropic direction for the SURFA (SA) or SURFB (SB) surface. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact, $\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}$, where the direction and surface are left off for clarity. When contacts with ORTHO_FRICTION are defined by segment sets, each segment allows the specification of an offset angle in degrees from the 1-2 side of the segment which locates the 1-direction for the friction. The offset angle is input as the first attribute of the segment in *SET_SEGMENT. The transverse direction, or 2-direction, is in the plane of the segment and is perpendicular to the 1-direction. The offset angle is taken as zero when part (set) IDs are used in the contact definition. See *DEFINE_FRICTION_ORIENTATION for remarks regarding the use of *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ORTHO_FRICTION.
FDI_SA or SB	Dynamic coefficient of friction in the local n orthotropic direction
DC*n*_SA or SB	Exponential decay coefficient for the local n direction
VC*n*_SA or SB	Coefficient for viscous friction in the local n direction. See the description for VC for mandatory Card 2 above.
LC*n*_SA or SB	Table ID of a two- or three-dimensional table (see *DEFINE_TABLE, *DEFINE_TABLE_2D, or *DEFINE_TABLE_3D) giving the friction coefficient in the local n direction as a function of the relative velocity, interface pressure, and optionally the temperature. In the 2D case, each curve in the 2D table definition defines the

VARIABLE	DESCRIPTION
	coefficient of friction as a function of the interface pressure corresponding to a particular value of the relative velocity. In the 3D case, each table in the 3D table definition corresponds to a given temperature, and that table gives the friction coefficient as a function of the relative velocity and interface pressure. Note that 3D tables are only available for Mortar contact.
OACS_SA or SB	If the default value, 0, is active, the frictional forces acting on a node sliding on a segment are based on the local directions of the segment. If OACS is set to unity, 1, the frictional forces acting on a node sliding on a segment are based on the local directions of the sliding node. No matter what the setting for OACS, the_SA coefficients are always used for SURFA nodes and the_SB coefficients for SURFB nodes.
LCFSA or SB	Optional load curve that gives the coefficient of friction as a function of the direction of relative motion, as measured in degrees from the first orthotropic direction. If this load curve is specified, the other parameters (FS, FD, DC, VC, LC) are ignored. This is currently only supported in the MPP version.
LCPSA or SB	Optional load curve that gives a scale factor for the friction coefficient as a function of interface pressure. This is only used if LCFSA (or SB) is defined.

Remarks:

1. **Orthotropic friction for mortar contact.** For Mortar contact, we consider what is going on in the plane of surface A with normal $\mathbf{n}$, and refer to Figure 11-14. The orthotropic friction law is based on an elliptic yield surface:

$$p \geq \sqrt{\left(\frac{t_{\parallel}}{\mu_{\parallel}}\right)^2 + \left(\frac{t_{\perp}}{\mu_{\perp}}\right)^2}$$

for the Coulomb friction law. This surface is a cone in the pressure-traction space as shown in Figure 11-14. We also mandate that the slip direction is associative

$$\mathbf{v} \sim \frac{\partial p}{\partial \mathbf{t}} \ ,$$

resulting in an angle δ between the slip and traction. Here p is the contact pressure, $\mathbf{t} = t_{\parallel}\mathbf{a}_{\parallel} + t_{\perp}\mathbf{a}_{\perp}$ is the traction vector and $\boldsymbol{\mu} = \{\mu_{\parallel}, \mu_{\perp}\}$ are the coefficients of friction in the orthonormal system $\mathbf{A} = \{\mathbf{a}_{\parallel}, \mathbf{a}_{\perp}\}$ of surface A. Similarly, we have the orthonormal system $\mathbf{B} = \{\mathbf{b}_{\parallel}, \mathbf{b}_{\perp}\}$ on surface B which is related to $\mathbf{A}$ by

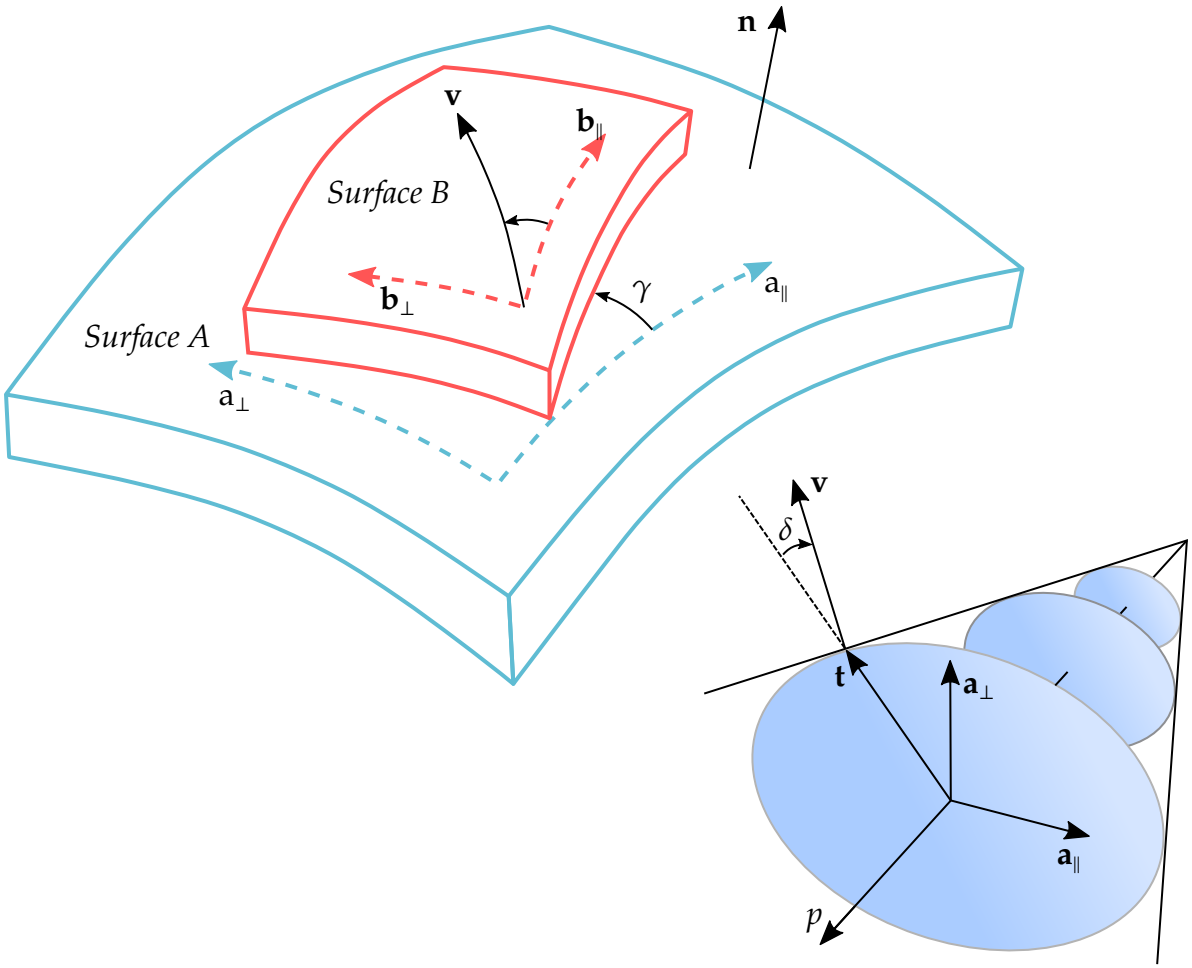


Figure 11-14. Definitions of terms for orthotropic friction in Mortar contact. Surface B with its main fibers in direction $\mathbf{b}_{\parallel}$ slides on surface A with its fibers in direction $\mathbf{a}_{\parallel}$. The contact pressure is p and the sliding velocity of surface A relative to B is in the direction $\mathbf{v}$. This results in a frictional traction, $\mathbf{t}$, on surface A directed (by and large) in the same direction as $\mathbf{v}$. The corresponding traction on surface B is in the opposite direction and of the same magnitude as $\mathbf{t}$.

rotating it an angle γ with respect to the normal $\mathbf{n}$ of surface A. First we need to mention how these systems, $\mathbf{A}$ and $\mathbf{B}$, are constructed in the first place.

We assume that the materials of the interacting surfaces are anisotropic. Then $\mathbf{a}_{\parallel}$ is the *first* material direction of the material constituting surface A and $\mathbf{b}_{\parallel}$ is likewise the corresponding first material direction on surface B. Therefore, currently the orthotropic directions for mortar contact are *not* set on the contact card but are inherited from the underlying materials. In Figure 11-14, $\mathbf{A}$ and $\mathbf{B}$ should not be confused with element specific directions, as this exposition is completely independent of finite elements.

The friction coefficients $\boldsymbol{\mu}$ are determined from the angle γ (see Figure 11-14) through direction cosines

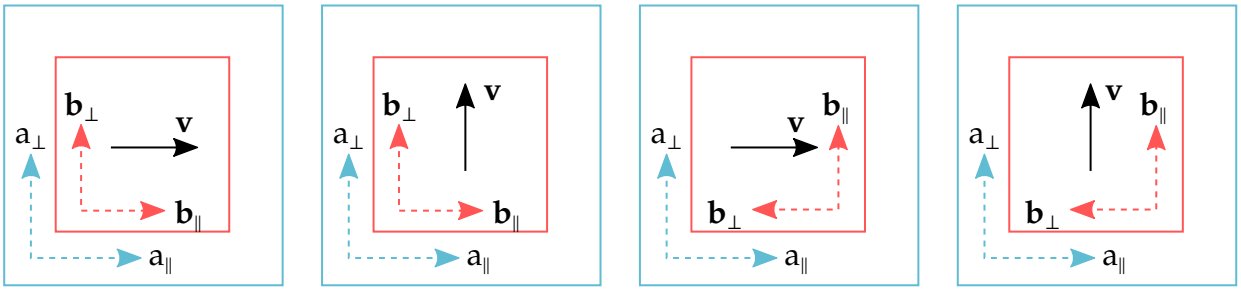


Figure 11-15. Interpretation of input parameters for Mortar orthotropic friction

$$\alpha = \cos^2\gamma = (\mathbf{a}_\parallel \cdot \mathbf{b}_\parallel)^2 = (\mathbf{a}_\perp \cdot \mathbf{b}_\perp)^2$$
$$\beta = \sin^2\gamma = (\mathbf{a}_\perp \cdot \mathbf{b}_\parallel)^2 = (\mathbf{a}_\parallel \cdot \mathbf{b}_\perp)^2$$

as

$$\mu_\parallel = \alpha\ \text{FS1_SA} + \beta\ \text{FS1_SB}$$
$$\mu_\perp = \alpha\ \text{FS2_SA} + \beta\ \text{FS2_SB}.$$

It follows that the interpretation of the input parameters that are relevant for Mortar orthotropic friction are (see Figure 11-15):

a) FS1_SA being the coefficient of friction for $\mathbf{v}$ in direction $\mathbf{a}_\parallel$ when $\mathbf{a}_\parallel$ is aligned with $\mathbf{b}_\parallel$.

b) FS1_SB being the coefficient of friction for $\mathbf{v}$ in direction $\mathbf{a}_\parallel$ when $\mathbf{a}_\parallel$ is aligned with $\mathbf{b}_\perp$.

c) FS2_SA being the coefficient of friction for $\mathbf{v}$ in direction $\mathbf{a}_\perp$ when $\mathbf{a}_\parallel$ is aligned with $\mathbf{b}_\parallel$.

d) FS2_SB being the coefficient of friction for $\mathbf{v}$ in direction $\mathbf{a}_\perp$ when $\mathbf{a}_\parallel$ is aligned with $\mathbf{b}_\perp$.

The effective friction coefficient will then be determined by the yield surface and flow rule as

$$\mu = \frac{\|\mathbf{t}\|}{p}$$

and thus, determines the relationship between contact pressure p and magnitude of sliding traction $\boldsymbol{t}$.

The behavior for relative sliding $\mathbf{v}$ in either of the main A directions $\mathbf{a}_\parallel$ and $\mathbf{a}_\perp$ is by design quite intuitive; the effective friction coefficients will be $\mu_\parallel$ and $\mu_\perp$, respectively, and the tractions $\mathbf{t}$ are collinear with the sliding. However, from the associative slip law the traction $\mathbf{t}$ and sliding $\mathbf{v}$ are not *always* pointing in the

exact same direction, and this is explained best by considering two situations. With prescribed sliding **v** in some other direction than $\mathbf{a}_{\parallel}$ or $\mathbf{a}_{\perp}$, the resulting traction vector **t** will have its main component in the direction of sliding, but also a (nonzero) component towards the direction with the *largest* friction coefficient. The corresponding tendency will be seen with a prescribed traction **t**, for which the resulting sliding direction **v** will tend towards the direction with the *smallest* friction coefficient. See Figure 11-16.

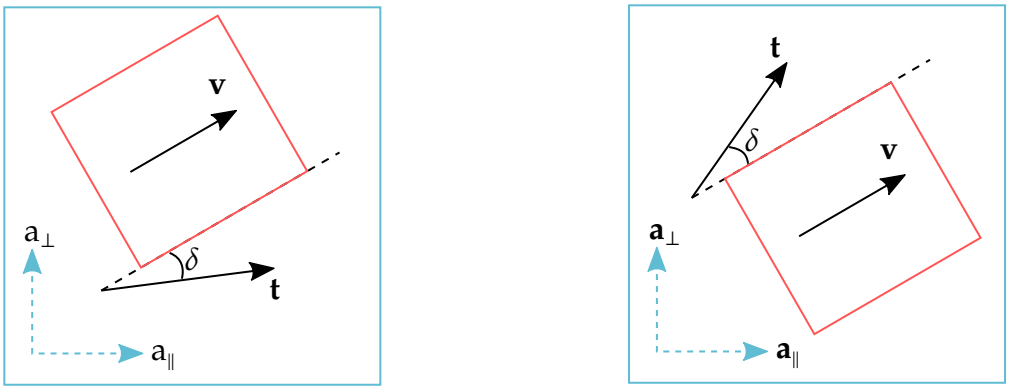


Figure 11-16. Illustration of how direction of sliding differs from the traction direction depending on the coefficients of friction

Optional Card A:

> **NOTE:** If Card 4 is required, then it must go before this card. (Card 4 is required for certain contact types.)

Optional Card A.

Card A	1	2	3	4	5	6	7	8
Variable	SOFT	SOFSCL	LCIDAB	MAXPAR	SBOPT	DEPTH	BSORT	FRCFRQ
Type	I	F	I	F	I	I	I	I
Default	0	.1	0	1.025	2	2	10-100	1
Remarks			type a13					

VARIABLE	DESCRIPTION
SOFT	Contact formulation: EQ.0: Standard penalty formulation EQ.1: Soft constraint penalty formulation. See About SOFT = 1. EQ.2: Segment-to-segment contact penalty formulation. See About SOFT = 2. EQ.4: Constraint approach for FORMING contacts. This formulation only applies to one-way forming contacts. You should use it when the penalty formulations result in large penetrations. The results, however, are sensitive to damping. EQ.6: Special contact algorithm to handle sheet blank edge (deformable) to gage pin (rigid shell) contact during implicit gravity loading. This applies to *CONTACT_FORMING_NODES_TO_SURFACE only. See remarks under About SOFT = 6.
SOFSCL	Scale factor for constraint forces of soft constraint option invoked with SOFT = 1 (default = 0.10). Values greater than 0.5 for single surface contact and 1.0 for one-way treatment are inadmissible.

VARIABLE	DESCRIPTION
LCIDAB	Load curve ID defining airbag thickness as a function of time for type a13 contact (*CONTACT_AIRBAG_SINGLE_SURFACE).
MAXPAR	Maximum parametric coordinate in segment search (values between 1.025 and 1.20 are recommended). This variable applies only to SMP; for MPP, see PARMAX on MPP 1. It does not apply to automatic contacts, except for AUTOMATIC_GENERAL. It also does not apply to segment-to-segment (SOFT = 2) and Mortar segment-to-segment contacts. Larger values can increase the cost. If zero, MAXPAR defaults to 1.025 for most contact options. The exceptions to this are listed in the table below.

Contact Option	Default
SINGLE_EDGE	1.300
SPOTWELD_BEAM_OFFSET	1.050
SPOTWELD_CONSTRAINED_OFFSET	1.006
SPOTWELD_WITH_TORSION	1.006
TIED_NODES_TO_SURFACE	1.070
TIED_NODES_..._CONSTRAINED_OFFSET	1.006
TIED_NODES_..._OFFSET	1.006
TIED_SHELL_..._BEAM_OFFSET	1.050
TIED_SHELL_..._CONSTRAINED_OFFSET	1.006
TIED_SHELL_..._OFFSET	1.006
TIED_SURFACE_TO_SURFACE(_FAILURE)	1.070
TIED_SURFACE_..._CONSTRAINED_OFFSET	1.006
TIED_SURFACE_..._OFFSET	1.006
AUTOMATIC_GENERAL(_...)	1.100

This factor allows for an increase in the size of the segments which may be useful at sharp corners. For the SPOTWELD and …_OFFSET options larger values can sometimes lead to numerical instabilities; however, a larger value is sometimes necessary to ensure that all nodes of interest are tied.

SBOPT — Segment-to-segment (SOFT = 2) contact options (see Fields used with SOFT = 2 under About SOFT = 2):

VARIABLE	DESCRIPTION
	EQ.1: Pinball edge-edge contact (not recommended) EQ.2: Assume planer segments (default) EQ.3: Warped segment checking EQ.4: Sliding option (not recommended when using DEPTH = 5, 25, and 35) EQ.5: Do options 3 and 4 (not recommended when using DEPTH = 5, 25, and 35)
DEPTH	Search depth in automatic contact to check for nodal penetration through the closest contact segments. A value of 1 (one segment) is sufficiently accurate for most crash applications and is much less expensive. By default, this value is 2 (two segments) for improved accuracy, except for *CONTACT_AUTOMATIC_GENERAL which has a default of 3. LT.0: |DEPTH| is the load curve ID defining searching depth as a function of time. (not available when SOFT = 2) See Fields used with SOFT = 2 under About SOFT = 2 for segment-to-segment contact options controlled by DEPTH.
BSORT	Number of cycles between bucket sorts. Values of 25 and 100 are recommended for contact types 4 (SINGLE_SURFACE) and 13 (AUTOMATIC_SINGLE_SURFACE), respectively. Values of 10-15 are okay for surface-to-surface and node-to-surface contact. If zero, LS-DYNA determines the interval. BSORT applies only to SMP (see BCKT on MPP 1 for MPP) except in the case of SOFT = 2 or for Mortar contact, in which case BSORT applies to both SMP and MPP. For Mortar contact the default is the value associated with NSBCS on *CONTROL_CONTACT. LT.0: |BSORT| is the load curve ID defining bucket sorting frequency as a function of time.
FRCFRQ	Number of cycles between contact force updates for penalty contact formulations. This option can provide a significant speed-up of the contact treatment. If used, values exceeding 3 or 4 are dangerous. Considerable care must be exercised when using this option, as this option assumes that contact does not change for FRCFRQ cycles. EQ.0: FRCFRQ is set to 1, and force calculations are performed each cycle (strongly recommended).

Remarks:

1. **Contact stiffness based on stability.** Setting SOFT = 1 or 2 on optional contact Card A will cause the contact stiffness to be determined based on stability considerations by taking into account the time step and nodal masses. This approach is generally more effective for contact between materials of dissimilar stiffness or dissimilar mesh densities.

2. **Switching between SOFT = 1 and SOFT = 2.** By adding "soft=2to1" on the LS-DYNA execution line, all occurrences of SOFT = 2 in a model will be changed to SOFT = 1. Conversely, by adding "soft=1to2" on the execution line, all occurrences of SOFT = 1 in a model will be changed to SOFT = 2 (other variables related to SOFT = 2, such as SBOPT and DEPTH are left unchanged).

About SOFT = 1:

The soft constraint formulation (SOFT = 1) may be helpful if the material constants of the elements that make up the surfaces in contact have a wide variation in the elastic bulk moduli because the interface stiffness is based on the nodal mass and the global time step size. This method of computing the interface stiffness will typically give a much higher stiffness value than would be obtained by using the bulk modulus. Therefore, this method is the preferred approach when soft foam materials interact with metals.

Note that with this method LS-DYNA computes two contact stiffnesses, $k_{\text{soft}=0}$ and k_{cs}. These stiffnesses are the contact stiffness calculated for SOFT = 0 and the stability contact stiffness, respectively. Generally, the maximum of these two stiffnesses is the stiffness for SOFT = 1. See the theory manuals for details about how these contact stiffnesses are calculated.

About SOFT = 2:

What is segment-to-segment contact
Traditionally, to determine contact, the nodes of one surface are checked to see if they penetrate the segments of another surface. The direction of the applied penalty force depends on the direction of the reference segment's normal vector at the contact point. Segment-to-segment contact differs from node-to-segment contact as contact forces are calculated for pairs of segments rather than a single node and a segment. It is supported by a subset of contact types (see When segment-to-segment contact is available). To calculate a penalty force and direction for each pair, the algorithm judges one segment to be the reference segment. Contact forces to oppose penetration are applied to nodes of both segments in the direction of the reference segment's normal vector. Optional frictional forces are in the plane of the reference segment. For ONE_WAY contact types, reference segments are always from SURFB. With all other types, the location and orientation of

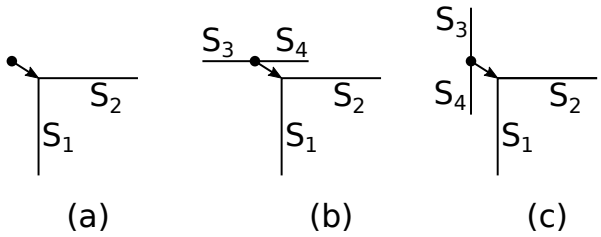


Figure 11-17. (a) illustrates the ambiguity of determing the normal vector for a node to segment contact at a corner since there is no knowledge about the shape of the surface impacting the corner. With segment-to-segment contact, this difficulty is resolved since it looks at segments coming into contact with segments. Thus LS-DYNA can judge the difference between cases (b) and (c).

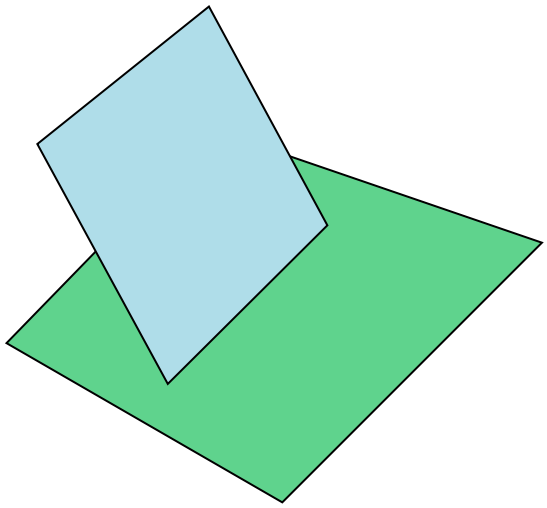


Figure 11-18. Example of two segments coming into contact at a T-intersection. The algorithms for segment-to-segment contact would choose the direction of the contact force as the normal direction to the segment in green.

the segments in a pair are used to choose a reference segment which may be from either SURFA or SURFB.

Why use segment-to-segment contact

In certain situations, node-to-segment contact can lead to undesirable behavior. For instance, if a node of one surface contacts another surface at a corner, determining the reference segment normal is ambiguous (see Figure 11-17). Segment-to-segment contact solves this problem. It looks at segments contacting segments and the directions of the segment normal vectors, thus making the contact direction less ambiguous.

Another problem arises when two segments come into contact at a T-intersection (see Figure 11-18). In traditional contact, the tracked node would be pushed away from the reference segment in the direction normal to the reference segment. This behavior may not always be physical, even in symmetric contact, when both sides are checked for penetration. With segment-to-segment contact, excluding one-way, the contact algorithm

determines the tracked and reference segments. With Figure 11-18 as an example, the algorithm determines that the green segment is the reference segment and the blue the tracked segment. Therefore, the normal direction for the contact force is chosen to be the normal vector of the green segment. Note that for one-way contact, the direction is always the normal vector to the segments in SURFB. Thus, if the segment in blue is in SURFB, unphysical results will occur.

When segment-to-segment contact is available
Segment-to-segment contact is for general shell, solid, and beam element contact. It is available for contact types with SURFACE_TO_SURFACE, ONE_WAY_SURFACE_TO_SURFACE, and SINGLE_SURFACE in the name that may additionally include AUTOMATIC, ERODING, and AIRBAG. When the contact type includes AUTOMATIC, the orientation of shell segment normal vectors is automatic. Otherwise, the segment or element orientations are used as input.

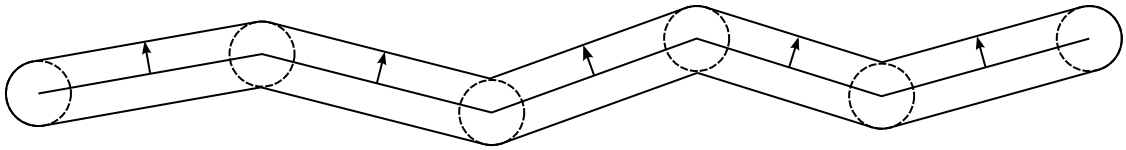
Note that including beam elements in the contact interface requires setting the IBEAMS2S flag on optional Card G to 1 and specifying the contact surfaces, SURFA and SURFB (if needed), with parts or part sets. Additionally, only DEPTH values (see Fields used with SOFT = 2 below) that support edge-to-edge contact (see Edge treatment below) are available, namely, DEPTH = 5, 15, 25, 35, and 55. If DEPTH is not set to any of these values, the segment-to-segment contact algorithm internally sets DEPTH to 55 for contact that includes beam elements.

Edge treatment
Edge-to-edge treatment with SOFT = 2 differs from the edge treatments with SOFT = 0 or 1. The edge treatment method for SOFT = 2 looks at penetration for each pair of segments and determines whether the penetrations look like surface or edge penetration. Thus, it is available for solid and shell elements. For SOFT = 0 or 1, edge treatment is only available for shells. SOFT = 2 also applies edge treatment to interior edges if the penetration occurs at a bend in the mesh. For SOFT = 0 or 1, interior edge penetration is not available except for with *CONTACT_AUTOMATIC_GENERAL_INTERIOR when using shell elements (see Remark 2 in General Remarks for details).

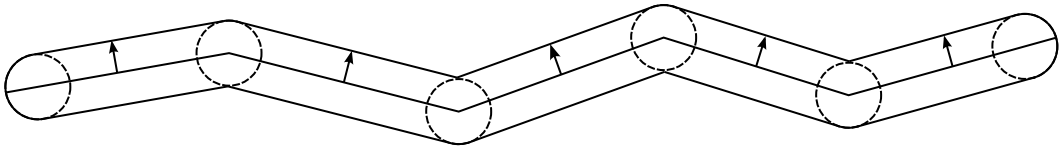
By default, we do not enable edge treatment for SOFT = 2. It depends on the setting of DEPTH. See Fields used with SOFT = 2 for which settings of DEPTH enable this feature.

Projecting the contact surface for shell thickness
To account for shell thickness, segment-to-segment contact uses segment-based projection on both the tracked and reference segments as shown in Figure 11-19. With this projection, half the contact thickness is projected normal to the shell midplane in each direction. Note that SHLOFF on Optional Card G and CNTCO on *CONTROL_SHELL can change the location of the reference plane for projection.

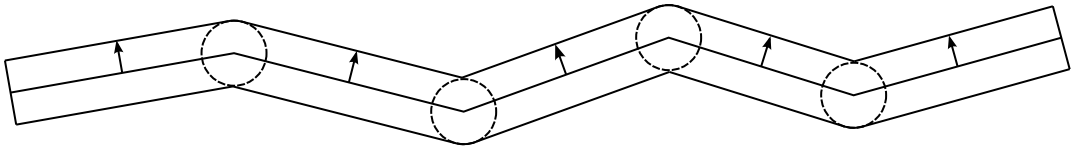
The treatment of external edges depends on SHLEDG on Optional Card D and SRNDE on Optional Card E with SHLEDG having priority. The internal edge treatment depends



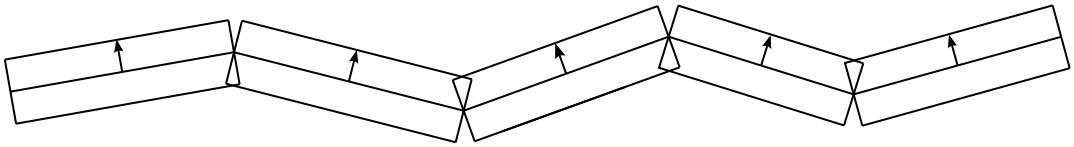
Projecting the contact surface with SHLEDG = 0 and SRNDE = 0



Projecting the contact surface with SHLEDG = 0 and SRNDE = 1



Projecting the contact surface with SHLEDG = 0 **an**d SRNDE = 2



Projecting the contact surface with SHLEDG = 1

Figure 11-19. Projecting the contact surface for SOFT = 2

only on SHLEDG. If SHLEDG is 0 on *CONTROL_CONTACT and on *CONTACT_..., the shell edges are rounded with a radius of half the contact thickness. However, setting SRNDE = 1 or 2 changes the behavior of the exterior edges (see Optional Card E for a description). If SHLEDG is 1, SRNDE is ignored and both exterior and interior edges are square and flush with the nodes. Note that for values of DEPTH = 23, 33, 35, and 55, the setting of SHLEDG is ignored and the behavior is like SHLEDG = 1 (see Fields used with SOFT = 2 for a discussion of DEPTH). The advantage of SHLEDG = 0 is that it better preserves energy than SHLEDG = 1 when the surfaces meet at an angle. However, as shells become smaller this advantage has become less noticeable, so SHLEDG = 1 works well in most cases.

Fields used with SOFT = 2

Fields that do *not* apply to the supported contact types are *not* used, such as FLANGL on optional Card C. Only the GRPABLE field is supported on the MPP cards. All fields on mandatory Cards 1, 2, and 3 are active, except for VSF and PENCHK. On optional Card A, some parameters have different meanings than they do for the default contact, which are discussed in detail below. MAXPAR on optional Card A is not supported. PENMAX, THKOPT, and SHLTHK on optional Card B are also not used (only the ISYM, I2D3D, SLDTHK, and SLDSTF parameters are active). IGNROFF on Card F is also not supported.

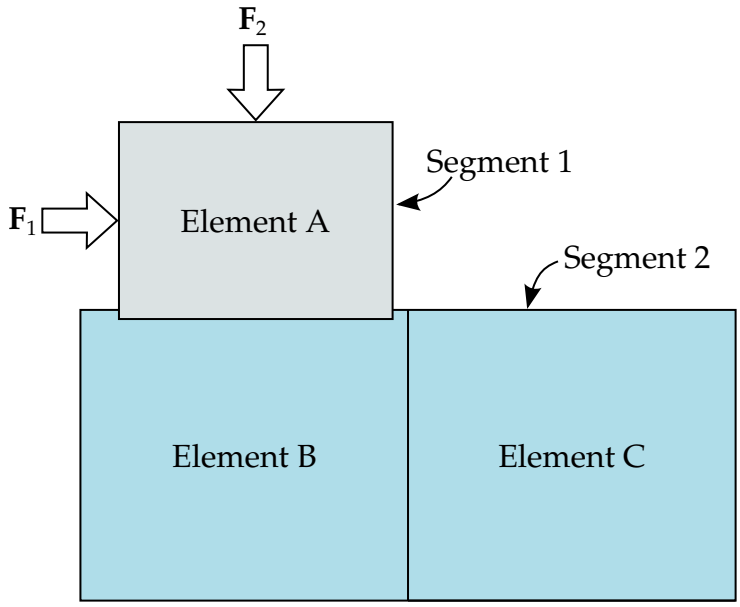


Figure 11-20. Example situation of when to use the sliding option (SBOPT = 3 or 5). Without the sliding option, the regular algorithm causes Segment 2 to become a tracked segment and Segment 1 to become a reference segment because offsets do not apply to solid elements. The sliding option prevents this unphysical contact force between those segments.

The SBOPT parameter on optional Card A controls several options. We do not recommend setting DEPTH = 1 for pinball edge-to-edge checking and only include it for backward compatibility. For edge-to-edge checking, we recommend setting DEPTH = 5 instead (see below). The warped segment option more accurately checks for penetration of warped surfaces.

The sliding option (SBOPT = 4 or 5) uses neighbor segment information to improve sliding behavior. It is useful for preventing segments from incorrectly catching nodes on a sliding surface. However, we do not recommend using it with DEPTH = 5, 25, and 35 because it can interfere with edge-to-edge checking and lead to incorrect behavior. Figure 11-20 illustrates an example situation ideal for using the sliding option. In this example, a block modeled with a solid element slides across a surface consisting of two solid elements. Because segments on solid elements do *not* have offsets, Element A must penetrate Element B for contact forces to apply. With the default algorithm, Segment 2 on Element C becomes a tracked segment, and Segment 1 on Element A becomes a reference segment as Element A moves to the right. Thus, the regular contact algorithm applies a nonphysical force in the normal direction of Segment 1. The sliding option prevents this unphysical contact force because it looks at the segments neighboring Segment 2. The orientation of the segment to the left of Segment 2 causes the algorithm to not apply a contact force in that direction.

The DEPTH parameter controls several additional options for segment-to-segment contact.

1. **DEPTH = 2 (default; but not recommended).** Surface penetrations measured at nodes are checked.

2. **DEPTH = 3.** Surface penetration is also measured at the edge. This option is more accurate than DEPTH = 2 and is good for a wide variety of simulations but does not check for edge-to-edge penetration.

3. **DEPTH = 5.** Both surface penetrations and edge-to-edge penetrations are checked.

4. **DEPTH = 13.** The penetration checking is the same as for DEPTH = 3, but the code has been tuned to conserve energy better.

5. **DEPTH = 15.** Behavior is the same as DEPTH = 5 from versions R4.2 and earlier. This option enables backward compatibility of results.

6. **DEPTH = 23.** The penetration checking is similar to DEPTH = 3 but uses different methods to try to improve robustness.

7. **DEPTH = 25 or 35.** The penetration checking is similar to DEPTH = 5 but uses different methods to try to improve robustness.

8. **DEPTH = 45.** The splitting pinball method [Belytschko and Yeh, 1993] is used. This method is more accurate at the cost of more CPU time. It is recommended when modeling complex contacts between parts comprised of shells. It does not apply to solid or thick shell parts, but such parts can be coated with null shells to make DEPTH = 45 available. It also includes edge-to-edge checking.

9. **DEPTH = 55.** The penetration checking is similar to DEPTH = 25 and 35, but it is robust in edge-to-edge checking. Thus, penetration is less likely.

10. **DEPTH = 1 or 4.** Airbag contact has two additional options, DEPTH = 1 and 4. DEPTH = 4 activates additional airbag logic that uses neighbor segment information when judging if the contact is between interior or exterior airbag surfaces. This option is not recommended and is maintained only for backward compatibility. Setting DEPTH = 1 suppresses all airbag logic.

About SOFT = 6:

SOFT = 6 contact addresses contact issues in situations where blank gage pins are narrow or small and blank meshes are coarse (Figure 11-21 left), leading to missing contact in some cases. This feature applies to gravity loading and forming of a sheet blank with an adaptive mesh. It can *only* be used with *CONTACT_FORMING_NODES_TO_SURFACE. In addition, the variable ORIENT in *CONTROL_CONTACT must be set to "4". Currently this feature is available in double precision, SMP only.

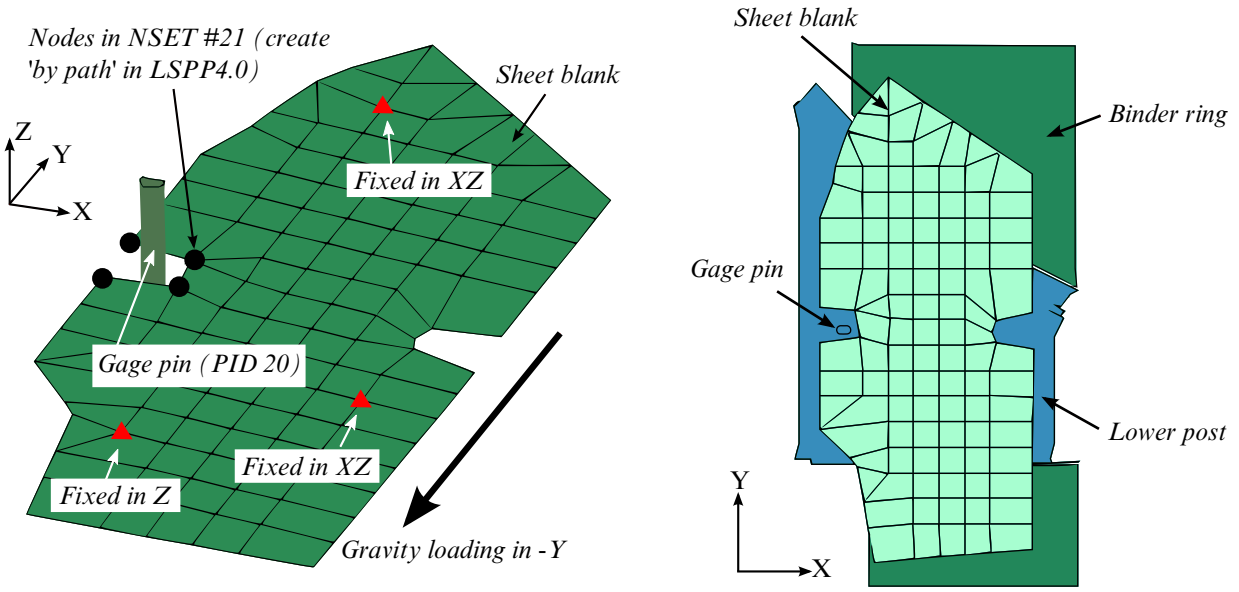
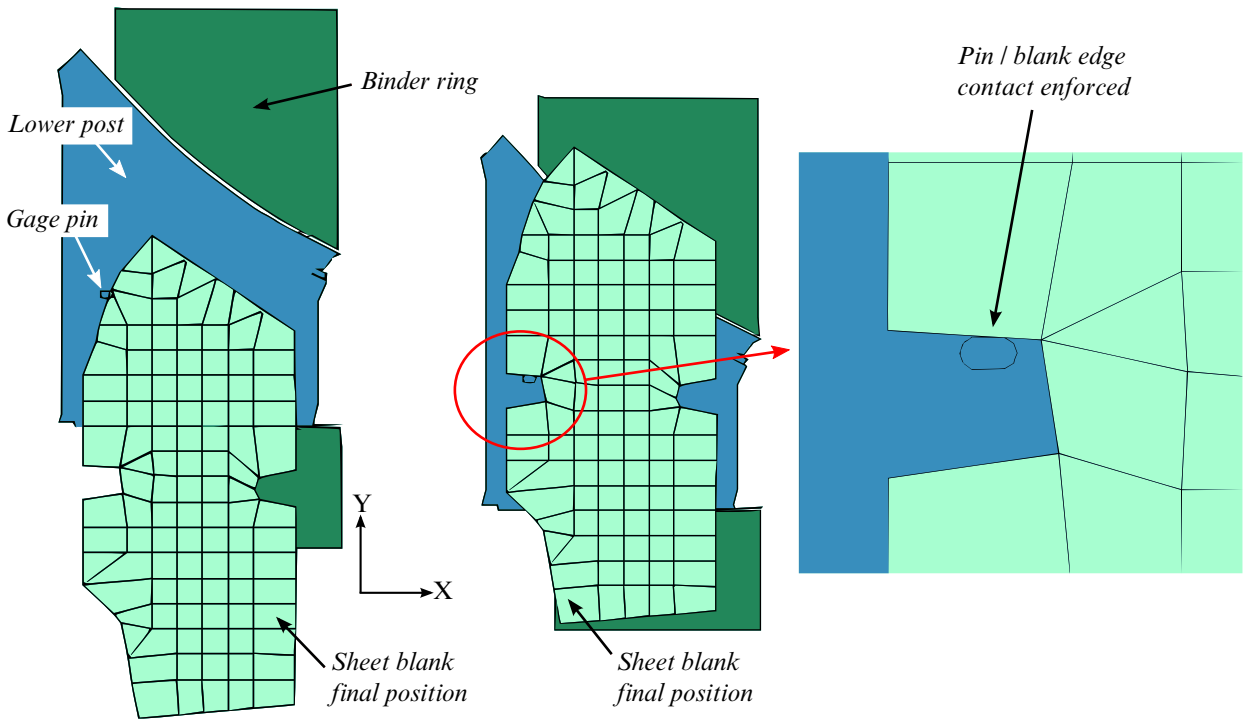


Figure 11-21. Illustrative/test model for SOFT = 6 (left) and initial blank position.

1. **SURFA type.** SURFA must be a node set that includes the nodes along the entire blank edge or the portion of the blank edge that will be in contact with gage pins (see Figure 11-21 left) or the part ID (not part set ID) of the blank. In the node set case, the nodes in the node set must be listed in consecutive order, as defined *by path* in the LS-PrePost application, under *Model → CreEnt → Cre → Set Data → *SET_NODE*. Note that the blank edge and the gage pins do not have any thickness.

2. **Example.** In the partial keyword example below, node set ID 21 (SURFATYP = 4) is in contact with the gage pin of part set ID 20. As shown in Figure 11-21 (left), with the boundary condition applied, a blank with a very coarse mesh is loaded with a body force. The left notch is anticipated to be in contact with the gage pins. The initial position (top view) of the test model is shown in Figure 11-21 (right) and the final gravity-loaded blank positions are shown in Figure 11-22 (left) without SOFT = 6, and in Figure 11-22 (middle and right) with SOFT = 6. Without SOFT = 6, the contact between the blank edge and the pin is missed completely.

```
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_FORMING
1
*CONTROL_IMPLICIT_GENERAL
1,0.2
*CONTROL_IMPLICIT_NONLINEAR
$  NSLOLVR    ILIMIT    MAXREF      DCTOL      ECTOL      RCTOL      LSTOL
         2         1      1200      0.000       0.00          0
$    dnorm   divflag   inistif
         0         2         0          1                     1
*SET_NODE_LIST
```



Gravity loading results without using SOFT=6; Pin/blank edge contact missed.

Gravity loading results using SOFT=6; Pin/blank edge contact sucessful.

Figure 11-22. Final blank position without (left) and with (right) SOFT = 6.

```
$ blank edge node set around the gage pin
21
1341,1342,1343,1344
*SET_PART_LIST
$ gage pin
20
20
*SET_PART_LIST
$ blank
13
13
*CONTACT_FORMING_NODES_TO_SURFACE
$     SURFA      SURFB   SURFATYP   SURFBTYP    SABOXID    SBBOXID       SAPR       SBPR
         21         20          4          2
$        FS         FD         DC          V        VDC     PENCHK         BT         DT
      0.125                                            20.          4
$      SFSA       SFSB       SAST       SBST      SFSAT      SFSBT        FSF        VSF

$      SOFT
          6
```

3. **Coarse meshes and initial contact penetration.** Sometimes a coarse blank mesh causes an initial contact penetration between the blank gaging hole edge and the gaging pin mesh. Starting with R11.0, this contact algorithm creates additional nodes on the blank along the hole edge and then moves the nodes to clear the initial penetration with the gage pin, as shown in Figure 11-23.

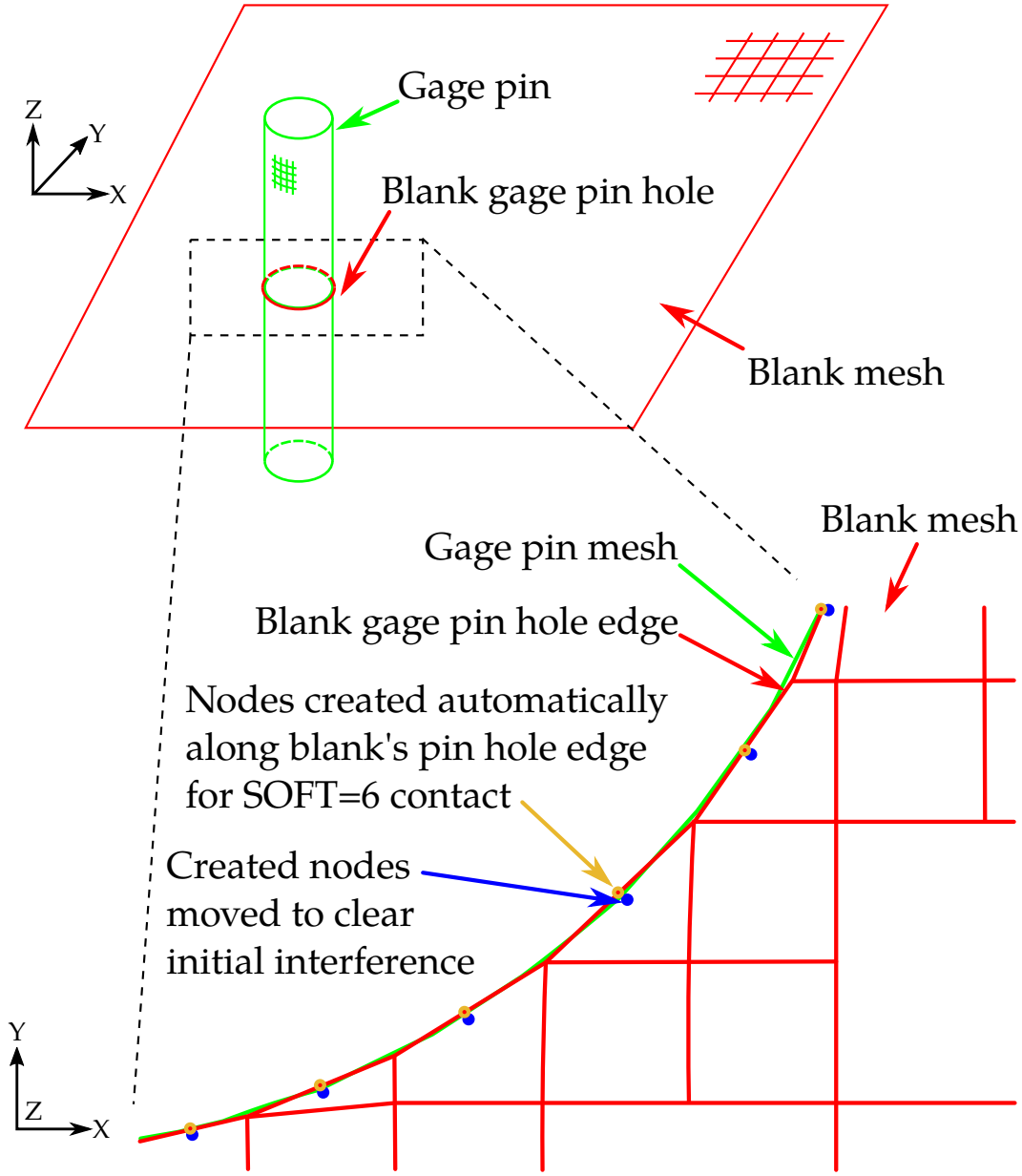


Figure 11-23. Improvement to SOFT = 6 for a coarse blank mesh

Optional Card B:

> **NOTE:** If using Optional Card B, then you must include Optional Card A. (Optional Card A may be a blank line.)

Optional Card B.

Card B	1	2	3	4	5	6	7	8
Variable	PENMAX	THKOPT	SHLTHK	SNLOG	ISYM	I2D3D	SLDTHK	SLDSTF
Type	F	I	I	I	I	I	F	F
Default	↓	0	0	0	0	0	0.0	0.0
Remarks		Old types 3, 5, 10	Old types 3, 5, 10					

VARIABLE	DESCRIPTION
PENMAX	For old types 3, 5, 8, 9, 10 (see Mapping of *CONTACT keyword option to "contact type" in d3hsp at the end of General Remarks) and Mortar contact, PENMAX is the maximum penetration distance. For contact types a3, a5, a10, 13, 15, and 26, see Table 11-2 for how the maximum penetration distance is determined. EQ.0.0: For old type contacts 3, 5, and 10, use small penetration search and value calculated from thickness and XPENE; see *CONTROL_CONTACT. EQ.0.0: For contact types a3, a5, and a10, the default value is 0.4 for shells and solids in SMP, 0.5 for solids in MPP, and 0.8 for shells in MPP. EQ.0.0: For contact types, 13 and 15, the default is 0.4 for shells, 0.4 for solids in SMP, and 0.5 for solids in MPP. EQ.0.0: For contact type 26, the default value is 10. EQ.0.0: For Mortar contact, the default is a characteristic size of the element; see Theory manual.
THKOPT	Thickness option for contact types 3, 5, and 10 (non-automatic surface-to-surface including one-way and non-automatic nodes-to-

VARIABLE	DESCRIPTION
	surface): EQ.0: Default is taken from the control card; see SHLTHK on *CONTROL_CONTACT. EQ.1: Thickness offsets are included. EQ.2: Thickness offsets are not included (old way).
SHLTHK	Define if and only if THKOPT above equals 1. Shell thickness is considered in non-automatic surface-to-surface and non-automatic node-to-surface type contact options, where options 1 and 2 below activate the new contact algorithms. The thickness offsets are always included in single surface, constraint method, and automatic contact types. EQ.0: Thickness is not considered. EQ.1: Thickness is considered, but rigid bodies are excluded. EQ.2: Thickness is considered including rigid bodies.
SNLOG	For SOFT = 0 or 1, SNLOG controls shooting node logic in thickness offset contact. With shooting node logic enabled, the tracked is moved back to the reference surface without applying any contact force in the first cycle that it penetrates the reference segment. EQ.0: Logic is enabled (default). EQ.1: Logic is skipped (sometimes recommended for metal forming calculations or for contact involving foam materials). For SOFT = 2, SNLOG controls the thick segment check (see Remark 1): EQ.0: Do a thick segment check but do not report the results. EQ.1: Do a thick segment check but do not report the results. EQ.2: Do a thick segment check and report the results to the d3hsp file. EQ.3: Do not do a thick segment check.
ISYM	Symmetry plane option: EQ.0: Off EQ.1: Do not include faces with normal boundary constraints (such as segments of brick elements on a symmetry

VARIABLE	DESCRIPTION
	plane). This option is important to retain the correct boundary conditions in the model with symmetry. For ERODING contacts this option may also be defined on Card 4.
I2D3D	Segment searching option: EQ.0: Search 2D elements (shells) before 3D elements (solids, thick shells) when locating segments. EQ.1: Search 3D (solids, thick shells) elements before 2D elements (shells) when locating segments.
SLDTHK	Optional solid element thickness. For non-Mortar contacts, a nonzero positive value will activate the contact thickness offsets in the contact algorithms where offsets apply. The contact treatment will then be equivalent to the case where null shell elements are used to cover the brick elements. The contact stiffness parameter below, SLDSTF, may also be used to override the default value. For Mortar contacts, SLDTHK is the offset from the solid element surface in the direction of the normal to the point where contact acts. Therefore, in this case a negative value for SLDTHK is well defined. SLDSTF is ignored for Mortar contact.
SLDSTF	Optional solid element stiffness. A nonzero positive value overrides the bulk modulus taken from the material model referenced by the solid element. For segment-to-segment contact (SOFT = 2), SLDSTF replaces the stiffness used in the penalty equation. This parameter does not apply to Mortar contacts.

Remarks:

1. **SNLOG.** For segment-to-segment contact (invoked by setting SOFT = 2 on Optional Card A), SNLOG controls the thick segment check. The thick segment check has always been performed during the initialization of single surface contact. It prevents possible unstable contact behavior that can occur when shell segments have thick offsets and segment dimensions are short relative to their thickness. When a pair of segments is found to be near each other when measured along the surface, and the segments in the pair are thick enough that their offsets could contact at a bend in the mesh, the pair is added to a list of pairs that will be skipped over during the simulation. Short and thick segments typically have enough stiffness that skipping contact between these pairs does not

compromise the solution. However, setting SNLOG = 3 switches off this check. Setting SNLOG = 2 leaves the check active but causes outputting to the d3hsp file a report of the pairs of elements that will not be checked for contact. Setting SNLOG = 0 or 1 leaves the check active with no report output.

Optional Card C:

> **NOTE:** If using Optional Card C, then you must define Optional Cards A and B. (Optional Cards A and B may be blank lines.)

Optional Card C.

Card C	1	2	3	4	5	6	7	8
Variable	IGAP	IGNORE	DPRFAC / MPAR1	DTSTIF / MPAR2	EDGEK		FLANGL	CID_RCF
Type	I	I	F	F	F		F	I
Default	1	0	0.0	0.0	0.0		0.0	0
Remarks		3	1	2	4			

VARIABLE	DESCRIPTION
IGAP	For Mortar contacts, IGAP is used to progressively increase contact stiffness for large penetrations, or use a linear relationship between penetration and contact pressure; see Remark 14 in the General Remarks section. For other contacts, IGAP can be used to improve implicit convergence at the expense of (1) creating some sticking if parts attempt to separate and (2) possibly under-reporting the contact force magnitude in the output files rcforc and ncforc. (Implicit only.) LT.0: Like IGAP = 1 except the maximum distance between contact surfaces at which stickiness is on is scaled by IGAP/10. EQ.1: Apply method to improve convergence (default). EQ.2: Do not apply method. GT.2: Set IGAP = 1 for first IGAP − 2 converged equilibrium states, then set IGAP = 2.
IGNORE	Ignore initial penetrations for the *CONTACT_AUTOMATIC options: LT.0: Applies only to the Mortar contact. When less than zero, the behavior is the same as for |IGNORE|, but contact between segments belonging to the same part is ignored.

VARIABLE	DESCRIPTION
	The main purpose of this option is to avoid spurious contact detections that otherwise could result for complicated geometries in a single surface contact, typically, when eliminating initial penetrations by interference. See IGNORE = 3 and IGNORE = 4.
	EQ.0: Take the default value from the fourth card of the *CONTROL_CONTACT input.
	EQ.1: Allow initial penetrations to exist by tracking the initial penetrations.
	EQ.2: Allow initial penetrations to exist by tracking the initial penetrations. However, penetration warning messages are printed with the original coordinates, and the recommended coordinates of each penetrating node are given. For Mortar contact, this is the default (see Remark 14 in the General Remarks section).
	EQ.3: Applies only to the Mortar contact. With this option initial penetrations are eliminated between time zero and the time specified by MPAR1. Intended for small initial penetrations. See Remark 14 in the General Remarks section.
	EQ.4: Applies only to the Mortar contact. With this option initial penetrations are eliminated between time zero and the time specified by MPAR1. In addition, a maximum penetration distance can be given as MPAR2, intended for large initial penetrations. See Remark 14 in the General Remarks section.
DPRFAC	Depth of penetration reduction factor for SOFT = 2 contact (see Remark 1 below):
	EQ.0.0: Initial penetrations are always ignored.
	GT.0.0.and.LT.1.0: Initial penetrations are penalized over time.
	GE.1.0: DPRFAC is a set ID used to limit the scope of DPRFAC.
	LE.-1.0: |DPRFAC| is the load curve ID defining DPRFAC as a function of time.
DTSTIF	Time step used in stiffness calculation for SOFT = 1 and SOFT = 2 contact (see Remark 2 below).
	EQ.0.0: Use the initial value that is used for time integration.

VARIABLE	DESCRIPTION
	GT.0.0: Use the value specified. GT.-1.0.and.LT.-0.01: Use a moving average of the solution time step (SOFT = 2 only). LE.-1.0: |DTSTIF| is the ID of a curve that defines DTSTIF as a function of time.
MPAR1	This field only applies to Mortar contact. If IGNORE = 2, MPAR1 corresponds to the initial contact pressure in interfaces with initial penetrations. For IGNORE = 3 and 4 it corresponds to the time of closure of initial penetrations. See Remark 14 in the General Remarks section. LE.-1.0: |MPAR1| is the ID of a curve defining the relative penetration reduction as a function of time. This is available for the IGNORE = 3 and 4 cases.
MPAR2	For Mortar contact and IGNORE = 4, MPAR2 corresponds to a penetration depth that must be at least the penetration occurring in the contact interface. See Remark 14 in the General Remarks section.
EDGEK	Scale factor for penalty stiffness of edge-to-edge contact when SOFT = 2 and DEPTH = 5, 15, 25, or 35 (see Remark 4): EQ.0.0: Use the default penalty stiffness. GT.0.0: Scale the stiffness by EDGEK.
FLANGL	Angle tolerance in radians for feature lines option in smooth contact: EQ.0.0: No feature line is considered for surface fitting in smooth contact. GT.0.0: Any edge with an angle between two contact segments bigger than this angle will be treated as a feature line during surface fitting in smooth contact.
CID_RCF	Coordinate system ID to output rcforc force resultants and ncforc data in a local system.

Remarks:

1. **DPRFAC.** DPRFAC is used only by segment-to-segment contact (SOFT = 2); see About SOFT = 2 under optional Card A for more details about segment-to-segment contact. By default, SOFT = 2 contact measures the initial penetration between segment pairs that are found to be in contact and subtracts the measured

value from the total penetration for as long as a pair of segments remains in contact. The penalty force is proportional to this modified value. This approach prevents shooting nodes but may allow unacceptable penetration. DPRFAC can be used to decrease the measured value over time until the full penetration is penalized. Setting DPRFAC = 0.01 causes ~1% reduction in the measured value each cycle. We recommend a small value, such as 0.001. DPRFAC does not apply to initial penetrations at the start of the calculation, only those that are measured at later times. This prevents nonphysical movement and energy growth at the start of the calculation.

The anticipated application for the load curve option is to reduce the initial penetrations at the end of a calculation if the final geometry is to be used for a subsequent analysis. To achieve this, the load curve should have a *y*-value of zero until a time near the end of the analysis and then ramp up to a positive value, such as 0.01, near the end of the analysis.

If the value of DPRFAC is greater or equal to 1.0, then the nearest integer will be used as a set ID and DPRFAC penetration reduction will only be done for pairs of segments that are both within that set. The set can be defined by either *SET_SHELL, *SET_SEGMENT, or *SET_PART. A search for a set with that set ID will be made in that order, and the search will stop when a matching set is found. The value of the first default attribute of that set (DA1) will be used as the DPRFAC value and should be either a small positive value or a negative value to reference a curve.

2. **DTSTIF.** DTSTIF is used only by the SOFT = 1 and SOFT = 2 contact options; see About SOFT = 1 and About SOFT = 2 under Optional Card A for more details. By default, when the SOFT option is active, the contact uses the initial solution time step to scale the contact stiffness. If the user sets DTSTIF to a nonzero value, the inputted value will be used. Because the square of the time step appears in the denominator of the stiffness calculation, a DTSTIF value larger than the initial solution time step reduces the contact stiffness and a smaller value increases the stiffness. This option could be used when one component of a larger model has been analyzed independently and validated. When the component is inserted into the larger model, the larger model may run at a smaller time step due to higher mesh frequencies. In the full model analysis, setting DTSTIF equal to the component analysis time step for the contact interface that treats the component will cause consistent contact stiffness between the analyses.

   The load curve option allows contact stiffness to be a function of time. This should be done with care as energy will not be conserved. A special case of the load curve option is when |DTSTIF| = LCTM on *CONTROL_CONTACT. LCTM sets an upper bound on the solution time step. For |DTSTIF| = LCTM, the contact stiffness time step value will track LCTM whenever the LCTM value is less than the initial solution time step. If the LCTM value is greater, the initial

solution time step is used. This option could be used to stiffen the contact at the end of an analysis. To achieve this, the LCTM curve should be defined such that it is larger than the solution time step until near the end of the analysis. Then the LCTM curve should ramp down below the solution time step, causing it to decrease and the contact to stiffen. A load curve value of 0.1 of the calculated solution time step will cause penetrations to reduce by about 99%. To prevent shooting nodes, the rate at which the contact stiffness increases is automatically limited. Therefore, to achieve a 99% reduction, the solution should be run for perhaps 1000 cycles with a small time step.

For segment-to-segment contact (SOFT = 2), setting DTSTIF less than or equal to -0.01 and greater than -1.0, causes the contact stiffness to be updated based on the current solution time step. Varying the contact stiffness during a simulation can cause energy growth. Thus, this option should be used with care when extra stiffness is needed to prevent penetration and the solution time step has dropped below the initial. Because quick changes in contact stiffness can cause shooting nodes, using a moving average of the solution time step can prevent this. The value of DTSTIF determines the number of terms in the moving average where $n = 100 \times$ (-DTSTIF) such that $n = 1$ for DTSTIF = -0.01 and $n = 100$ for DTSTIF = -0.999. Setting DTSTIF = -1.0 triggers the load curve option described in the previous paragraph, so DTSTIF cannot be smaller than -0.999 for this option.

3. **IGNORE with SOFT = 2.** When SOFT = 2 on Optional Card A, the treatment of initial penetrations is always like IGNORE = 1 in that initial penetrations are ignored when calculating penalty forces. If SOFT = 2 and IGNORE = 2, then a report of initial penetrations will be written to the messag file(s) in the first cycle.

4. **EDGEK.** When SOFT = 2 and DEPTH = 5, 15, 25, or 35 on Optional Card A, the EDGEK parameter will scale the contact penalty stiffness when contact is between segment edges if EDGEK > 0.0. This is true for both shell segments and solid element segments. Surface contact stiffness is unaffected by EDGEK.

Optional Card D:

> **NOTE:** If using Optional Card D, then you must include Optional Cards A, B, and C. (Optional Cards A, B, and C may be blank lines.)

Optional Card D.

Card D	1	2	3	4	5	6	7	8
Variable	Q2TRI	DTPCHK	SFNBR	FNLSCL	DNLSCL	TCSO	TIEDID	SHLEDG
Type	I	F	F	F	F	I	I	I
Default	0	0.0	0.0	0.0	0.0	0	0	0
Remarks	1	2	3	5	5		4	

VARIABLE	DESCRIPTION
Q2TRI	Flag to split quadrilateral contact segments into two triangles (only available when SOFT = 2): EQ.0: Off (default) EQ.1: On for all SURFA shell segments EQ.2: On for all SURFB shell segments EQ.3: On for all shell segments EQ.4: On for all shell segments of material type 34
DTPCHK	Time interval between shell penetration reports (only available for SOFT = 2): EQ.0.0: Off (default) GT.0.0: Check and report segment penetrations at time intervals equal to DTPCHK LT.0.0: Check and report segment penetrations at time intervals equal to |DTPCHK|. In addition, the calculation stops with an error at $t = 0$ if any intersections are initially present.
SFNBR	Scale factor for neighbor segment contact (only available when

VARIABLE	DESCRIPTION
	SOFT = 2) EQ.0.0: Off (default) GT.0.0: Check neighbor segments for contact. LT.0.0: Neighbor segment checking with improved energy balance when |SFNBR| < 1000. |SFNBR| ≥ 1000 activates a split-pinball based neighbor contact with a penalty force scale factor of |SFNBR + 1000|. For example, the force scale factor used is 2 when SFNBR = -1002.
FNLSCL	Scale factor for nonlinear force scaling, f. See Remark 5. This field only applies to segment-to-segment contact invoked with SOFT = 2 on Optional Card A (see About SOFT = 2 under Optional Card A).
DNLSCL	Distance for nonlinear force scaling, d. See Remark 5. This field only applies to segment-to-segment contact invoked with SOFT = 2 on Optional Card A (see About SOFT = 2 under Optional Card A).
TCSO	Flag to consider only contact segments (not all attached elements) when computing the contact thickness for a node or segment: EQ.0: Off (default) EQ.1: Only consider segments in the contact definition This flag only applies to shell elements. It is supported for contact types NODES_TO_SURFACE, SURFACE_TO_SURFACE, ONE_WAY_SURFACE_TO_SURFACE, and SINGLE_SURFACE, including the AUTOMATIC variants for all these types. It is only supported for SOFT = 0 and 1.
TIEDID	Flag for incremental displacement update for tied contacts (see Remark 4 below): EQ.0: Off (default) EQ.1: On
SHLEDG	Flag for assuming edge shape for shells when measuring penetration. This is available for segment-to-segment contact (SOFT = 2). EQ.0: Default to SHLEDG on *CONTROL_CONTACT EQ.1: Shell edges are assumed to be square and are flush with the nodes.

VARIABLE	DESCRIPTION
	EQ.2: Shell edges are assumed to be round with a radius equal to half the shell thickness. The edge centers lie on the lines between the segment nodes and extend outward by the radius.
	Note that for DEPTH values of 23, 33, 35, and 55, the setting of SHLEDG is ignored and the behavior is like SHLEDG = 1.

Remarks:

1. **Q2TRI.** Setting Q2TRI to a nonzero value causes quadrilateral shell segments to be split into two triangles. Only the contact segments are split. The elements are not changed. This option is only available for segment-to-segment contact which is activated by setting SOFT = 2. See About SOFT = 2 under optional Card A for more details about segment-to-segment contact.

2. **DTPCHK (penetration check).** Setting DTPCHK to a positive value causes a penetration check to be done periodically with the interval equal to DTPCHK. The check looks for shell segments that are penetrating the mid-plane of another shell segment. It does not report penetration of thickness offsets. The penetrating pairs are reported to the messag file or files for MPP. If at least one penetration is found, the total number of pairs is reported to the screen output. This option is only available for segment-to-segment contact which is activated by setting SOFT = 2. See About SOFT = 2 under optional Card A for more details about segment-to-segment contact.

3. **SFNBR.** SFNBR is a scale factor for optional neighbor segment contact checking. This is available only in segment-to-segment (SOFT = 2) contact (see About SOFT = 2 under optional Card A for more details about segment-to-segment contact). This is helpful when a mesh folds, such as during compression folding of an airbag. Only shell element segments are checked. Setting SFNBR to a negative value modifies the neighbor checking to improve energy balance. When used, a value between -0.5 and -1.0 is recommended.

4. **Round off in OFFSET and TIEBREAK.** There have been several issues with tied OFFSET contacts and AUTOMATIC_TIEBREAK contacts with offsets creating numerical round-off noise in stationary parts. By computing the interface displacements incrementally rather than using total displacements, the round-off errors that occur using single precision are eliminated. The incremental approach (TIEDID = 1) is available for the following contact types:

   TIED_SURFACE_TO_SURFACE_OFFSET

   TIED_NODES_TO_SURFACE_OFFSET

TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET

TIED_SHELL_EDGE_TO_SURFACE_OFFSET

AUTOMATIC_…_TIEBREAK

AUTOMATIC_SINGLE_SURFACE_TIED

5. **FNLSCL and DNLSCL.** FNLSCL = f and DNLSCL = d invoke alternative contact stiffness scaling options. These variables apply only when SOFT = 2 (see About SOFT = 2 under optional Card A for more details about segment-to-segment contact).

When FNLSCL > 0 and DNLSCL > 0, this feature scales the stiffness by the depth of penetration to provide smoother initial contact and a larger contact force as the depth of penetration exceeds DNLSCL. The stiffness k is scaled by the relation

$$k \rightarrow kf\sqrt{\frac{\delta}{d}} \ ,$$

where δ is the depth of penetration, making the penalty force proportional to the 3/2 power of the penetration depth. Adding a small amount of surface damping (such as VDC = 10) is advised with this feature.

For FNLSCL < 0, and DNLSCL > 0, an alternative stiffness scaling scheme is used,

$$k \rightarrow k\left[\frac{0.01|f|A_o}{d(d-\delta)}\right] \ .$$

Here A_0 is the overlap area of segments in contact. For δ greater than $0.9d$, the stiffness is extrapolated to prevent it from going to infinity.

For FNLSCL > 0, and DNLSCL = 0, the contact is scaled by the overlap area:

$$k \rightarrow kf\left(\frac{A_o}{A_m}\right) \ .$$

Here A_m is the mean area of all the contact segments in the contact interface. This third option can improve friction behavior, particularly when the FS = 2 option is used.

Optional Card E:

> **NOTE:** If using Optional Card E, then you must define Optional Cards A, B, C, and D. (Optional Cards A, B, C, and D may be blank lines.)

Optional Card E.

Card E	1	2	3	4	5	6	7	8
Variable	SHAREC	CPARM8	IPBACK	SRNDE	FRICSF	ICOR	FTORQ	REGION
Type	I	I	I	I	F	I	I	I
Default	0	0	0	0	1.	0	0	0
Remarks	1		2	3	5			4

VARIABLE	DESCRIPTION
SHAREC	Shared constraint flag (only available for segment-to-segment contact which is activated with SOFT = 2): EQ.0: Segments that share constraints are not checked for contact. EQ.1: Segments that share constraints are checked for contact.
CPARM8	This variable is similar to CPARM8 in *CONTACT_…_MPP but applies to SMP and not to MPP. CPARM8 for SMP only controls the treatment of spot weld beams in *CONTACT_AUTOMATIC_GENERAL. EQ.0: Spot weld (type 9) beams are not considered in the contact even if included in SURFA. EQ.2: Spot weld (type 9) beams are considered in the contact if included in SURFA.
IPBACK	If set to a nonzero value, create a "backup" penalty, tied contact for this interface. This option applies to constrained tied contacts only. See Remark 2.
SRNDE	Flag for non-extended exterior shell segment edges. See Remark 3 below for further information and restrictions.

VARIABLE	DESCRIPTION
	EQ.0: Exterior shell edges have their usual treatment where the contact surface extends beyond the shell edge. EQ.1: The contact surface is rounded at exterior shell edges but does not extend beyond the shell edges. EQ.2: The shell edges are square.
FRICSF	Scale factor for frictional stiffness (available for SOFT = 2 only).
ICOR	If set to a nonzero value, VDC is the coefficient of restitution expressed as a percentage. First and foremost, this option is intended only for contact between independent, unconstrained rigid bodies, and its application to any other contact situation will not render the desired effect. When SOFT = 0 or 1, ICOR applies to AUTOMATIC_NODES_TO_SURFACE, AUTOMATIC_SURFACE_TO_SURFACE and AUTOMATIC_SINGLE_SURFACE. ICOR applies to all SOFT = 2 contacts. Two proprietary implementations are available, depending on the sign of ICOR. A positive ICOR should give satisfactory results for most applications.
FTORQ	FTORQ controls the transmittal of moments across the contact interface for some contact types. The moments addressed by FTORQ are those due to contact friction (or tangential forces in the case of tied/tiebreak contact). For MPP with SOFT ≠ 2, AUTOMATIC_GENERAL, AUTOMATIC_SURFACE_TO_SURFACE, AUTOMATIC_SINGLE_SURFACE, and AUTOMATIC_NODES_TO_SURFACE are supported for nonzero FTORQ. For AUTOMATIC_GENERAL, only FTORQ = 1 works and causes the transmission of moments due to beam-to-beam friction. Setting FTORQ = 1 or 2 for AUTOMATIC_SURFACE_TO_SURFACE, AUTOMATIC_SINGLE_SURFACE, and AUTOMATIC_NODES_TO_SURFACE leads to transmitting a moment due to tied/tiebreak, contact friction, or damping. In the case of AUTOMATIC_SINGLE_SURFACE, FTORQ = 1 differs only from FTORQ = 2 in that nodal moments are distributed to the nodes of the contact segment if the segment belongs to a shell element; otherwise, the moment is transmitted as self-balanced nodal forces rather than as nodal moments. For AUTOMATIC_SURFACE_TO_SURFACE and AUTOMATIC_NODES_TO_SURFACE, FTORQ = 1 is the same as FTORQ = 2. For SMP with SOFT ≠, 2 FTORQ = 1 and 2 are supported for AU-

VARIABLE	DESCRIPTION
	TOMATIC_SURFACE_TO_SURFACE_TIEBREAK and AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK. FTORQ = 2 is supported for AUTOMATIC_SURFACE_TO_SURFACE and AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE. SOFT = 2 contacts support FTORQ = 2. FTORQ = 1 is not supported.
REGION	The ID of a *DEFINE_REGION which will delimit the volume of space where this contact is active. See Remark 4 below.

Remarks:

1. **SHAREC.** The SHAREC flag is a segment-to-segment contact (SOFT = 2) option that allows contact checking of segment pairs that share a multi-point constraint or rigid body. Sharing a constraint is defined as having at least one node of each segment that belongs to the same constraint.

2. **IPBACK.** The IPBACK flag is only applicable to constraint-based tied contacts (TIED with no options, or with CONSTRAINED_OFFSET). An identical penalty-based contact is generated with type OFFSET, except in the case of SHELL_EDGE constrained contact which generates a BEAM_OFFSET type. The ID of the generated interface will be set to the ID of the original interface plus 1 if that ID is available; otherwise, one more than the maximum used contact ID. For nodes successfully tied by the constraint interface, the extra penalty tying should not cause problems, but nodes dropped from the constraint interface due to rigid body or other conflicting constraints will be handled by the penalty contact. In MPP, nodes successfully tied by the constraint interface are skipped during the penalty contact phase.

3. **SRNDE.** Contact at shell edges is, by default, treated by adding cylindrical caps along the free edges, with the radius of the cylinder equal to half the thickness of the segment. This has the side effect of extending the segment at the free edges, which can cause problems. Setting SRNDE = 1 "rounds over" the (through the thickness) corners of the element instead of extending it. The edges of the segment are still rounded, but the overall size of the contact area is not increased. The effect is as if the free edge of the segment was moved in toward the segment by a distance equal to half the segment thickness, and then the old cylindrical treatment was performed. Setting SRNDE = 2 will treat the shell edges as square, with no extension. This variable has no effect on shell-edge-to-shell-edge interaction in AUTOMATIC_GENERAL; for that, see CPARM8 on the MPP Card.

When SOFT = 0 or SOFT = 1, SRNDE = 1 and 2 are available for the AUTOMATIC_SINGLE, AUTOMATIC_GENERAL, NODES_TO_SURFACE, and SURFACE_TO_SURFACE contacts in both SMP and MPP. However, in MPP, SRNDE = 1 and 2 cause enabling the groupable algorithm (see GRPBLE on MPP2), even if it is disabled.

When SOFT = 2, both the SRNDE = 1 and SRNDE = 2 options are available with all keywords that are supported when SOFT = 2. Both options are available for SMP, MPP, and hybrid versions, and with and without the MPP groupable option. However, for DEPTH = 23, 33, 35, and 55, SRNDE is ignored, and the behavior is like SHLEDG = 1. For all other values of DEPTH, SHLEDG takes precedence over SRNDE. SRNDE is ignored unless SHLEDG is 0 on both *CONTACT and *CONTROL_CONTACT.

When SOFT = 0 or SOFT = 1, the contact algorithm is based on the node-to-surface contact. The SRNDE options are applied to the reference segment in the contact pair, but not the tracked node. For SOFT = 2, the SRNDE options are applied to both segments in the contact pair.

4. **REGION.** Setting a nonzero value for REGION does not limit or in any way alter the list of SURFA/SURFB nodes and segments. This option should not be used for that purpose. For efficiency, the smallest possible portion of the model should be defined as SURFA or SURFB using the normal mechanisms for specifying the SURFB and SURFA surfaces. Setting a nonzero value will, however, result in contact outside the REGION being ignored. As SURFA and SURFB nodes and segments pass into the indicated REGION, contact for them will become active. As they pass out of the REGION, they will be skipped in the contact calculation. This option is currently available in MPP for all SOFT options, and in SMP when SOFT = 2. Contacts of type AUTOMATIC_SINGLE_SURFACE, AUTOMATIC_..._TO_SURFACE, and ERODING_… can be modified using REGION.

5. **FRICSF.** The FRICSF factor is an optional factor to scale the frictional stiffness. FRICSF is available only when SOFT = 2 on optional Card A. With penalty contact, the frictional force is a function of the stiffness, the sliding distance, and the Coulomb limit.

Optional Card F:

> **NOTE:** If using Optional Card F, then you must include Optional Cards A, B, C, D and E. (Optional Cards A, B, C, D and E may be blank lines.)

Optional Card F.

Card F	1	2	3	4	5	6	7	8
Variable	PSTIFF	IGNROFF		FSTOL	2DBINR	SSFTYP	SWTPR	TETFAC
Type	I	I		F	I	I	I	F
Default	0	0		2.0	0	0	0	0.0
Remarks	1					2		

VARIABLE	DESCRIPTION
PSTIFF	Flag to choose the method for calculating the penalty stiffness. This is available for segment-to-segment contact activated by setting SOFT = 2 on optional Card A (see About SOFT = 2 for details). EQ.0: Default determined by the setting of PSTIFF on *CONTROL_CONTACT. EQ.1: Based on nodal masses or on material density and segment dimensions. The segment mass is taken as the larger of the two values calculated. EQ.2: Based on material density and segment dimensions
IGNROFF	Flag to ignore the thickness offset for shells in the calculation of the shell contact surface. As an example, this allows shells to be used for meshing tooling surfaces without modifying the positions of the nodes to compensate for the shell thickness. This variable applies to automatic contact types, excluding segment-to-segment (SOFT = 2) contacts. This flag only applies to SMP. EQ.0: Use default thicknesses. EQ.1: Ignore the SURFA side thickness. EQ.2: Ignore the SURFB side thickness. EQ.3: Ignore the thickness of both sides. This option is only

VARIABLE	DESCRIPTION
	supported in SMP.
FSTOL	Tolerance used with the SMOOTH option for determining which segments are considered flat. The value is in degrees and approximately represents half the angle between adjacent segments.
2DBINR	Flag to indicate that 2D belts initially inside retractors are involved in the contact. This is only available for SURFACE_TO_SURFACE contact of segment-to-segment contact (SOFT = 2). EQ.0: No 2D belt initially inside a retractor is involved. EQ.1: 2D belts initially inside retractors are involved.
SSFTYP	Flag to determine how the SSF option on *PART_CONTACT behaves when SOFT = 2 on optional card A: EQ.0: Use SSF from the tracked segment as determined by the SOFT = 2 algorithm (see Remark 2). EQ.1: Use the larger of the SSF values. EQ.2: Use the smaller of the SSF values.
SWTPR	Flag to use tapered shell contact segments adjacent to segments that are thinned by the SPOTHIN option on *CONTROL_CONTACT. This option is only available when SOFT = 2 on Optional Card A. EQ.0: Use full thickness constant segments. EQ.1: Use tapered segments.
TETFAC	Scale factor for the computed volume of tetrahedral solid elements for the mass calculation in SOFT = 2 contact (see Remark 3): EQ.0.0: Same behavior as in R11 for backward compatibility. GT.0.0.AND.LE.0.5: Scale factor is 0.5. GT.0.5: Scale factor is TETFAC.

Remarks:

1. **PSTIFF.** See Remark 6 on *CONTROL_CONTACT for an explanation of the PSTIFF option. Specifying PSTIFF here overrides the behavior defined by PSTIFF on *CONTROL_CONTACT.

2. **SSFTYP.** SSFTYP affects how the SSF field on *PART_CONTACT is used when the contact formulation is segment-to-segment (SOFT = 2 on Optional Card A). It only applies when the segments in contact belong to different parts, and the parts have different penalty scale factors. The SSF field on *PART_CONTACT assigns a penalty scale factor to a given part. For one-way segment-to-segment contact, the SSF for SURFA is used when SSFTYP = 0. For all other segment-to-segment contacts, the contact algorithm judges for each pair of segments in contact which segment is the tracked segment and which is the reference segment. The reference segment in the pair determines the contact force direction. If SSFTYP = 0, then the SSF of the tracked segment is used. If SSFTYP = 1, then the larger of the two scale factors is used regardless of the contact type.

3. **TETFAC.** TETFAC is a factor applied when calculating the mass using the material density and segment dimensions (see PSTIFF). By default, 3/8 of the mass of the tetrahedron is used for the contact segment when calculating the mass this way. For $0 < \text{TETFAC} \leq 0.5$, half the mass of a tetrahedral element is used. For hexahedrons, using half the mass (the default behavior for hexahedrons) is reasonable. However, for tetrahedrons, a larger value than 0.5 is preferable because several tets fit into one hex. Therefore, a TETFAC value around 3.0 to 5.0 should make the contact stiffness more comparable with hex meshes.

Optional Card G:

> **NOTE:** If using Optional Card G, then you must include Optional Cards A, B, C, D, E, and F. (Optional Cards A, B, C, D, E, and F may be blank lines.)

Optional Card G.

Card G	1	2	3	4	5	6	7	8
Variable		SHLOFF	CSPOTHIN	IBEAMS2S	TIE_OPT			
Type		F	F	I	I			
Default		0.0	0.0	0	0			

VARIABLE	DESCRIPTION
SHLOFF	Flag affecting the location of the contact surfaces for shells when NLOC is nonzero in *SECTION_SHELL or *PART_COMPOSITE, or when OFFSET is specified using *ELEMENT_SHELL_OFFSET. This flag is basically a local version of CNTCO on *CONTROL_SHELL but can have modifications as seen in SHLOFF = 2. Thus, set this field to a value greater than 0 to enable the desired local behavior. EQ.0: The setting of CNTCO on *CONTROL_SHELL determines the contact reference plane. EQ.1: The contact reference plane coincides with the shell reference surface. This option is equivalent to CNTCO = 1 on *CONTROL_SHELL. EQ.2: This option only applies to segment-to-segment (SOFT = 2). It activates a faster version of CNTCO = 3 or 4, depending on the value of CNTCO on *CONTROL_SHELL. If CNTCO = 0, 1, or 3, then the behavior matches CNTCO = 3 but faster. If CNTCO = 2 or 4, then the behavior matches CNTCO = 4 but faster.
CSPOTHIN	Contact-specific spot weld thinning scale factor to override the scale factor defined in the SPOTHIN option on *CONTROL_CONTACT. This field is only active if the SPOTHIN option is enabled and is available for all contacts that support the SPOTHIN option. If set to 0.0, the scale factor from SPOTHIN is used. Setting

VARIABLE	DESCRIPTION
	CSPOTHIN to 1.0 effectively disables spotweld thinning for this contact. Note that 0.0 ≤ CSPOTHIN ≤ 1.0.
IBEAMS2S	Flag specifying whether beams are included segment-to-segment (SOFT = 2) contact: EQ.0: Beams are not included in the segment-to-segment contact. EQ.1: Beams are included in the segment-to-segment contact. Including beams in the contact does not depend on the contact keyword name, meaning it works for all contact keywords supported for segment-to-segment contact. To include beams in the contact, SURFATYP and/or SURFBTYP must be set to 2, 3, 5, or 6. If the value of DEPTH is not 5, 15, 25, 35, or 55, it is automatically changed to 55, and a warning message is issued.
TIE_OPT	Value of TIEOPT (see *CONTROL_CONTACT) for this contact. This field only applies to tied contacts. If zero, then TIE_OPT inherits its value from TIEOPT on *CONTROL_CONTACT.

General Remarks: *CONTACT

1. **Force output.** Contact force data can be recorded with *DATABASE_RCFORC, *DATABASE_NCFORC, and *DATABASE_BINARY_INTFOR. Note that LS-DYNA only outputs data between a SURFA and SURFB surface. Self-contact, such as single surface contact, only has SURFA and thus no data is output by default. In this case, a *CONTACT_FORCE_TRANSDUCER_*OPTION* is needed to extract contact forces from contact types that may not otherwise be recordable.

2. **AUTOMATIC_GENERAL compared to AUTOMATIC_SINGLE_SURFACE.** *CONTACT_AUTOMATIC_GENERAL is a single surface contact similar to *CONTACT_AUTOMATIC_SINGLE_SURFACE. Both types are automatic, so they consider shell thickness and beam thickness by offsetting the contact interface from the shell midplane and the beam centerline (see Item 4 under Allowed Values for Option 1).

   *CONTACT_AUTOMATIC_GENERAL differs from *CONTACT_AUTOMATIC_SINGLE_SURFACE in how it handles beam contact and shell edge contact. *CONTACT_AUTOMATIC_GENERAL includes treatment of beam-to-beam contact in which it checks the entire length of the beam for penetration, not just the nodes of the beam. For shells, it checks for penetration along the entire length of the exterior (unshared) shell edges. *CONTACT_AUTOMATIC_GENERAL essentially adds null beams to the exterior edges of shell parts so that edge-to-edge treatment of the shell parts is handled by virtue of contact through the automatically generated null beams. By adding the word INTERIOR to *CONTACT_AUTOMATIC_GENERAL, the contact algorithm goes a step further by adding null beams to all the shell meshlines, both along the exterior, unshared edges and the interior, shared shell edges. The EDGEONLY option skips the node-to-surface contact and does only the edge-to-edge and beam-to-beam contact.

   Another distinction is the default value of PENMAX (see Optional Card A). The value of PENMAX helps determine the value of penetration that triggers the release of a penetrating node (see Table 11-2). For AUTOMATIC_GENERAL, the default value of PENMAX causes the penetration value to be effectively unlimited, while for AUTOMATIC_SINGLE_SURFACE the default leads to a release condition of about half an element thickness.

   By default, the AUTOMATIC_GENERAL checks for nodal penetration through the three closest segments (DEPTH = 3 on Optional Card A). For AUTOMATIC_SINGLE_SURFACE, the default search depth is 2 segments. The three segment check is more expensive but may be more robust for contact in corners.

   Last, segment-to-segment contact invoked with SOFT = 2 on Optional Card A is supported for AUTOMATIC_SINGLE_SURFACE but not for AUTOMATIC_GENERAL.

3. **Edge-to-edge contact treatment.** For SOFT = 0 and 1, only AUTOMATIC_GENERAL and SINGLE_EDGE include edge-to-edge contact treatment. This treatment is available for shell elements. AUTOMATIC_GENERAL is a single surface contact that includes edge treatment by checking for penetration along the entire length of the exterior (unshared) shell edges. Appending INTERIOR to AUTOMATIC_GENERAL, additionally causes the edge contact to include interior (meshline) edges. AUTOMATIC_GENERAL_EDGEONLY performs edge-to-edge contact on exterior edges but excludes checking nodes-to-surface contact.

   SINGLE_EDGE contact is an old contact algorithm that only handles edge-to-edge contact. In other words, it has edge contact but excludes checking nodes-to-surface contact. It also only applies to exterior edges. As it is based on single surface contact, only SURFA is defined for this contact. The edge-to-edge treatment is limited in that contact only occurs between edges whose normal vectors (that lie in the plane of their respective shells) point toward each other (see the theory manual for illustration). Thus, AUTOMATIC_GENERAL is the preferred contact for edge treatment when SOFT = 0 or 1.

   When using a segment-to-segment (SOFT = 2) contact type, the checking of edge-to-edge penetration does not depend on the specific *CONTACT keyword name. Instead, edge-to-edge checking is included when the DEPTH parameter is set to 5, 15, 25, 35, 45, or 55 (see Fields used with SOFT = 2 in About SOFT = 2 under Optional Card A for details). Edge-to-edge treatment with SOFT = 2 differs from the edge treatments with SOFT = 0 or 1. The edge treatment method for SOFT = 2 looks at penetration for each pair of segments and determines whether the penetrations look like surface or edge penetration. Thus, it is available for solid and shell elements. SOFT = 2 also applies edge treatment to interior edges if the penetration occurs at a bend in the mesh.

   For Mortar segment-to-segment contact, edge-to-edge contact is enabled for the automatic Mortar contacts for shell elements. It only has edge treatment for external edges, except in the case of *MAT_034. For *MAT_034, edge-to-edge contact on internal edges is activated if the angle between two adjacent segments exceeds an internally set value. See Remark 14 for a more detailed discussion of Mortar contact.

4. **Tying distance criterion.** Tying will only work if the surfaces are near each other. The criterion used to determine whether a tracked node is tied down is that it must be "close". For shell elements, "close" is less than a distance, δ, defined as:

$$\delta_1 = 0.60 \times (\text{thickness of tracked node} + \text{thickness of reference segment})$$
$$\delta_2 = 0.05 \times \min(\text{reference segment diagonals})$$
$$\delta = \max(\delta_1, \delta_2)$$

If a node is further away, it will not be tied, and a warning message will be printed. For solid elements, the node thickness is zero, and the segment thickness is the element volume divided by the segment area; otherwise, the same procedure is used.

If there is a large difference in element areas between the two sides of the contact interface, δ_2 may be too large, causing the unexpected projection of nodes that should not be tied. This can occur when adaptive remeshing is used. To avoid this difficulty, the SURFA and SURFB thicknesses can be specified as negative values on Card 3 in which case

$$\delta = \text{abs}(\delta_1) \ .$$

5. **Tied contact types for different situations**. For tying solids-to-solids, that is, for situations where none of the nodes have rotational degrees-of-freedom, use TIED_NODES_TO_SURFACE and TIED_SURFACE_TO_SURFACE type contacts. These contact types may include the OFFSET or CONSTRAINED_OFFSET option.

   For tying shells-to-shells and beams-to-shells, that is, for situations where all the nodes have rotational degrees-of-freedom, use TIED_SHELL_EDGE_TO_SURFACE type contacts. This contact type may include the OFFSET, CONSTRAINED_OFFSET, or BEAM_OFFSET option. For beam spot welds, you can use *CONTACT_SPOTWELD or *CONTACT_SPOTWELD_WITH_TORSION for beam-to-shell contact. When *MAT_SPOTWELD is used for the beam material, the material model can include failure for the spot weld.

   TIED_SHELL_EDGE_TO_SOLID is intended for tying shell edges to solids or beam ends to solids.

   Tied contacts with failure include TIEBREAK contacts and TIED_SURFACE_TO_SURFACE_FAILURE. Segment orientation is important for distinguishing the direction of tension from compression when defining these contacts.

6. **Tied contact types and the implicit solver.** Nonphysical results have been observed when the implicit time integrator is used for models that combine tied contact formulations with automatic single point constraints on solid element rotational degrees of freedom (AUTOSPC on *CONTROL_IMPLICIT_SOLVER). The following subset of tied interfaces support a *strongly objective* mode (discussed in the next paragraph) and are verified to behave correctly with the implicit time integrator:

   a) TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET
      TIED_SURFACE_TO_SURFACE_CONSTRAINED_OFFSET

   b) TIED_NODES_TO_SURFACE_OFFSET
      TIED_SURFACE_TO_SURFACE_OFFSET

c) TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET

d) TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET

The first two of these ignore rotational degrees of freedom, while the third and fourth constrain rotations. The first and third are constraint-based; while the second and fourth are penalty-based. These four contact types are intended to cover most use scenarios.

Setting IACC = 1 on *CONTROL_ACCURACY activates the *strongly objective formulation* for the contacts mentioned above (as well as for the non-offset options as a side effect, namely, *CONTACT_TIED_NODES_TO_SURFACE, *CONTACT_TIED_SURFACE_TO_SURFACE, *CONTACT_TIED_SHELL_EDGE_TO_SURFACE, and *CONTACT_TIED_SHELL_EDGE_TO_SOLID). When active, forces and moments transform correctly under superposed rigid body motions within a single implicit step. Additionally, this formulation applies rotational constraints consistently *when, and only when, necessary*. In particular, strong objectivity is implemented so that tracked nodes without rotational degrees of freedom are not rotationally constrained, while tracked nodes with bending and torsional rotations are rotationally constrained. Additionally, strong objectivity ensures that the constraint is physically correct.

For a reference node belonging to a *shell*, the tracked node's bending rotations (rotations in the plane of the reference segment) are constrained to match the reference segment's rotational degrees of freedom; for reference nodes *not belonging to a shell*, the tracked node's bending rotations are constrained to the reference segment rotation as determined from its individual nodal translations. The tracked node's torsional rotations (rotations with respect to the normal of the reference segment) are *always* constrained based on the reference segment's torsional rotation as determined from its individual nodal translations, thus avoiding the relatively weak drilling mode of shells. This tied contact formulation properly treats bending *and* torsional rotations. Since the tracked node's rotational degrees of freedom typically come from shell or beam elements, the most frequently used options are:

TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET

TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET

The other two classes of "non-rotational" formulations:

TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET
TIED_SURFACE_TO_SURFACE_CONSTRAINED_OFFSET

TIED_NODES_TO_SURFACE_OFFSET
TIED_SURFACE_TO_SURFACE_OFFSET

are included for situations where rotations do not need to be constrained at all. See the LS-DYNA Theory Manual for further details.

7. **Tying to rigid bodies.** The following tied contact types are constraint-based and *will not* work with rigid bodies:

   TIED_NODES_TO_SURFACE

   TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET

   TIED_SURFACE_TO_SURFACE

   TIED_SURFACE_TO_SURFACE_CONSTRAINED_OFFSET

   TIED_SURFACE_TO_SURFACE_FAILURE (SMP only)

   TIED_SHELL_EDGE_TO_SURFACE

   TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET

   TIED_SHELL_EDGE_TO_SOLID

   SPOTWELD

   SPOTWELD_WITH_TORSION

   The following tied contact types are penalty-based and *will* work with rigid bodies:

   SPOTWELD_WITH_TORSION_PENALTY

   TIED_NODES_TO_SURFACE_OFFSET

   TIED_SHELL_EDGE_TO_SURFACE_OFFSET

   TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET

   TIED_SURFACE_TO_SURFACE_OFFSET

   Also, it may sometimes be advantageous to use the CONSTRAINED_EXTRA_NODE_*OPTION* instead for tying deformable nodes to rigid bodies because, in this latter case, the tied nodes may be an arbitrary distance away from the rigid body.

8. **SPOTWELD_WITH_TORSION contact.** The contact algorithm for tying spot welds with torsion, SPOTWELD_WITH_TORSION, must be used with care. Parts tied by this option should be subjected to stiffness proportional damping of approximately ten percent, meaning input a coefficient of 0.10. This can be defined for each part on the *DAMPING_PART_STIFFNESS input. Stability

problems may arise with this option if damping is not used. This comment applies also to the PENALTY keyword option.

9. **Forming contact type and surface thickness.** For "Forming" contact, the surface thickness of SURFB is ignored (SURFB should be the rigid tooling while SURFA should be the blank). Furthermore, SURFB can be offset away from the blank by setting a negative (meaning opposite of the positive normal of the SURFB surface) value for SBST on Mandatory Card 3. This offsets SURFB from the surface midplane by |SBST|/2 in the direction opposite to the SURFB positive normal direction. A tool and die can be quickly offset (virtually, not physically) with this field.

10. **Airbag interactions.** Modeling airbag interactions with structures and occupants using the actual fabric thickness, which is approximately 0.30 mm, may result in a contact breakdown, leading to inconsistent occupant behavior between different machines. Based on our experience, using a two-way automatic type contact definition, meaning AUTOMATIC_SURFACE_TO_SURFACE, between any airbag to structure/occupant interaction and setting the airbag fabric contact thickness to at least 10 times the actual fabric thickness has helped improve contact behavior and eliminates the machine inconsistencies. Due to a large stiffness difference between the airbag and the interacting materials, we recommend the soft constraint option (SOFT = 1) or the segment-to-segment option (SOFT = 2). Note that with the above contact definition, only the airbag materials should be included in any *AIRBAG_SINGLE_SURFACE definitions to avoid duplicate contact treatment that can lead to numerical instabilities.

11. **Constraint contact.** The following two contact types are constraint-based and must be used with care. The surface and the nodes that are constrained to a surface are not allowed to be used in any other CONSTRAINT_… contact definition:

    CONSTRAINT_NODES_TO_SURFACE

    CONSTRAINT_SURFACE_TO_SURFACE

    If, however, contact must be defined from both sides as in sheet metal forming, one of these contact definitions can be a CONSTRAINT type; the other one could be a standard penalty type such as SURFACE_TO_SURFACE or NODES_TO_SURFACE.

12. **Penalty-based contact types and shell thickness.** The following penalty-based contact types consider shell thickness, meaning the contact surfaces for shells are offset from the shell midplane. This offset distance can be modified using variables on Card 3 of *CONTACT. The SHLTHK option on the *CONTROL_CONTACT card is ignored for these contact types.

AIRBAG_SINGLE_SURFACE

AUTOMATIC_GENERAL

AUTOMATIC_GENERAL_INTERIOR

AUTOMATIC_NODES_TO_SURFACE

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

AUTOMATIC_SINGLE_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE

SINGLE_SURFACE

A thickness that is too small may result in loss of contact, while an unrealistically large thickness may result in a degradation in speed during the bucket sorts as well as nonphysical behavior. To address the latter behavior, the default setting of SSTHK in *CONTROL_CONTACT may sometimes cause the contact thickness to be reduced for certain single surface contacts (see Remark 2 of *CONTROL_CONTACT).

13. **Projecting the contact surface for shell thicknesses.** For SOFT = 0 and 1, the method for projecting the contact surface to account for shell thickness depends on the choice of contact and the choice between SMP and MPP. The two implemented methods for projection are nodal normal projection and segment-based projection. For a discussion of projecting the contact thickness for SOFT = 2, see Projecting the contact surface for shell thickness under About SOFT = 2 in the remarks following Optional Card A.

    For SMP, segment-based projection is used in contact types:

    AIRBAG_SINGLE_SURFACE

    AUTOMATIC_GENERAL

    AUTOMATIC_NODES_TO_SURFACE

    AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

    AUTOMATIC_SINGLE_SURFACE

    AUTOMATIC_SURFACE_TO_SURFACE

    FORMING_NODES_TO_SURFACE

    FORMING_ONE_WAY_SURFACE_TO_SURFACE

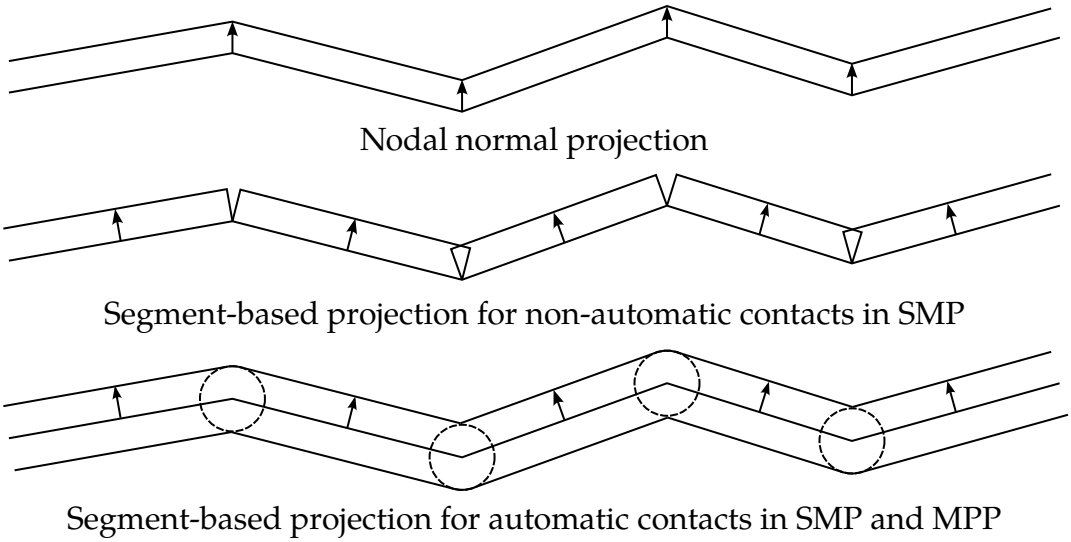


Figure 11-24. Nodal normal projection and segment-based projection of shell elements

FORMING_SURFACE_TO_SURFACE

The remaining contact types in SMP use nodal normal projections if projections are used (see THKOPT and SHLTHK on Optional Card B). For MPP, all automatic contacts use segment-based projection, while all the others use nodal normal projections (if projections are used).

Figure 11-24 illustrates the different types of projection. Note that for the MPP version of nodal normal projection, the vectors are computed such that the surface looks like a segment-based projection with the surfaces extended to remove the gaps. Segment-based projection is used for automatic contacts in both SMP and MPP, and for this kind of contact, the projection occurs in both directions automatically. The gaps are automatically closed for these contacts using circles tangent to the projections. In SMP, the gaps for non-automatic contacts projected with segment-based projection can be closed with MAXPAR on Optional Card A.

The method used can influence the accuracy and cost of the calculation. The main advantage of nodal projections for non-automatic contacts is that a continuous contact surface is obtained, which is much more accurate in applications such as metal forming. The disadvantages of nodal projections are the higher costs due to the nodal normal calculations, difficulties in treating T-intersections and other geometric complications, and the need for consistent orientation of contact surface segments.

For the treatment of the free edges of shell surfaces, see SRNDE on Optional Card E.

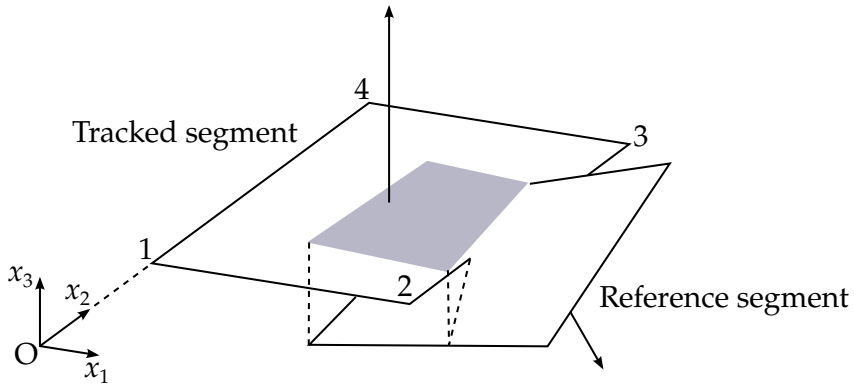


Figure 11-25. Illustration of Mortar segment-to-segment contact

Note that SHLOFF on Optional Card G and CNTCO on *CONTROL_SHELL can change the location of the reference plane for the projection.

14. **Overview of Mortar contact.** Mortar contact, invoked by appending the suffix MORTAR to FORMING_SURFACE_TO_SURFACE, AUTOMATIC_SURFACE_TO_SURFACE, or AUTOMATIC_SINGLE_SURFACE, is a segment-to-segment penalty-based contact. For two segments on each side of the contact interface that are overlapping and penetrating, a consistent nodal force assembly taking into account the individual shape functions of the segments is performed; see Figure 11-25 for an illustration. A *CONTACT_FORCE_TRANSDUCER_PENALTY can extract forces from Mortar contacts, *but the SURFA and SURFB sides must then be defined through parts or part sets*. For the automatic Mortar contacts, support for eroded solid and shell elements is automatically invoked without the need for an ERODING option. It will also create new edge segments exposed to the exterior as shell elements erode, but beam elements are not yet supported. As for transducers, eroding requires the SURFA and SURFB sides to be defined through parts or part sets. In this respect, the results with this contact may be more accurate, especially when considering contact with elements of higher order.

By appending the suffix TIED to the *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR keyword or the suffix TIEBREAK_MORTAR (only OPTION = 2, 4, 6, 7, 8, and 9 are supported) to the *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE keyword, the contact is treated as a tied contact interface with optional failure in the latter case. For MORTAR_TIED_WELD, parts can be welded together to create a tied contact at the welded zones; segment pairs are dynamically created based on a weld criterion to make up a tied contact element. For all these Mortar tied contacts, the tied SURFA segments can be viewed in the intfor file by specifying NTIED on *DATABASE_EXTENT_INTFOR, a convenient way to know which segments. This contact is intended for implicit analysis in particular but is nevertheless supported for explicit analysis as well. For explicit analysis, the bucket sort frequency is 100 if not specified.

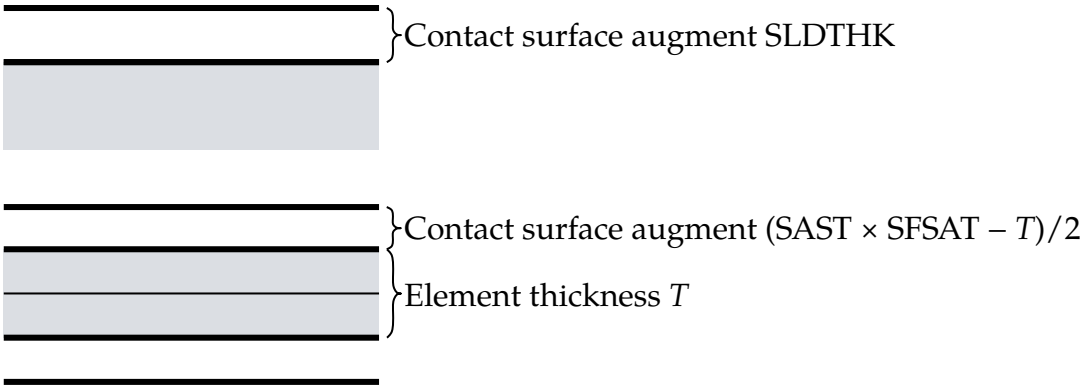


Figure 11-26. Illustration of contact surface location for automatic Mortar contact, solids on top and shells below.

FORMING *Mortar* contact, in contrast to other forming contacts, does *not* assume a rigid SURFB side, but, if this side consists of shell elements, the normal vector should be oriented towards the SURFA side. Furthermore, no shell thickness is considered on the SURFB side. The SURFA side is assumed to be a deformable part, and the orientation of the elements does not matter. However, each FORMING contact definition should be such that contact occurs with ONE deformable SURFA side only, which obviously leads to multiple contact definitions if two-sided contact is presumed.

AUTOMATIC contact is supported for solids, shells, thick shells, and beams. Here, the thicknesses are considered for both rigid and deformable parts. Flat edge contact is supported for shell elements, and contact with beams occurs on the lateral surface area as well as on the end tip. The contact assumes that the beam has a cylindrical shape[1] with a cross-sectional area coinciding with that of the underlying beam element. The contact surface can be augmented with the aid of parameters SAST and SFSAT for shells and beams, while SLDTHK is used for solids and thick shells. For shells/beams, SAST corresponds to the contact thickness of the element (SBST likewise for the SURFB side); by default, this is the same as the element thickness. This parameter can be scaled with the aid of SFSAT (SFSBT for the SURFB side) to adjust the location of the contact surface; see Figure 11-26.

For all unilateral Mortar contacts (meaning non-tied contacts), FS and FD on Mandatory Card 2 are supported and work like similar non-Mortar contacts. If implicit statics is used, the dynamic friction coefficient is set equal to the static coefficient.

For solids PENMAX on Optional Card B can be used to determine the maximum penetration. It also determines the search depth for finding contact pairs. If set,

[1] Internally the cylinder is faceted as described in the Theory Manual, chapter Mortar Contact.

it should correspond to a characteristic thickness in the model. Also, the contact surface can be adjusted with the aid of SLDTHK if it is important to reduce the gap between parts; see Figure 11-26. This may be of interest if initial gaps result in free objects undergoing rigid body motion and thus preventing convergence in implicit. SLDTHK may be set to a negative number, meaning that the contact surface will be offset in the negative direction of the normal by the amount specified. This option may be useful, for instance, if mesh coarseness prevents concentric cylinders from rotating freely with respect to each other.

For the TIED option, the criterion for tying two contact surfaces is, by default, that the distance should be less than 0.05 × T, that is, within 5% of the element thickness (characteristic size for solids). In this case, PENMAX can be used to set the tying distance, meaning if PENMAX is positive, then segments are tied if the distance is less than PENMAX.

If initial penetrations are detected (reported in the messag file), then, by default, these penetrations will be given zero contact stress. This treatment is due to IGNORE = 2 being the default for Mortar contact. IGNORE = 2 behaves differently for Mortar contacts than for other contacts. For this option, the penetrations are not tracked, but the contact surface is fixed at its initial location. In addition, for IGNORE = 2, an initial contact pressure can be imposed on the interface by setting MPAR1 to the desired contact pressure. Any rigid body motion due to initial contact gaps can be properly eliminated with these settings. If you want initial penetrations to result in contact pressure (the "old" IGNORE = 0 behavior), use IGNORE = 2 and set MPAR1 to a large number; the contact stress cannot be higher than that allowed for the given penetration. IGNORE = 1 is similar to IGNORE = 2, except the penetrations will be tracked. Thus, if the surfaces separate, the contact surface locations are updated accordingly. In the event of surfaces separating enough to form gaps, the contact is restored to as if there were no initial penetrations in the first place.

A third option is IGNORE = 3, for which prestress can be applied. This allows initial penetrations to exist which are closed during the time between zero and the value given by MPAR1. It works like the INTERFERENCE option, except that the closure is linear in time. Initial penetrations, however, must be small enough for the contact algorithm to detect them. For this option, MPAR1 can be chosen negative, meaning that the reduction factor is specified as a function of time with a load curve. This load curve should increase from 0 to 1, where 1 means that all initial penetrations have been removed.

For large penetrations IGNORE = 4 is recommended (this can only be used if the SURFA side consists of solid elements). This option does pretty much the same thing as IGNORE = 3, but you may provide a penetration depth with MPAR2. This depth must be at least as large as (and preferably in the order of) the maximum initial penetration in the contact interface; otherwise, an error termination

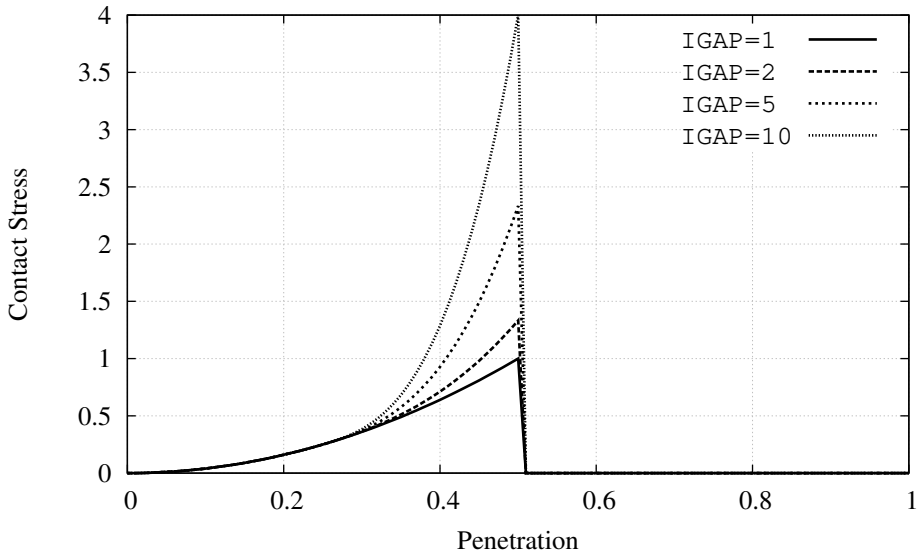


Figure 11-27. Mortar contact stress as function of penetration

will be the result. This parameter helps the contact algorithm locate the contact surface and thus estimate the initial penetration. With this option, the contact surfaces are pushed back and placed in incident contact at places where initial penetrations are present which can be done for (more or less) arbitrary initial penetration depths. Like IGNORE = 3, the contact surfaces will be restored linearly in the time given by MPAR1.

Mortar contacts in implicit analysis can cause the contact pressure to locally be very high, leading to the release of large enough penetrations in subsequent steps. Penetration information can be requested on MINFO on *CONTROL_OUTPUT which issues a warning if there is a danger of this happening. To prevent contact release, you may increase IGAP which penalizes large penetrations without affecting small penetration behavior and thereby overall implicit performance. Figure 11-27 shows the contact pressure as function of penetration for the Mortar contact, including the effect of increasing IGAP. It also shows that for sufficiently large penetrations, the contact is not detected in subsequent steps, which is something to avoid. In addition, a negative IGAP can be used to render a linear relationship between the penetration and contact pressure. This might be helpful if relatively large penetrations are observed for small contact pressures.

Table 11-1 lists fields on the mandatory and optional cards that apply to MORTAR contact. Any parameter not mentioned in this list is ignored. See a similar table on the manual page for *CONTROL_CONTACT.

Table 11-1. Fields related to Mortar contact

Data Card	Comment
Mandatory Card 1	Apply as for any other contact. Parts or part sets are recommended. Contact stiffness is taken from the SURFA side of the contact.
Mandatory Card 2	Except for PENCHK, these fields also apply as for any other contact. PENCHK is ignored.
Mandatory Card 3	Except for SFSB, FSF and VSF, these also apply as for any other contact. SFSB, FSF and VSF are ignored.
Optional Card A	Only BSORT applies. All others are ignored. BSORT is the bucket sort frequency.
Optional Card B	Only PENMAX and SLDTHK apply. All others are ignored. PENMAX is the contact search depth for solid element segments, and SLDTHK is the distance by which solid element segments are offset in the direction of the segment normal.
Optional Card C	First four parameters apply, others are ignored. IGAP is a stiffness parameter for large penetrations which is usually not needed. IGNORE deals with initial penetrations as described above, and MPAR1/MPAR2 are parameters associated with the choice of IGNORE.
Optional Cards D, E, F, and G	No parameter applies. Shell edges are always flat.

Table 11-2. Criterion for node release for nodal points which have penetrated too far. This criterion does not apply to SOFT = 2 contact. Larger penalty stiffnesses are recommended for the contact interface which allows nodes to be released. For node-to-surface type contacts (5, 5a), the element thicknesses which contain the node determines the nodal thickness. The parameter is defined on the *CONTROL_CONTACT input.

INTERFACE TYPE ID	PENCHK	ELEMENT TYPE	FORMULA FOR RELEASE OF PENETRATING NODAL POINT
1, 2, 6, 7	-		
3, 5, 8, 9, 10 (without thickness)	0	solid	d = PENMAX if PENMAX > 0 d = 10^{10} if PENMAX = 0
		shell	d = PENMAX if PENMAX > 0 d = 10^{10} if PENMAX = 0
	1	solid	d = XPENE × thickness of solid element
		shell	d = XPENE thickness of shell element
	2	solid	d = 0.05 × minimum diagonal length
		shell	d = 0.05 × minimum diagonal length
3, 5, 10 (thickness), 17 and 18	-	solid	d = XPENE × thickness of solid element
		shell	d = XPENE × thickness of shell element
a3, a5, a10	-	solid	d = PENMAX × thickness of solid element *[default: PENMAX = 0.4 in SMP and 0.5 in MPP]*
		shell (SMP)	d = PENMAX × (tracked thickness + reference thickness) for PENMAX ≥ 0.4. d = 0.4 × (tracked thickness + reference thickness) for PENMAX < 0.4. *[default: PENMAX = 0.4]*
		shell (MPP)	d = ½ × (PENMAX + 1) × (tracked thickness + reference thickness) *[default: PENMAX = 0.8]*
13, 15	-	solid (SMP)	d = PENMAX × thickness of solid element *[default: PENMAX = 0.4]*
		solid (MPP)	d = PENMAX × thickness of solid element if PENMAX ≥ 0.5 d = 0.5 × thickness of solid element if PENMAX < 0.5 *[default: PENMAX = 0.5]*
		shell	d = PENMAX × (tracked thickness + reference thickness) if PENMAX ≥ 0.4 d = 0.4 × (tracked thickness + reference thickness) if PENMAX < 0.4 *[default: PENMAX = 0.4]*
4	-	solid	d = 0.5 × thickness of solid element

INTERFACE TYPE ID	PENCHK	ELEMENT TYPE	FORMULA FOR RELEASE OF PENETRATING NODAL POINT
		shell	d = 0.4 × (tracked thickness + reference thickness)
26	———	solid	d = PENMAX × thickness of solid element *[default: PENMAX = 10.0]*
		shell	d = PENMAX × (tracked thickness + reference thickness) *[default: PENMAX = 10.0]*

Mapping of *CONTACT keyword option to "contact type" in d3hsp:

Structured Input Type ID	Keyword Name
a 13	AIRBAG_SINGLE_SURFACE
26	AUTOMATIC_GENERAL
i 26	AUTOMATIC_GENERAL_INTERIOR
a 5	AUTOMATIC_NODES_TO_SURFACE
t 5	AUTOMATIC_NODES_TO_SURFACE_TIEBREAK
a 10	AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE
t 10	AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIE-BREAK
13	AUTOMATIC_SINGLE_SURFACE
a 3	AUTOMATIC_SURFACE_TO_SURFACE
18	CONSTRAINT_NODES_TO_SURFACE
17	CONSTRAINT_SURFACE_TO_SURFACE
23	DRAWBEAD
16	ERODING_NODES_TO_SURFACE
14	ERODING_SURFACE_TO_SURFACE
15	ERODING_SINGLE_SURFACE
27	FORCE_TRANSDUCER_CONSTRAINT
25	FORCE_TRANSDUCER_PENALTY
m 5	FORMING_NODES_TO_SURFACE
m 10	FORMING_ONE_WAY_SURFACE_TO_SURFACE
m 3	FORMING_SURFACE_TO_SURFACE
5	NODES_TO_SURFACE
5	NODES_TO_SURFACE_INTERFERENCE
10	ONE_WAY_SURFACE_TO_SURFACE
20	RIGID_NODES_TO_RIGID_BODY

Structured Input Type ID	Keyword Name
21	RIGID_BODY_ONE_WAY_TO_RIGID_BODY
19	RIGID_BODY_TWO_WAY_TO_RIGID_BODY
22	SINGLE_EDGE
4	SINGLE_SURFACE
1	SLIDING_ONLY
p 1	SLIDING_ONLY_PENALTY
3	SURFACE_TO_SURFACE
3	SURFACE_TO_SURFACE_INTERFERENCE
8	TIEBREAK_NODES_TO_SURFACE
9	TIEBREAK_SURFACE_TO_SURFACE
6	TIED_NODES_TO_SURFACE
o 6	TIED_NODES_TO_SURFACE_OFFSET
c 6	TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET
7	TIED_SHELL_EDGE_TO_SURFACE or SPOTWELD
o 7	TIED_SHELL_EDGE_TO_SURFACE_OFFSET
c 7	TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET or SPOTWELD_CONSTRAINED_OFFSET
b 7	TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET or SPOTWELD_BEAM_OFFSET
s 7	SPOTWELD_WITH_TORSION
2	TIED_SURFACE_TO_SURFACE
o 2	TIED_SURFACE_TO_SURFACE_OFFSET
c 2	TIED_SURFACE_TO_SURFACE_CONSTRAINED_OFFSET

Contact Examples:

```
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONTACT_NODES_TO_SURFACE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Make a simple contact that prevents the nodes in part 2 from
$  penetrating the segments in part 3.
$
*CONTACT_NODES_TO_SURFACE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         2         3         3         3
$
$       fs        fd        dc        vc       vdc    penchk        bt        dt

$
$     sfsa      sfsb      sast      sbst     sfsat      sfsbt       fsf       vsf

$
$    surfatyp, surfbtyp = 3  id's specified in surfa and surfb are parts
$      surfa = 2 use nodes in part 2 in the contact
$      surfb = 3 use segments in part 3 in the contact
$
$  Use defaults for all parameters.
$
$$$$  Optional Cards A and B not specified (default values will be used).
$
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONTACT_SINGLE_SURFACE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Create a single surface contact between four parts: 28, 97, 88 and 92
$     - create a part set with set ID = 5, list the four parts
$     - in the *CONTACT_SINGLE_SURFACE definition specify:
$          surfatyp = 2  which means surfa is a part set
$          surfa = 5  use part set 5 for defining the contact surfaces
$
$  Additional contact specifications described below.
$
*CONTACT_SINGLE_SURFACE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         5                   2
$       fs        fd        dc        vc       vdc    penchk        bt        dt
      0.08      0.05        10                  20                        40.0
$     sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf

$
$       fs = 0.08  static coefficient of friction equals 0.08
$       fd = 0.05  dynamic coefficient of friction equals 0.05
$       dc =   10  exponential decay coefficient, helps specify the transition
$                     from a static slide to a very dynamic slide
$      vdc =   20  viscous damping of 20% critical (damps out nodal
```

```
$                     oscillations due to the contact)
$       dt = 40.0  contact will deactivate at 40 ms (assuming time unit is ms)
$
$$$$  Optional Cards A and B not specified (default values will be used).
$
$
*SET_PART_LIST
$      sid
         5
$     pid1      pid2      pid3      pid4
         28        97        88        92
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *CONTACT_DRAWBEAD
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a draw bead contact:
$    - the draw bead is to be made from the nodes specified in node set 2
$    - the reference segments are those found in the box defined by box 2
$        that are in part 18
$    - include SURFA and SURFB forces in interface file (sapr, sbpr = 1)
$
*CONTACT_DRAWBEAD
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         2        18         4         3                 2         1         1
$
$       fs        fd        dc        vc       vdc    penchk        bt        dt
      0.10
$
$     sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf

$
$$$$ Card 4 required because it's a drawbead contact
$
$  lcdidrf    lcidnf     dbdth     dfscl    numint
         3             0.17436       2.0
$
$  lcdidrf =       3  load curve 3 specifies the bending component of the
$                     restraining force per unit draw bead length
$    dbdth = 0.17436  draw bead depth
$    dfscl =     2.0  scale load curve 3 (lcdidrf) by 2
$
$$$$  Optional Cards A and B not specified (default values will be used).
$
*DEFINE_BOX
$    boxid       xmm       xmx       ymn       ymx       zmn       zmx
         2 0.000E+00 6.000E+00 6.000E+00 1.000E+02-1.000E+03 1.000E+03
$
*SET_NODE_LIST
$      sid       da1       da2       da3       da4
         2
$     nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
      2580      2581      2582      2583      2584      2585      2586      2587
      2588      2589      2590
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
         3
$                a                   o
$             DEPTH           FORC/LGTH
```

```
0.000E+00           0.000E+00
1.200E-01           1.300E+02
1.500E-01           2.000E+02
1.800E-01           5.000E+02
```

*CONTACT_ADD_WEAR

Purpose: Associate a wear model to a contact interface.

Wear is associated with friction, so the frictional coefficient *must* be nonzero for the associated contact interface. This feature calculates the wear depth, sliding distance and possibly user defined wear history variables according to the specified model which are then written to the intfor database (see *DATABASE_EXTENT_INTFOR) for post-processing. Note that this data is *not* written unless the parameter NWEAR and/or NWUSR are set on the *DATABASE_EXTENT_INTFOR card. *H*-adaptive remeshing is supported with this feature. Implicit analysis is supported, for which mortar is the preferred contact.

For positive CID values, this keyword does not affect the results of a simulation and only provides wear quantities for post-processing. For negative CID values, the contact surface is perturbed internally based on the wear depth so that the contact behavior is affected by the wear on the surface.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	WTYPE	P1	P2	P3	P4	P5	P6
Type	I	I	F	F	F	F	F	F
Default	none	0	none	none	↓	↓	↓	↓

User Defined Wear Parameter Cards. Define as many cards as needed to define P1 parameters if and only if WTYPE < 0.

Card 2	1	2	3	4	5	6	7	8
Variable	W1	W2	W3	W4	W5	W6	W7	W8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
CID	Contact interface ID, see *CONTACT_… LT.0: Perturb contact surface according to wear values (see Remark 5). GT.0: Calculate wear properties for post-processing only.

VARIABLE	DESCRIPTION
WTYPE	Wear law: LT.0: User defined wear law; value specifies type used in subroutine. EQ.0: Archard's wear law
P1	First wear parameter: WTYPE.EQ.0: Dimensionless scale factor k. If negative, the absolute value specifies a table ID with $k = k(p, \dot{d})$ as a function of contact pressure $p \geq 0$ and relative sliding velocity $\dot{d} \geq 0$. The table should be defined such that for each specified value of $\dot{d}$, there exists a curve such that k (ordinate) is a function of p (abscissa). WTYPE.LT.0: Number of user wear parameters for this interface
P2	Second wear parameter: WTYPE.EQ.0: SURFA surface hardness parameter H_A. If negative, the absolute value specifies a curve ID with $H_A = H_A(T_A)$ as function of SURFA node temperature T_A. WTYPE.LT.0: Number of user wear history variables per contact node. These can be output to the intfor file; see NWUSR on *DATABASE_EXTENT_INTFOR.
P3	Third wear parameter: WTYPE.EQ.0: SURFB surface hardness parameter H_B. If negative, the absolute value specifies a curve ID with $H_B = H_B(T_B)$ as function of SURFB node temperature T_B. WTYPE.LT.0: Not used
P4 - P6	Not used
Wn	nth user defined wear parameter.

Remarks:

1. **Archard's Wear Law.** Archard's wear law (WTYPE = 0) states that the wear depth w at a contact point evolves with time as

$$\dot{w} = k\frac{p\dot{d}}{H}$$

where $k > 0$ is a dimensionless scale factor, $p \geq 0$ is the contact interface pressure, $\dot{d} \geq 0$ is the relative sliding velocity of the points in contact and $H > 0$ is the surface hardness (force per area). The wear depth for a node in contact is incremented in accordance with this formula, accounting for the different hardness of the SURFA and SURFB side, H_A and H_B, respectively. By using negative numbers for wear parameters P1, P2 or P3, the corresponding parameter is defined by a table or a curve. For P1, the value of k is taken from a table with contact pressure p and sliding velocity $\dot{d}$ as arguments, while for P2 or P3, the corresponding hardness H is taken from curves with the associated contact nodal temperature T as argument. That is, the SURFA side hardness will be a function of the SURFA side temperature, and vice versa.

2. **User Defined Wear Laws.** Customized wear laws may be specified as a user-defined subroutine called userwear. This subroutine is called when WTYPE < 0. This subroutine is passed wear parameters (see P1) for this interface as well as number of wear history variables (see P2) per contact node. The wear parameters are defined on additional cards (see WN) and the history variables are updated in the user subroutine. The history variables can be output to the intfor file, see NWUSR on *DATABASE_EXTENT_INTFOR. WTYPE may be used to distinguish between different wear laws, and consequently any number of different laws can be implemented within the same subroutine. For more information, we refer to the source code which contains extensive commentaries and two sample wear laws.

3. **Using Wear Laws.** Only one wear law per contact interface can be specified. The procedure for activating this feature involves

   a) Using the present keyword to associate wear to a contact interface

   b) Setting NWEAR and/or NWUSR on the *DATABASE_EXTENT_INTFOR card.

   c) Having a contact interface with friction of a type that is supported. If SOFT = 2 on optional card A of the contact data, then any valid keyword option is supported. If SOFT = 0 or SOFT = 1, then the following list is supported.

      *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE

*CONTACT_FORMING_SURFACE_TO_SURFACE

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE

*CONTACT_FORMING_SURFACE_TO_SURFACE_MORTAR

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR

*CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR

4. **Limitations.** The following are not supported with this keyword.

   a) _SMOOTH option

   b) MPP "groupable" option

5. **Negative CID Option.** For this option, the wear is accounted for by perturbing the contact surface internally (not by updating the node positions). As such, no changes to the element geometry will be observed due to wear, however increased part penetrations and stress redistributions will occur, as well as redistribution of contact pressure and wear depth in the intfor results. The method is only physically justified for relatively small wear depths. Care must therefore be taken in selecting the wear parameters to ensure that the wear depth does not exceed the maximum penetration associated with the contact. This option is available for SMP, MPP and Hybrid versions when SOFT = 2 on optional card A of the *CONTACT card but is only available for the SMP version when SOFT = 0 or 1.

*CONTACT_AUTO_MOVE

Purpose: Automatically move the surfb surface in a contact definition to close an unspecified gap between the surfa and surfb surfaces. The gap may result from loading the surfa part with an initial gravity load. The gap will be closed at a specified time to save CPU time. The surfb surface in metal forming applications is typically the upper cavity while the surfa part is the blank. This feature is only applicable to sheet metal forming application.

Cards 1	1	2	3	4	5	6	7	8
Variable	ID	CONTID	VID	LCID	ATIME	OFFSET		
Type	I	I	I	I	F	F		
Default	none	none	none	0	0.0	0.0		

VARIABLE	DESCRIPTION
ID	Move ID for this automatic move input: GT.0: Velocity controlled tool kinematics (the variable VAD = 0 in *BOUNDARY_PRESCRIBED_MOTION_RIGID) LT.0: Displacement controlled tool kinematics (VAD = 2)
CONTID	Contact ID, as in *CONTACT_FORMING_..._ID, which defines the surfa and surfb part set IDs.
VID	Vector ID of a vector oriented in the direction of movement of the surfb surface, as in *DEFINE_VECTOR. The origin of the vector is unimportant since the direction cosines of the vector are computed and used.
LCID	Load curve defining tooling kinematics, either by velocity as a function of time or by displacement as a function of time. This load curve will be adjusted automatically during a simulation to close the empty tool travel.
ATIME	Activation time specifying the moment the surfb surface (tool) will be moved
OFFSET	Time at which a surfb surface will move to close a gap distance, which may happen following the move of another surfb surface. This is useful for sequential multiple flanging or press hemming

VARIABLE	DESCRIPTION
	simulations. Simulation time (CPU) is much faster based on the shortened tool travel (no change to the termination time).

Example: gravity loading and closing with implicit static

Referring to the partial input deck below and Figure 11-28, a combined simulation of gravity loading and binder closing of a fender outer is demonstrated on the NUMISHEET 2002 benchmark. In this multistep implicit static set up, the blank is allocated 0.3 "time" units (3 implicit steps for DT0 = 0.1) to be loaded with gravity. At the end of gravity loading, a gap of 12mm was created between the upper die and the blank (see Figure 11-29). The upper die is set to be moved at 0.3 "time" units, closing the gap caused by the gravity effect on the blank (see Figure 11-30 left). An intermediate closing state is shown at $t = 0.743$ (see Figure 11-30 right) while the final completed closing is shown in Figure 11-31. It is noted that the upper die is controlled with displacement (VAD = 2) in the shape of a right triangular in the displacement versus "time" space as defined by load curve #201, and the ID in *CONTACT_AUTO_MOVE is set to "-1".

```
*PARAMETER
R grvtime       0.3
R endtime       1.0
R diemv      145.45
*CONTROL_TERMINATION
&endtime
*CONTROL_IMPLICIT_FORMING
2,2,100
*CONTROL_IMPLICIT_GENERAL
$   IMFLAG        DT0
         1      0.10
*CONTROL_ACCURACY
         1         2
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
11
....
....
....
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$#     pid       dof       vad      lcid        sf       vid     death     birth
         2         3         2       201 -1.000000         0       0.0     0.000
*CONTACT_AUTO_MOVE
$       ID    ContID       VID      LCID     ATIME
        -1        11        89       201  &grvtime
*DEFINE_VECTOR
89,0.0,0.0,0.0,0.0,0.0,-10.0
*DEFINE_CURVE
201
0.0,0.0
&grvtime,0.0
1.0,&diemv
```

Similarly, "velocity" controlled tool kinematics is also enabled. In the example keyword below, the "velocity" profile is ramped up initially and then kept constant. Note that the variable VAD in *BOUNDARY is set to "0", and ID in *CONTACT_AUTO_MOVE is set to positive "1" indicating it is a velocity boundary condition.

```
*PARAMETER
R grvtime       0.3
R tramp       0.001
R diemv      145.45
R clsv       1000.0
*PARAMETER_EXPRESSION
R tramp1  tramp+gravtime
R endtime tramp1+(abs(diemv)-0.5*clsv*tramp)/clsv
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
11
....
....
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$#     pid       dof       vad      lcid        sf       vid     death     birth
         2         3         0       201 -1.000000         0       0.0     0.000
*CONTACT_AUTO_MOVE
$       ID    ContID       VID      LCID     ATIME
         1        11        89       201  &grvtime
*DEFINE_VECTOR
89,0.0,0.0,0.0,0.0,0.0,-10.0
*DEFINE_CURVE
201
0.0,0.0
0.2,0.0
&tramp1,&clsv
&endtime,&clsv
```

Example: tool delay in sequential flanging process with explicit dynamic:

The following example demonstrates the use of the variable OFFSET. As shown in Figure 11-32 (left), a total of 5 flange steels are auto-positioned initially according to the initial blank shape. Upon closing the pressure pad, the first set of 4 flanging steels moves to home, completing the first stage of the stamping process (see Figure 11-32 right).

The gap created by the completion of the first flanging process is closed automatically at a time defined using variables ATIME/OFFSET (see Figure 11-33 left). During the second stage of the process, flanging steel *&flg5pid* moves to home completing the final flanging (see Figure 11-33 right). An excerpt from the input deck for this model can be found below. This deck was created using LS-PrePost's eZ-Setup feature, with two additional keywords added:*CONTACT_AUTO_MOVE and *DEFINE_VECTOR.

Flanging steel #5 is set to move in a cam angle defined by vector #7 following the completion of the flanging (straight down) process of flanging steel #2. The variables ATIME and OFFSET in *CONTACT_AUTO_MOVE are both defined as &endtim4, which is calculated based on the automatic positioning of tools/blank using *CONTROL_FORMING_AUTOPOSITION. At a defined time, flanging steel #5 'jumps' into position to where it just comes into contact with the partially formed down-standing flange, saving some CPU times (see Figure 11-33 left). Flanging steel #5 continues to move to its home position completing the simulation (see Figure 11-33 right). The CPU time saving is 27% in this case.

```
*KEYWORD
```

```
*PARAMETER
...
*PART

  &flg5pid  &flg5sec  &flg5mid
...
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$ Local coordinate system for flanging steel #5 move direction
*DEFINE_COORDINATE_SYSTEM
$#     cid        xo        yo        zo        xl        yl        zl
  &flg5cid  -5.09548   27.6584  -8.98238  -5.43587   26.8608  -9.48034
$#      xp        yp        zp
  -5.82509   27.5484  -8.30742
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$ Auto positioning
*CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET
$      SID       CID       DIR      MPID  POSITION   PREMOVE     THICK  PARORDER
...
  &flg5sid  &flg5cid         3  &blk1sid        -1             &bthick    flg5mv
*PART_MOVE
$    PID            XMOV            YMOV            ZMOV     CID   IFSET
&flg5sid             0.0             0.0         &flg5mv&flg5cid       1
...
*MAT_RIGID
$      MID        RO         E        PR         N    COUPLE         M     ALIAS
  &flg5mid 7.830E-09 2.070E+05      0.28
$      CMO      CON1      CON2
        -1  &flg5cid    110111
$LCO or A1        A2        A3        V1        V2        V3
  &flg5cid
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTACT_AUTO_MOVE
$       ID    CONTID       VID      LCID     ATIME    OFFSET
         1         7         7        10  &endtim4  &endtim4
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH
         7       0.0       0.0       0.0-0.5931240 0.5930674-0.5444952
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$      CID
         7
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
  &blk1sid  &flg5sid         2         2                             1         1
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
...
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$                    Tool kinematics
$  -------------------------closing
*BOUNDARY_PRESCRIBED_MOTION_RIGID_local
...
  &flg5pid         3         0         4       1.0         0  &endtim4
$  -------------------------flanging
*BOUNDARY_PRESCRIBED_MOTION_RIGID_local
...
  &flg5pid         3         0        10       1.0         0            &endtim4
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*END
```

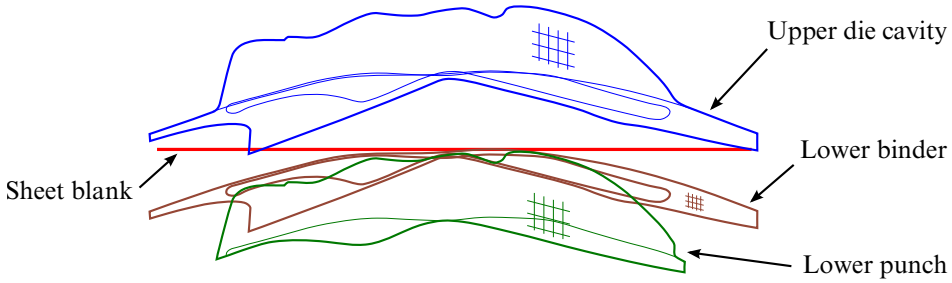


Figure 11-28. Initial parts auto-positioned at $t = 0.0$.

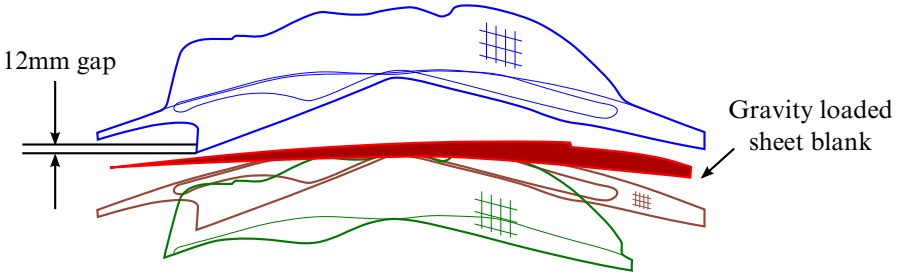


Figure 11-29. Gravity loading on blank at $t = 0.2$.

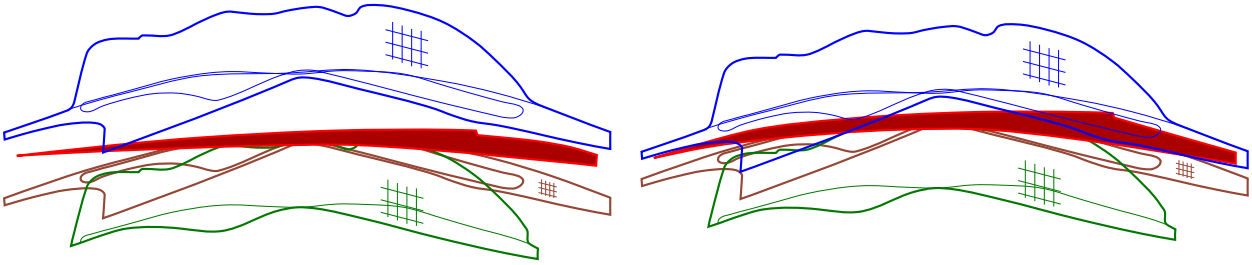


Figure 11-30. Upper die moved down at $t = 0.3$ to close the gap (left); continue closing at $t = 0.743$ (right).

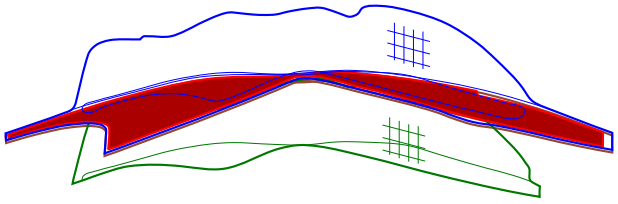


Figure 11-31. Closing complete at $t = 1.0$.

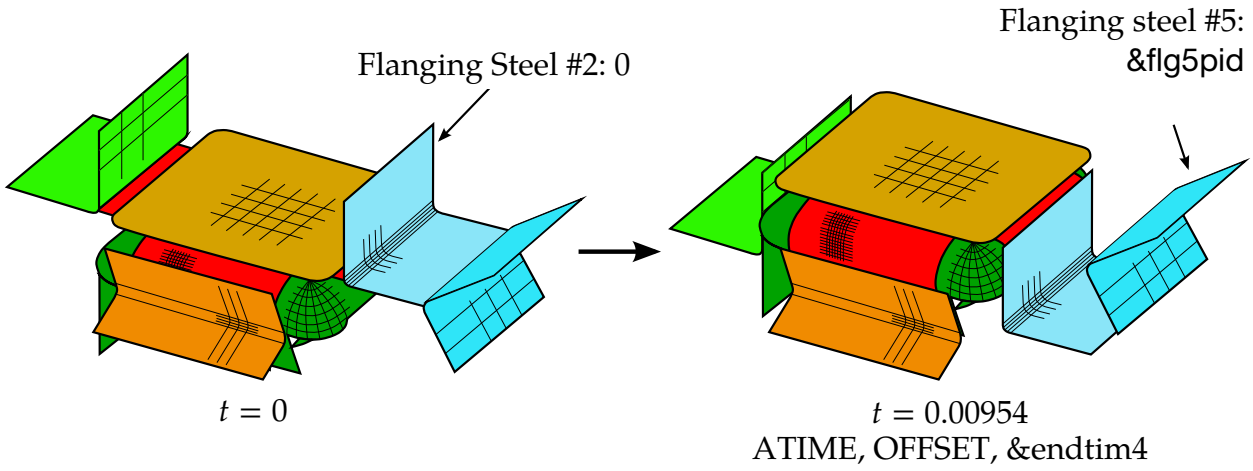


Figure 11-32. A sequential flanging process (left); first set of flanging steels reaching home (right).

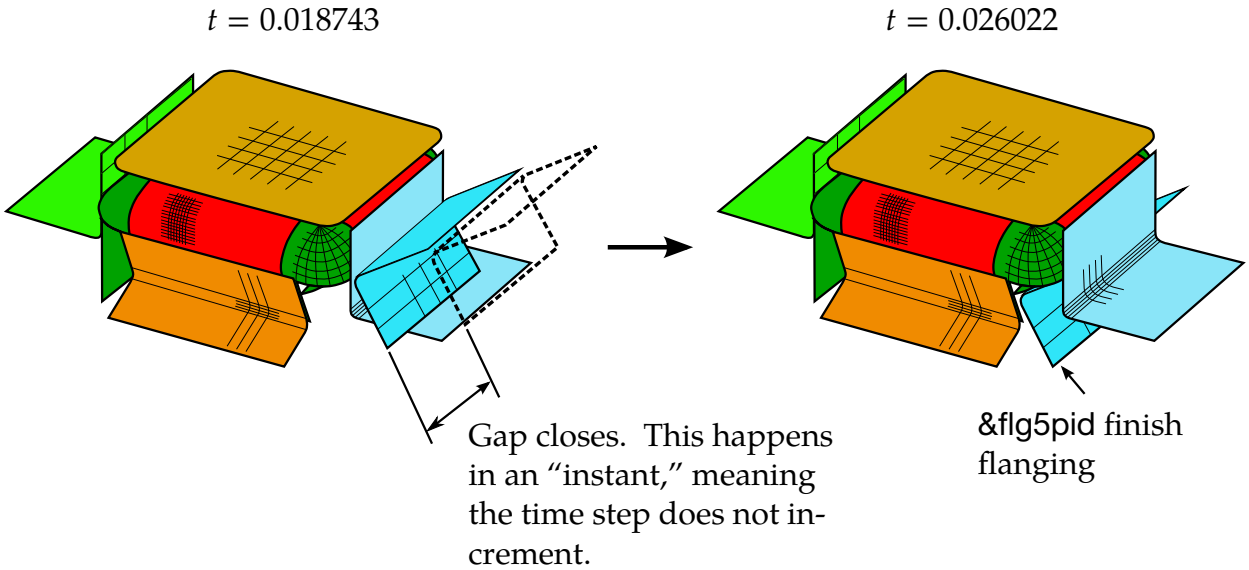


Figure 11-33. Closing the empty travel (left); flanging steel &flg5pid completes flanging process (right).

*CONTACT_COUPLING

Purpose: Define a coupling surface for MADYMO to couple LS-DYNA with deformable and rigid parts within MADYMO. In this interface, MADYMO computes the contact forces acting on the coupling surface, and LS-DYNA uses these forces in the update of the motion of the coupling surface for the next time step. Contact coupling can be used with other coupling options in LS-DYNA.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Set Cards. Include on card for each coupled set. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	SID	STYPE						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
SID	Set ID for coupling. See Remark 1 below.
STYPE	Set type: EQ.0: part set EQ.1: shell element set EQ.2: solid element set EQ.3: thick shell element set

Remarks:

1. **Coupling Surface.** Only one coupling surface can be defined. If additional surfaces are defined, the coupling information will be added to the first definition.

2. **Units.** The units and orientation can be converted by using the *CONTROL_COUPLING keyword. It is not necessary to use the same system of units in MADYMO and in LS-DYNA if unit conversion factors are defined.

*CONTACT_ENTITY

Purpose: Define a contact entity. Geometric contact entities treat the impact between a deformable body, defined as a set of tracked nodes or nodes in a shell part set, and a rigid body. The shape of the rigid body is determined by attaching geometric entities. Contact between these geometric entities and the tracked nodes is a penalty formulation. The penalty stiffness is optionally maximized within the constraint of the Courant criterion. As an alternative, a finite element mesh made with shells can be used as geometric entity. Also, axisymmetric entities with arbitrary shape made with multi-linear polygons are possible. The latter is particularly useful for metal forming simulations. See *DATABASE_GCEOUT for contact force output.

> **WARNING:** If the problem being simulated involves dynamic motion of the entity, care should be taken to ensure that the inertial properties of the entity are correct. These properties may need to be specified with *PART_INERTIA.

Card Summary:

Card 1. This card is required.

PID	GEOTYP	SURFA	SURFATYP	SF	DF	CF	INTORD

Card 2. This card is required.

BT	DT	SO	GO	ITHK	SAPR		

Card 3. This card is required.

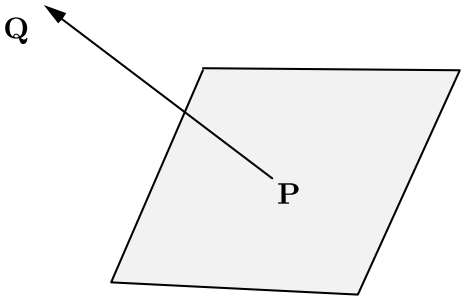
XC	YC	ZC	AX	AY	AZ		

Card 4. This card is required.

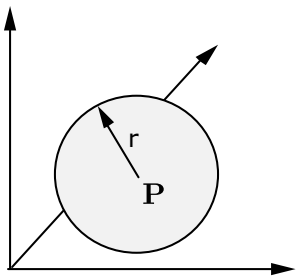
BX	BY	BZ					

Card 5. This card is required.

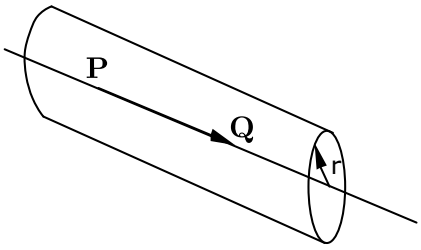
INOUT	G1	G2	G3	G4	G5	G6	G7



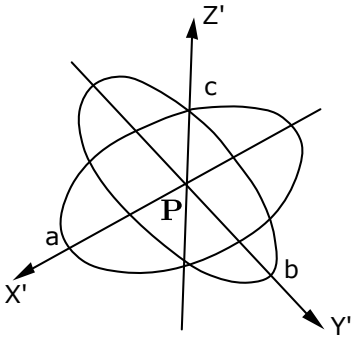
GEOTYP = 1: Infinite Plane



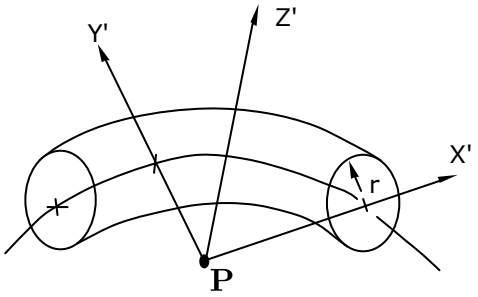
GEOTYP = 2: Sphere



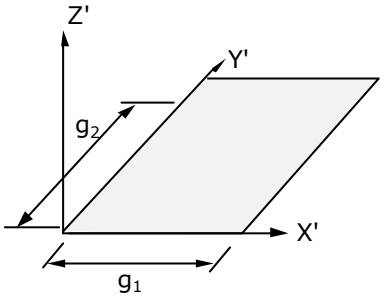
GEOTYP = 3: Infinite Cylinder



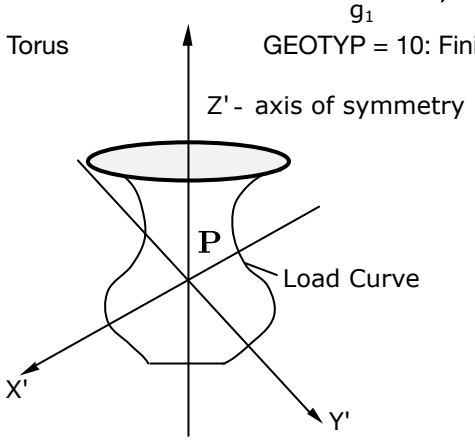
GEOTYP = 4: Hyperellipsoid



GEOTYP = 5: Torus



GEOTYP = 10: Finite Plane



GEOTYP = 11: Load Curve

Figure 11-34. Contact Entities

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PID	GEOTYP	SURFA	SURFATYP	SF	DF	CF	INTORD
Type	I	I	I	I	F	F	F	I
Default	none	none	none	0	1.	0.	0.	0

VARIABLE	DESCRIPTION
PID	Part ID of the rigid body to which the geometric entity is attached, see *PART.
GEOTYP	Type of geometric entity (see Figure 11-34): EQ.1: Plane, EQ.2: Sphere, EQ.3: Cylinder, EQ.4: Ellipsoid, EQ.5: Torus, EQ.6: CAL3D/MADYMO Plane, see Appendix I, EQ.7: CAL3D/MADYMO Ellipsoid, see Appendix I, EQ.8: VDA surface, see Appendix L, EQ.9: Rigid body finite element mesh (shells only), EQ.10: Finite plane, EQ.11: Load curve defining line as surface profile of axisymmetric rigid bodies.
SURFA	Tracked surface set ID, see *SET_NODE_*OPTION*, *PART, or *SET_PART.
SURFATYP	Tracked surface set type: EQ.0: Node set EQ.1: Part EQ.2: Part set

VARIABLE	DESCRIPTION
SF	Penalty scale factor. Useful to scale maximized penalty.
DF	Damping option (see description for *CONTACT_*OPTION*): EQ.0: No damping, GT.0: Viscous damping in percent of critical, e.g., 20 for 20% damping, LT.0: DF must be a negative integer. -DF is the load curve ID giving the damping force as a function of relative normal velocity (see Remark 1 below).
CF	Coulomb friction coefficient. See Remark 2 below. EQ.0: No friction GT.0: Constant friction coefficient LT.0: CF must be a negative integer. -CF is the load curve ID giving the friction coefficient as a function of time.
INTORD	Integration order (tracked materials only). This option is not available with entity types 8 or 9 where only nodes are checked: EQ.0: check nodes only, EQ.1: 1 point integration over segments, EQ.2: 2×2 integration, EQ.3: 3×3 integration, EQ.4: 4×4 integration, EQ.5: 5×5 integration. This option allows a check of the penetration of the rigid body into the deformable (tracked) material. Then virtual nodes at the location of the integration points are checked.

Card 2	1	2	3	4	5	6	7	8
Variable	BT	DT	SO	GO	ITHK	TPR		
Type	F	F	I	I	I	I		
Default	0.	10^{20}	0	0	0	0		

VARIABLE	DESCRIPTION
BT	Birth time
DT	Death time
SO	Flag to use penalty stiffness as in surface-to-surface contact: EQ.0: Contact entity stiffness formulation, EQ.1: Surface to surface contact method, EQ.2: Normal force is computed via a constraint-like method. The contact entity is considered to be infinitely massive, so this is recommended only for entities with constrained motion. LT.0: SO must be an integer. -SO is the load curve ID giving the force versus the normal penetration. See Remark 1.
GO	Flag for automatic meshing of the contact entity for entity types 1 - 5 and 10 - 11. GO = 1 creates null shells for visualization of the contact entity. Note these shells have mass and will affect the mass properties of the rigid body PID unless *PART_INERTIA is used for the rigid body. EQ.0: Mesh is not generated. EQ.1: Mesh is generated.
ITHK	Flag for considering thickness for shell tracked nodes (applies only to entity types 1, 2, 3; SURFATYP must be set to zero). EQ.0: Shell thickness is not considered. EQ.1: Shell thickness is considered.
SAPR	Include the tracked side in *DATABASE_BINARY_INTFOR interface force files; valid only when SURFATYP > 0: EQ.1: Tracked side forces included.

Card 3	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	AX	AY	AZ		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0		

Card 4	1	2	3	4	5	6	7	8
Variable	BX	BY	BZ					
Type	F	F	F					
Default	0.	0.	0.					

VARIABLE	DESCRIPTION
XC	x-center, x_c. See Remark 3.
YC	y-center, y_c. See Remark 3.
ZC	z-center, z_c. See Remark 3.
AX	x-direction for local axis **A**, A_x. See Remark 3.
AY	y-direction for local axis **A**, A_y. See Remark 3.
AZ	z-direction for local axis **A**, A_z. See Remark 3.
BX	x-direction for local axis **B**, B_x. See Remark 3.
BY	y-direction for local axis **B**, B_y. See Remark 3.
BZ	z-direction for local axis **B**, B_z. See Remark 3.

Card 5	1	2	3	4	5	6	7	8
Variable	INOUT	G1	G2	G3	G4	G5	G6	G7
Type	I	F	F	F	F	F	F	F
Default	0	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
INOUT	In-out flag. Allows contact from the inside or the outside (default) of the entity: EQ.0: Tracked nodes exist outside of the entity. EQ.1: Tracked nodes exist inside the entity.
G1	Entity coefficient g_1 (CAL3D/MADYMO plane or ellipse number) for coupled analysis (see Appendix I). See Geometric Contact Entity Coefficients below.
G2	Entity coefficient g_2. See Geometric Contact Entity Coefficients below.
G3	Entity coefficient g_3. See Geometric Contact Entity Coefficients below.
G4	Entity coefficient g_4. See Geometric Contact Entity Coefficients below.
G5	Entity coefficient g_5. See Geometric Contact Entity Coefficients below.
G6	Entity coefficient g_6. See Geometric Contact Entity Coefficients below.
G7	Entity coefficient g_7. See Geometric Contact Entity Coefficients below.

Remarks:

1. **Damping and Contact Load Curves.** The optional load curves that are defined for damping versus relative normal velocity and for force versus normal penetration should be defined in the positive quadrant. The sign for the damping

force depends on the direction of the relative velocity and the treatment is symmetric if the damping curve is in the positive quadrant. If the damping force is defined in the negative and positive quadrants, the sign of the relative velocity is used in the table look-up.

2. **Friction Coefficient.** If at any time the friction coefficient is ≥ 1.0, the force calculation is modified to a constraint like formulation which allows no sliding. This is only recommended for entities with constrained motion since the mass of the entity is assumed to be infinite.

3. **Geometric Entity Coordinate System.** The coordinates, (x_c, y_c, z_c), are the positions of the local origin of the geometric entity in global coordinates. The entity's local A-axis is determined by the vector (A_x, A_y, A_z) and the local B-axis by the vector (B_x, B_y, B_z).

   Cards 3 and 4 define a local to global transformation. The geometric contact entities are defined in a local system and transformed into the global system. For the ellipsoid, this is necessary because it has a restricted definition for the local position. For the plane, sphere, and cylinder, the entities can be defined in the global system and the transformation becomes $(x_c, y_c, z_c) = (0,0,0)$, $(A_x, A_y, A_z) = (1,0,0)$, and $(B_x, B_y, B_z) = (0,1,0)$.

Geometric Contact Entity Coefficients:

Figure 11-34 shows the definitions of the geometric contact entities. The relationships between the entity coefficients and the Figure 11-34 variables are as described below. Note that (P_x, P_y, P_z) defines a point and (Q_x, Q_y, Q_z) is a direction vector.

GEOTYP = 1

$g_1 = P_x$
$g_2 = P_y$
$g_3 = P_z$
$g_4 = Q_x$
$g_5 = Q_y$
$g_6 = Q_z$
$g_7 = L$

If automatic generation is used, a square plane of length L on each edge is generated which represents the infinite plane. If generation is inactive, then g_7 may be ignored.

GEOTYP = 2

$g_1 = P_x$
$g_2 = P_y$
$g_3 = P_z$
$g_4 = r$

GEOTYP = 3

$g_1 = P_x$
$g_2 = P_y$
$g_3 = P_z$
$g_4 = Q_X$
$g_5 = Q_y$
$g_6 = Q_z$
$g_7 = r$

If automatic generation is used, a cylinder of length $\sqrt{Q_x^2 + Q_y^2 + Q_z^2}$ and radius r is generated which represents the infinite cylinder. The midpoint of the displayed cylinder is at point P.

GEOTYP = 4

$g_1 = P_x$
$g_2 = P_y$
$g_3 = P_z$
$g_4 = a$
$g_5 = b$
$g_6 = c$
$g_7 = n$ (order of the ellipsoid)

GEOTYP = 5

g_1 = Radius of torus
$g_2 = r$
g_3 = number of elements along minor circumference
g_4 = number of elements along major circumference

GEOTYP = 8

g_1 = Blank thickness (option to override true thickness)
g_2 = Scale factor for true thickness (optional)
g_3 = Load curve ID defining thickness versus time. (optional)

GEOTYP = 9

g_1 = Shell thickness (option to override true thickness).
g_2 = Scale factor for true thickness (optional)
g_3 = Load curve ID defining thickness versus time. (optional)

Note: The shell thickness specification is necessary if the tracked surface is generated from solid elements.

GEOTYP = 10

g_1 = Length of edge along X' axis
g_2 = Length of edge along Y' axis

GEOTYP = 11

g_1 = Load curve ID defining axisymmetric surface profile about the Z-axis. Load curves defined by the keywords *DEFINE_CURVE or *DEFINE_CURVE_ENTITY can be used.

g_2 = Number of elements along circumference

EQ.0: default set to 10

g_3 = Number of elements along axis

EQ.0: default set to 20

EQ.-1: the elements generated from points on the load curve

g_4 = Number of sub divisions on load curve used to calculate contact

EQ.0: default set to 1000

*CONTACT_EXCLUDE_INTERACTION

Purpose: Specify portions of a contact interface to exclude from having a contact interaction. The portions of the interface are specified with sets. If a single set is given, the contact algorithm excludes all contact with segments in that set. If two sets are defined, the contact algorithm excludes only contact between segments in set A from contact with segments in set B. This option is only available for the segment-to-segment contact that is invoked by setting SOFT = 2 on optional Card A of *CONTACT.

Card 1	1	2	3	4	5	6	7	8
Variable	CEID	CID						
Type	I	I						
Default	none	none						

Set Cards. Include as many lines as needed to specify different portions of the interface to be excluded from contact. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	SIDA	SIDB	TYPEA	TYPEB				
Type	I	I	I	I				
Default	0	0						

VARIABLE	DESCRIPTION
CEID	Contact exclusion ID for output only
CID	Contact interface ID for limiting exclusions to an interface
SIDA	Set ID of set A
SIDB	Set ID of set B
TYPEA	ID type of set SIDA: EQ.0: Segment set EQ.1: Shell element set

VARIABLE	DESCRIPTION
	EQ.2: Part set
TYPEB	ID type of set SIDB: EQ.0: Segment set EQ.1: Shell element set EQ.2: Part set

Remarks:

There are two ways to use this keyword. If both SIDA and SIDB are defined, interface CID excludes from contact all segment pairs where one segment is in set SIDA and the other in set SIDB. The second way is to define either SIDA or SIDB but not both. When this is done, contact interface CID excludes from contact any segment pair if either segment is in SIDA or SIDB.

During initialization, segments in SIDA or SIDB are compared with SURFA and SURFB, and a list of matches is made for exclusion. If CID is a single-surface type contact, SURFA is searched for matches with either SIDA or SIDB or both. If CID is a surface-to-surface type contact, SURFA is searched for matches with SIDA, and SURFB is searched for matches with SIDB.

CONTACT_FORCE_TRANSDUCER_*OPTION1_{OPTION2}

Purpose: Create a force transducer to measure the contact force of a 3D contact. LS-DYNA only outputs data between a SURFA and SURFB surface. Self-contact, such as single surface contact, only has a SURFA definition. In the case of self-contact, force transducers provide a means to record the self-contact forces. Force transducers are not needed for other types of contact. However, they may be useful for extracting the contact forces for any part of the model. For instance, if you have two-way contact, you may only want to extract the forces on a section of SURFB. Force transducers do not apply any force to the model nor do they have any effect on the solution.

This keyword is *not* implemented for 2D contact.

Available options for *OPTION1* are:

CONSTRAINT

PENALTY

The CONSTRAINT option extracts forces from constraint-based contact types while the PENALTY option extracts forces only from penalty-based contacts.

Available options for *OPTION2* are:

ID

In the output files, the contact ID identifies the results from the force transducer measurements. If the contact ID is undefined, the default ID is determined by the sequence of the contact definitions, that is, the first contact definition has an ID of 1, the second, 2, and so forth. The ID and heading are picked up by some of the peripheral LS-DYNA codes to aid in post-processing.

Card ID. Additional card for ID keyword option

Card ID	1	2	3	4	5	6	7	8
Variable	CID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SURFA	SURFB	SURFATYP	SURFBTYP	SABOXID	SBBOXID	SAPR	SBPR
Type	I	I	I	I	I	I	I	I
Default	none	optional	none	optional	optional	optional	0	0

This card must be included as a blank line.

Card 2	1	2	3	4	5	6	7	8
Variable								
Type								

This card must be included as a blank line.

Card 3	1	2	3	4	5	6	7	8
Variable								
Type								

Optional Card A. Include this blank card if defining optional Card C.

Card A	1	2	3	4	5	6	7	8
Variable								
Type								

Optional Card B. Include this blank card if defining optional Card C.

Card B	1	2	3	4	5	6	7	8
Variable								
Type								

Optional Card C. This card is optional.

Card C	1	2	3	4	5	6	7	8
Variable								CID_RCF
Type								I
Default								0

VARIABLE	DESCRIPTION
CID	Contact interface ID. This must be a unique number.
HEADING	Interface descriptor. We suggest using unique descriptions.
SURFA	Segment set ID, node set ID, part set ID, part ID, or shell element set ID specifying the SURFA side of the force transducer; see *SET_SEGMENT, *SET_NODE_*OPTION*, *PART, *SET_PART or *SET_SHELL_*OPTION*. EQ.0: Includes all parts.
SURFB	Segment set ID, node set ID, part set ID, part ID, or shell element set ID specifying the SURFB side of the force transducer. A SURFB set is not required for force transducers, See Remark 1.
SURFATYP	ID type of SURFA: EQ.0: Segment set ID EQ.1: Shell element set ID EQ.2: Part set ID EQ.3: Part ID EQ.4: Node set ID. See Remark 2.

VARIABLE	DESCRIPTION
	EQ.5: Include all (SURFA is ignored) EQ.6: Part set ID for exempted parts. All non-exempted parts are included in the force transducer. EQ.7: Branch ID; see *SET_PART_TREE.
SURFBTYP	ID type of SURFB: EQ.0: Segment set ID EQ.1: Shell element set ID EQ.2: Part set ID EQ.3: Part ID EQ.4: Node set ID. See Remark 2. EQ.5: Include all (SURFB is ignored). EQ.6: Part set ID for exempted parts. All non-exempted parts are included in the force transducer. EQ.7: Branch ID; see *SET_PART_TREE.
SABOXID	Include in force transducer definition only those SURFA nodes/segments within box SABOXID (corresponding to BOXID in *DEFINE_BOX), or if SABOXID is negative, only those SURFA nodes/segments within contact volume |SABOXID| (corresponding to CVID in *DEFINE_CONTACT_VOLUME). SABOXID can be used only if SURFATYP is set to 2 or 3, that is, SURFA is a part ID or part set ID
SBBOXID	Include in force transducer definition only those SURFB segments within box SBBOXID (corresponding to BOXID in *DEFINE_BOX), or if SBBOXID is negative, only those SURFB segments within contact volume |SBBOXID| (corresponding to CVID in *DEFINE_CONTACT_VOLUME). SBBOXID can be used only if SURFBTYP is set to 2 or 3, that is, SURFB is a part ID or part set ID.
SAPR	Include the SURFA side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files, and optionally in the dynain file for wear: EQ.0: Do not include. EQ.1: SURFA side forces included. EQ.2: Same as 1 but also allows for SURFA nodes to be written as *INITIAL_CONTACT_WEAR to dynain; see NCYC on

VARIABLE	DESCRIPTION
	*INTERFACE_SPRINGBACK_LSDYNA.
SBPR	Include the SURFB side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files, and optionally in the dynain file for wear: EQ.0: Do not include. EQ.1: SURFB side forces included. EQ.2: Same as 1, but also allows for SURFB nodes to be written as *INITIAL_CONTACT_WEAR to dynain; see NCYC on *INTERFACE_SPRINGBACK_LSDYNA.
CID_RCF	Coordinate system ID to output rcforc force resultants and ncforc data in a local system

Remarks:

1. **SURFB set with force transducers.** A SURFB set ID (SURFB and SURFBTYP) is optional for force transducers. If the contact forces acting between two particular surfaces are needed, a SURFB surface should be defined. In this case, only the contact forces applied between SURFA and SURFB are recorded. Otherwise, the contact forces between SURFA and any surface are output. If a transducer is used for extracting forces from Mortar contacts, the SURFA and SURFB sides *must* be defined through parts or part sets; segment and node sets will not gather the correct data.

> **NOTE:** The SURFB surface option of FORCE_TRANSDUCER is only implemented for the PENALTY option. It works only in conjunction with Mortar contacts, AUTOMATIC_SURFACE_TO_SURFACE, ERODING_SINGLE_SURFACE, AUTOMATIC_SINGLE_SURFACE contact types, and groupable AUTOMATIC_GENERAL contact types (MPP only), except as noted in the next remark.

2. **SURFA/SURFB node sets.** Using node sets to specify the surfaces requires the PENALTY keyword option. Additionally, if either SURFA or SURFB is a node set, both must be node sets. Node sets allow the transducer to give correct results for eroding materials. The node sets should include all nodes that may be exposed as erosion occurs.

3. **Force output.** Contact force data can be recorded with *DATABASE_RCFORC, *DATABASE_NCFORC, and *DATABASE_BINARY_INTFOR. The contact energy data for force transducers can be recorded with *DATABASE_SLEOUT, but this data is only recorded for MPP.

CONTACT_GEBOD_*OPTION

Purpose: Define contact interaction between the segment of a GEBOD dummy and parts or nodes of the finite element model. This implementation follows that of the contact entity, however, it is specialized for the dummies. Forces may be output using the *DATABASE_GCEOUT command. See *COMPONENT_GEBOD and Appendix N for further details.

Conventional *CONTACT_*OPTION* treatment (surface-to-surface, nodes-to-surface, etc.) can also be applied to the segments of a dummy. To use this approach, the part ID assignments must be first determined by running the model through LS-DYNA's initialization phase.

The following options are available and refer to the ellipsoids which comprise the dummy. Options involving HAND are *not* applicable for the child dummy since its lower arm and hand share a common ellipsoid.

LOWER_TORSO
MIDDLE_TORSO
UPPER_TORSO
NECK
HEAD
LEFT_SHOULDER
RIGHT_SHOULDER
LEFT_UPPER_ARM
RIGHT_UPPER_ARM
LEFT_LOWER_ARM
RIGHT_LOWER_ARM
LEFT_HAND
RIGHT_HAND
LEFT_UPPER_LEG
RIGHT_UPPER_LEG
LEFT_LOWER_LEG
RIGHT_LOWER_LEG
LEFT_FOOT
RIGHT_FOOT

Card 1	1	2	3	4	5	6	7	8
Variable	DID	SURFA	SURFATYP	SF	DF	CF	INTORD	
Type	I	I	I	F	F	F	I	
Default	none	none	none	1.	20.	0.5	0	

VARIABLE	DESCRIPTION
DID	Dummy ID, see *COMPONENT_GEBOD_*OPTION*.
SURFA	Tracked surface set ID, see *SET_NODE_*OPTION*, *PART, or *SET_PART.
SURFATYP	SURFA set type: EQ.0: Node set EQ.1: Part EQ.2: Part set
SF	Penalty scale factor. Useful to scale maximized penalty.
DF	Damping option, see description for *CONTACT_*OPTION*: EQ.0.0: No damping, GT.0.0: Viscous damping in percent of critical, e.g., 20 for 20% damping, LT.0.0: DF must be an integer. -DF is the load curve ID giving the damping force as a function of relative normal velocity (see Remark 1 below).
CF	Coulomb friction coefficient (see Remark 2 below). Assumed to be constant.
INTORD	Integration order (tracked materials only). EQ.0: Check nodes only, EQ.1: 1 point integration over segments, EQ.2: 2×2 integration, EQ.3: 3×3 integration, EQ.4: 4×4 integration, EQ.5: 5×5 integration. This option allows a check of the penetration of the dummy segment into the deformable (tracked) material. Then virtual nodes at the location of the integration points are checked.

Card 2	1	2	3	4	5	6	7	8
Variable	BT	DT	SO					
Type	F	F	I					
Default	0.	10^{20}	0					

VARIABLE	DESCRIPTION
BT	Birth time
DT	Death time
SO	Flag to use penalty stiffness as in surface-to-surface contact: EQ.0: Contact entity stiffness formulation, EQ.1: Surface to surface contact method, LT.0: In this case SO must be an integer. |SO| references the load curve ID which defines the force as a function of the normal penetration. See Remark 1.

Remarks:

1. **Load Curves.** The optional load curves that are defined for damping as a function relative normal velocity and for force as a function of normal penetration should be defined in the positive quadrant. The sign for the damping force depends on the direction of the relative velocity and the treatment is symmetric if the damping curve is in the positive quadrant. If the damping force is defined in the negative and positive quadrants, the sign of the relative velocity is used in the table look-up.

2. **Friction and Contact.** Insofar as these ellipsoidal contact surfaces are continuous and smooth it may be necessary to specify Coulomb friction values larger than those typically used with faceted contact surfaces.

*CONTACT_GUIDED_CABLE_{OPTION1}_{OPTION2}

Purpose: Define a sliding contact that guides 1D elements, such as springs, trusses, and beams, along a path defined by a set of nodes. Only one 1D element can be in contact with any given node in the node set at a given time. If for some reason, a node is in contact with multiple 1D elements, one guided contact definition must be used for each contact. The ordering of the nodal points and 1D elements in the input is arbitrary.

OPTION1 specifies that a part set ID is given with the single option:

<BLANK>

SET

If not used, a part ID is assumed.

OPTION2 specifies that the first card read defines the heading and ID number of the contact interface and takes the single option:

ID

Title Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	CID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
CID	Contact interface ID. This must be a unique number.
HEADING	Interface descriptor. We suggest using unique descriptions.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	PID/PSID	SOFT	SSFAC	FRIC	ENDTOL		
Type	I	I	I	F	F	F		
Default	none	none	0	1.0	none	↓		

VARIABLE	DESCRIPTION
NSID	Node set ID that guides the 1D elements
PID/PSID	Part ID or part set ID if SET is included in the keyword line
SOFT	Flag for soft constraint option. Set to 1 for soft constraint.
SSFAC	Stiffness scale factor for penalty stiffness value. The default value is unity. This applies to SOFT set to 0 and 1.
FRIC	Contact friction
ENDTOL	Tolerance, in length units, applied at the ends of the cable elements beyond which contact will pass to the next cable element. The default is 0.002 times the element length.

*CONTACT_INTERIOR

Purpose: Define interior contact for solid elements. Frequently, when soft materials are compressed under high pressure, the solid elements used to discretize these materials may invert leading to negative volumes and error terminations. To keep these elements from inverting, it is possible to consider interior contacts between layers of interior surfaces made up of the faces of the solid elements. Since these interior surfaces are generated automatically, the part (material) ID's for the materials of interest are defined here, prior to the interface definitions.

Define as many cards as necessary. Input ends at the next keyword ("*") card. Multiple instances of this keyword may appear in the input.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID1	PSID2	PSID3	PSID4	PSID5	PSID6	PSID7	PSID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
PSID*	Part set ID for which interior contact is desired.

Four attributes should be defined for each part set:

Attribute 1: PSF, penalty scale factor (Default = 1.00).

Attribute 2: Activation factor, F_a (Default = 0.10). When the crushing of the element reaches F_a times the initial thickness, the contact algorithm begins to act.

Attribute 3: ED, Optional modulus for interior contact stiffness.

Attribute 4: TYPE, Formulation for interior contact.

EQ.1.0: Default, recommended for uniform compression

EQ.2.0: Designed to control the combined modes of shear and compression. Works for type 1 brick formulation and type 10 tetrahedron formulation.

Define each part set with the *SET_PART_COLUMN option to specify independent attribute values for each part in the part set.

Remarks:

The interior penalty is determined by the formula:

$$K = \frac{\text{SLSFAC} \times \text{PSF} \times \text{Volume}^{2/3} \times \text{E}}{\text{Min. Thickness}}$$

where SLSFAC is the value specified on the *CONTROL_CONTACT card, volume is the volume of the brick element, E is a constitutive modulus, and min. thickness is approximately the thickness of the solid element through its thinnest dimension. If ED is defined above, the interior penalty is then given instead by:

$$K = \frac{\text{Volume}^{2/3} \times \text{ED}}{\text{Min. Thickness}}$$

where the scaling factors are ignored. Generally, ED should be taken as the locking modulus specified for the foam constitutive model.

Caution should be observed when using this option since if the time step size is too large an instability may result. The time step size is not affected by the use of interior contact.

*CONTACT_RIGID_SURFACE

Purpose: Define rigid surface contact. The purpose of rigid surface contact is to model large rigid surfaces, such as road surfaces, with nodal points and segments that require little storage and are written out at the beginning of the binary databases. The rigid surface motion, which can be optionally prescribed, is defined by a displacement vector which is written with each output state. The nodal points defining the rigid surface must be defined with *NODE_RIGID_SURFACE. These rigid nodal points do not contribute degrees-of-freedom.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	PSID	BOXID	SSID	FS	FD	DC	VC
Type	I	I	I	I	F	F	F	F
Default	none	none	0	none	0.	0.	0.	0.

Card 2	1	2	3	4	5	6	7	8
Variable	LCIDX	LCIDY	LCIDZ	FSLCID	FDLCID			
Type	I	I	I	I	I			
Default	0	0	0	0	0			

Card 3	1	2	3	4	5	6	7	8
Variable	SFS	STTHK	SFTHK	XPENE	BSORT	CTYPE		
Type	F	F	F	F	I	I		
Default	1.0	0.0	1.0	4.0	10	0		

VARIABLE	DESCRIPTION
CID	Contact interface ID. This must be a unique number.

VARIABLE	DESCRIPTION
PSID	Part set ID of all parts that may contact the rigid surface. See *SET_PART.
BOXID	Include only nodes of the part set that are within the specified box, see *DEFINE_BOX, in contact. If BOXID is zero, all nodes from the part set, PSID, will be included in the contact.
SSID	Segment set ID defining the rigid surface. See *SET_SEGMENT.
FS	Static coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact, $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ If FSLCID is defined, see below, then FS is overwritten by the value from the load curve.
FD	Dynamic coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact, $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ If FDLCID is defined, see below, then FD is overwritten by the value from the load curve.
DC	Exponential decay coefficient. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$
VC	Coefficient for viscous friction. This is necessary to limit the friction force to a maximum. A limiting force is computed, $$F_{\text{lim}} = \text{VC} \times A_{\text{cont}} \ .$$ A_{cont} being the area of the segment contacted by the node in contact. The suggested value for VC is to use the yield stress in shear VC $= \sigma_0/\sqrt{3}$ where σ_0 is the yield stress of the contacted material.
LCIDX	Load curve ID defining x-direction motion. If zero, there is no motion in the x-coordinate system.
LCIDY	Load curve ID defining y-direction motion. If zero, there is no motion in the y-coordinate system.
LCIDZ	Load curve ID defining z-direction motion. If zero, there is no motion in the z-coordinate system.

VARIABLE	DESCRIPTION
FSLCID	Load curve ID defining the static coefficient of friction as a function of interface pressure. This option applies to shell segments only.
FDLCID	Load curve ID defining the dynamic coefficient of friction as a function of interface pressure. This option applies to shell segments only.
SFS	Scale factor on default tracked surface penalty stiffness, see also *CONTROL_CONTACT.
STTHK	Optional thickness for tracked surface (overrides true thickness). This option applies to contact with shell, solid, and beam elements. True thickness is the element thickness of the shell elements. Thickness offsets are not used for solid element unless this option is specified.
SFTHK	Scale factor for tracked surface thickness (scales true thickness). This option applies only to contact with shell elements. True thickness is the element thickness of the shell elements.
XPENE	Contact surface maximum penetration check multiplier. If the penetration of a node through the rigid surface exceeds the product of XPENE and the tracked node thickness, the node is set free. EQ.0.0: default is set to 4.0.
BSORT	Number of cycles between bucket sorts. The default value is set to 10 but can be much larger, e.g., 50-100, for fully connected surfaces.
CTYPE	Contact formulation: EQ.0: Equivalent to the ONE_WAY_SURFACE_TO_SURFACE formulation (default) EQ.1: Penalty formulation. If the tracked surface belongs to a rigid body, this formulation must be used.

Remarks:

Thickness offsets do not apply to the rigid surface. There is no orientation requirement for the segments in the rigid surface, and the surface may be assembled from disjoint, but contiguous, arbitrarily oriented meshes. With disjoint meshes, the global searches must be done frequently, about every 10 cycles, to ensure a smooth movement of a tracked node between mesh patches. For fully connected meshes this frequency interval can be safely set to 50-200 steps between searches.

The modified binary database, d3plot, contains the road surface information prior to the state data. This information includes:

NPDS = Total number of rigid surface points in problem.

NRSC = Total number of rigid surface contact segments summed over all definitions.

NSID = Number of rigid surface definitions.

NVELQ = Number of words at the end of each binary output state defining the rigid surface motion. This equals 6 × NSID if any rigid surface moves or zero if all rigid surfaces are stationary.

PIDS = An array equal in length to NPDS. This array defines the ID for each point in the road surface.

XC = An array equal in length to 3 × NPDS. This array defines the global x, y, and z coordinates of each point.

For each road surface define the following NSID sets of data:

ID = Rigid surface ID.

NS = Number of segments in rigid surface.

IXRS = An array equal in length to 4 × NS. This is the connectivity of the rigid surface in the internal numbering system.

At the end of each state, 6 × NVELQ words of information are written. For each road surface the x, y, and z displacements and velocities are written. If the road surface is fixed, a null vector should be output. Skip this section if NVELQ = 0. LS-PrePost currently displays rigid surfaces and animates their motion.

*CONTACT_SPG

Purpose: Define contact between SPG particles from different SPG parts or self-contact of SPG particles from the same SPG part. This keyword was developed for high-speed deformations, such as projectile impact penetration problems.

Card 1	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Card 2	1	2	3	4	5	6	7	8
Variable	ISELF1	ISELF2	ISELF3	ISELF4	ISELF5	ISELF6	ISELF7	ISELF8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Card 3	1	2	3	4	5	6	7	8
Variable	PFAC1	PFAC2	PFAC3	PFAC4	PFAC5	PFAC6	PFAC7	PFAC8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Card 4	1	2	3	4	5	6	7	8
Variable	FS	FD	DC	NFREQ				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
PID*i*	Part ID of the SPG parts in particle contact.
ISELF*i*	Self-contact flag for PID*i*: EQ.0: No self-contact allowed for PIDi EQ.1: Self-contact allowed for PIDi
PFAC*i*	Penalty factor *i*
FS	Static coefficient of friction
FD	Dynamic coefficient of friction
DC	Exponential decay coefficient. The frictional coefficient is assumed to be dependent on the relative velocity, v_{rel}, of the surfaces in contact: $\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}$.
NFREQ	Contact searching frequency

*CONTACT_1D

Purpose: Define one-dimensional slide lines for rebar in concrete.

Card 1	1	2	3	4	5	6	7	8
Variable	NSIDR	NSIDC	ERR	SIGC	GB	SMAX	EXP	
Type	I	I	F	F	F	F	F	
Default	none	none	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
NSIDR	Nodal set ID for the rebar nodes that slide along the concrete; see *SET_NODE.
NSIDC	Nodal set ID for the concrete nodes that the rebar nodes may slide along; see *SET_NODE.
ERR	External radius of rebar
SIGC	Unconfined compressive strength of concrete, f_c
GB	Bond shear modulus
SMAX	Maximum shear strain
EXP	Exponent in damage curve

Remarks:

With this option the concrete is defined with solid elements and the rebar with truss elements, each with their own unique set of nodal points. A string of spatially consecutive nodes related to the truss elements (NSIDR) may slide along another string of spatially consecutive nodes related to the solid elements (NSIDC). The sliding commences after the rebar debonds.

The bond between the rebar and concrete is assumed to be elastic perfectly plastic. The maximum allowable slip strain is given as:

$$u_{\max} = \text{SMAX} \times e^{-\text{EXP} \times D}$$

where D is the damage parameter $D_{n+1} = D_n + \Delta u$. The shear force, acting on area A_S, at time $n + 1$ is given as:

$$f_{n+1} = \min[f_n - \mathrm{GB} \times A_s \times \Delta u, \mathrm{GB} \times A_s \times u_{\max}]$$

***CONTACT_2D**_*OPTION1_{OPTION2}_{OPTION3}*

Purpose: Define a two-dimensional contact interface or slide line. This option is to be used with two-dimensional solid and shell elements using the plane stress, plane strain or axisymmetric formulations; see *SECTION_SHELL and *SECTION_BEAM.

All the two-dimensional contacts are supported in SMP. Only *CONTACT_2D_AUTOMATIC_SINGLE_SURFACE and *CONTACT_2D_AUTOMATIC_SURFACE_TO_SURFACE are supported for MPP, with and without the option MORTAR.

OPTION1 specifies the contact type. The following options activate kinematic constraints and should be used with deformable materials only but may be used with rigid bodies if the rigid body is the reference surface, and all rigid body motions are prescribed. Kinematic constraints are recommended for high pressure hydrodynamic applications.

SLIDING_ONLY

TIED_SLIDING

SLIDING_VOIDS

AUTOMATIC_TIED_ONE_WAY

The following option uses both kinematic constraints and penalty constraints.

AUTOMATIC_TIED

The following options are penalty-based. These methods have no rigid-material limitations. We recommend them for lower pressure solid mechanics applications.

PENALTY_FRICTION

PENALTY

AUTOMATIC_SINGLE_SURFACE

AUTOMATIC_SINGLE_SURFACE_MORTAR

AUTOMATIC_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_IN_CONTINUUM

The following two contacts are used for SPH particles in contact with two-dimensional continuum elements (shell formulations 13, 14, 15). These contacts do not apply to two-dimensional shell elements (beam formulations 7, 8).

NODE_TO_SOLID

NODE_TO_SOLID_TIED

The following option is used to measure contact forces that are reported as RCFORC output.

FORCE_TRANSDUCER

OPTION2 specifies a thermal contact and takes the single option:

THERMAL

Only the AUTOMATIC contact options: SINGLE_SURFACE, SURFACE_TO_SURFACE, and ONE_WAY_SURFACE_TO_SURFACE including the Mortar contact versions may be used with the THERMAL option.

OPTION3 specifies that the first card to read defines the title and ID number of contact interface and takes the single option:

TITLE

Title Card. Additional card for the TITLE keyword potion.

Card Title	1	2	3	4	5	6	7	8
Variable	CID	TITLE						
Type	I	A70						

Two-dimensional contact may be divided into 3 groups, each with a unique input format.

1. The first group were adopted from LS-DYNA2D and originated in the public domain version of DYNA2D from the Lawrence Livermore National Laboratory. Contact surfaces are specified as ordered sets of nodes. These sets define either contact surfaces or slide lines. The keyword options for the first group are:

   SLIDING_ONLY

   TIED_SLIDING

   SLIDING_VOIDS

   PENALTY_FRICTION

PENALTY

> **NOTE:** We do not recommend TIED_SLIDING, PENALTY_-FRICTION and PENALTY since easier to use automatic options with same functionality exist in the second group.

2. The second group contains the automatic contacts. These contact surfaces may be defined using part sets or unordered node sets. Segment orientations are determined automatically. The keywords for these are:

   AUTOMATIC_SINGLE_SURFACE

   AUTOMATIC_SINGLE_SURFACE_MORTAR

   AUTOMATIC_SURFACE_TO_SURFACE

   AUTOMATIC_SURFACE_TO_SURFACE_MORTAR

   AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

   AUTOMATIC_SURFACE_IN_CONTINUUM

   AUTOMATIC_TIED

   AUTOMATIC_TIED_ONE_WAY

   FORCE_TRANSDUCER

3. The third group is for contact between SPH particles and continuum elements:

   NODE_TO_SOLID

   NODE_TO_SOLID_TIED

Each of the 3 groups has a section below with a description of input and additional remarks.

CONTACT_2D_[SLIDING, TIED, & PENALTY]_*OPTION

This section documents the *CONTACT_2D variations derived from DYNA2D:

SLIDING_ONLY

TIED_SLIDING

SLIDING_VOIDS

PENALTY_FRICTION

PENALTY

Card 1	1	2	3	4	5	6	7	8
Variable	SURFA	SURFB	TBIRTH	TDEATH				
Type	I	I	F	F				
Default	none	none	0.	10^{20}				

Card 2	1	2	3	4	5	6	7	8
Variable	EXT_PAS	THETA1	THETA2	TOL_IG	PEN	TOLOFF	FRCSCL	ONEWAY
Type	I	F	F	F	F	F	F	F
Default	none	none	none	0.001	0.1	0.025	0.010	0.0

Friction Card. Additional card for the PENALTY_FRICTION keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	FRIC	FRIC_L	FRIC_H	FRIC_S				
Type	F	F	F	F				

VARIABLE	DESCRIPTION
SURFA	Nodal set ID for the SURFA nodes, see *SET_NODE. The surface specified with SURFA must be to the left of the surface specified with SURFB. For nonsymmetric contact, this surface is the tracked surface (all contacts in this section except PENALTY and PENALTY_FRICTION).
SURFB	Nodal set ID for the SURFB nodes, see *SET_NODE. For nonsymmetric contact, this surface is the reference surface (all contacts in this section except PENALTY and PENALTY_FRICTION).
TBIRTH	Birth time for contact
TDEATH	Death time for contact
EXT_PAS	Slide line extension bypass option. EQ.0: Extensions are used. EQ.1: Extensions are not used.
THETA1	Angle in degrees of slide line extension at first SURFB node. EQ.0: Extension remains tangent to first SURFB segment.
THETA2	Angle in degrees of slide line extension at last SURFB node. EQ.0: Extension remains tangent to last SURFB segment.
TOL_IG	Tolerance for determining initial gaps. EQ.0.0: default set to 0.001
PEN	Scale factor or penalty. EQ.0.0: default set to 0.10
TOLOFF	Tolerance for stiffness insertion for implicit solution only. The contact stiffness is inserted when a node approaches a segment a distance equal to the segment length multiplied by TOLOFF. The stiffness is increased as the node moves closer with the full stiffness being used when the nodal point finally makes contact. EQ.0.0: default set to 0.025.
FRCSCL	Scale factor for the interface friction. EQ.0.0: Default set to 0.010

VARIABLE	DESCRIPTION
ONEWAY	Flag for one way treatment. If set to 1.0, the nodal points on SURFA are constrained to SURFB. This option is generally recommended if SURFB is rigid. EQ.1.0: Activate one way treatment.
FRIC	Coefficient of friction
FRIC_L	Coefficient of friction at low velocity
FRIC_H	Coefficient of friction at high velocity
FRIC_S	Friction factor for shear

Remarks:

1. **Option descriptions.** The following are descriptions of the DYNA2D variations:

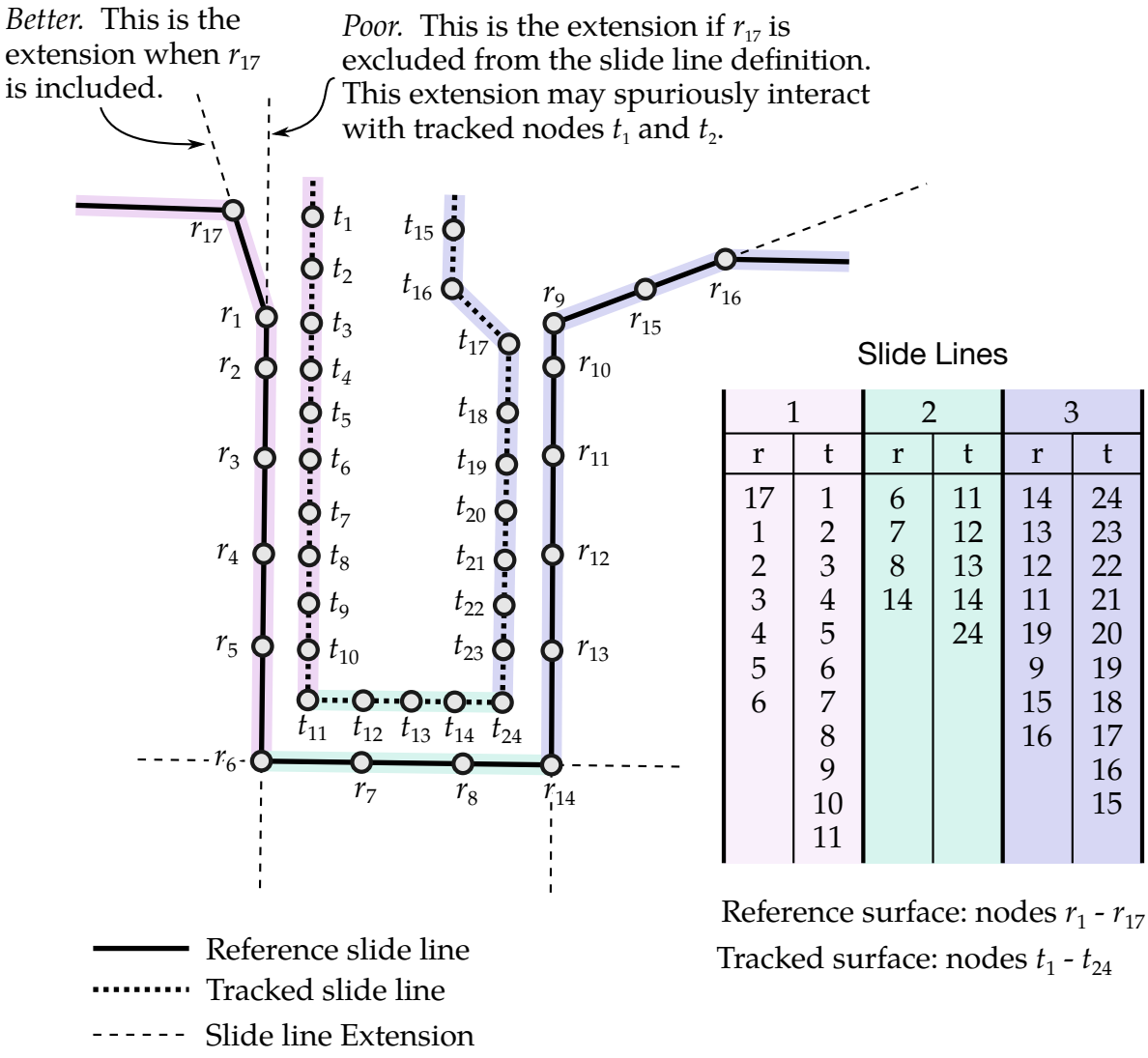
   a) The SLIDING_ONLY option is a two-surface method based on a kinematic formulation. The two surfaces can slide arbitrarily large distances without friction but are not permitted to separate or interpenetrate. Surfaces should be initially in contact. This option performs well when extremely high interface pressures are present. The more coarsely meshed surface should be chosen as the reference surface (SURFB) for best performance.

   b) The TIED_SLIDING option joins two parts of a mesh with differing mesh refinement. It is a kinematic formulation, so the more coarsely meshed surface should be chosen as the reference surface (SURFB).

   c) The SLIDING_VOIDS option is a kinematic formulation without friction which permits two surfaces to separate if tensile forces develop across the interface. The surfaces may be initially in contact or initially separated.

   d) The PENALTY_FRICTION and PENALTY options are penalty formulations, so the designation of SURFA and SURFB is not important. The two bodies may be initially separate or in contact. A rate-dependent Coulomb friction model is available for PENALTY_FRICTION.

2. **Slide line.** Consider two slide line surfaces in contact. It is necessary to designate one as a tracked surface and the other as a reference surface. Nodal points defining the tracked surface are called tracked nodes, and similarly, nodes defining the reference surface are called reference nodes. Each tracked-reference surface combination is referred to as a slide line.



1		2		3	
r	t	r	t	r	t
17	1	6	11	14	24
1	2	7	12	13	23
2	3	8	13	12	22
3	4	14	14	11	21
4	5		24	19	20
5	6			9	19
6	7			15	18
	8			16	17
	9				16
	10				15
	11				

Figure 11-35. *Slide line Example.* Note: (1) as recommended, for 90° angles each facet is assigned a distinct slide line; (2) the reference slide line is more coarsely meshed; (3) the tracked slide line is to the left of the reference (following the node ordering, see inset table); (4) as shown for tracked nodes 1 and 2 it is important the slide line extension does not spuriously come into contact.

3. **Modeling precautions.** Many potential problems with the options can be avoided by observing the following precautions:

   a) Metallic materials should contain the reference surface along high explosive-metal interfaces.

   b) SLIDING_ONLY type slide lines are appropriate along high explosive-metal interfaces. The penalty formulation is not recommended along such interfaces.

c) If one surface is more finely zoned, it should be used as the tracked surface (SURFA). If penalty slide lines, PENALTY and PENALTY_FRICTION, are used, then the tracked-reference distinction is irrelevant.

d) A tracked node may have more than one reference segment and may be included as a member of a reference segment if a slide line intersection is defined.

e) Angles in the reference side of a slide line that approach 90° must be avoided.

Whenever such angles exist in a reference surface, two or more slide lines should be defined. This procedure is illustrated in Figure 11-35. An exception for the foregoing rule arises if the surfaces are tied. In this case, only one slide line is needed.

f) Whenever two surfaces are in contact, the smaller of the two surfaces should be used as the tracked surface. For example, in modeling a missile impacting a wall, the contact surface on the missile should be used as the tracked surface.

g) Care should be used when defining a reference surface to prevent the extension from interacting with the solution. In Figures 11-35 and 11-36, slide line extensions are shown.

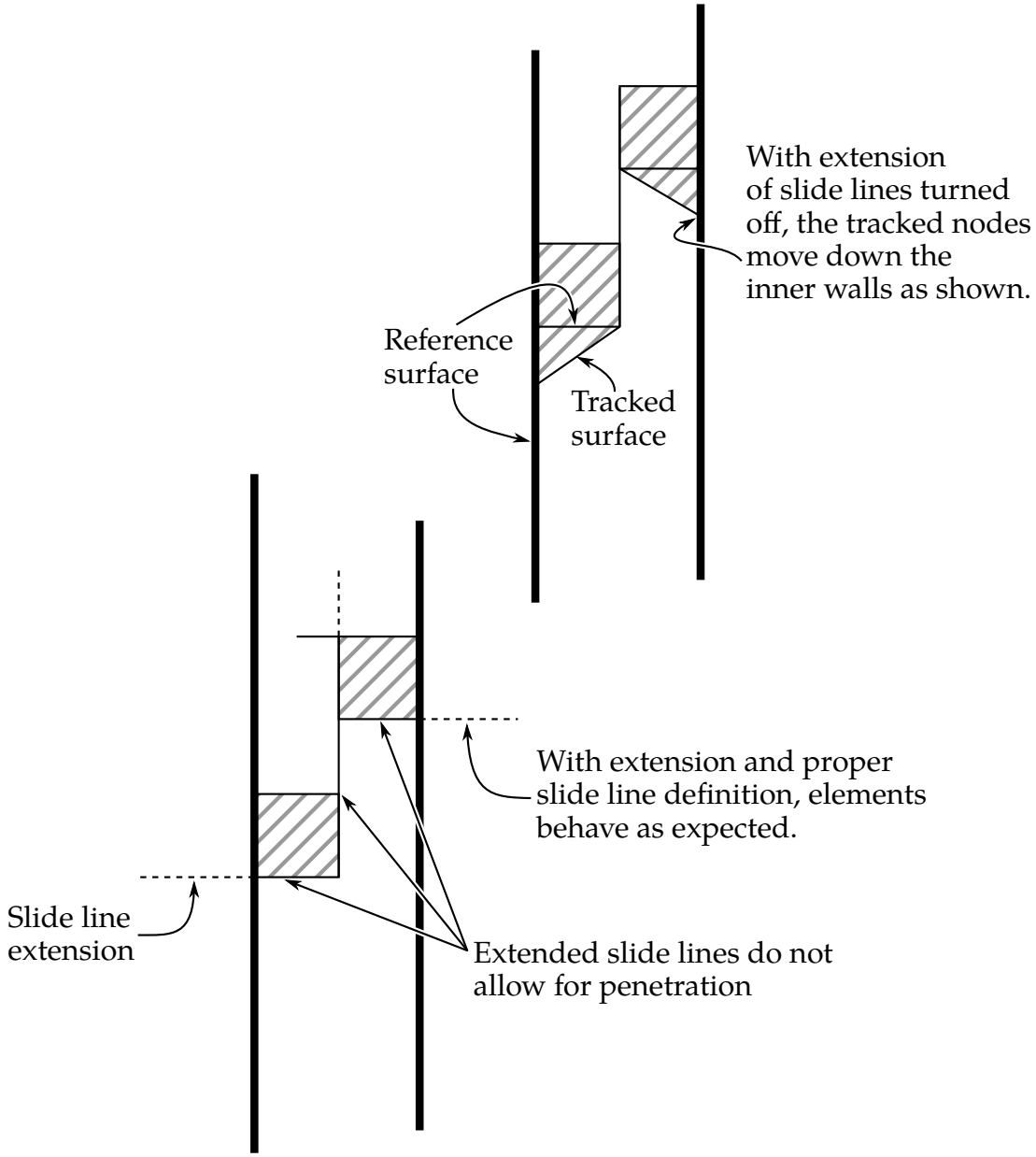


Figure 11-36. *With and without extension.* Extensions may be turned off by setting EXT_PAS (card 2), but, when turned off, tracked nodes may "leak" out as shown in the upper version of the figure.

CONTACT_2D_[AUTOMATIC & FORCE_TRANSDUCER]_*OPTION

This section documents the following variations of *CONTACT_2D:

AUTOMATIC_SINGLE_SURFACE

AUTOMATIC_SINGLE_SURFACE_MORTAR (see Remark 12)

AUTOMATIC_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR (see Remark 12)

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_IN_CONTINUUM (see Remark 11)

AUTOMATIC_TIED

AUTOMATIC_TIED_ONE_WAY

FORCE_TRANSDUCER (see Remark 2)

Card 1	1	2	3	4	5	6	7	8
Variable	SURFA	SURFB	SFACT	FREQ	FS	FD	DC	
Type	I	I	F	I	F	F	F	
Default	none	↓	1.0	50	0.	0.	0.	
Remarks	1, 2	1, 2						

Card 2	1	2	3	4	5	6	7	8
Variable	TBIRTH	TDEATH	SOA	SOB	NDA	NDB	COF	INIT
Type	F	F	F	F	I	I	I	I
Default	0.	10^{20}	1.0	1.0	0	0	0	0
Remarks			4	4	3	3, 11		6

Automatic Thermal Card. Additional card for keywords with both the AUTOMATIC and THERMAL options. For example, *CONTACT_2D_AUTOMATIC_..._THERMAL_..... . See Remark 7.

Card 3	1	2	3	4	5	6	7	8
Variable	K	RAD	H	LMIN	LMAX	CHLM	BC_FLAG	
Type	F	F	F	F	F	F	I	
Default	none	none	none	none	none	1.0	0	

Automatic Optional Card 1. Optional card for the AUTOMATIC keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	VC	VDC	IPF	SLIDE	ISTIFF	TIEDGAP	IGAPCL	TIETYP
Type	F	F	I	I	I	R	I	I
Default	0.	10.0	0	0	0		0	0
Remarks				8	9	10		10

Automatic Optional Card 2. Optional card for the AUTOMATIC keyword option. This card can only be used if Card 4 is included, but Card 4 can be left blank.

Card 5	1	2	3	4	5	6	7	8
Variable	SLDSOA	SLDSOB	TDPEN					
Type	F	F	F					
Default	0.	0.	0.					
Remarks	13	13						

VARIABLE	DESCRIPTION
SURFA	Set ID for SURFA. If SURFA > 0, a part set is assumed; see *SET_PART. If SURFA < 0, a node set with ID equal to the absolute value of SURFA is assumed; see *SET_NODE. For nonsymmetric contact, this surface is the tracked surface.
SURFB	Set ID to define the SURFB surface. If SURFB > 0, a part set is assumed; see *SET_PART. If SURFB < 0, a node set with ID equal to the absolute value of SURFB is assumed; see *SET_NODE. Do not define for single surface contact. For nonsymmetric contact, this surface is the reference surface.
SFACT	Scale factor for the penalty force stiffness
FREQ	Search frequency. The number of timesteps between bucket sorts. For implicit contact this parameter is ignored, and the search frequency is 1. EQ.0: Default set to 50.
FS	Static coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact according to the relation given by: $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$
FD	Dynamic coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ This parameter does not apply to mortar contact.
DC	Exponential decay coefficient. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ This parameter does not apply to mortar contact.
TBIRTH	Birth time (contact surface becomes active at this time): LT.0.0: Birth time is set to |TBIRTH|. When negative, birth time is followed during the dynamic relaxation phase of the calculation. After dynamic relaxation has completed, contact is activated regardless of the value of TBIRTH.

VARIABLE	DESCRIPTION
	EQ.0.0: Birth time is inactive, meaning contact is always active
	GT.0.0: If TDEATH = -9999, TBIRTH is interpreted as the curve defining multiple pairs of birth-time/death-time. Otherwise, if TDEATH > 0, birth time applies both during and after dynamic relaxation.
TDEATH	Death time (contact surface is deactivated at this time):
	LT.0.0: If TDEATH = -9999, TBIRTH is interpreted as the curve ID defining multiple pairs of birth-time/death-time. Otherwise, negative TDEATH indicates that contact is inactive during dynamic relaxation. After dynamic relaxation the birth and death times are followed and set to |TBIRTH| and |TDEATH|, respectively.
	EQ.0.0: TDEATH defaults to 10^{20}.
	GT.0.0: TDEATH sets the time at which the contact is deactivated.
SOA	Surface offset from midline for two-dimensional shells of SURFA surface:
	EQ.0.0: Default to 1.0.
	GT.0.0: Scale factor applied to actual thickness
	LT.0.0: Absolute value is used as the offset
SOB	Surface offset from midline for 2D shells of SURFB surface:
	EQ.0.0: Default to 1.0.
	GT.0.0: Scale factor applied to actual thickness
	LT.0.0: Absolute value is used as the offset
NDA	Normal direction flag for two-dimensional shells of SURFA surface:
	EQ.0: Normal direction is determined automatically.
	EQ.1: Normal direction is in the positive direction.
	EQ.-1: Normal direction is in the negative direction.
NDB	Normal direction flag for two-dimensional shells of SURFB surface:
	EQ.0: Normal direction is determined automatically.

VARIABLE	DESCRIPTION
	EQ.1: Normal direction is in the positive direction.
	EQ.-1: Normal direction is in the negative direction.
COF	Closing/opening flag for implicit contact: EQ.0: Recommended for most problem where gaps are only closing. EQ.1: Recommended when gaps are opening to avoid sticking. This parameter does not apply to mortar contact.
INIT	Special processing during initialization: EQ.0: No special processing EQ.1: Forming option
K	Thermal conductivity (k) of fluid between the slide surfaces. If a gap with a thickness l_{gap} exists between the slide surfaces, then the conductance due to thermal conductivity between the slide surfaces is $$h_{cond} = \frac{k}{l_{gap}}$$ Note that LS- DYNA calculates l_{gap} based on deformation.
RAD	Radiation factor, f, between the slide surfaces. A radiant-heat-transfer coefficient (h_{rad}) is calculated (see *BOUNDARY_RADIATION). If a gap exists between the slide surfaces, then the contact conductance is calculated by $$h = h_{cond} + h_{rad}$$
H	Heat transfer conductance (h_{cont}) for closed gaps. Use this heat transfer conductance for gaps in the range: $$0 \le l_{gap} \le l_{min} \ ,$$ where l_{min} is GCRIT defined below.
LMIN	Critical gap (l_{min}), use the heat transfer conductance defined (HTC) for gap thicknesses less than this value.
LMAX	No thermal contact if gap is greater than this value (l_{max}).

VARIABLE	DESCRIPTION
CHLM	Multiplier used on the element characteristic distance for the search routine. The characteristic length is the largest interface surface element diagonal. EQ.0: Default set to 1.0
BC_FLAG	Thermal boundary condition flag: EQ.0: Thermal boundary conditions are on when parts are in contact. EQ.1: Thermal boundary conditions are off when parts are in contact.
VC	Coefficient for viscous friction. This is used to limit the friction force to a maximum. A limiting force is computed $F_{\text{lim}} = \text{VC} \times A_{\text{cont}}.$ A_{cont} being the area of contacted between segments. The suggested value for VC is to use the yield stress in shear: $\text{VC} = \frac{\sigma_o}{\sqrt{3}}$, where σ_o is the yield stress of the contacted material.
VDC	Viscous damping coefficient in percent of critical for explicit contact. This parameter does not apply to Mortar contact.
IPF	Initial penetration flag (see Remark 14): EQ.0: Allow initial penetrations to remain. GE.1: Push apart initially penetrated surfaces.
SLIDE	Sliding option: EQ.0: Off EQ.1: On
ISTIFF	Stiffness scaling option: EQ.0: Use default option. EQ.1: Scale stiffness using segment masses and explicit time step (default for explicit contact). EQ.2: Scale stiffness using segment stiffness and dimensions (default for implicit contact).

VARIABLE	DESCRIPTION
TIEDGAP	Search gap for tied contacts. EQ.0: Default, use 1% of the SURFB segment length GT.0: Use the input value LT.0: Use –TIEDGAP % of the SURFB segment length.
IGAPCL	Flag to close gaps in tied contact: EQ.0: Default, allow gaps to remain EQ.1: Move SURFA nodes to SURFB segments to close gaps
TIETYP	Flag to control constraint type of tied contact: EQ.0: Default, use kinematic constraints when possible EQ.1: Use only penalty type constraints
SLDSOA	Solid surface offset for the SURFA surface
SLDSOB	Solid surface offset for the SURFB surface
TDPEN	Time span of penetration removal for 2D Mortar contacts. Each initial penetration will be gradually reduced linearly in time, so that it is removed by time TDPEN. For instance, if a given penetration has an initial distance D, then the penetration distance over time, $d(t)$ will be $d(t) = \frac{D}{\text{TDPEN}}(\text{TDPEN} - t)$. This is the interference option analogue to MPAR1 for IGNORE = 3 in 3D automatic Mortar contacts.

Remarks:

1. **Penalty force methods.** The SINGLE_SURFACE, SURFACE_TO_SURFACE, and ONE_WAY_SURFACE_TO_SURFACE options use penalty forces to prevent penetration between two-dimensional shell elements and external faces of two-dimensional continuum elements. Contact surfaces are defined using SURFA and SURFB to reference either part sets or node sets. If part sets are used, all elements and continuum faces of the parts in the set are included in contact. If node sets are used, elements or continuum faces that have both nodes in the set are included in the contact surface. The SINGLE_SURFACE option uses only the SURFA set and checks for contact between all elements and continuum faces in the set. If SURFA is blank or zero, contact will be checked for all elements and continuum faces in the model. With the other options, both SURFA and SURFB are required.

2. **Force transducer.** The FORCE_TRANSDUCER option should be used in conjunction with at least one AUTOMATIC contact option. It does nothing to prevent penetration but measures the forces generated by other contact definitions. The FORCE_TRANSDUCER option requires only SURFA. SURFB is optional. If only SURFA is defined, the force transducer measures the resultant contact force on all the elements and continuum faces in the SURFA surface. If both SURFA and SURFB are defined, then the force transducer measures contact forces between the elements and continuum faces in the SURFA and SURFB surfaces. The measured forces are included in the rcforc output. In the case of an axisymmetric analysis, values output to rcforc and ncforc are in units of force per radian (this includes both shell types 14 and 15).

3. **Normal direction.** By default, the normal direction of 2D shell elements is evaluated automatically for SINGLE_SURFACE, SURFACE_TO_SURFACE and ONE_WAY_SURFACE_TO_SURFACE contact. You can override the automatic algorithm using NDA or NDB, and contact will occur with the positive or negative face of the element.

4. **Shell thickness.** By default, the true thickness of two-dimensional shell elements is taken into account for the SURFACE_TO_SURFACE, SINGLE_SURFACE, and ONE_WAY_SURFACE_TO_SURFACE options. You can override the true thickness by using SOA and SOB. If the surface offset is reduced to a small value, the automatic normal direction algorithm may fail, so it is best to specify the normal direction using NDA or NDB.

5. **AUTOMATIC contact and rrosion.** For all AUTOMATIC contact options, eroding materials are treated by default. At present, subcycling is not possible.

6. **Thin Solid Parts and Contact Interface.** The INIT parameter activates a forming option that is intended for implicit solutions of thin solid parts when backside segments may interfere with the solution. It automatically removes backside segments during initialization. Alternatively, the user can input INIT = 0 and use node set input to limit the contact interface to just the front of a thin part.

7. **THERMAL option.** For the THERMAL option:

   $h = h_{\text{cont}}$, if the gap thickness is $0 \le l_{\text{gap}} \le l_{\text{min}}$

   $h = h_{\text{cond}} + h_{\text{rad}}$, if the gap thickness is $l_{\text{min}} \le l_{\text{gap}} \le l_{\text{max}}$

   $h = 0$, if the gap thickness is $l_{\text{gap}} > l_{\text{max}}$

8. **Sliding with kinks and vorners.** The SLIDE parameter activates a sliding option which uses additional logic to improve sliding when surfaces in contact have kinks or corners. This option is off by default.

9. **Penalty stiffness control.** The ISTIFF option allows control of the equation used in calculating the penalty stiffness. For backward compatibility, the default values are different for implicit and explicit solutions. When ISTIFF = 1 is used, the explicit time step appears in the stiffness equation, regardless of whether the calculation is implicit or explicit.

10. **Kinematic constraints.** The TIED_ONE_WAY contact creates two degree of freedom translational kinematic constraints to nodes on the SURFA surface that are initially located on or near SURFB segments. The TIED option creates kinematic constraints between SURFA nodes and SURFB segments and creates penalty constraints between SURFB nodes and SURFA segments. With either contact option, a kinematic constraint may be switched to penalty if there is a conflict with another constraint. The TIEDGAP parameter determines the maximum normal distance from a segment to a node for a constraint to be formed. Nodes will not be moved to eliminate an initial gap, and the initial gap will be maintained throughout the calculation. If TIETYP = 1, then only penalty constraints will be used.

11. **SURFACE_IN_CONTINUUM option.** Note that the SURFACE_IN_CONTINUUM option has been deprecated in favor of the *CONSTRAINED_LAGRANGE_IN_SOLID keyword, which allows coupling between fluids and structures. However, this option is maintained to provide backward compatibility for existing data.

    For the SURFACE_IN_CONTINUUM option, penalty forces prevent the flow of tracked element material (the continuum) through the reference surfaces. Flow of the continuum tangent to the surface is permitted. Only two-dimensional solid parts are permitted in the tracked part set. Both 2D solid and 2D shell parts are permitted in the reference part set. Flow through two-dimensional shell elements is prevented in both directions by default. If NDB is set to ±1, flow in the direction of the normal is permitted. The thickness of two-dimensional shell elements is ignored.

    When using the SURFACE_IN_CONTINUUM option, you do not need to mesh the continuum around the structure because contact is not with continuum nodes but with material in the interior of the continuum elements. The algorithm works well for Eulerian or ALE elements since the structure does not interfere with remeshing. However, a structure will usually not penetrate the surface of an ALE continuum since the nodes are Lagrangian normal to the surface. Therefore, if using an ALE fluid, the structure should be initially immersed in the fluid and remain immersed throughout the calculation. Penetrating the surface of an Eulerian continuum is not a problem.

12. **Mortar contact.** Mortar contact (MORTAR) is available for implicit and explicit calculations in SMP and MPP. The apparent behavior compared to non-Mortar

contact is very similar; the difference lies in details concerning the constitutive relation (contact stress as a function of the relative motion of contact surfaces) and the kinematics (the relative motion of contact surfaces as a function of nodal coordinates). Mortar contact is designed for continuity and smoothness, which is beneficial for an implicit solution scheme. It is intended to enhance robustness for implicit. For details regarding the two-dimensional mortar contact, see the LS-DYNA Theory Manual.

13. **Segment thickness.** By default, segments on the surface of two-dimensional continuum elements have zero thickness, so the contact surface is the line between the nodes. The contact surface can be optionally offset from this line by using the SLDSOA and SLDSOB parameters. If set to a positive number, SLDSOA is the offset for two-dimensional continuum element segments on the SURFA surface, and SLDSOB is the offset for two-dimensional continuum element segments on the SURFB surface. The SOA and SOB parameters provide similar options for segments on two-dimensional shell elements.

14. **IPF.** If a segment pair is detected to be in contact, and the penetration depth exceeds the penetration that has occurred since the last solution cycle, then the excess initial penetration will not be penalized for as long as the segment pair remains in contact. This is done to prevent spikes in contact forces and possible shooting nodes. If IPF > 0, then the excess penetration will be penalized, but the penalty force is ramped up from zero. The value of IPF is the additional percentage of excess initial penetration will be penalized each cycle. For example, IPF = 1 penalizes 1% and IPF = 2 penalizes 2%. We do not recommend larger values. The IPF flag is available for the SURFACE_TO_SURFACE and SINGLE_SURFACE type contact options when used in explicit dynamic solutions. For Mortar contacts we recommend using TDPEN instead of IPF.

CONTACT_2D_NODE_TO_SOLID_*OPTION

This section documents the following variations of *CONTACT_2D:

NODE_TO_SOLID

NODE_TO_SOLID_TIED

Card 1	1	2	3	4	5	6	7	8
Variable	SPH	SOLID	TBIRTH	TDEATH				
Type	I	I	F	F				
Default	none	None	0.	10^{20}				

Card 2	1	2	3	4	5	6	7	8
Variable	SOFT	MAXPAR	VC	OFFD	PEN	FS	FD	DC
Type	I	F	F	F	F	F	F	F
Default	0	1.05	0.0	0.0	↓	0.0	0.0	0.0

VARIABLE	DESCRIPTION
SPH	Nodal set ID or part set ID for the SPH nodes, If SPH > 0, a nodal set ID is assumed. If SPH < 0, a part set ID is assumed.
SOLID	Solid part set ID. SOLID < 0 since only part set is allowed.
TBIRTH	Birth time for contact
TDEATH	Death time for contact
SOFT	Soft constraint option: EQ.0: Penalty formulation EQ.1: Soft constraint formulation

VARIABLE	DESCRIPTION
	The soft constraint may be necessary if the material constants of the parts in contact have a wide variation in the elastic bulk moduli. In the soft constraint option, the interface stiffness is based on the nodal mass and the global time step size. The soft constraint option is also recommended for axisymmetric simulations.
MAXPAR	Maximum parametric coordinate in segment search (values between 1.025 and 1.20 are recommended). If zero, the default is set to 1.05. This factor allows an increase in the size of the segments which may be useful at sharp corners.
VC	Coefficient for viscous friction. This is used to limit the friction force to a maximum. A limiting force is computed $$F_{\text{lim}} = \text{VC} \times A_{\text{cont}} \ .$$ A_{cont} being the area of contacted between segments. The suggested value for *VC* is to use the yield stress in shear: $$\text{VC} = \frac{\sigma_o}{\sqrt{3}} \ ,$$ where σ_o is the yield stress of the contacted material.
OFFD	Contact offset distance for SPH nodes. It does not currently apply to tied contacts. Recommended to be half of the original particle spacing in contact direction.
PEN	Scale factor for penalty. EQ.0.0: default set to 1.0 for penalty formulation, or 0.1 for soft constraint formulation.
FS	Static coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact according to the relationship given by: $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$
FD	Dynamic coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$
DC	Exponential decay coefficient. The frictional coefficient is assumed

VARIABLE	DESCRIPTION
	to be dependent on the relative velocity v_{rel} of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{rel}\rvert}.$$

Remarks:

NODE_TO_SOLID contact is a penalty-based contact type used only for SPH particles with solid elements using the plane stress, plane strain or axisymmetric formulation. NODE_TO_SOLID_TIED contact is used only for SPH particles tied with solid elements, an offset of distance h (smooth length) is adopted for each SPH particle.

*CONTROL

The keyword control cards are optional and can be used to change defaults. They can also activate solution options, such as mass scaling, adaptive remeshing, and the implicit solver (see the *CONTROL_IMPLICIT section for details). We do, however, suggest defining the *CONTROL_TERMINATION card. *The order of the control cards in the input file is arbitrary. To avoid ambiguities, define no more than one control card of each type.*

The available control cards follow in alphabetical order:

*CONTROL_ACCURACY

*CONTROL_ACOUSTIC

*CONTROL_ACOUSTIC_COUPLING

*CONTROL_ACOUSTIC_SPECTRAL

*CONTROL_ADAPSTEP

*CONTROL_ADAPTIVE

*CONTROL_ADAPTIVE_CURVE

*CONTROL_AIRBAG

*CONTROL_ALE

*CONTROL_BULK_VISCOSITY

*CONTROL_CHECK_SHELL

*CONTROL_CHECK_SOLID

*CONTROL_COARSEN

*CONTROL_CONSTRAINED

*CONTROL_CONTACT

*CONTROL_COUPLING

*CONTROL_CPG

*CONTROL_CPG_OUTPUT

*CONTROL_CPM

*CONTROL_CPU

*CONTROL_DEBUG

*CONTROL_DISCRETE_ELEMENT

*CONTROL_DYNAMIC_RELAXATION

*CONTROL_EFG

*CONTROL_ENERGY

*CONTROL_EOS_USER_LIBRARY

*CONTROL_EXPLICIT_THERMAL_ALE_COUPLING

*CONTROL_EXPLICIT_THERMAL_BOUNDARY

*CONTROL_EXPLICIT_THERMAL_CONTACT

*CONTROL_EXPLICIT_THERMAL_INITIAL

*CONTROL_EXPLICIT_THERMAL_OUTPUT

*CONTROL_EXPLICIT_THERMAL_PROPERTIES

*CONTROL_EXPLICIT_THERMAL_SOLVER

*CONTROL_EXPLOSIVE_SHADOW

*CONTROL_FORMING_AUTO_NET

*CONTROL_FORMING_AUTOCHECK

*CONTROL_FORMING_AUTOPOSITION_PARAMETER

*CONTROL_FORMING_BESTFIT

*CONTROL_FORMING_DRAWBEAD_TRANSITION

*CONTROL_FORMING_HOME_GAP

*CONTROL_FORMING_INITIAL_THICKNESS

*CONTROL_FORMING_MAXID

*CONTROL_FORMING_ONESTEP

*CONTROL_FORMING_OUTPUT

*CONTROL_FORMING_PARAMETER_READ

*CONTROL_FORMING_POSITION

*CONTROL_FORMING_PRE_BENDING

*CONTROL_FORMING_PROJECTION

*CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS

*CONTROL_FORMING_SCALE

*CONTROL_FORMING_SCRAP_FALL

*CONTROL_FORMING_SHELL_TO_TSHELL

*CONTROL_FORMING_STONING

*CONTROL_FORMING_STRAIN_RATIO_SMOOTH

*CONTROL_FORMING_TEMPLATE

*CONTROL_FORMING_TIPPING

*CONTROL_FORMING_TRAVEL

*CONTROL_FORMING_TRIM_MERGE

*CONTROL_FORMING_TRIM_SOLID_REFINEMENT

*CONTROL_FORMING_TRIMMING

*CONTROL_FORMING_UNFLANGING

*CONTROL_FORMING_USER

*CONTROL_FREQUENCY_DOMAIN

*CONTROL_HOURGLASS

*CONTROL_IMPLICIT_AUTO

*CONTROL_IMPLICIT_BUCKLE

*CONTROL_IMPLICIT_CONSISTENT_MASS

*CONTROL_IMPLICIT_DYNAMICS

*CONTROL_IMPLICIT_EIGENVALUE

*CONTROL_IMPLICIT_FORMING

*CONTROL_IMPLICIT_GENERAL

*CONTROL_IMPLICIT_INERTIA_RELIEF

*CONTROL_IMPLICIT_JOINTS

*CONTROL_IMPLICIT_MODAL_DYNAMIC

*CONTROL_IMPLICIT_MODAL_DYNAMIC_DAMPING

*CONTROL_IMPLICIT_MODAL_DYNAMIC_MODE

*CONTROL_IMPLICIT_MODES

*CONTROL_IMPLICIT_ORDERING

*CONTROL_IMPLICIT_RESIDUAL_VECTOR

*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS

*CONTROL_IMPLICIT_SOLUTION

*CONTROL_IMPLICIT_SOLVER

*CONTROL_IMPLICIT_SSD_DIRECT

*CONTROL_IMPLICIT_STABILIZATION

*CONTROL_IMPLICIT_STATIC_CONDENSATION

*CONTROL_IMPLICIT_TERMINATION

*CONTROL_LSDA

*CONTROL_MAPPING_SHIFT_BY_DT

*CONTROL_MAT

*CONTROL_MPP_CONTACT_GROUPABLE

*CONTROL_MPP_DECOMPOSITION_ADJUST_MTYPE_COST

*CONTROL_MPP_DECOMPOSITION_ADJUST_PART_COST

*CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS

*CONTROL_MPP_DECOMPOSITION_AUTOMATIC

*CONTROL_MPP_DECOMPOSITION_BAGREF

*CONTROL_MPP_DECOMPOSITION_CHECK_SPEED

*CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE

*CONTACT_MPP_DECOMPOSITION_DEFORMED_GEOMETRY

*CONTROL_MPP_DECOMPOSITION_DISABLE_UNREF_CURVES

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_SPH_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_ELCOST

*CONTROL_MPP_DECOMPOSITION_FILE

*CONTROL_MPP_DECOMPOSITION_FLAG_STRESS_STRAIN_CURVE

*CONTROL_MPP_DECOMPOSITION_METHOD

*CONTROL_MPP_DECOMPOSITION_NODISTRIBUTE_DES_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_NUMPROC

*CONTROL_MPP_DECOMPOSITION_OUTDECOMP

*CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_RCBLOG

*CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION

*CONTROL_MPP_DECOMPOSITION_SCALE_CONTACT_COST

*CONTROL_MPP_DECOMPOSITION_SCALE_FACTOR_SPH

*CONTROL_MPP_DECOMPOSITION_SHOW

*CONTROL_MPP_DECOMPOSITION_TRANSFORM_KEYWORD

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION

*CONTROL_MPP_IO_LSTC_REDUCE

*CONTROL_MPP_IO_NOBEAMOUT

*CONTROL_MPP_IO_NOD3DUMP

*CONTROL_MPP_IO_NODUMP

*CONTROL_MPP_IO_NOFULL

CONTROL_MPP_IO_NOTIEDIO

*CONTROL_MPP_IO_SWAPBYTES

*CONTROL_MPP_MATERIAL_MODEL_DRIVER

*CONTROL_MPP_PFILE

*CONTROL_MPP_REBALANCE

*CONTROL_NONLOCAL

*CONTROL_OUTPUT

*CONTROL_PARALLEL

*CONTROL_PORE_AIR

*CONTROL_PORE_FLUID

*CONTROL_PZELECTRIC

*CONTROL_REFERENCE_CONFIGURATION

*CONTROL_REFINE_ALE

*CONTROL_REFINE_ALE2D

*CONTROL_REFINE_MPP_DISTRIBUTION

*CONTROL_REFINE_SHELL

*CONTROL_REFINE_SOLID

*CONTROL_REMESHING

*CONTROL_REQUIRE_REVISION

*CONTROL_RIGID

*CONTROL_SEGMENTS_IN_ALE_COUPLING

*CONTROL_SHELL

*CONTROL_SOLID

*CONTROL_SOLUTION

*CONTROL_SPH

*CONTROL_SPH_INCOMPRESSIBLE

*CONTROL_SPOTWELD_BEAM

*CONTROL_STAGED_CONSTRUCTION

*CONTROL_START

*CONTROL_STEADY_STATE_ROLLING

*CONTROL_STRUCTURED

*CONTROL_SUBCYCLE

*CONTROL_TERMINATION

*CONTROL_THERMAL_EIGENVALUE

*CONTROL_THERMAL_FORMING

*CONTROL_THERMAL_NONLINEAR

*CONTROL_THERMAL_SOLVER

*CONTROL_THERMAL_TIMESTEP

*CONTROL_TIMESTEP

*CONTROL_UNITS

*CONTROL_2D_REMESHING_REGION

*CONTROL_ACCURACY

Purpose: Define control parameters that can improve the accuracy of the calculation.

Card 1	1	2	3	4	5	6	7	8
Variable	OSU	INN	PIDOSU	IACC	EXACC	SRTFLG		
Type	I	I	I	I	F	I		
Default	0	↓	optional	0	0.0	0		

VARIABLE **DESCRIPTION**

OSU — Global flag for 2nd order objective stress updates (see Remark 1 below). Generally, for explicit calculations only those parts undergoing large rotations, such as rolling tires, need this option. Objective stress updates can be activated for a subset of part IDs by defining the part set in columns 21-30.

EQ.0: Off (default)

EQ.1: On

INN — Invariant node numbering for shell and solid elements. (See Remarks 2 and 3).

EQ.-4: On for both shell and solid elements except triangular shells

EQ.-2: On for shell elements except triangular shells

EQ.1: Off (default for explicit)

EQ.2: On for shell and thick shell elements (default for implicit)

EQ.3: On for solid elements

EQ.4: On for shell, thick shell, and solid elements

PIDOSU — Part set ID for objective stress updates. If this part set ID is given only those part IDs listed will use the objective stress update; therefore, OSU is ignored.

IACC — Implicit accuracy flag, turns on some specific accuracy considerations in implicit analysis at an extra CPU cost. See Remark 4.

EQ.-1: Off

VARIABLE	DESCRIPTION
	EQ.0: On (only for implicit) EQ.1: On (only for implicit) EQ.2: On (partially also for explicit, for compatibility when switching between implicit and explicit)
EXACC	Explicit accuracy parameter: EQ.0.0: Off (default) GT.0.0: On (see Remark 5)
SRTFLG	Flag to process parts, contacts, nodal rigid bodies, and elements in a sequence sorted by their respective user IDs, regardless of the order they appear in the input files. When turned on, it ensures consistent results if the user modifies the order of these entities in the input files, such as by changing the order of include files between runs of the same model. See Remark 6. EQ.0: Off (default) EQ.1: On

Remarks:

1. **Second-order objective stress update.** Second-order objective stress updates are occasionally necessary. Some examples include spinning bodies such as turbine blades in a jet engine, high velocity impacts generating large strains in a few time steps, and large time step sizes due to mass scaling in metal forming. There is a significantly added cost, which is due in part to the added cost of the second-order terms in the stress update when the Jaumann rate is used and the need to compute the strain-displacement matrix at the mid-point geometry. This option is available for the following element types:

   a) one-point brick elements

   b) the selective-reduced integrated brick element which uses eight integration points

   c) fully integrated plane strain and axisymmetric volume-weighted (type 15) 2D solid elements

   d) the thick shell elements

   e) Belytschko-Tsay shell elements

f) Belytschko-Tsay shell elements with warping stiffness

g) Belytschko-Chiang-Wong shell elements

h) S/R Hughes-Liu shell elements

i) type 16 fully integrated shell element

2. **Invariant node numbering for shell elements.** Invariant node numbering for shell and thick shell elements affects the choice of the local element shell coordinate system. The orientation of the default local coordinate system is based on the shell normal vector and the direction of the 1-2 side of the element. If the element numbering is permuted, the results will change in irregularly shaped elements. With invariant node numbering, permuting the nodes shifts the local system by an exact multiple of 90 degrees. In spite of its higher costs [<5%], the invariant local system is recommended for several reasons. First, element forces are nearly independent of node sequencing; secondly, the hourglass modes will not substantially affect the material directions; and, finally, stable calculations over long time periods are achievable. The INN parameter has no effect on thick shell form 2, which is always invariant, and thick shell form 3, which is never invariant.

3. **Invariant node numbering for solid elements.** Invariant node numbering for solid elements is available for anisotropic materials only. This option has no effect on solid elements of isotropic material. This option is recommended when solid elements of anisotropic material undergo significant deformation.

4. **Implicit calculations.** All other things being equal, a single time step of an implicit analysis usually involves a larger time increment and deformation than an explicit analysis. Many of the algorithms in LS-DYNA have been heavily optimized for explicit analysis in ways that are inappropriate for implicit analysis. While an implicit analysis, by default, invokes many measures to ensure accuracy, certain corrections associated with unusual applications or with large computational expense are invoked only by setting IACC = 1. A list of features that are included with this option follows at the end of this remarks section. The explicit column indicates features that are also active when IACC = 2. IACC = 2 is primarily intended for switching between implicit and explicit and maintaining compatibility between features. As of release R16, the corrections are on by default for implicit calculations. To turn off the corrections, set IACC = -1.

5. **EXACC.** The EXACC option is developed to improve the numerical accuracy for an explicit analysis. Currently, nodal coordinates are computed and stored in double precision in all versions. In most cases, this is sufficiently accurate and EXACC is recommended to be off. However, in some cases, particularly when initial coordinates are large and displacements are small, EXACC can increase the accuracy of computations. To use this option, the EXACC parameter should

be set to a positive value which is a characteristic element length for the mesh. For example, if the typical edge length in a model is 5.0 mm and the length units are millimeters, then set EXACC = 5.0.

To translate a model or a component of a model, keyword *INCLUDE_TRANSFORM is recommended because the 10-digit or 20-digit formats in keyword *NODE are not enough to represent a double precision value without round off error. Using *INCLUDE_TRANSFORM together with EXACC can yield a solution that is more consistent with the original result before the transformation.

Element features activated for IACC = 1	Implicit	Explicit
Strong objectivity, meaning that large rotations do not induce spurious strains and stresses, enforced for ◦ 1D seatbelts ◦ 2nd order shells (types 23 and 24) ◦ 1st order shells (types -16, 4 and 16) ◦ 1st order solids (types -2, -1, 1, 2, 13, 15, and 16) ◦ cohesive solids (types 19 and 20) ◦ beam elements (types 1, 2, and 9)	Yes	No
Assumed strain formulation of 2nd order triangular shell (type 24)	Yes	No
Thickness is involved in solution process for shell thickness update (see ISTUPD on *CONTROL_SHELL) and not imposed as a post-processing step. This is for shell type 16 and allows contact to act on updated geometry, thus avoiding large contact stresses at beginning of steps.	Yes	No
Allow beam types 1, 2, and 11 in implicit linear analyses	Yes	No
Linear element formulations in linear implicit, meaning that results are scaled linearly with respect to scaling of boundary conditions, of ◦ 1st order shells (types 13, 15, 16) ◦ 1st order solids (types -2, -1, 1, 2).	Yes	No
Switch low-order quadrilateral shell elements (types 1, 2, 6, 7, 8, 10, 11) to type 16, solid element type 1 to type 2 for material 83, and beam types 4 and 5 to type 1. ESORT on *CONTROL_SOLID and *CONTROL_SHELL are still honored, and a nonzero ISOLID/ISHELL/IBEAM on *CONTROL_IMPLICIT_EIGENVALUE has precedence and thus overrides the automatic switch mentioned here.	Yes	No
Activate solid element 13 for all materials	Yes	Yes
Improved representation of torsional and rotational modes in eigenvalue analysis for beams (types 1, 2, and 13). See ITORM = 2	Yes	No

Element features activated for IACC = 1	Implicit	Explicit
on *SECTION_BEAM.		

Contact features activated for IACC = 1	Implicit	Explicit
Strong objectivity in tied contacts listed on *CONTACT, meaning that large rotations will not induce contact stresses. These contacts also include bending and torsional constraints whenever those are physically justified.	Yes	Yes

Material features activated for IACC = 1	Implicit	Explicit
Large strain/temperature accuracy in some materials (types 4 shells/solids, 60 solids, 106 solids).	Yes	No
Stiffness smoothing of tension/compression transition in material 83, for enhanced implicit convergence characteristics.	Yes	No
Accurate Jacobi iterations in some hyperelastic material models (type 30 solids, 77 solids, 83 solids), for better strain assessment and implicit convergence characteristics.	Yes	No
Fully iterative plasticity in some metallic material models (type 3 solids, 24 shells/solids, 123 shells/solids), for enhanced accuracy and implicit convergence.	Yes	No
Consistent tangent modulus in material 24, accounting for relatively large plastic strains.	Yes	No
Switch FORM to 14 for fabric material, *MAT_FABRIC, when not set to either 14 or -14.	Yes	No
Switch material 57 to an equivalent material 83 whenever applicable, due to better implicit treatment.	Yes	No

Miscellaneous feature activated for IACC=1	Implicit	Explicit
Steady state thermal solution allowed in coupled simulations, for compatibility with implicit mechanical statics; see ATYPE on *CONTROL_THERMAL_SOLVER.	Yes	No
Consistent nodal forces from body loads for high order elements, accounting for isoparametric shape functions; see *LOAD_BODY.	Yes	No
Allow point constraints on rigid body nodes, without transferring it to rigid body center of mass. Use with care, so as to not overconstrain the system; see *BOUNDARY_SPC.	Yes	No

Miscellaneous feature activated for IACC=1	Implicit	Explicit
Strongly objective joint stiffness formulation; see *CONSTRAINED_JOINT_STIFFNESS.	Yes	No
Sparse matrix treatment of control volume airbags for efficiency; see *AIRBAG.	Yes	No

6. **Enabling/disabling SRTFLG with the command line.** The command line option srtflg= activates and deactivates SRTFLG by setting srtflg=1 and srtflg=0, respectively. This command line option overrides the value specified in *CONTROL_ACCURACY

*CONTROL_ACOUSTIC

Purpose: Define control parameters for transient acoustic solutions.

Card 1	1	2	3	4	5	6	7	8
Variable	MACDVP							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
MACDVP	Calculate the nodal displacements and velocities of *MAT_ACOUSTIC volume elements for inclusion in d3plot and time-history files. EQ.0: Acoustic nodal motions will not be calculated EQ.1: Acoustic nodal motions will be calculated

Remarks:

1. **Solution Costs.** Acoustic simulations using *MAT_ACOUSTIC volume elements (ELFORM = 8 and ELFORM = 14) solve for the displacement potential. The infinitesimal motions of the acoustic nodes can then be found from the gradient of the displacement and velocity potentials. This is purely a post-processing endeavor and has no effect on the predicted pressures and structural response. Calculating these motions will, however, roughly double the cost of the acoustic solution, so it is not done by default.

2. **Model Validity.** The acoustic theory underpinning *MAT_ACOUSTIC volume elements presumes infinitesimal motions. In the presence of larger motions the pressure calculations will proceed regardless, but the calculation of acoustic nodal motions can then be unreliable.

*CONTROL_ACOUSTIC_COUPLING

Purpose: Alter default parameters for keywords *BOUNDARY_ACOUSTIC_COUPLING_MISMATCH and *BOUNDARY_ACOUSTIC_COUPLING_SPECTRAL. Changing these parameters is not generally recommended.

Card 1	1	2	3	4	5	6	7	8
Variable	MACCPL	ACECF1	ACECF2	ACECF3	ACECF4			
Type	I	F	F	I	F			
Default	3	1.5	0.79	0.5	0.95			

VARIABLE	DESCRIPTION
MACCPL	Coupling method: EQ.3: Projection with areal equilibration to enhance enforcement of the zero moments. The equilibration test examines the moments generated by the coupling matrix when the acoustic pressure is constant. If these moments are not zero (rare), then an attempt is made to adjust the coupling coefficients, so the moments are minimized. EQ.4: Projection with no areal equilibration
ACECF1	Multiplier on proximity test
ACECF2	Angle between normal vectors in an orientation test
ACECF3	Multiplier on ceiling test
ACECF4	Area equilibration threshold. The equilibration test is skipped when the accumulated area of the coupling matrix is less than the area of the structural face. This normally occurs with partial coverage.

*CONTROL_ACOUSTIC_SPECTRAL

Purpose: Request an acoustic spectral element analysis instead of the default isoparametric, acoustic finite element analysis. This keyword is only available for double precision. See Appendix W for a list of keywords this feature supports.

Card 1	1	2	3	4	5	6	7	8
Variable	MASEORD	MASEHRF	MASEKFL	MASEIGX	MASEPLT			
Type	I	I	I	I	I			
Default	none	0	0	1	0			

VARIABLE	DESCRIPTION
MASEORD	Spectral element integration order (2 ≤ MASEORD ≤15). See Remark 2.
MASEHRF	Optional *h*-refinement (see Remark 3): EQ.0: No splitting unless tetrahedra or pentahedra are present EQ.1: Split all elements once into hexahedra EQ.2: Split each element a second time into 8 hexahedra EQ.3: Split each element a second time into 27 hexahedra
MASEKFL	Dump flag for *h*-refined and spectral element meshes: EQ.1: Dump keyword deck of acoustic mesh after *h*-refinement EQ.10: Dump keyword deck of spectral acoustic element mesh (generated by LS-DYNA) EQ.11: Dump both meshes for review
MASEIGX	Approach to element time step calculation (see Remark 4): EQ.1: Gerschgorin theorem EQ.2: Maximum element eigenvalue
MASEPLT	Flag to output a high-resolution plot state format: EQ.0: Not output EQ.4: Ensight binary

Remarks:

1. **Elements Types.** This keyword applies to all elements with ELFORM = 8 in the model. Those elements may be hexahedra, tetrahedra, or pentahedra. No acoustic pyramids may be used in spectral element solutions.

2. **Integration Order.** Spectral elements have extra degrees of freedom. Second-order integration results in 27 degrees of freedom per element. 15^{th} order integration results in 4096 degrees of freedom per element. Only the degrees of freedom at the corner nodes are visible to you for d3plot visualization. One element per wavelength with 8^{th} order integration typically gives a very accurate solution over hundreds of cycles of time.

3. ***h*-refinement.** If pentahedra or tetrahedra are used anywhere in the acoustic fluid mesh, then all acoustic fluid elements are split once into hexahedra. This feature supports additional splitting. For example, you may use it to accommodate the extreme mesh refinement that is often required in ultrasonic wave propagation problems. With this field you do not have to generate and manipulate an extremely large and unwieldy mesh.

4. **Critical Time Step.** The Gerschgorin theorem is a faster estimation method and will yield a more conservative time step. Typically, the conservative time step is also less dispersive and more accurate.

*CONTROL_ADAPSTEP

Purpose: Define control parameters for contact interface force update during each adaptive cycle.

Card 1	1	2	3	4	5	6	7	8
Variable	FACTIN	DFACTR						
Type	F	F						
Default	1.0	0.01						

VARIABLE	DESCRIPTION
FACTIN	Initial relaxation factor for contact force during each adaptive remesh. To turn this option off set FACTIN = 1.0. Unless stability problems occur in the contact, FACTIN = 1.0 is recommended since this option can create some numerical noise in the resultant tooling forces. A typical value for this parameter is 0.10.
DFACTR	Incremental increase of FACTIN during each time step after the adaptive step. FACTIN is not allowed to exceed unity. A typical value might be 0.01.

Remarks:

1. **Contact Types.** This command applies to contact with thickness offsets including contact types:

   *CONTACT_FORMING_…_

   *CONTACT_NODES_TO_SURFACE_

   *CONTACT_SURFACE_TO_SURFACE

   *CONTACT_ONE_WAY_SURFACE_TO_SURFACE.

*CONTROL_ADAPTIVE

Purpose: Activate adaptive meshing for applications, such as sheet metal forming or bulk metal forming. The field ADPOPT in *PART identifies the part(s) whose mesh is to be adapted and, to some extent, the type of adaptivity applied. The field ADPTYP in this keyword, *CONTROL_ADAPTIVE, also helps determine the type of mesh adaptivity to perform. For alternative forms of adaptivity based solely on mesh refinement, see *CONTROL_REFINE_*OPTION*. Note that remeshing hyperelastic materials or material models based on a total Lagrangian formulation may lead to numerical instabilities or inaccurate results.

Available adaptivity types:

- Shell splitting or shell *h*-adaptivity
- Solid *r*-adaptivity in which a new mesh of tetrahedrons is created
- *r*-adaptivity of continuum shells in which a new mesh of quads is created
- Refinement of a composite sandwich, comprised of a solid core merged to shell face plates, by setting the fields IFSAND to 1 and ADPTYP to 1 or 2
- 3D axisymmetric (or orbital) adaptivity of axisymmetric geometries comprised of hexahedral and/or pentahedral solids
- Shell mesh fusion

Related keywords include:

1. *CONTROL_ADAPTIVE_CURVE: refines mesh along a curve. This feature tends to generate too many elements along the curve and therefore, negatively affects the computational speed.

2. *DEFINE_CURVE_TRIM: when used together with *CONTROL_ADAPTIVE_CURVE, pre-refines mesh in the area within a specific distance (TCTOL) from both sides of a curve. With this feature you have more control over how many elements will be generated along the curve.

3. *DEFINE_BOX_ADAPTIVE: uses one box for fission and another box for fusion. Boxes can translate or remain stationary. Applies to both shell *h*-adaptivity and tetrahedral *r*-adaptivity.

4. *DEFINE_BOX_NODES_ADAPTIVE: defines a moving tube along a tool path for mesh fission in front of the tool and fusion behind the tool. This is useful in cases where the tool path is curved, such as in roller hemming simulations.

5. *DEFINE_CURVE_BOX_ADAPTIVITY: defines a polygon adaptive box within which you can control mesh refinement level. This feature is useful when deformation is concentrated in a localized region (such as in a line die simulation).

Card Summary:

Card 1a. Include this card if ADPTYP = 1, 2 or 4 (*h*-adaptivity for shells).

ADPFREQ	ADPTOL	ADPTYP	MAXLVL	TBIRTH	TDEATH	LCADP	IOFLAG

Card 1b. Include this card if ADPTYP = 7 (3D *r*-adaptive remeshing of solid elements).

ADPFREQ		ADPTYP		TBIRTH	TDEATH	LCADP	ADPD3P

Card 1c. Include this card if |ADPTYP| = 8 (2D *r*-adaptive remeshing for plane stress, plane strain, and axisymmetric continuum elements).

ADPFREQ	ADPTOL	ADPTYP	MAXLVL	TBIRTH	TDEATH	LCADP	

Card 2a. This card is optional. It is read if ADPTYP = 1, 2, or 4.

ADPSIZE	ADPASS	IREFLG	ADPENE	ADPTH	MEMORY	ORIENT	MAXEL

Card 2b. This card is optional. It is read if ADPTYP = 7.

ADPSCL			ADPENE	ADPDAM	MEMORY	ADPMIC	ADPTGEN

Card 2b.1. Include this card if ADPDAM > 1 on Card 2b.

DAM1							

Card 2c. This card is optional. It is read if |ADPTYP| = 8

ADPSIZE	ADPASS				MEMORY		MAXEL

Card 3a. This card is optional. It is read if ADPTYP = 1, 2, or 4.

IADPN90	IADPGH	NCFREQ	IADPCL	ADPCTL	CBIRTH	CDEATH	LCLVL

Card 3b. This card is optional and blank. It is read if ADPTYP = 7 (for ADPDAM < 2), -8, or 8. Do not include this card if ADPDAM > 1 on Card 2b.

Card 4a. This card is optional. It is read if ADPTYP = 1 or 2.

					D3TRACE		IFSAND

Card 4b. This card is optional. It is read if ADPTYP = 4.

				ADPERR	D3TRACE		

Card 4c. This card is optional. It is read if ADPTYP = 7.

					D3TRACE	IADPCF	

Card 4d. This card is optional. It is read if |ADPTYP| = 8.

CNLA			MMM2D		D3TRACE		

Card 5. This card is optional.

INMEMRY							

Data Card Definitions:

This card is included if ADPTYP = 1, 2, or 4.

Card 1a	1	2	3	4	5	6	7	8
Variable	ADPFREQ	ADPTOL	ADPTYP	MAXLVL	TBIRTH	TDEATH	LCADP	IOFLAG
Type	F	F	I	I	F	F	I	I
Default	none	10^{20}	1	3	0.0	10^{20}	0	0

VARIABLE	DESCRIPTION
ADPFREQ	Time interval between adaptive refinements; see Figures 12-1 and 12-2.
ADPTOL	Adaptive error tolerance; ADPTOL is in degrees for ADPTYP set to 1 or 2
ADPTYP	Adaptive options. ADPTYP = 1, 2 and 4 refer to *h*-adaptivity for shells. ADPTYP = 1 and 2 refer to *h*-adaptivity for shell / solid / shell sandwich composites. EQ.1: Angle change in degrees per adaptive refinement relative to the surrounding shells for each shell to be refined EQ.2: Total angle change in degrees relative to the surrounding shells for each shell to be refined. For example, if APDTOL = 5 degrees, the shell will be refined to the second level when the total angle change reaches 5 degrees. When the angle change is 10 degrees, the shell will be

VARIABLE	DESCRIPTION
	refined to the third level. EQ.4: Adapts when the shell error in the energy norm, Δe, exceeds ADPTOL/100 times the mean energy norm within the part, which is estimated as: $$\Delta e = \left(\int_{\Omega_e} \frac{\|\Delta\sigma\|^2}{E} d\Omega \right)^{1/2} .$$ Here E is the Young's modulus. The error of the stresses, $\Delta\sigma$, is defined as the difference between the recovered solution $\sigma^\star$ and the numerical solution, σ^h, that is, $\Delta\sigma \equiv \sigma^\star - \sigma^h$. Various recovery techniques for $\sigma^\star$ and error estimators for Δe are defined by ADPERR. This option works for shell types 2, 4, 16, 18, and 20.
MAXLVL	Maximum number of refinement levels. Values of 1, 2, 3, 4, … allow a maximum of 1, 4, 16, 64, … shells, respectively, to be created for each original shell. The refinement level can be overridden by *DEFINE_BOX_ADAPTIVE or *DEFINE_SET_ADAPTIVE.
TBIRTH	Birth time at which the adaptive remeshing begins; see Figures 12-1 and 12-2.
TDEATH	Death time at which the adaptive remeshing ends; see Figures 12-1 and 12-2.
LCADP	Load curve ID, defining how the adaptive interval is changed as a function of time. If this option is nonzero, the ADPFREQ will be replaced by LCADP. The x-axis is time, and the y-axis is the varied adaptive time interval.
IOFLAG	Flag to generate adaptive mesh at exit, including *NODE, *ELEMENT_SHELL_THICKNESS, *BOUNDARY_*OPTION*, and *CONSTRAINED_ADAPTIVITY, which is to be saved in the file, adapt.msh: EQ.1: Generate h-adapted mesh.

Include this card if ADPTYP = 7.

Card 1b	1	2	3	4	5	6	7	8
Variable	ADPFREQ		ADPTYP		TBIRTH	TDEATH	LCADP	ADPD3P
Type	F		I		F	F	I	I
Default	none		1		0.0	10^{20}	0	0

VARIABLE	DESCRIPTION
ADPFREQ	Time interval between adaptive refinements; see Figures 12-1 and 12-2.
ADPTYP	Adaptive options: EQ.7: 3D *r*-adaptive remeshing for solid elements. The adaptive remeshing process uses tetrahdrons (solid formulation 10 or 13, or if EFG, formulation 42), or in the case of 3D axisymmetry (orbital) adaptivity, hexahedral and pentahedral elements. The process generates a completely new mesh initialized from the old mesh using a least squares approximation. The algorithm bases the mesh size on the minimum and maximum edge lengths defined on the input for *CONTROL_REMESHING. This option remains under development, and we are improving its reliability on complex geometries.
TBIRTH	Birth time at which the adaptive remeshing begins; see Figures 12-1 and 12-2.
TDEATH	Death time at which the adaptive remeshing ends; see Figures 12-1 and 12-2.
LCADP	Load curve ID, defining how the adaptive time interval is changed as a function of time. If this option is nonzero, LCADP replaces ADPFREQ. The *x*-axis is time, and the *y*-axis is the varied adaptive time interval.
ADPD3P	Flag to control the output of an adaptive mesh and d3plot in 3D *r*-adaptivity: EQ.0: Do not output an adaptive mesh output at exit and do not write d3plot files before and after each adaptive step.

VARIABLE	DESCRIPTION
	EQ.1: Output the adaptive mesh at exit, but do not write d3plot files before and after each adaptive step.
	EQ.2: Output d3plot files after each adaptive step, but do not output an adaptive mesh at exit.
	EQ.3: Output d3plot files before and after each adaptive step, but do not output an adaptive mesh at exit. Use this option for metal flow lines.

Include this card if |ADPTYP| = 8.

Card 1c	1	2	3	4	5	6	7	8
Variable	ADPFREQ	ADPTOL	ADPTYP	MAXLVL	TBIRTH	TDEATH	LCADP	
Type	F	F	I	I	F	F	I	
Default	none	10^{20}	1	3	0.0	10^{20}	0	

VARIABLE	DESCRIPTION
ADPFREQ	Time interval between adaptive refinements; see Figures 12-1 and 12-2.
ADPTOL	Characteristic element size
ADPTYP	Adaptive options: EQ.±8: 2D *r*-adaptive remeshing for plane stress, plane strain, and axisymmetric continuum elements, that is, shell formulations 12 through 15. A completely new mesh is generated which is initialized from the old mesh using a least squares approximation. The mesh size is currently based on the value, ADPTOL, which gives the characteristic element size. This option is based on earlier work by Dick and Harris [1992]. If ADPTYP is negative, then self-contacting material will not be merged together. The self-merging is often preferred since it eliminates sharp folds in the boundary; however, if the sharp fold is being simulated, unexpected results are generated.
MAXLVL	Maximum number of refinement levels. Values of 1, 2, 3, 4, … allow a maximum of 1, 4, 16, 64, … shells, respectively, to be created

VARIABLE	DESCRIPTION
	for each original shell. The refinement level can be overridden by *DEFINE_BOX_ADAPTIVE or *DEFINE_SET_ADAPTIVE.
TBIRTH	Birth time at which the adaptive remeshing begins; see Figures 12-1 and 12-2.
TDEATH	Death time at which the adaptive remeshing ends; see Figures 12-1 and 12-2.
LCADP	Load curve ID, defining how the adaptive interval is changed as a function of time. If this option is nonzero, LCADP replaces ADPFREQ. The *x*-axis is time, and the *y*-axis is the varied adaptive time interval.

This card is optional. The keyword reader reads it if ADPTYP = 1, 2, or 4.

Card 2a	1	2	3	4	5	6	7	8
Variable	ADPSIZE	ADPASS	IREFLG	ADPENE	ADPTH	MEMORY	ORIENT	MAXEL
Type	F	I	I	F	F	I	I	I
Default	inactive	0	0	0.0	inactive	inactive	0	inactive

VARIABLE	DESCRIPTION
ADPSIZE	Minimum shell size to be adapted based on element edge length. If undefined, the edge length limit is ignored. LT.0: Absolute value defines the minimum characteristic element length to be adapted based on square root of the element area, that is, instead of comparing the shortest element edge with ADPSIZE, it compares the square root of the element area with |ADPSIZE|, whenever ADPSIZE is defined by a negative value.
ADPASS	One or two pass flag for *h*-adaptivity: EQ.0: Two pass adaptivity as shown in Figure 12-1 EQ.1: One pass adaptivity as shown in Figure 12-2
IREFLG	If positive, the mesh is refined uniformly by IREFLG levels at time = TBIRTH. A value of 1, 2, 3, … creates 4, 16, 64, … shells,

VARIABLE	DESCRIPTION
	respectively, for each original shell. MAXLVL must be greater than or equal to IREFLG for this to work. If negative, |IREFLG| is taken as a curve ID. The curve specifies the minimum element size as a function of time. If the ordinate values (minimum element size) are positive, those values will override other element size criteria. If the ordinate values are negative, the absolute value of the ordinate is the element size used for refinement.
ADPENE	For shells, *h*-adapt the mesh when the FORMING contact surfaces approach or penetrate the tooling surface depending on whether the value of ADPENE is positive (*approach*) or negative (*penetrates*), respectively. The tooling adaptive refinement is based on the curvature of the tooling. If ADPENE is positive the refinement generally occurs before contact takes place; consequently, it is possible that the parameter ADPASS can be set to 1 in invoke the one pass adaptivity.
ADPTH	Thickness below which adaptive remeshing begins: EQ.0.0: This parameter is ignored. GT.0.0: Absolute shell thickness level below which adaptive remeshing should begin. LT.0.0: |ADPTH| is the element thickness reduction ratio. If the ratio of the element thickness to the original element thickness is less than 1.0 + ADPTHK, the element will be refined. This option works only if ADPTOL is nonzero. If thickness-based adaptive remeshing is desired without angle changes, then set ADPTOL to a large angle for ADPTYP = 1 or 2.
MEMORY	This flag can have two meanings depending on whether the memory environmental variable is or is not set. The command "setenv LSTC_MEMORY auto" (or for bourne shell "export LSTC_MEMORY=auto") sets the memory environmental variable which causes the solver to expand memory automatically. Note that automatic memory expansion is not always 100% reliable depending on the machine and operating system level; consequently, it is not yet the default. To see if this is set on a particular machine type the command "env". If the environmental variable *is not set* then when memory usage reaches this percentage, MEMORY, further adaptivity is prevented to avoid exceeding the memory

VARIABLE	DESCRIPTION
	specified at execution time. Caution is necessary since memory usage is checked after each adaptive step, and, if the memory usage increases by more than the residual percentage, 100-PERCENT, the calculation will terminate. If the memory environmental variable *is set*, then when the number of words of memory allocated reaches or exceeds this value, MEMORY, further adaptivity is stopped.
ORIENT	This option applies to the FORMING contact option only. If this flag is set to one (1), the user orientation for the contact interface is used. If this flag is set to zero (0), the solver sets the global orientation of the contact surface the first time a potential contact is observed after the birth time. If tracked nodes are found on both sides of the contact surface, the orientation is set based on the principle of "majority rules." Experience has shown that this principle is not always reliable.
MAXEL	If this number of shells is exceeded, adaptivity is stopped.

This card is optional. It is read if ADPTYP = 7.

Card 2b	1	2	3	4	5	6	7	8
Variable	ADPSCL			ADPENE	ADPDAM	MEMORY	ADPMIC	ADPTGEN
Type	F			F	I	I	I	F
Default	0.0			0.0	0	inactive	0	0.0

VARIABLE	DESCRIPTION
ADPSCL	Strain rate scale factor. See Remark 12. EQ.0.0: No strain rate scaling GT.0.0: Strain rate scale factor
ADPENE	For three-dimensional *r*-adaptive solid remeshing (ADPOPT = 2 in *PART), the mesh refinement is based on the curvature of the tooling when ADPENE is positive. See Remark 9.
ADPDAM	Type of damage accumulation in the workpiece (ee Remark 11):

VARIABLE	DESCRIPTION
	EQ.0: No damage accumulation EQ.1: Ratio of effective plastic strain to failure plastic strain EQ.2: Cockcroft-Latham damage
MEMORY	This flag can have two meanings depending on whether the memory environmental variable is or is not set. The command "setenv LSTC_MEMORY auto" (or for bourne shell "export LSTC_MEMORY=auto") sets the memory environmental variable which causes the solver to expand memory automatically. Note that automatic memory expansion is not always 100% reliable depending on the machine and operating system level; consequently, it is not yet the default. To see if this is set on a particular machine type the command "env". If the environmental variable *is not set* then when memory usage reaches this percentage, MEMORY, further adaptivity is prevented to avoid exceeding the memory specified at execution time. Caution is necessary since memory usage is checked after each adaptive step, and, if the memory usage increases by more than the residual percentage, 100-PERCENT, the calculation will terminate. If the memory environmental variable *is set*, then when the number of words of memory allocated reaches or exceeds this value, MEMORY, further adaptivity is stopped.
ADPMIC	Option to compute microstructure evolution in hot forging simulations: EQ.0: No microstructure evolution (default) EQ.1: Activate microstructure evolution. Use *MAT_ADD_MICRO to input material constants
ADPTGEN	Option for selecting the tetrahedral volume remesher: EQ.-1.0: Do not use the alternative remesher. EQ.0.0: Use the alternative remesher when the classic remesher fails (default). EQ.1.0: Use the alternative remesher with default meshing parameter 1.06. GT.1.0: Use the alternative remesher with meshing parameter ADPTGEN. See Remark 14.

Include this card ADPTYP = 7 if ADPDAM > 1.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	DAM1							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
DAM1	Critical Cockcroft-Latham damage value (include if ADPDAM = 2)

This card is optional. It is read if |ADPTYP| = 8.

Card 2c	1	2	3	4	5	6	7	8
Variable	ADPSIZE	ADPASS				MEMORY		MAXEL
Type	F	I				I		I
Default	inactive	0				inactive		inactive

VARIABLE	DESCRIPTION
ADPSIZE	Minimum shell size to be adapted based on element edge length. If undefined, the edge length limit is ignored. LT.0: Absolute value defines the minimum characteristic element length to be adapted based on square root of the element area, that is, instead of comparing the shortest element edge with ADPSIZE, it compares the square root of the element area with |ADPSIZE|, whenever ADPSIZE is defined by a negative value.
ADPASS	One or two pass flag for *h*-adaptivity: EQ.0: Two pass adaptivity as shown in Figure 12-1 EQ.1: One pass adaptivity as shown in Figure 12-2
MEMORY	This flag can have two meanings depending on whether the memory environmental variable is or is not set. The command

VARIABLE	DESCRIPTION
	"setenv LSTC_MEMORY auto" (or for bourne shell "export LSTC_MEMORY=auto") sets the memory environmental variable which causes the solver to expand memory automatically. Note that automatic memory expansion is not always 100% reliable depending on the machine and operating system level; consequently, it is not yet the default. To see if this is set on a particular machine type the command "env". If the environmental variable *is not set* then when memory usage reaches this percentage, MEMORY, further adaptivity is prevented to avoid exceeding the memory specified at execution time. Caution is necessary since memory usage is checked after each adaptive step, and, if the memory usage increases by more than the residual percentage, 100-PERCENT, the calculation will terminate. If the memory environmental variable *is set* then when the number of words of memory allocated reaches or exceeds this value, MEMORY, further adaptivity is stopped.
MAXEL	If this number of shells is exceeded, adaptivity is stopped.

This card is optional. It is read if ADPTYP = 1, 2, or 4.

Card 3a	1	2	3	4	5	6	7	8
Variable	IADPN90	IADPGH	NCFREQ	IADPCL	ADPCTL	CBIRTH	CDEATH	LCLVL
Type	I	I	I	I	F	F	F	I
Default	0	0	none	1	none	0.0	10^{20}	0

VARIABLE	DESCRIPTION
IADPN90	Fission control flag around radii: GT.0: Maximum number of shells after fission covering the entire radius from starting tangent to ending tangent EQ.-1: This setting works with look-forward adaptivity, making more consistent mesh adaptivity along the radius from starting tangent to ending tangent. The actual number of elements covering the radius, will be controlled by ADPSIZE and MAXLVL. Note this setting also works to prevent the "kinks" that are likely to happen along the draw

VARIABLE	DESCRIPTION
	wall in the deep drawing scenario, under which the parameter ADPFREQ needs to be set fine enough for fission as the blank draws into the die radius. Also see Remark 5.
IADPGH	Fission flag for neighbor splitting: EQ.0: Split all neighbor shells EQ.1: Do not split neighbor shells
NCFREQ	Frequency of fission to fusion steps. For example, if NCFREQ = 4, then fusion will occur on the fourth, eighth, twelfth, etc., fission steps, respectively. If this option, is used NCFREQ > 1 is recommended.
IADPCL	Fusion will not occur until the fission level reaches IADPCL. Therefore, if IADPCL = 2 and MAXLVL = 5, any shell can be split into 256 shells. If the surface flattens out, the number of elements will be reduced if the fusion option is active, i.e., the 256 elements can be fused and reduced to 16.
ADPCTL	Adaptivity error tolerance in degrees for activating fusion. It follows the same rules as ADPTYP as defined in Card 2a.
CBIRTH	Birth time for adaptive fusion. If ADPENE > 0, look-ahead adaptivity is active. In this case, fission, based on local tool curvature, will occur while the blank is still relatively flat. The time value given for CBIRTH should be set to a time later in the simulation after the forming process is well underway.
CDEATH	Death time for adaptive fusion
LCLVL	Load curve ID of a curve that defines the maximum refinement level as a function of time

This card is optional and blank. It is read if ADPTYP = 7 (for ADPDAM < 2), 8, or -8. Include this card if including Cards 4c or 4d. Do not include this card if ADPDAM > 1 for ADPTYP = 7.

Card 3b	1	2	3	4	5	6	7	8
Variable								
Type								

This card is optional. It is read if ADPTYP = 1 or 2.

Card 4a	1	2	3	4	5	6	7	8
Variable						D3TRACE		IFSAND
Type						I		I
Default						0		0

VARIABLE	DESCRIPTION
D3TRACE	Output flag: EQ.0: No additional output states EQ.1: A d3plot state will be output just before and after an adaptive step even though it may not be requested. You may want this output so that the LS-PrePost particle trace algorithm will work in the case of adaptivity.
IFSAND	Set this flag to "1" for forming of sandwich composites. For details, see Remark 6.

This card is optional. It is read if ADPTYP = 4.

Card 4b	1	2	3	4	5	6	7	8
Variable					ADPERR	D3TRACE		
Type					I	I		
Default					0	0		

VARIABLE	DESCRIPTION
ADPERR	3-digit number, as "XYY", where "X" and "YY" define the options for the recovery techniques and the error estimators, respectively: For X: EQ.0: Superconvergent patch recovery (SPR) (default) EQ.1: The least square fit of the stress to the nodes (Global L2) EQ.2: Error density SPR, as $\Delta\tilde{e} = \Delta e / \text{Area}_{\text{element}}$ EQ.3: Self-weighted SPR, as $\Delta\hat{e} = \sqrt{\Delta e \times e}$ For YY: EQ.00: Energy norm (default) EQ.01: Cauchy σ_x EQ.02: σ_y EQ.03: σ_z EQ.04: τ_{xy} EQ.05: τ_{yz} EQ.06: τ_{zx} EQ.07: Effective plastic strain, ε_{ep} EQ.08: Pressure EQ.09: von Mises EQ.10: Principal deviator stress S_{11} EQ.11: S_{22} EQ.12: S_{33} EQ.13: Tresca

VARIABLE	DESCRIPTION
	EQ.14: Principal stress σ_{11} EQ.15: σ_{22} EQ.16: σ_{33} EQ.20: User subroutine uadpval to extract the numerical solutions for recovery and uadpnorm to provide an error estimator.
D3TRACE	Output flag: EQ.0: No additional output states EQ.1: A d3plot state will be output just before and after an adaptive step even though it may not be requested. You may want this output so that the LS-PrePost particle trace algorithm will work in the case of adaptivity.

This card is optional. It is read if ADPTYP = 7.

Card 4c	1	2	3	4	5	6	7	8
Variable						D3TRACE	IADPCF	
Type						I	I	
Default						0	0	

VARIABLE	DESCRIPTION
D3TRACE	Output flag: EQ.0: No additional output states EQ.1: A d3plot state will be output just before and after an adaptive step even though it may not be requested. You may want this output so that the LS-PrePost particle trace algorithm will work in the case of adaptivity.
IADPCF	Flag to enable adaptive user control files: EQ.0: No user control files EQ.1: Perform run-time control on 3D adaptivity through control files

This card is optional. It is read if |ADPTYP| = 8.

Card 4d	1	2	3	4	5	6	7	8
Variable	CNLA			MMM2D		D3TRACE		
Type	F			I		I		
Default	110.0			0		0		

VARIABLE	DESCRIPTION
CNLA	Limit angle for corner nodes. See Remark 10. GT.0.0: CNLA is the limit angle. Simplified boundary lines for straight sections are used as remeshing basis. LT.0.0: |CNLA| is the limit angle and accurate boundary lines are used as remeshing basis (recommended, with CNLA = -110 being a good choice).
MMM2D	If non-zero, common boundaries of all adapted parts will be merged. This is true even if the parts did not share a boundary at the beginning of the calculation but come into contact later.
D3TRACE	Output flag: EQ.0: No additional output states EQ.1: A d3plot state will be output just before and after an adaptive step even though it may not be requested. You may want this output so that the LS-PrePost particle trace algorithm will work in the case of adaptivity.

This card is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	INMEMRY							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
INMEMRY	Flag to determine the way shell *h*-adaptivity is performed (see Remark 8): EQ.0: Traditional out-of-core adaptivity (default). EQ.1: In-core adaptivity (under development). This approach is only supported in MPP and only for ADPTYP = 2. It does not apply to composite sandwich *h*-adaptivity.

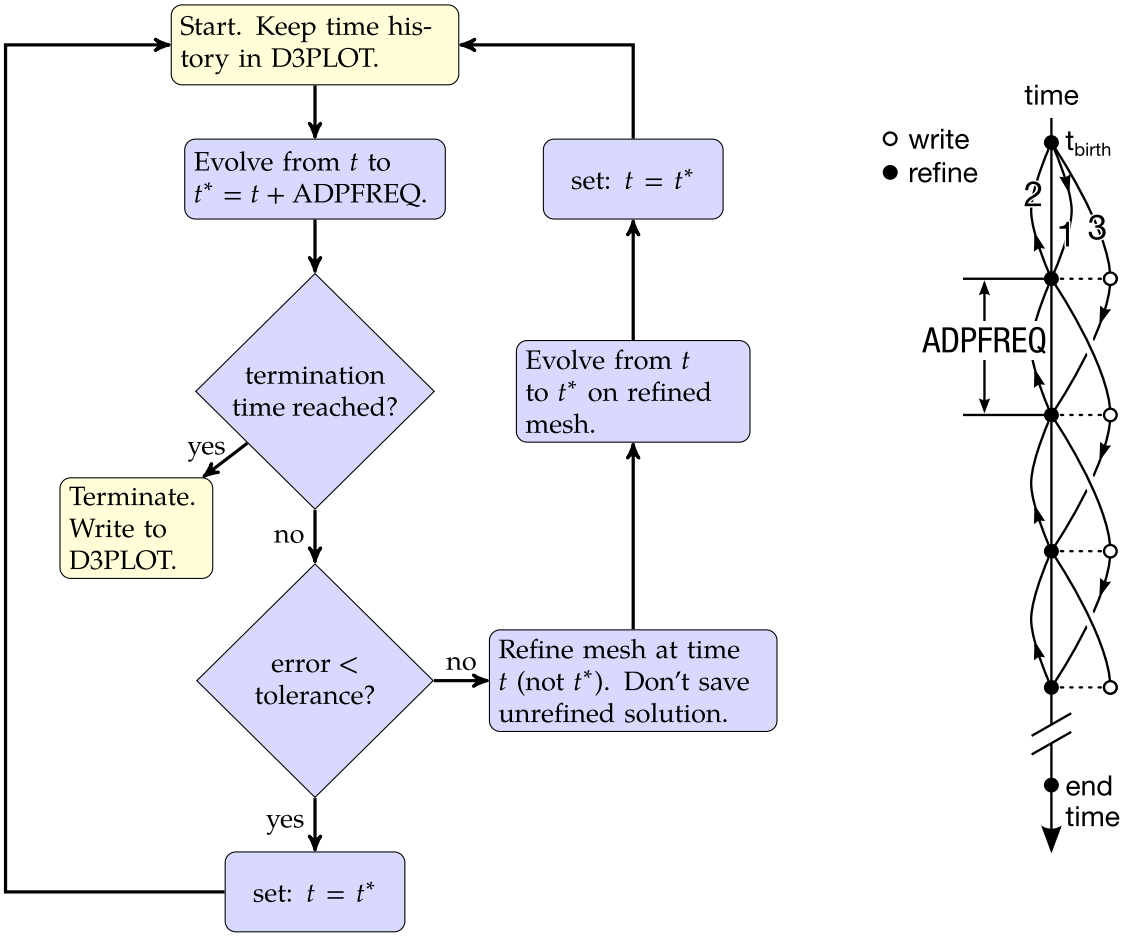


Figure 12-1. Flowchart for ADPASS = 0. While this option is *sometimes* more accurate, ADPASS = 1 is *much* less expensive and recommended when used *with* ADPENE.

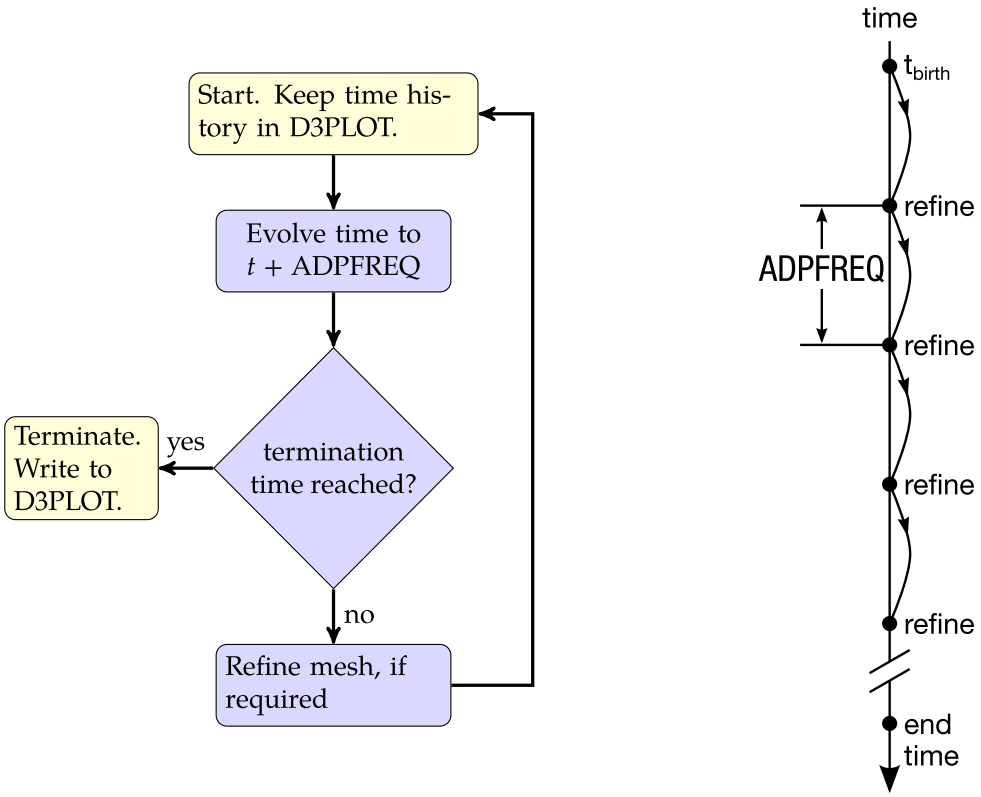


Figure 12-2. Flow chart for ADPASS = 1. This algorithm may be summarized as "periodically refine." This method is recommended over ADPASS = 0 when used *with* ADPENE, which implements look ahead.

Remarks:

Remarks about *h*-adaptivity

1. **Restarting.** The d3dump and runrsf files contain all information necessary to restart an adaptive run. This did not work in version 936.

2. **Related field in *PART.** For this control card to work, the field ADPOPT = 1 must be set in the *PART definition. Otherwise, adaptivity will not function.

3. **Contact types and options.** For adaptivity to work optimally, the parameter SNLOG = 1, must be set on Optional Card B in the *CONTACT Section. On disjoint tooling meshes the contact option *CONTACT_FORMING_… is recommended.

4. **Root ID (RID) file.** After the adaptive run is completed, a file named adapt.rid is left on disk. This file contains the root ID of all elements created during the calculation, and it does not need to be kept if it is not used in post-processing.

5. **Note about IADPN90 field.** For all metal forming simulations, set IADPN90 to -1.

6. **Mesh adaptivity for sandwiched parts.** Mesh adaptivity can be applied to sandwich composites consisting of layer(s) of solid elements (core) sandwiched by one layer of shell elements each on the top and bottom surfaces of the core. Nodes must be shared at the solid-to-shell interfaces. Prior to R11.0, this mesh adaptivity is limited to only one layer of solid elements with in-plane mesh refinements for both solids and shells. Starting with R11.0 [Zhu et al, 2017], it applies to multiple layers of solid elements; see Figure 12-3.

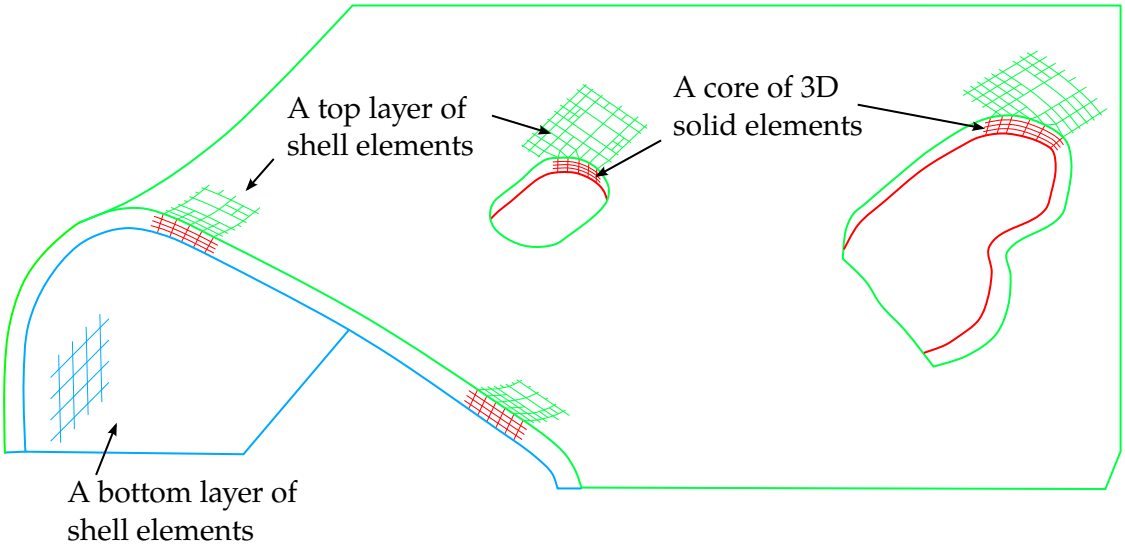


Figure 12-3. Activate sandwich composite mesh adaptivity by setting IFSAND = 1. Before R11.0, adaptivity is limited to only one layer of solid elements; starting with R11.0 adaptive meshing applies to multiple layers of solid elements.

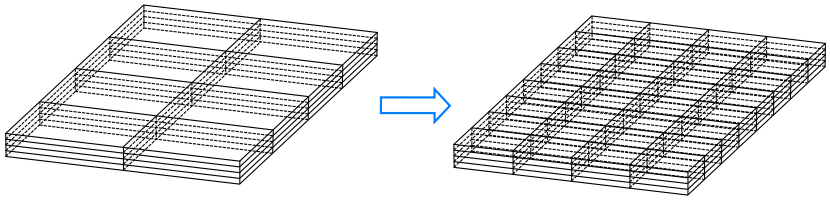


Figure 12-4. Mesh fission for sandwich part. Sweeping the top and bottom adaptive shell elements through the thickness, a total of 32 hexahedral elements are refined into 128 of the same type. The number of layers of the solids always remains the same.

After adaptive refinement of the shell faces, the solid elements of the core are created by sweeping the shells through the thickness (Figure 12-4). So, if quad shells are used and refined into smaller quads, matching hexahedral solids are created for the core. Similarly, if triangular shells are used and refined into

smaller triangular shells, matching pentahedral solids are created for the core. The number of layers of the solids in the core does not change due to adaptivity. By allowing multiple layers of solids in the thickness direction, different materials may be used through the core thickness. Moreover, this adaptive approach serves to provide better resolution in local areas of interest while keeping the computational cost to a reasonable level.

The adapted sandwich composite may be trimmed by setting ITYP = 1 in *CONTROL_FORMING_TRIMMING and with *DEFINE_CURVE_TRIM. Trimming of sandwiched parts allows for multiple layers of solids.

In a typical forming setup, the following cards need to be changed to activate the sandwiched part mesh adaptivity:

```
*CONTROL_ADAPTIVE
$# adpfreq    adptol    adptyp    maxlvl    tbirth    tdeath     lcadp    ioflag
   0.00223       4.0         2         4     0.0001.0000E+20         0         0
$# adpsize    adpass    ireflg    adpene     adpth    memory    orient     maxel
   0.90000         1           10.00000     0.000         0         0         0
$# ladpn90    ladpgh    ncfred    ladpcl    adpctl    cbirth    cdeath     lclvl
        -1         0         0         1     0.000     0.0001.0000E+20         0
$                                                                        IFSAND
                                                                              1
*PART
Mid-core layer of solid elements
$       PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1                                       1
Top layer of shell elements
       100       100         1                                       1
Bottom layer of shell elements
       101       100         1                                       1
```

Note: IFSAND in *CONTROL_ADAPTIVE is set to “1” to activate the composite sandwich adaptivity; each ADPOPT under *PART is set to “1” to activate the adaptivity.

7. **Mesh fusion in MPP.** Starting with R11.0, mesh fusion in MPP is available. A mesh fusion example is shown in Figure 12-5, and a partial keyword example is provided below:

```
*CONTROL_ADAPTIVE
$  ADPFREQ    ADPTOL    ADPTYP    MAXLVL    TBIRTH    TDEATH     LCADP    IOFLAG
    0.0024       4.0         2         3       0.0      70.0         0         1
$  ADPSIZE    ADPASS    IREFLG    ADPENE    ADPTH     MEMORY     ORIENT   MAXEL
       0.9         1         0       5.0       0.0
$  IADPN90    IADPGH    NCFREQ    IADPCL    ADPCTL    CBIRTH     CDEATH   LCLVL
        -1         0         2         0       8.0       0.00       70.0
```

In this keyword, NCFREQ defines the fusion frequency, ADPCTL defines the fusion criterion, and CBIRTH and CDEATH define when the fusion starts and ends, respectively.

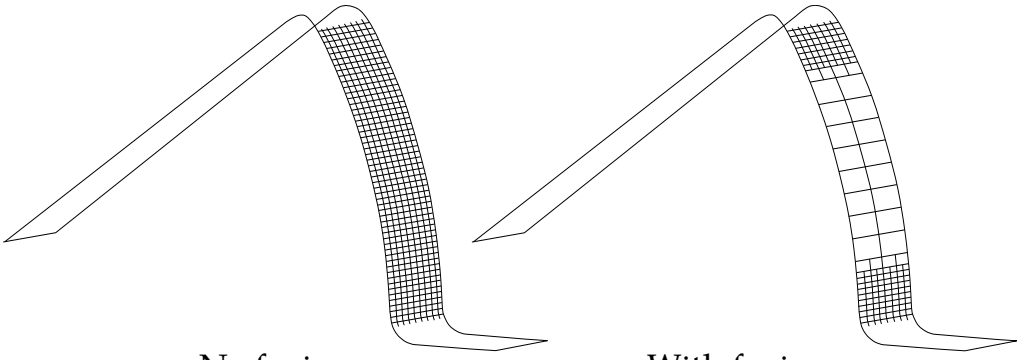


Figure 12-5. Comparison of final mesh pattern between adaptive mesh without fusion and with fusion, on a hat-section draw-bending (half-symmetric model shown).

Based on an extensive study [Fan et al, 2017], the fusion feature reduces the computation time notably (average around 25%) and has little effects on formability analysis, such as thinning and effective strain predictions. The mesh fusion effect on springback prediction is found to be smaller than 10%. Therefore, you can apply fusion extensively in all formability related simulations since the leading formability indicators are little affected while the calculation can be sped-up by a factor of 25%. In springback simulations, however, you should use fusion with caution. Using this feature depends on the required simulation accuracy. You can apply this feature if the springback results are to be used for a quick and rough estimation; however, if the results are to be used for compensating dies and for re-machining, then fusion may not be appropriate

8. **Adaptivity algorithm.** Out-of-core adaptivity requires dumping the mesh out to disk and restarting the program with a newly created input deck at each adaptive step. This method is computationally expensive due to the I/O, keyword processing, and MPP decomposition time. In-core adaptivity defined with INMEMRY = 1 does not require exiting the solution loop and can maximize the performance of the CPUs employed. This method is under development, so many features are not yet supported. It is currently supported for basic metal forming. It also maintains the loading condition from *LOAD_SHELL_SET upon adaptivity.

Remarks about *r*-adaptivity

9. **Contact and ADPENE.** In three dimensions when ADPENE > 0, the solid part to be adapted is assumed to be on the SURFA side of a contact while the "tooling", consisting of a shell surface, is assumed to be on the SURFB side of that same contact. ADPENE > 0 represents a distance from the tooling surface within which the adapted mesh refinement of the SURFA part is influenced by the radius of curvature of the tooling surface. This feature is currently *unavailable* in SMP and for SOFT = 2 in *CONTACT.

10. **CNLA.** In two dimensions *r*-adaptive remeshing (|ADPTYP| = 8), the generated mesh should have a node at each corner so that the corners are not smooth. By default, the mesher will assume a corner wherever the interior angle between adjacent edges is smaller than 110 degrees. Setting CNLA larger than 110 enables angles larger than 110 to be corners. Care should be taken to avoid an unnecessarily large value of CNLA as this may prevent the mesher from generating smooth meshes.

11. **Damage accumulation.** The solver calculates the damage value in the workpiece undergoing 3D *r*-adaptivity (ADPTYP = 7) with the criterion specified with ADPDAM. The damage is uncoupled from the material response and indicates critical failure spots in the workpiece. Input the material properties needed to determine damage in the data cards for the material model (such as stress triaxiality dependent failure strain) and set additional constants with Card 2b.1 of this keyword. A history variable stores the damage value. The location of the history variable depends on the material type. The d3hsp file provides the location. For example, it is history variable #28 for *MAT_224 and history variable #22 *MAT_106. NEIPH in *DATABASE_EXTENT_BINARY controls the number of history variables output to d3plot. To output this damage value, set to NEIPH to a value greater than the history variable # for the material under consideration. Currently, this feature is available for *MAT_024, *MAT_106, *MAT_107, and *MAT_224.

12. **Strain rate scaling**. To speed up the forging simulation, the tool velocity is usually increased. For strain-rate-dependent materials, the strain rate needs to be scaled down to obtain the correct stress response in the workpiece. Currently, this feature is available for *MAT_024, *MAT_106, *MAT_107, and *MAT_224.

13. **Limitations of *r*-adaptivity.** In 2D/3D *r*-adaptivity, the adaptive parts are remeshed at each adaptive step, followed by an automatic restart to continue the simulation with the new mesh. However, some features and keywords are not supported or can be restricted for *r*-adaptive simulations. For instance, most keywords defined on old node IDs and element IDs are not supported for *r*-adaptivity, unless explicitly stated otherwise. Predefined MPP decomposition based on an old mesh using *CONTROL_MPP_DECOMPOSITION_FILE, also, does not work for *r*-adaptive simulations. This keyword causes the job to crash during the automatic restart.

14. **ADPTGEN > 1.0 (alternative tetrahedral volume remesher).** When ADPTGEN > 1.0, the alternative remesher adopts the value of ADPTGEN as a meshing quality control parameter. This parameter is the maximum allowable radius-edge ratio for the tetrahedral volume remeshing. Larger values of ADPTGEN cause the remesher to create fewer tetrahedral elements with a faster transition between the fine and coarse regions than smaller values. For problems with localized large deformations, we recommend using smaller values for ADPTGEN.

*CONTROL_ADAPTIVE_CURVE

Purpose: To refine the element mesh along a curve during or prior to a sheet metal forming simulation. All curves defined by the keyword *DEFINE_CURVE_TRIM are used in the refinement. This option provides additional refinement to that generated by *CONTROL_ADAPTIVE. Additionally, pre-mesh refinement along a curve with specific distance/range on both sides of the curve can be modeled when this keyword is used together with *DEFINE_CURVE_TRIM_3D (by activating the field TCTOL). Lastly, this keyword can be used to refine the mesh along a curve during trimming when used together with the keyword *ELEMENT_TRIM. This feature only applies to shell elements and *h*-adaptivity.

Card 1	1	2	3	4	5	6	7	8
Variable	IDSET	ITYPE	N	SMIN	ITRIOPT			
Type	I	I	I	F	I			

VARIABLE	DESCRIPTION
IDSET	Set ID
ITYPE	Set type: EQ.1: IDSET is shell set ID. EQ.2: IDSET is part set ID.
N	Refinement option: EQ.1: Refine until there are no adaptive constraints remaining in the element mesh around the curve, subjected to the maximum refinement level of 5. GT.1: Refine no more than N levels.
SMIN	If the element dimension is smaller than this value, do not refine.
ITRIOPT	Option to refine an enclosed area of a trim curve. EQ.0: Refine the elements along the trim curve. EQ.1: Refine the elements along the trim curve and enclosed by the trim curve. Under the keyword *DEFINE_CURVE_TRIM_3D, the variable TCTOL must be set to "2" and a seed node NSEED1 must be defined inside the curve loop.

Adaptive mesh refinement along a curve during the beginning of a simulation:

The Figure 12-6 top right mesh refinement illustrates mesh adaptivity along an enclosed curve as done by the partial input example below. Since the mesh refinement is controlled by either the refinement level N or smallest element size SMIN, care should be taken so not too many elements are generated in the model.

The partial keyword input example below refines the mesh by four levels along both sides of the curve defined by the IGES file adpcurves.iges. If an element edge length is shorter than 0.3 mm, then the element is not refined.

```
*INCLUDE
drawn.dynain
*DEFINE_CURVE_TRIM_3D
$      TCID     TCTYPE       TFLG       TDIR      TCTOL
         1          2
adpcurves.iges
*CONTROL_ADAPTIVE_CURVE
$     IDSET      ITYPE          N       SMIN
         1          2          4        0.3
```

Adaptive mesh refinement inside a curve loop during the beginning of a simulation:

Figure 12-6 bottom left mesh refinement illustrates refining the mesh inside the closed curve from the partial input below. The original mesh (Figure 12-6 top left) is refined by either six levels or to no smaller than 3.2 mm element edge length inside the curve loop defined by the IGES file area.iges. TCTYPE may be set to either 1 or 2, but TCTOL must be set to 2, and NSEED1 (106877) must be defined inside the curve loop. Note this feature is available starting in Revision 115142.

```
*KEYWORD
*INCLUDE
incoming.dynain
*CONTROL_TERMINATION
0.0
*parameter
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ enter refinement level:
I reflvl            6
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$  enter smallest element length:
R minsize         3.2
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE_TRIM_3D
$      TCID     TCTYPE       TFLG       TDIR      TCTOL     TRDIS     NSEED1
         1          2                                 2               106877
$$$$$       enter IGES curve file name (make sure it's very close to the blank):
area.iges
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_ADAPTIVE_CURVE
$   PartSET      ITYPE          N       SMIN    ITRIOPT
         1          2    &reflvl   &minsize          1
```

Adaptive mesh along a curve with refinement controlled by a distance during the

beginning of a simulation:

This feature was added to limit the number of elements created.

When *CONTROL_CURVE_TRIM_3D is used with *CONTROL_ADAPTIVE_CURVE, TCTOL is interpreted as the total width the mesh will be refined centered on the defined curve during the beginning of a simulation; in other words, on each side of the curve the mesh will be refined to a distance of half TCTOL. Figures 12-7 and 12-8 illustrate this. This feature *only* works with the 3D option.

The curve needs to be sufficiently close to the part. Since the curve is often made from some feature lines of forming tools, the curve must be re-positioned closer to the blank, or better yet, projected onto the blank; otherwise the refinement will not take place. The curve can be moved closer to the part with LS-PrePost 4.0 using *GeoTol* → *Project* → *Closest Proj* → *Project to Element* → *By Part*.

The partial input example below refines the mesh 2.0 mm (TCTOL = 4.0) on each side of the curve defined by the file adpcurves.iges. The maximum refinement level is 4, and the minimum element size allowed is 0.3 mm.

```
*INCLUDE
drawn.dynain
*DEFINE_CURVE_TRIM_3D
$      TCID    TCTYPE      TFLG      TDIR     TCTOL
         1         2         0         0     4.000
adpcurves.iges
*CONTROL_ADAPTIVE_CURVE
$     IDSET     ITYPE         N      SMIN
         1         2         4       0.3
```

Mesh refinement along a curve is very useful during line die simulations. For example, in a flanging simulation, a trimmed blank, that is mostly flat in the flanging break line in draw die, can be refined using a curve generated from the trim post radius. In LS-PrePost 4.0, the curve can be generated using *Curve* → *Spline* → *From Mesh* → *By Edge*, check *Prop*, and defining a large *Ang* to create a continuous curve along element edge. This curve can then be projected onto the blank mesh using *GeoTol* → *Project* feature which will then be used as the curve file adpcurves.iges here. The mesh pre-refinement along curves are implemented in the flanging process starting with LS-PrePost4.0 *eZSetup* for metal forming applications. In LS-PrePost4.3 *eZSetup*, improvements are made so adaptive mesh refinement along a curve can be made without the need to define any tools.

In Figures 12-9, 12-10, 12-11, 12-12and 12-13, mesh pre-refinement along a curve is demonstrated on a fender outer case in which the effect of different TCTOL values can be seen.

The keyword *INCLUDE_TRIM is recommended to be used at all times to include the dynain file from a previous simulation, except when the to-be-adapted sheet blank has no stress and strain information; that is, no *INITIAL_STRESS_SHELL, and *INITIAL_-

STRAIN_SHELL cards present in the sheet blank keyword or dynain file. For this case, the keyword *INCLUDE must be used instead.

Adaptive mesh refinement along a curve during trimming:

When the keyword *ELEMENT_TRIM is present, this keyword is used to refine meshes during a trimming simulation. Coarse meshes along the trim curve can be refined prior to trimming, causing a more detailed and distinctive trim edge. A partial example input deck is shown below:

```
*INCLUDE_TRIM
drawn.dynain
*ELEMENT_TRIM
         1
*DEFINE_CURVE_TRIM_2D
$#     TCID    TCTYPE      TFLG      TDIR     TCTOL      TOLN    NSEED1    NSEED2
         1         2                    0     0.250         1
doubletrim.iges
*DEFINE_TRIM_SEED_POINT_COORDINATES
$     NSEED        X1        Y1        Z1        X2        Y2        Z2
         1  -184.565    84.755
*CONTROL_ADAPTIVE_CURVE
$#    IDSET     ITYPE         N      SMIN   ITRIOPT
         1         2         3       3.0         0
*CONTROL_CHECK_SHELL
$#     PSID    IFAUTO    CONVEX      ADPT     ARATIO    ANGLE      SMIN
         1         1         1         1       0.25    150.0      0.18
```

The keyword *ELEMENT_TRIM defines a deformable part set to be trimmed. The keyword *DEFINE_CURVE_TRIM_2D defines the trim curve and type, along with trim tolerance, etc. The keyword *DEFINE_TRIM_SEED_POINT_COORDINATES indicates which side of the part will remain after trimming by specifying a seed node. The keyword *CONTROL_ADAPTIVE_CURVE specifies the adaptive mesh refinement level and minimum element size along the trim curve. Finally, the keyword *CONTROL_CHECK_SHELL repairs and fixes trimmed elements, so they are suitable for next simulation. More details can be found in each of the corresponding keyword manual sections.

Revision Information:

1. Revision 65630: use of TCTOL as a distance for mesh refinement (when used together with *CONTROL_ADAPTIVE_CURVE). TCTOL is the distance from the curve to the edge of the refinement.

2. Revision 113756: some improvements to TCTOL. TCTOL becomes the distance of the entire width of the refinement.

3. Revision 115142: ITRIOPT = 1 and TCTOL = 2 are available for mesh refinement along and inside a curve loop.

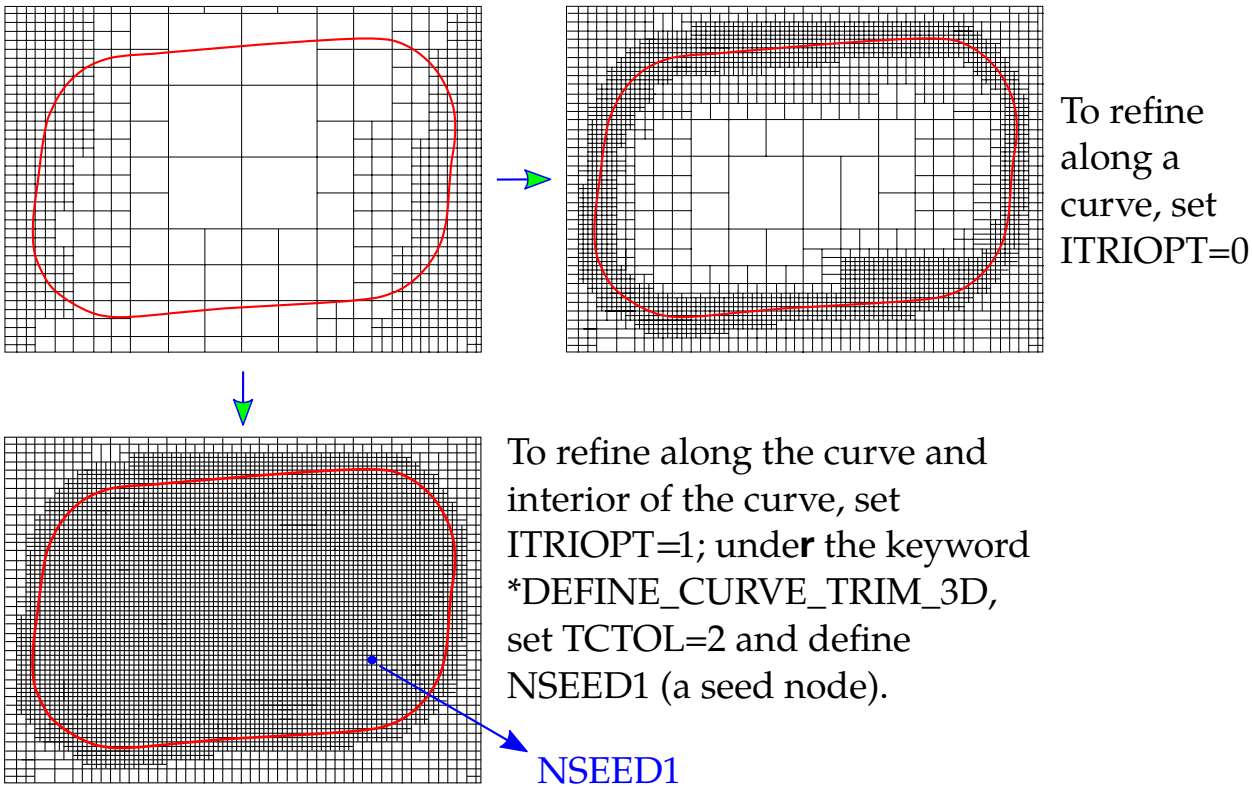


Figure 12-6. Mesh refinement along a curve (top right); along a curve and interior of the curve (bottom left).

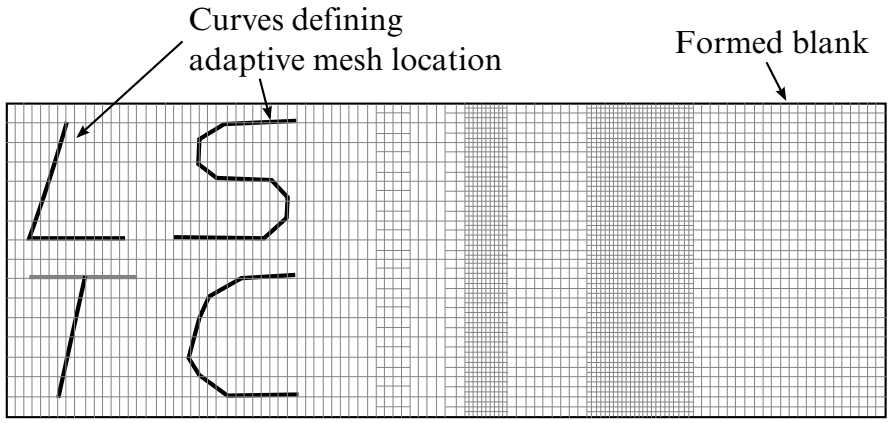


Figure 12-7. Curves can be discontinuous and in one IGES file.

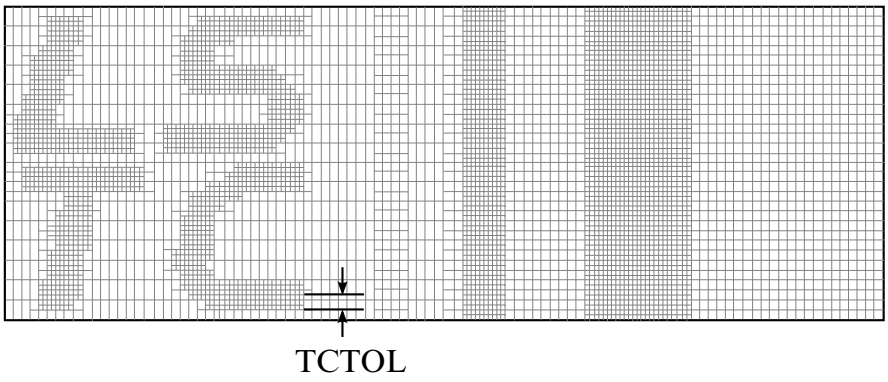


Figure 12-8. Define variable TCTOL to limit the mesh adaptivity area.

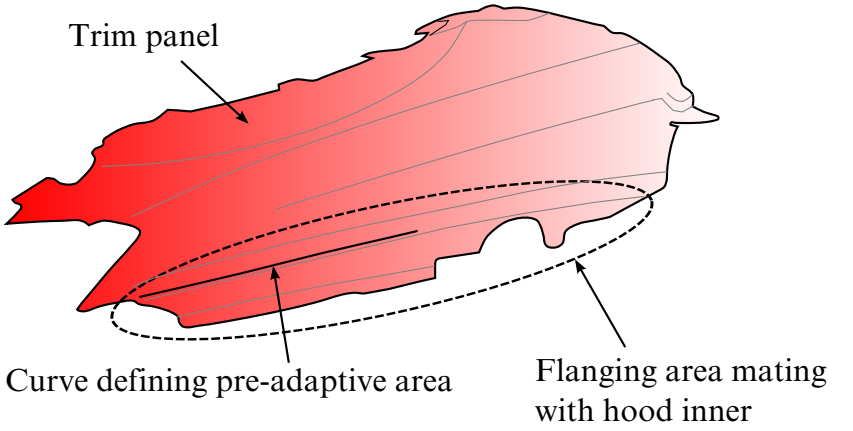


Figure 12-9. A complex mesh refinement example (NUMISHEET2002 Fender).

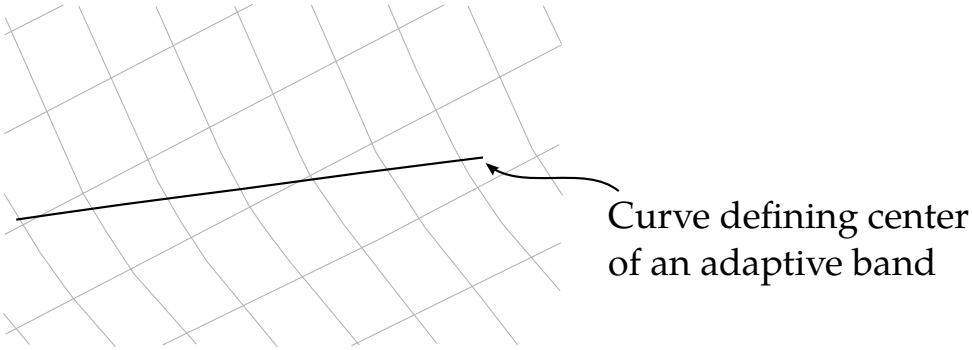


Figure 12-10. Original mesh with target curve defined.

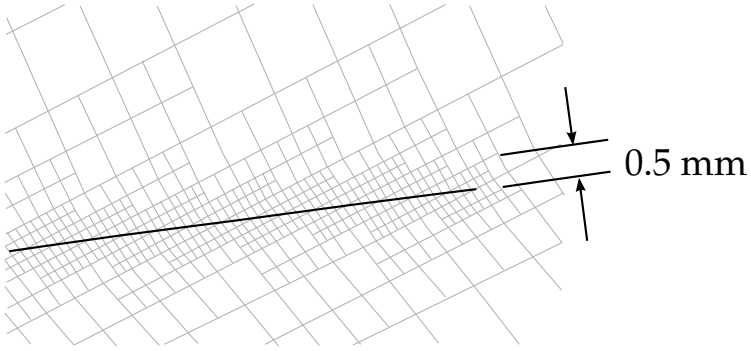


Figure 12-11. Mesh refinement along the target curve with TCTOL = 1.0.

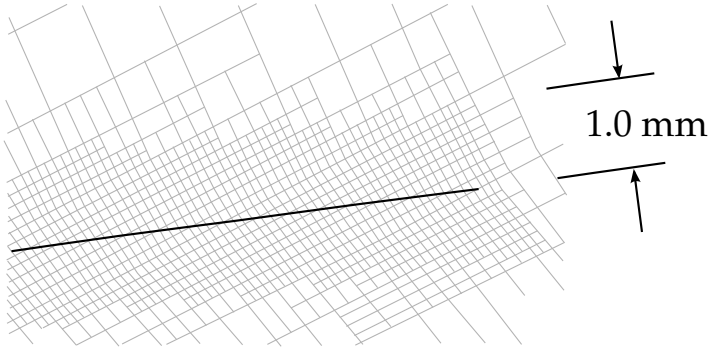


Figure 12-12. Mesh refinement along the target curve with TCTOL = 2.0.

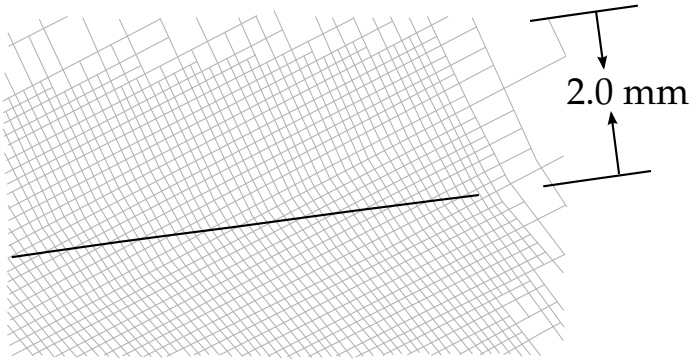


Figure 12-13. Mesh refinement along the target curve with TCTOL = 4.0.

*CONTROL_AIRBAG

Purpose: Global control parameters for CV AIRBAG.

Card 1	1	2	3	4	5	6	7	8
Variable	CKERR							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
CKERR	Flag to check the bag of a CV airbag for (a) open (free) edges and (b) segments not associated with shell or solid elements: EQ.0: Do not check (default). EQ.1: Check for free edges and segments not associated with elements. If the airbag surface contains a free edge, LS-DYNA will output the nodes of the free edge to d3hsp, issue a warning, and *continue* the run. If a segment in the segment set defining the airbag is not associated with an element, LS-DYNA will output the segment to d3hsp, issue an error message, and *terminate* the run. EQ.2: Check for free edges and segments not associated with elements. If the airbag surface contains a free edge, LS-DYNA will output the nodes of the free edge to d3hsp, issue a warning, and *terminate* the run. If a segment in the segment set defining the airbag is not associated with an element, LS-DYNA will output the segment to d3hsp, issue an error message, and *terminate* the run.

Remarks:

CKERR = 1 or 2 causes LS-DYNA to check the integrity of the airbag model. The airbag must be specified as a closed bag (no free edges) using a part set or segments set (SIDTYP of *AIRBAG_*OPTION*) for correct results. If it has a free edge, then the calculated values for the airbag, such as pressure, volume, and temperature, will be incorrect. If the airbag is defined using a segment set, the segments in the set must be associated with a element. Otherwise, LS-DYNA cannot capture the physics of the airbag.

*CONTROL_ALE

Purpose: Set global control parameters for the Arbitrary Lagrangian-Eulerian (ALE) and Eulerian calculations. This command is required when solid element formulation 5, 6, 7, 11, or 12 is used. Parallel processing using SMP is not recommended when using these element formulations; rather, it is better to use MPP for good parallel processing performance. See *CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS.

Card 1	1	2	3	4	5	6	7	8
Variable	DCT	NADV	METH	AFAC	BFAC	CFAC	DFAC	EFAC
Type	I	I	I	F	F	F	F	F
Default	1	1	2	0.0	0.0	0.0	0.0	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	AAFAC	VFACT	PRIT	EBC	PREF	NSIDEBC
Type	F	F	F	F	I	I	F	I
Default	0.0	10^{20}	1.0	10^{-6}	0	0	0.0	none
Remarks			obsolete					

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	NCPL	NBKT	IMASCL	CHECKR	BEAMIN	MMGPREF	PDIFMX	DTMUFAC
Type	I	I	I	F	F	I	F	F
Default	1	50	0	0.0	0.0	0	0.0	0.0

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	OPTIMPP	IALEDR	BNDFLX	MINMAS				
Type	I	I	I	F				
Default	0	0	0	10^{-5}				

VARIABLE	DESCRIPTION
DCT	Flag to invoke alternate advection logic for ALE (see Remark 2): NE.-1: Use default advection logic. EQ.-1: Use alternate (improved) advection logic; generally recommended, especially for simulation of explosives. Note that for S-ALE, DCT is ignored, and the alternative advection option is always used.
NADV	Number of cycles between advections (almost always set to 1).
METH	Advection method: EQ.1: Donor cell with Half Index Shift (HIS), first-order accurate. EQ.2: Van Leer with HIS, second-order accurate (default). EQ.-2: Van Leer with HIS: Additionally, the monotonicity condition is relaxed during the advection process to better preserve *MAT_HIGH_EXPLOSIVE_BURN material interfaces (see Remark 3). EQ.3: Donor cell with HIS modified to conserve total energy over each advection step, in contrast to METH = 1, which conserves internal energy (see Remark 4). EQ.4: Van Leer with HIS modified to conserve total energy over each advection step, in contrast to METH = 2, which conserves internal energy (see Remark 4). EQ.6: Finite volume method with flux-corrected transport. Only supported for ideal gases: the finite volume method is only applied to ALE elements fully filled with materials using *EOS_IDEAL_GAS or *EOS_001 for ideal gases. The advection in mixed ALE elements is handled by a

VARIABLE	DESCRIPTION
	donor cell method.
AFAC	ALE smoothing weight factor - Simple average EQ.-1.0: Turn smoothing off. See Remark 5.
BFAC	ALE smoothing weight factor - Volume weighting
CFAC	ALE smoothing weight factor - Isoparametric
DFAC	ALE smoothing weight factor - Equipotential
EFAC	ALE smoothing weight factor - Equilibrium
START	Start time for ALE smoothing or start time for ALE advection if smoothing is not used.
END	End time for ALE smoothing or end time for ALE advection if smoothing is not used. LT.0.0: The ALE mesh is removed after |END|.
AAFAC	ALE advection factor (donor cell options, default = 1.0). This field is obsolete.
VFACT	Volume fraction limit for stresses in single material and void formulation. All stresses are set to zero for elements with lower volume fraction than VFACT. EQ.0.0: Set to default 10^{-6}
PRIT	A flag to turn on or off the pressure equilibrium iteration option for multi-material elements (see Remark 6): EQ.0: Off (default) EQ.1: On
EBC	Automatic Eulerian boundary condition (see Remark 7): EQ.-2: Generate *ALE_ESSENTIAL_BOUNDARY with slip condition for mesh boundaries that do not already have segment boundary conditions defined with *BOUNDARY_NON_REFLECTING, *LOAD_BLAST_SEGMENT_SET, *BOUNDARY_SPC_SET, *LOAD_SEGMENT, or *LOAD_SEGMENT_SET applied.

VARIABLE	DESCRIPTION
	EQ.0: Off EQ.1: On with stick condition EQ.2: On with slip condition
PREF	A pseudo reference pressure equivalent to an environmental pressure that is applied to the free surfaces of the ALE domain or mesh (see Remark 8) LT.0: |PREF| is a *DEFINE_CURVE ID giving the reference pressure as a function of time. Refer to Remark 8 to determine which applications are suitable for this option. GE.0: Constant reference pressure value.
NSIDEBC	Node set ID (NSID) for a node set that is to be excluded from the EBC constraint.
NCPL	Number of Lagrangian cycles between coupling calculations. This is typically done every cycle; therefore, its default is 1.
NBKT	Number of Lagrangian cycles between global bucket-sort searches to locate the position of the Lagrangian structure (mesh) relative to the ALE fluid (mesh). The default is 50. LT.0: |NBKT| is a *DEFINE_CURVE ID defining a table: time vs NBKT as defined above. EQ.0: NBKT = 50 (default). If the mesh is moving, NBKT is adapted for the buckets to follow the mesh more closely. GT.0: NBKT remains constant.
IMASCL	A flag for turning ON/OFF mass scaling for ALE parts. The global mass scaling control parameter DT2MS for the *CONTROL_TIMESTEP keyword must be nonzero. If the ALE time step becomes smaller than the mass scaling time step (DT2MS), then IMASCL has the following effects: EQ.0: No mass scaling for ALE parts (default). The DT2MS time step will be used for the calculation. Note that because the DT2MS time step is larger than the ALE time step, the CFL condition will not be satisfied in the ALE domain, potentially causing instability. Print out a maximum of 20 warnings. EQ.1: No mass scaling for ALE parts. Stop the run.

VARIABLE	DESCRIPTION
	EQ.2: Do mass scaling for ALE parts (the result may not be correct due to this scaling). EQ.3: No mass scaling for ALE parts. Use the ALE time step. This time step may be small enough to substantially increase the calculation's run time.
CHECKR	A parameter for reducing or eliminating an ALE pressure locking pattern. It may range from 0.01 to 0.1 (See Remark 9).
BEAMIN	Flag to align the dynamics of plane strain and axisymmetric beams in 2D FSI ALE models to their shell counterparts in 3D FSI ALE models: EQ.0.0: Off (default) EQ.1.0: On
MMGPREF	A flag to select the method for assigning a reference pressure to multiple ALE multi-material groups (see Remark 8). EQ.0: Off (default). PREF applies to every AMMG in the model. LT.0: |MMGPREF| is an ID of either a curve defined using *DEFINE_CURVE or a table defined using *DEFINE_TABLE. If it is a curve, then its abscissa contains the multi-material group IDs (AMMGID), and the ordinates are the constant reference pressure values associated with each of those AMMGIDs. This is the PREF = constant case for each AMMG. If it is a table, then a load curve for PREF as a function of time must be defined for each AMMGID (the value in the table) in the model.
PDIFMX	Maximum of pressure difference between neighboring ALE elements under which the nodal forces are zeroed out: EQ.0: Off (default) GT.0: On
DTMUFAC	Scale a time step called DTMU that depends on the dynamic viscosity μ, the initial density ρ, and an element characteristic length l: $$\text{DTMU} = \frac{\rho l^2}{2\mu}$$ DTMU is emitted by the element to the solver as an element time step, thereby making DTMU an upper bound on the global time

VARIABLE	DESCRIPTION
	step. EQ.0: Off (default) GT.0: On
OPTIMPP	Optimize the MPP communications in the penalty coupling (*CONSTRAINED_LAGRANGE_IN_SOLID, CTYPE = 4) and group ALE parts together for the element processing. EQ.0: Off (default) EQ.1: On
IALEDR	Include ALE computations in the dynamic relaxation analysis (*CONTROL_DYNAMIC_RELAXATION). EQ.0: Off (default) EQ.1: On
BNDFLX	Multi-Material ALE group set ID selecting only the materials in elements at mesh boundaries with influxes that can flow in. By default, when the flow is inwards at the boundary faces of ALE elements, every material in these elements flows in. This option can select only a few of these ALE groups. EQ.0: Off (default) GT.0: *SET_MULTI-MATERIAL_GROUP_LIST ID EQ.-1: No influx
MINMAS	Factor of the minimum mass allowed in an element (see Remark 11): MINMAS×initial density×material volume

Remarks:

1. **Recommended starting point settings.** Although this keyword has many fields, only a few are required input. As a starting point, you can try setting DCT = -1, NADV = 1, METH = 1, AFAC = -1, and the rest as "0". When needed, PREF should also be defined. These settings are adequate for most cases. Depending on the physics, you may also need to change METH to 2 or 3.

2. **The DCT field.** For the general ALE solver in both 2D and 3D, the DCT field specifies the advection scheme. By default, the solver uses the original advection scheme. Setting DCT = -1 invokes the improved advection scheme. The S-ALE

solver in both 2D and 3D always uses the improved advection scheme (DCT is ignored).

We recommend DCT = -1 over the default scheme, especially for simulating explosives. This scheme includes the following major changes:

a) Relaxes an artificial limit on the expansion ratio limit. The default limit improves stability in some situations but can overestimate the explosive impulse.

b) Corrects redundant out-flux of material at corner elements. The redundancy can lead to negative volume.

c) Removes several artificial constraints in the advection which were originally implemented to assist in stability but are no longer needed.

3. **METH = -2.** The METH = -2 advection type is the same as METH = 2 with only one exception. It employs a looser constraint on monotonicity requirement during ALE advection. When METH = 2, for each advection process along three directions (front/back, top/bottom, left/right), the maximum/minimum values for advected history variables in the three elements along that direction are capped. METH = -2 relaxed the monotonicity condition so that the advected value is capped at the maximum/minimum value in the element itself and its neighboring 26 elements. This option, in certain conditions, can better preserve the material interface for materials defined with *MAT_HIGH_EXPLOSIVE_-BURN.

4. **METH = 3 and 4 for conserving total energy.** Generally, it is not possible to conserve both momentum and kinetic energy (KE) at the same time. Typically, internal energy (IE) is conserved, and KE may not be, resulting in some KE loss (hence, total energy loss). For many analyses, this energy loss is tolerable, but for airbag applications, it may lead to the reduction of the inflating potential of the inflator gas. METH = 3 and 4 try to eliminate this loss in KE over the advection step by storing any lost KE under IE, thus conserving the total energy of the system.

5. **Smoothing factors.** All the smoothing factors (AFAC, BFAC, CFAC, DFAC, EFAC) are generally most applicable to ELFORM = 5 (single material ALE formulation). The ALE smoothing feature is *not* supported by MPP versions.

6. **The PRIT field.** Most of the fast transient applications do not need this feature. It could be used in specific slow dynamic problems for which material constitutive laws with very different compressibility are linear, and the stresses in multi-material elements require to be balanced.

7. **The EBC field.** This option is used for EULER formulations. It automatically defines velocity boundary condition constraints for the user. The constraints, once defined, are applied to all nodes on free surfaces of an Eulerian domain. For problems where the normal velocity of the material at the boundary is zero, such as injection molding problems, the automatic boundary condition parameter is set to 2. This will play the same role as the nodal single point constraint. For EBC = 1, the material velocity of all free surface nodes of an Eulerian domain is set to zero.

8. **Environmental pressure.** By default, the pressure outside the ALE mesh is assumed to be zero. Any material with a pressure higher than zero will tend to flow out of the ALE domain. Therefore, when there is any ALE material with initialized pressure greater than zero, defined using the *EOS keyword, PREF should be defined to prevent that ALE material from leaving the ALE domain.

   To provide the effect of environmental pressure loading on all the free surfaces of the ALE mesh, the code performs an equivalent calculation. It subtracts PREF from the diagonal components of the stress tensor of each material before computing the internal forces. Thus, defining PREF is equivalent to applying *LOAD_SEGMENT cards to balance the internal pressure along the free ALE mesh boundaries. By default, PREF is applied to all the materials in the ALE domain. When PREF is a constant value, then a fixed value of ambient pressure is applied. When PREF is a load curve, then the ambient pressure is a function of time.

   The load curve option provides a way to gradually ramp up the FSI coupling in pressurized containers. The conventional way is to assign the high pressure to gas inside the tank, 1 bar of pressure to air outside, and the same 1 bar to PREF. But this would lead to a sudden loading at time 0, caused by the large pressure difference between inside/outside. The alternative approach is to assign the high pressure to both the inside gas and the outside air and use the load curve option. PREF could simply be a straight line, with high pressure at time 0 and 1 bar at some later time. With this setting, the outside air would start to flow out of the ALE mesh due to the pressure imbalance between the reference (ambient) pressure and eventually reach the 1 bar value. This air-leaking-out would effectively gradually load up the container.

   MMGPREF cannot be used to set the initial pressure for a material. The initial pressure must be set using the *EOS or *MAT keyword for the material. The shift of the stresses by PREF cannot be seen in the LS-PrePost fringe of the pressures.

   When MMGPREF < 0, there are 2 possible cases, assigning (a) PREF = constant or (b) PREF = PREF(t) (reference pressure as a function of time) to each AMMG.

```
[Example 1]
Consider for example, if a model has 3 ALE groups:
```

```
- AMMG1 = air        (with reference pressure for AMMG1 = PREF1 = 1.0 bar)
- AMMG2 = explosive  (with reference pressure for AMMG2 = PREF2 = 0.0 bar)
- AMMG3 = water      (with reference pressure for AMMG3 = PREF3 = 0.0 bar)

Here is how it may be defined in the input file.
PREF    = 0.0 bar
MMGPREF = -LCID
where LCID is the ID of the following curve:
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*DEFINE_CURVE
       LCID
$              AMMGID                PREF
                    1                 1.0
                    2                 0.0
                    3                 0.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--

[Example 2]
Consider a model also having 3 ALE groups where each group may have a
different reference pressure as a function of time.  One scenario may be that
of a pre-pressurized container.  Another may be the simulation of some
reservoir conditions.  Assume for this example that we have a pressurized
container.  All 3 AMMGs are intialized to 4 bars at t=0.0. Then, we lower the
environmental pressure of the oustside air from 4.0 bars to 1.0 over a short
time.  Over this duration, the FSI will have time to build up the pre-stressed
state in the container before another dynamic process is introduced, such as
an impact.

- AMMG1 = air outside                          (with PREF1=PREF1(t))
- AMMG2 = pressurized gas    inside container (with PREF2=PREF2(t))
- AMMG3 = pressurized liquid inside container (with PREF3=PREF3(t))

$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*DEFINE_TABLE
$     TBID
       101
$              AMMGID
                    1
                    2
                    3
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
$ The 1st curve immediately following the table corresponds to AMMG1, the 2nd
$ curve to AMMG2, and the 3rd curve to AMMG3, respectively.
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*DEFINE_CURVE
$     LCID
        11
$                time            PREF1(t)
               0.000                 4.0
               0.001                 1.0
               1.000                 1.0
*DEFINE_CURVE
$     LCID
        12
$                time            PREF2(t)
               0.000                 1.0
               0.001                 1.0
               1.000                 1.0
*DEFINE_CURVE
$     LCID
        13
$                time            PREF3(t)
               0.000                 1.0
               0.001                 1.0
               1.000                 1.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
```

9. **CHECKR field for one-point integration.** Due to one-point integration, ALE elements may experience spatial instability in the pressure field referred to as checkerboarding. CHECKR is a scale for diffusive flux calculation to alleviate this problem.

10. **Pressure checkerboarding.** Because the internal forces are located at the nodes, while the pressure is stored at the element center, sometimes a "checkerboard pattern" arises in the pressure distribution. It is a kind of locking effect that normally occurs only in problems having very small volumetric strains, i.e., at small pressures. CHECKR is designed to alleviate this problem.

11. **MINMAS for materials with large expansions.** Materials with large expansions could have element masses smaller than MINMAS. In such cases, MINMAS needs to be decreased to avoid artificial mass increases.

*CONTROL_BULK_VISCOSITY

Purpose: Reset the default values of the bulk viscosity coefficients globally. This may be advisable for shock wave propagation and some materials. Bulk viscosity is used to treat shock waves. A viscous term, q, is added to the pressure to smear the shock discontinuities into rapidly varying but continuous transition regions. With this method the solution is unperturbed away from a shock, the Hugoniot jump conditions remain valid across the shock transition, and shocks are treated automatically.

Card 1	1	2	3	4	5	6	7	8
Variable	Q1	Q2	TYPE	BTYPE	TSTYPE			
Type	F	F	I	I	I			
Default	1.5	.06	1	0	0			

VARIABLE	DESCRIPTION
Q1	Default quadratic viscosity coefficient.
Q2	Default linear viscosity coefficient.
TYPE	Default bulk viscosity type, IBQ (default = 1): EQ.-2: Same as -1 but the internal energy dissipated by the viscosity in the shell elements is computed and included in the overall energy balance. EQ.-1: Same as 1 but also includes viscosity in shell formulations 2, 4, 10, 16, and 17. The internal energy is not computed in the shell elements. EQ.1: Standard bulk viscosity. Solid elements only and internal energy is always computed and included in the overall energy balance. EQ.2: Richards-Wilkins bulk viscosity. Two-dimensional plane strain and axisymmetric solid elements only. Internal energy is always computed and included in the overall energy balance.
BTYPE	Beam bulk viscosity type (default = 0): EQ.0: The bulk viscosity is turned off for beams.

VARIABLE	DESCRIPTION
	EQ.1: The bulk viscosity is turned on for beam types 1 and 11. The energy contribution is not included in the overall energy balance.
	EQ.2: The bulk viscosity is turned on for beam type 1 and 11. The energy contribution is included in the overall energy balance.
TSTYPE	Bulk viscosity for thick shells (default = 0):
	EQ.0: The bulk viscosity is turned off for thick shells.
	EQ.1: The bulk viscosity is turned on for thick shells 5, 6, and 7.

Remarks:

The bulk viscosity creates an additional additive pressure term given by:

$$q = \begin{cases} \rho l\left(Q_1 l\dot{\varepsilon}_{kk}^2 - Q_2 a\dot{\varepsilon}_{kk}\right) & \dot{\varepsilon}_{kk} < 0 \\ 0 & \dot{\varepsilon}_{kk} \geq 0 \end{cases}$$

where Q_1 and Q_2 are dimensionless input constants which default to 1.5 and 0.06, respectively, l is a characteristic length given as the square root of the area in two dimensions and as the cube root of the volume in three, and a is the local sound speed. See Chapter 21 in the LS-DYNA Theory Manual for more details.

The Richards-Wilkins, see [Richards 1965, Wilkins 1976], bulk viscosity considers the directional properties of the shock wave which has the effect of turning off the bulk viscosity in converging geometries minimizing the effects of "q-heating." The standard bulk viscosity is active whenever the volumetric strain rate is undergoing compression even though no shock waves are present.

*CONTROL_CHECK_SHELL

Purpose: Check for various problems in the mesh after trimming for metal forming simulations. It only fixes boundary elements. This keyword should be included in a trimming input deck.

Part cards. Include one card for each part or part set to be checked. The next keyword ("*") card terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	IFAUTO	CONVEX	ADPT	ARATIO	ANGLE	SMIN	
Type	I	I	I	I	F	F	F	
Default	0	0	1	1	0.25	150.0	0.0	

VARIABLE	DESCRIPTION
PSID	Part or part set ID to be checked: EQ.0: Do not check GT.0: Part ID LT.0: Part set ID
IFAUTO	Flag to automatically correct bad elements: EQ.0: Write warning message only EQ.1: Fix bad element, write message
CONVEX	Check element convexity (internal angles less than 180 degrees). See Remark 2. EQ.0: Do not check EQ.1: Check
ADPT	Check adaptive constraints: EQ.0: Do not check EQ.1: Check
ARATIO	Minimum allowable aspect ratio. Elements which do not meet minimum aspect ratio test will be treated according to IFAUTO above.

VARIABLE	DESCRIPTION
ANGLE	Maximum allowable internal angle. Elements which fail this test will be treated according to IFAUTO above.
SMIN	Minimum element size. Elements which fail this test will be treated according to IFAUTO above. See Remark 4.

Remarks:

1. **Metal Forming Applications.** Shell element integrity checks which have been identified as important in metal forming applications are performed. These checks can improve springback convergence and accuracy. This keyword will repair bad elements created, for example, during trimming operations.

2. **Convexity.** If the convexity test is activated, all failed elements will be fixed regardless of IFAUTO.

3. **Mesh Connectivities.** In addition to illegal constraint definitions, checks are performed for mesh connectivities which have been found to cause convergence trouble in implicit springback applications.

4. **SMIN.** Variable SMIN should be set between 1/4 and 1/3 of the smallest pre-trim element length.

*CONTROL_CHECK_SOLID

Purpose: Check for various problems in the solid mesh after trimming for metal forming simulations. This keyword only fixes boundary elements. It should be included in a trimming input deck. The solid elements must be extruded from shell elements.

Part cards. Include one card for each part or part set to be checked. The next keyword ("*") card terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID		CONVEX		ARATIO	ANGLE	SMIN	VMIN
Type	I		I		F	F	F	F
Default	0		1		0.1	150.0	0.05	0.0

VARIABLE	DESCRIPTION
PSID	Part or part set ID to be checked: EQ.0: Do not check GT.0: Part ID LT.0: Part set ID
CONVEX	Check element convexity (internal angles less than 180 degrees): EQ.0: Do not check EQ.1: Check
ARATIO	Minimum allowable aspect ratio
ANGLE	Maximum allowable internal angle
SMIN	Minimum element size
VMIN	Minimum element volume

Remarks:

1. **SMIN.** Variable SMIN should be set between 1/4 and 1/3 of the smallest pre-trim element length.

*CONTROL_COARSEN

Purpose: Adaptively de-refine (coarsen) a shell mesh by selectively merging four adjacent elements into one. Adaptive constraints are added and removed as necessary.

Card 1	1	2	3	4	5	6	7	8
Variable	ICOARSE	ANGLE	NSEED	PSID	SMAX			
Type	I	F	I	I	F			
Default	0	none	0	0	0			

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
ICOARSE	Coarsening flag: EQ.0: do not coarsen (default) EQ.1: coarsen mesh at beginning of simulation for forming model EQ.2: coarsen mesh at beginning of simulation for crash model
ANGLE	Allowable angle change between neighboring elements. Adjacent elements which are flat to within ANGLE degrees are merged. Suggested starting value is 8.0 degrees.
NSEED	Number of seed nodes (optional). EQ.0: use only automatic searching. GT.0: the number of seed nodes with which to supplement the search algorithm. See Remark 2. NSEED must be an integer less than or equal to 8.

VARIABLE	DESCRIPTION
PSID	Part set ID. All the parts defined in this set will be prevented from been coarsened.
SMAX	Maximum element size. For ICOARSE = 2, no elements larger than this size will be created.
N1, …, N8	Optional list of seed node IDs for extra searching. If no seed nodes are specified, leave Card 2 blank.

Remarks:

1. **Coarsened Mesh Input Deck.** Coarsening is performed at the start of a simulation. The first plot state represents the coarsened mesh. By setting the termination time to zero and including the keyword *INTERFACE_SPRINGBACK_-LSDYNA, a keyword input deck can be generated containing the coarsened mesh.

2. **Seed Nodes.** By default, an automatic search is performed to identify elements for coarsening. In some meshes, isolated regions of refinement may be overlooked. Seed nodes can be identified in these regions to assist the automatic search. Seed nodes identify the central node of a four-element group which is coarsened into a single element if the angle criterion is satisfied.

3. **Non-Coarsened Regions.** The keyword *DEFINE_BOX_COARSEN can be used to indicate regions of the mesh which are protected from coarsening.

*CONTROL_CONSTRAINED

Purpose: Define global control parameters for constraint-related properties.

Card 1	1	2	3	4	5	6	7	8
Variable	SPRCHK	SPRSMD	SPRSRCH	SKPCNRB				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
SPRCHK	SPR2/SPR3 initialization check: EQ.0: Automatically increase search radius to find enough nodes (default) EQ.1: Same as 0 but also write a warning EQ.2: Error termination if not enough nodes found immediately
SPRSMD	Shear moment distribution behavior for SPR3: EQ.0: Distributed as force pairs EQ.1: Distributed as nodal moments (old behavior)
SPRSRCH	Search method for SPR2 and SPR3: EQ.0: Include nodes inside the search radius. EQ.1: Include not only nodes inside the search radius but also all nodes from elements inside the search radius. This search method generally leads to more nodes involved in the connection. In addition, this option invokes a different search algorithm from SPRSRCH = 0 when solid element parts are involved: nodes on the surface facing the SPR connector are detected, leading to automatically finding enough nodes. Because the default algorithm depends on a user-specified search cylinder, it sometimes does not detect enough nodes for coarse meshes. Also, sometimes, the default algorithm finds nodes beneath the surface of the solid element part. Thus, this alternative algorithm for elements leads to more robust search results in general.

VARIABLE	DESCRIPTION
SKPCNRB	Flag to skip the stress calculation of a part based on element type when it is part of a nodal rigid body, *CONSTRAINED_NODAL_-RIGID_BODY. If nonzero, the value can be any combination of the numbers 1 through 3. For example, to skip the calculation for shell and solid elements, input 12 or 21. EQ.0: Do not skip. EQ.1: Skip for shell elements EQ.2: Skip for solid elements EQ.3: Skip for beam elements

*CONTROL_CONTACT

Purpose: Change defaults for computation with contact surfaces.

Card Summary:

Unless noted otherwise in the description of the field, the optional cards (Cards 3 through 7) apply only to the following contact types:

SINGLE_SURFACE

AUTOMATIC_GENERAL

AUTOMATIC_SINGLE_SURFACE

AUTOMATIC_NODES_TO_…

AUTOMATIC_SURFACE_…

AUTOMATIC_ONE_WAY_…

ERODING_SINGLE_SURFACE

The friction coefficients SFRIC, DFRIC, EDC, and VFC are active only when *PART_CONTACT is invoked with FS = -1 in *CONTACT, and the corresponding frictional coefficients in *PART_CONTACT are set to zero. This keyword's TH, TH_SF, and PEN_SF override the corresponding parameters in *CONTACT but do not override corresponding nonzero parameters in *PART_CONTACT.

Card 1. This card is required.

SLSFAC	RWPNAL	ISLCHK	SHLTHK	PENOPT	THKCHG	ORIEN	ENMASS

Card 2. This card is required.

USRSTR	USRFRC	NSBCS	INTERM	XPENE	SSTHK	ECDT	TIEDPRJ

Card 3. This card is optional.

SFRIC	DFRIC	EDC	VFC	TH	TH_SF	PEN_SF	PTSCL

Card 4. This card is optional.

IGNORE	FRCENG	SKIPRWG	OUTSEG	SPOTSTP	SPOTDEL	SPOTHIN	DIR_TIE

Card 5. This card is optional.

ISYM	NSEROD	RWGAPS	RWGDTH	RWKSF	ICOV	SWRADF	ITHOFF

Card 6. This card is optional.

SHLEDG	PSTIFF	ITHCNT	TDCNOF	FTALL		SHLTRW	IGACTC

Card 7. This card is optional.

IREVSPT		COHTIEM	TIEOPT	STROBJ	DEFBLK		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SLSFAC	RWPNAL	ISLCHK	SHLTHK	PENOPT	THKCHG	ORIEN	ENMASS
Type	F	F	I	I	I	I	I	I
Default	.1	none	1	0	1	0	1	0

VARIABLE	DESCRIPTION
SLSFAC	Scale factor for sliding interface penalties, SLSFAC: EQ.0.0: Default = 0.1.
RWPNAL	Scale factor for rigid wall penalties (see *RIGIDWALL) that treats nodal points interacting with rigid walls. The penalties are set so that an absolute value of unity should be optimal; however, this penalty value may be very problem-dependent. If rigid/deformable materials switching is used, this option should be used if the switched materials interact with rigid walls. If you have IGA parts in your model, see Remark 10. LT.0.0: All nodes are treated by the penalty method. This is set to -1.0 for implicit calculations. Since seven (7) variables are stored for each possible tracked node (see NSID on *RIGIDWALL_PLANAR/GEOMETRIC), only the nodes that may interact with the wall should be included in the node list. EQ.0.0: The constraint method is used, and nodal points that

VARIABLE	DESCRIPTION
	belong to rigid bodies are not considered. GT.0.0: Rigid body nodes are treated by the penalty method, and all other nodes are treated by the constraint method.
ISLCHK	Initial penetration check in contact surfaces with indication of initial penetration in output files (see Remark 3): EQ.1: No checking (default) EQ.2: Full check of initial penetration is performed.
SHLTHK	Flag for consideration of shell thickness offsets in non-automatic surface-to-surface and non-automatic nodes-to-surface type contacts. Shell thickness offsets are always included in single surface, constraint-based, automatic surface-to-surface, and automatic nodes-to-surface contact types (see Remarks 1 and 2): EQ.0: Thickness is not considered. EQ.1: Thickness is considered, but rigid bodies are excluded. EQ.2: Thickness is considered, including rigid bodies.
PENOPT	Penalty stiffness value option (applies to the Standard Penalty Formulation and the Soft Constraint Penalty Formulation, that is, SOFT = 0 and 1 on *CONTACT_*OPTION*). For the default calculation of the penalty value, please refer to the LS-DYNA Theory Manual. EQ.1: Minimum of reference segment and tracked node (default for most contact types) EQ.2: Use the reference segment stiffness (old way). EQ.3: Use the tracked node value. EQ.4: Use the tracked node value, area or mass weighted. EQ.5: Same as 4 but inversely proportional to the shell thickness. This may require special scaling and is not generally recommended. PENOPT = 4 and 5 can be used for metal forming calculations. In general, PENOPT = 2 - 5 should be avoided if both the tracked nodes and the reference segments belong to deformable parts.
THKCHG	Shell thickness changes considered in single surface contact (see Remark 1):

VARIABLE	DESCRIPTION
	EQ.0: No consideration (default) EQ.1: Shell thickness changes are included. EQ.2: Applies to MPP only. Shell thickness changes are included, but a different algorithm is used than for THKCHG = 1. This method is more consistent with the way the initial contact thickness is computed.
ORIEN	Optional automatic reorientation of contact interface segments during initialization. See Remark 4. EQ.1: Active for automated (part) input only (default). Contact surfaces are given by *PART definitions. EQ.2: Active for manual (segment) and automated (part) input EQ.3: Inactive for non-forming contact EQ.4: Inactive for *CONTACT_FORMING types and *CONTACT_DRAWBEAD
ENMASS	Flag for treatment of eroded nodes in contact. An eroded node is a node that is no longer attached to any element after an element deletion. ENMASS is not supported by all contact types; it is suggested that the user toggle on "Show Deleted Nodes" in LS-PrePost when postprocessing to display eroded nodes as particles, and in so doing, determine if ENMASS affects the contact behavior. ENMASS is not supported when SOFT = 2 on Optional Card A of *CONTACT. EQ.0: Eroded nodes are not considered in the contact algorithm. EQ.1: Eroded nodes of solid elements remain active in the contact algorithm. EQ.2: Eroded nodes of solid and shell elements remain active in the contact algorithm.

Card 2	1	2	3	4	5	6	7	8
Variable	USRSTR	USRFRC	NSBCS	INTERM	XPENE	SSTHK	ECDT	TIEDPRJ
Type	I	I	I	I	F	I	I	I
Default	0	0	10-100	0	4.0	0	0	0

VARIABLE	DESCRIPTION
USRSTR	Storage per contact interface for user supplied interface control subroutine; see Appendix F. If zero, no input data is read, and no interface storage is permitted in the user subroutine. This storage should be large enough to accommodate input parameters and any history data. This input data is available in the user supplied subroutine.
USRFRC	Storage per contact interface for user supplied interface friction subroutine; see Appendix G. If zero, no input data is read, and no interface storage is permitted in the user subroutine. This storage should be large enough to accommodate input parameters and any history data. This input data is available in the user supplied subroutine.
NSBCS	Number of cycles between contact searching using three-dimensional bucket searches. Using the default value for this field is strongly recommended. For mortar contact (option MORTAR on the CONTACT card), the default is 100. For MPP, this field is ignored when SOFT = 0 and 1, and only BCKT on MPP 1 of *CONTACT_*OPTION*_... applies in those cases.
INTERM	Flag for intermittent searching in old surface-to-surface contact using the interval specified as NSBCS above: EQ.0: Off EQ.1: On
XPENE	Contact surface maximum penetration check multiplier. If the small penetration checking option, PENCHK, on the contact surface control card is active, then nodes whose penetration then exceeds the product of XPENE and the element thickness are set free, see *CONTACT_*OPTION*_...

VARIABLE	DESCRIPTION
	EQ.0.0: Set to default which is 4.0.
SSTHK	Flag for determining default contact thickness for shells in single surface contact types. This variable does not apply when SOFT = 2. See Remark 2. EQ.0: Default contact thickness may be controlled by shell thickness or by shell edge length (default). EQ.1: Default contact thickness is equal to the shell thickness.
ECDT	Time step size override for eroding contact: EQ.0: Contact time size may control DT. EQ.1: Contact is not considered in DT determination.
TIEDPRJ	Bypass projection of SURFA nodes to SURFB surface in types: *CONTACT_TIED_NODES_TO_SURFACE *CONTACT_TIED_SHELL_EDGE_TO_SURFACE *CONTACT_TIED_SURFACE_TO_SURFACE Tied interface options: EQ.0: Eliminate gaps by projection nodes EQ.1: Bypass projection. Gaps create rotational constraints that can substantially affect results.

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	SFRIC	DFRIC	EDC	VFC	TH	TH_SF	PEN_SF	PTSCL
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

VARIABLE	DESCRIPTION
SFRIC	Default static coefficient of friction (see *PART_CONTACT)

VARIABLE	DESCRIPTION
DFRIC	Default dynamic coefficient of friction (see *PART_CONTACT)
EDC	Default exponential decay coefficient (see *PART_CONTACT)
VFC	Default viscous friction coefficient (see *PART_CONTACT)
TH	Default contact thickness (see *PART_CONTACT)
TH_SF	Default thickness scale factor (see *PART_CONTACT)
PEN_SF	Default local penalty scale factor (see *PART_CONTACT)
PTSCL	Scale factor on the contact stress exerted onto shell formulations 25, 26, and 27. When DOF = 3, the scale factor also applies to shell formulations 2, 4, and 16.

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	IGNORE	FRCENG	SKIPRWG	OUTSEG	SPOTSTP	SPOTDEL	SPOTHIN	DIR_TIE
Type	I	I	I	I	I	I	F	I
Default	0	0	0	0	0	0	inactive	0

VARIABLE	DESCRIPTION
IGNORE	Ignore initial penetrations for the *CONTACT_AUTOMATIC options. In the SMP contact, this flag is not implemented for the AUTOMATIC_GENERAL option. "Initial" in this context refers to the first time step that a penetration is encountered. This option can also be specified for each interface on Optional Card C of *CONTACT_.... The value defined here will be the default.

EQ.0: Move nodes to eliminate initial penetrations in the model definition.

EQ.1: Allow initial penetrations to exist by tracking the initial penetrations.

EQ.2: Allow initial penetrations to exist by tracking the initial penetrations. However, penetration warning messages are printed with the original coordinates and the

VARIABLE	DESCRIPTION
	recommended coordinates for each penetrating node given.
FRCENG	Flag to activate the calculation of frictional sliding energy: EQ.0: Do not calculate. EQ.1: Calculate frictional energy in contact and store as "Surface Energy Density" in the binary INTFOR file. Convert mechanical frictional energy to heat when doing a coupled thermal-mechanical problem. When PKP_SEN = 1 on the keyword card *DATABASE_EXTENT_BINARY, it is possible to identify the energies generated on the upper and lower shell surfaces, which is important in metal forming applications. This data is mapped after each *h*-adaptive remeshing. EQ.2: Same behavior as above (set to 1) except that frictional energy is not converted to heat.
SKIPRWG	Flag not to display a stationary rigid wall by default: EQ.0: Generate four extra nodes and one shell element to visualize a stationary planar rigid wall. EQ.1: Do not generate stationary rigid walls.
OUTSEG	Flag to determine whether to output for contact type *CONTACT_SPOTWELD each beam spot weld SURFA node and its SURFB segment into the d3hsp file: EQ.0: No, do not write out this information. EQ.1: Yes, write out this information.
SPOTSTP	Flag to determine whether an error termination should occur if a spot weld node or face, which is related to a *MAT_SPOTWELD beam or solid element, respectively, cannot be found on the SURFB surface: EQ.0: No error termination. Silently delete the weld and continue. EQ.1: Print error message and terminate. EQ.2: No error termination. Delete the weld, print a message, and continue. EQ.3: No error termination; keep the weld. This is not

VARIABLE	DESCRIPTION
	recommended as it can lead to instabilities.
SPOTDEL	This option controls the behavior of spot welds when the parent element erodes. When SPOTDEL is set to 1, the beam or solid spot weld is deleted, and the tied constraint is removed when the parent element erodes. The parent element is the element to which the SURFA node is attached using the TIED interface. This option also works for SPRs, namely, they automatically fail if at least one of the parent elements fails. To avoid instabilities, this option should be set to 1 whenever the parent element is expected to erode. EQ.0: Do not delete the spot weld beam, solid element, or SPR. EQ.1: Delete the spot weld elements or SPRs when the attached shells on one side of the element fail. GT.1: Delete the SPR when SPOTDEL nodes are attached to failed elements in the search radius. On vector processors, this option can significantly slow down the calculation if many weld elements fail since the vector lengths are reduced. On non-vector processors, the cost penalty is minimal.
SPOTHIN	Optional thickness scale factor. If active, define a factor greater than zero, but less than one. Premature failure of spot welds can occur due to contact of the spot-welded parts in the vicinity of the spot weld. This contact creates tensile forces in the spot weld. Although this may seem physical, the compressive forces generated in the contact are large enough to fail the weld in tension before failure is observed in an experimental test. With this option, the thickness of the parts in the vicinity of the weld is automatically scaled, the contact forces do not develop, and the problem is avoided. We recommend setting the IGNORE option to 1 or 2 if SPOTHIN is active. In MPP, this option applies to all non-Mortar contacts that have SINGLE_SURFACE, AUTOMATIC_GENERAL, SURFACE_TO_SURFACE, or NODES_TO_SURFACE in the name. In SMP, it only applies to the AUTOMATIC_SINGLE_SURFACE option. See Remark 5.
DIR_TIE	Directional tie for *MPP non-groupable* tied contacts. If this flag is set to 1, then each node in the SURFA side of a tied contact is associated with outward normal vectors. A node belonging to the surface of a shell, solid, or thick shell surface is associated with one or more vectors depending on the angle between adjacent segments. Typically, the node of a flat surface will have *one* normal vector, the

VARIABLE	DESCRIPTION
	node of an edge will have *two* normal vectors, and the node of a corner will have *three* normal vectors. Nodes on a cylinder may have *one or two* normal vectors, depending on the mesh resolution of the faceted surface. When deciding which SURFB segment to tie to, the algorithm gives preference to those segments for which the direction from the SURFA node to the SURFB segment has a positive dot product with at least one of the aforementioned normal vectors. Therefore, a SURFA node will not necessarily tie to the closest SURFB segment, but to the "correct" segment. This feature avoids nonphysical tie situations or even zero solid element volumes as a result.

This card is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	ISYM	NSEROD	RWGAPS	RWGDTH	RWKSF	ICOV	SWRADF	ITHOFF
Type	I	I	I	F	F	I	F	I
Default	0	0	1	0.	1.0	0	0.	0

VARIABLE	DESCRIPTION
ISYM	Symmetry plane option default for automatic segment generation when contact is defined by part IDs: LT.0: |ISYM| is a node set on the symmetry boundary, supported and recommended for Mortar contact. This will allow for a correct treatment of segments close to the symmetry face/edge. See Remark 8. EQ.0: Off EQ.1: Do not include faces with normal boundary constraints (for example, segments of brick elements on a symmetry plane). This option is important to retain the correct boundary conditions in the model with symmetry.
NSEROD	Flag to invoke old method for ERODING_NODES_TO_SURFACE (SMP only):

VARIABLE	DESCRIPTION
	EQ.0: Use two-way algorithm (default) EQ.1: Use one-way algorithm (old method)
RWGAPS	Flag to add rigid wall gap stiffness (see parameter RWGDTH below): EQ.1: Add gap stiffness (default). EQ.2: Do not add gap stiffness.
RWGDTH	Death time for gap stiffness. After this time, the gap stiffness is no longer added.
RWKSF	Rigid wall penalty scale factor for contact with deformable parts during implicit calculations. This value is independent of SLSFAC and RWPNAL. If RWKSF is also specified in *RIGIDWALL_PLANAR, the stiffness is scaled by the product of the two values.
ICOV	Invokes the covariant formulation of Konyukhov and Schweizerhof in the FORMING contact option. EQ.0: Standard formulation (default) EQ.1: Covariant contact formulation
SWRADF	Spot weld radius scale factor for neighbor segment thinning: EQ.0: Neighbor segments are not thinned (default). GT.0: The radius of a beam spot weld is scaled by SWRADF when searching for close neighbor segments to thin.
ITHOFF	Flag for offsetting thermal contact surfaces for thick thermal shells: EQ.0: No offset. If thickness is not included in the contact, the heat will be transferred between the mid-surfaces of the corresponding contact segments (shells). EQ.1: Offsets are applied so that contact heat transfer is always between the outer surfaces of the contact segments (shells).

This card is optional.

Card 6	1	2	3	4	5	6	7	8
Variable	SHLEDG	PSTIFF	ITHCNT	TDCNOF	FTALL		SHLTRW	IGACTC
Type	I	I	I	I	I		F	I
Default	0	0	0	0	0		0.	0

VARIABLE	DESCRIPTION
SHLEDG	Flag for assuming edge shape for shells when measuring penetration. This field is available for segment-to-segment contact (see SOFT = 2 on *CONTACT). EQ.0: Shell edges are assumed round (default). EQ.1: Shell edges are assumed square and are flush with the nodes.
PSTIFF	Flag to choose the method for calculating the penalty stiffness. This field is available for segment-to-segment contact (see SOFT = 2 on *CONTACT). See Remark 6. EQ.0: Based on material density and segment dimensions (default). EQ.1: Based on nodal masses or on material density and segment dimensions. The segment mass is taken as the larger of the two values calculated.
ITHCNT	Thermal contact heat transfer methodology: LT.0: Conduction evenly distributed (pre R4) EQ.0: Set to the default, which is 1 EQ.1: Conduction weighted by shape functions, reduced integration EQ.2: Conduction weighted by shape functions, full integration
TDCNOF	Tied constraint offset contact update option: EQ.0: Update velocities and displacements from accelerations. EQ.1: Update velocities and accelerations from displacements. This option is recommended only when there are large

VARIABLE	DESCRIPTION
	angle changes where the default does not maintain a constant offset to a small tolerance. This latter option is not as stable as the default and may require additional damping for stability. See *CONTROL_BULK_VISCOSITY and *DAMPING_PART_STIFFNESS.
FTALL	Option to output contact forces to RCFORC for all two surface force transducers when the force transducer surfaces overlap. See Remark 7. EQ.0: Output to the first force transducer that matches (default). EQ.1: Output to all force transducers that match.
SHLTRW	Optional shell thickness scale factor for contact with rigid walls. Shell thickness is not considered when SHLTRW = 0.0 (default). SHLTRW = 0.5 will result in an offset of half a shell thickness in contact with rigid walls.
IGACTC	Options to use isogeometric shells for contact detection for contact involving isogeometric shells: EQ.0: Contact between interpolated nodes and interpolated shells EQ.1: Contact between interpolated nodes and isogeometric shells

This card is optional.

Card 7	1	2	3	4	5	6	7	8
Variable	IREVSPT		COHTIEM	TIEOPT	STROBJ	DEFBLK		
Type	I		I	I	I	F		
Default	0		0	0	0	0.0		

VARIABLE	DESCRIPTION
IREVSPT	Flag to revert the spot weld thinning behavior where beam and brick spot welds share nodes with shell parts instead of being tied to the shells: EQ.0: Thinning at shared nodes will be done as it has been in all

VARIABLE	DESCRIPTION
	versions after R9.3.1. EQ.1: Behavior reverts to that of R9.3.1. In this version and previous versions, spot weld thinning was not done.
COHTIEM	Flag to treat how the mass from SURFB of a tied contact affects the time step estimation of cohesive elements: EQ.0: No treatment EQ.1: Assuming the cohesive element's nodes are on SURFA of a tied contact, LS-DYNA includes the mass from SURFB when estimating the cohesive element's time step. Note that groupable tied contacts are not currently supported with this option.
TIEOPT	Option for constrained tied contact formulations to circumvent some of the shortcomings present in the standard implementations: EQ.0: Standard implementation EQ.1: Standard implementation, except that a groupable effect is invoked. See Remark 11. This option is automatically applied for selective mass scaling. EQ.2: Alternative implementation. See Remark 11.
STROBJ	Flag for strong objectivity (frame invariance) in non-groupable single surface contacts with SOFT = 0 or 1. Strong objectivity incurs an extra cost but may improve results. EQ.0: Do not turn on strong objectivity. EQ.1: Turn on strong objectivity.
DEFBLK	Default bulk modulus used for the contact stiffness calculation. This field applies to all penalty-based contacts that use the bulk modulus to calculate the contact stiffness. Thus, it applies when using the standard penalty formulation (SOFT = 0), the soft constraint penalty formulation (SOFT = 1), and Mortar segment-to-segment contact. See Remark 12.

Remarks:

1. **Shell thickness change.** The shell thickness change option (ISTUPD) must first be activated in *CONTROL_SHELL for shell thickness to change. Secondly, for

a single surface contact to recognize the change in shell thickness, THKCHG must be set to 1. For surface-to-surface contacts with shell thickness offsets, that is, *CONTACT_SURFACE_TO_SURFACE with SHLTHK set to 1 or 2, or *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE, the shell thickness change is recognized by default, and THKCHG is not used.

2. **Default contact thickness for single surface contacts.** For single-surface contact types,

   SINGLE_SURFACE

   AUTOMATIC_SINGLE_SURFACE

   AUTOMATIC_GENERAL

   AUTOMATIC_GENERAL_INTERIOR

   ERODING_SINGLE_SURFACE,

   the default contact thickness is taken as the smaller of two values, namely:

   - shell thickness, and
   - 40% of the minimum edge length.

   The minimum edge length is calculated as the minimum of the distances from N4 to N1, N1 to N2, and N2 to N3. N3 to N4 is neglected because the shell is possibly triangular. For shells that have a large thickness, the "40% of the minimum edge length" criterion may control the contact thickness. This edge length criterion is an ad hoc criterion included to help prevent solution instability or add a slow down in the contact searching in the case where the shell thickness-to-edge-length ratio is large.

   If SSTHK = 1 or if SOFT = 2 (Optional Card A in *CONTACT), the edge length criterion is not considered, that is, the default contact thickness is simply the shell thickness.

3. **Initial penetration check.** As of version 950 the initial penetration check option is always performed regardless of the value of ISLCHK. If you do not want to remove initial penetrations, then set the contact birth time (see *CONTACT_…) so that the contact is not active at time 0.0.

4. **Automatic reorientation.** Automatic reorientation requires offsets between the segments on each side of the contact interface. The reorientation is based on segment connectivity, and once all segments are oriented consistently based on connectivity, a check is made to see if the surfaces on each side of the contact interface face each other based on the right-hand rule. If not, all segments in a given surface are reoriented. This procedure works well for non-disjoint

surfaces. If the surfaces are disjoint, the AUTOMATIC contact options, which do not require orientation, are recommended. For FORMING contact types, automatic reorientation works for disjoint surfaces.

5. **Neighbor segment thinning option.** Setting SPOTHIN and SWRADF greater than zero activates a neighbor segment thinning option. The radius of a spot weld is scaled by SWRADF, and then a search is made for shell segments that are neighbors of the tied shell segments that are touched by the weld but not tied by it. The SWRADF option is available for beam element welds, solid element welds, and solid weld assemblies. For solids and solid assemblies, the weld radius is calculated assuming a circular shape with an equivalent area.

6. **Segment masses for penalty stiffness.** Segment-to-segment contact (see SOFT = 2 on *CONTACT) calculates a penalty stiffness based on the solution time step and the masses of the segments in contact. By default, segment masses are calculated using the material density of the element associated with the segment and the volume of the segment. This method does not take into account added mass introduced by lumped masses or mass scaling and can lead to stiffness that is too low. Therefore, a second method (PSTIFF = 1) was added which estimates the segment mass using the nodal masses. Note that for PSTIFF = 1, the algorithm compares the mass calculated by the two methods and uses the larger value. Setting a PSTIFF value here sets the default values of PSTIFF for all interfaces. The PSTIFF option can also be specified for individual contact interfaces by defining PSTIFF on Optional Card F of *CONTACT.

7. **Force transducer search option.** Force transducers between two surfaces measure the contact force from any contact interfaces that generate force between the SURFA and SURFB surfaces of the force transducer. When contact is detected, a search is made to see if the contact force should be added to any two surface force transducers. By default, when a force transducer match is found, the force is added, and the search terminates. When FTALL = 1, the search continues to check for other two surface force transducer matches. This option is useful when the SURFA and SURFB force transducer surfaces overlap. If there is no overlap, the default is recommended.

8. **Symmetry option for Mortar contact.** By specifying a node set through ISYM < 0, a segment will be ignored in the Mortar contact if *all* nodes in the segment are contained in the specified node set. Edge treatment will also be avoided on the interface between the symmetry plane and the true contact surface. *Important note:* When using this option, it is important that segments *close* to the symmetry plane but *not in it* have at least one node on the true contact surface. For instance, if a quarter symmetry model is used, avoid using a wedge element at the center as this is likely to render a triangular contact segment with all of its nodes in the symmetry node set.

9. **MORTAR contact fields.** The following table lists parameters that in some sense are related to MORTAR contact. Any parameter that is not mentioned in this list is ignored. See also a similar table on *CONTACT.

Data Card	Comment
Card 1	Only SLSFAC applies, and it applies as for any other contact. Other parameters are ignored. Shell thickness offsets are always included, except for SURFB surface thickness with the FORMING option. For the FORMING option, the normal of the SURFB surface must point towards the SURFA side of the contact. For any contact, shell thickness changes are considered if ISTUPD on *CONTROL_SHELL is used. Offset of the shell reference surface is considered if NLOC on *SECTION_SHELL is used, regardless of the value of CNTCO on *CONTROL_SHELL.
Card 2	USRFRC and NSBCS apply as for any other contact; all other parameters are ignored.
Card 3	All parameters apply as for any other contact.
Card 4	IGNORE = 0/1/2 applies as described on *CONTACT, as does FRCENG. Other parameters are ignored.
Card 5	ISYM < 0 applies, which indicates a node set on symmetry planes. Other parameters are ignored.
Card 6	No parameters apply. Edges in automatic contact are always flat.
Card 7	Only DEFBLK applies. All other parameters are ignored.

10. **RWPNAL and IGA.** For IGA parts specified with either *IGA_SHELL or *IGA_SOLID, the rigid wall is enforced through the interpolation nodes generated for the IGA parts. The interpolation nodes are treated like nodal points that belong to rigid bodies and are treated with the penalty method. The actual control points for the IGA parts are removed from the list of possible tracked nodes.

    If RWPNAL = 0.0, rigid nodes are not included in the contact, but the interpolation nodes are still enforced by the penalty method using a rigid wall penalty scale factor.

11. **TIEOPT.** When invoking the standard algorithm with groupable effects (TIEOPT = 1), the following happens for any of the constrained tied contacts in the model:

    a) SMP and MPP tied contacts become identical, and each finds the same node-segment pairs.

b) The segment sides of different tied contact interfaces are allowed to share nodes, which otherwise would require the GROUPABLE option.

c) The SHELL_EDGE_TO_SOLID_CONSTRAINED_OFFSET option becomes available, which otherwise is silently converted to a SHELL_EDGE_TO_-SOLID without offset.

The alternative algorithm (TIEOPT = 2) also causes the above items to occur. Additionally, for the alternative algorithm, the tied contacts automatically detect the nature of involved nodes and convert any invalid constraints involving rotational degrees of freedom to suitable alternatives, so "any" tied option is valid for "any" situation. A list of pairs that are changed is output.

12. **DEFBLK.** When estimating the contact stiffness, LS-DYNA calculates a bulk modulus for each part. If this calculation results in zero (for example, when using *MAT_NULL without a defined Young's modulus), LS-DYNA automatically replaces it with the largest bulk modulus found in the model. This behavior can cause variations in the contact response depending on which parts are included in the model, even if they are not in contact. To ensure consistent results, specify a nonzero DEFBLK to set a default bulk modulus for such parts. If DEFBLK is zero, the treatment is as before this field was implemented, meaning the model's largest bulk modulus is used for these parts.

*CONTROL_COUPLING

Purpose: Change defaults for MADYMO3D/CAL3D coupling; see Appendix I.

Card 1	1	2	3	4	5	6	7	8
Variable	UNLENG	UNTIME	UNFORC	TIMIDL	FLIPX	FLIPY	FLIPZ	SUBCYL
Type	F	F	F	F	I	I	I	I
Default	1.	1.	1.	0.	0	0	0	1

VARIABLE	DESCRIPTION
UNLENG	Unit conversion factor for length. MADYMO3D/GM-CAL3D lengths are multiplied by UNLENG to obtain LS-DYNA lengths.
UNTIME	Unit conversion factor for time, UNTIME. MADYMO3D/GM-CAL3D time is multiplied by UTIME to obtain LS-DYNA time.
UNFORC	Unit conversion factor for force, UNFORC. MADYMO3D/GM-CAL3D force is multiplied by UNFORC to obtain LS-DYNA force.
TIMIDL	Idle time during which CAL3D or MADYMO is computing and LS-DYNA remains inactive. Important for saving computer time.
FLIPX	Flag for flipping *X*-coordinate of CAL3D/MADYMO3D relative to the LS-DYNA model: EQ.0: off, EQ.1: on.
FLIPY	Flag for flipping *Y*-coordinate of CAL3D/MADYMO3D relative to the LS-DYNA model: EQ.0: off, EQ.1: on.
FLIPZ	Flag for flipping *Z*-coordinate of CAL3D/MADYMO3D relative to the LS-DYNA model: EQ.0: off, EQ.1: on.

VARIABLE	DESCRIPTION
SUBCYL	CAL3D/MADYMO3D subcycling interval (# of cycles): EQ.0: Set to 1, GT.0: SUBCYL must be an integer equal to the number of LS-DYNA time steps between each CAL3D/MADYMO3D step. Then the position of the contacting rigid bodies is assumed to be constant for n LS-DYNA time steps. This may result in some increase in the spikes in contact, thus this option should be used carefully. As the CAL3D/MADYMO3D programs usually work with a very small number of degrees of freedom, not much gain in efficiency can be achieved.

*CONTROL_CPG

Purpose: Global control parameters for CPG (Continuum-based Gas Particle method).

Card Summary:

Card 1. This card is required.

	NCPC		VERB		NSLIP	IBCHK	ICORR

Card 2. This card is optional.

ITURB							

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable		NCPC		VERB		NSLIP	IBCHK	ICORR
Type		I		I		I	I	I
Default		5		0		0	1	1

VARIABLE	DESCRIPTION
NCPC	Number of cycles between point cloud checks. A negative value points to a load curve ID giving the number of cycles as a function of time. See Remark 1.
VERB	CPG verbosity control: EQ.0: No CPG screen/stdout output EQ.1: CPG output displayed on screen/stdout EQ.2: Same as 1, except the information is also output in the mes0000 file
NSLIP	Slip condition flag: EQ.0: Free slip condition applied at walls EQ.1: Non-slip condition applied at walls

VARIABLE	DESCRIPTION
IBCHK	Mesh integrity check: EQ.1: Verify that the external mesh is closed and the normal vectors are consistently oriented (default). EQ.2: Turn off integrity check.
ICORR	Correction of density and total energy: EQ.1: Thermodynamic balance correction on (default). See Remark 2. EQ.2: Turn off correction.

Optional Card.

Card 2	1	2	3	4	5	6	7	8
Variable	ITURB							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ITURB	Turbulence model: EQ.0: Off EQ.1: Standard $k-\varepsilon$ model

Remarks:

1. **Point cloud check.** CPG relies on a point cloud sampling of the entire volume. As the airbag deforms, the point cloud needs to be updated with the creation and deletion of CPG particles and the redistribution of particles across processors. If no deformation occurs throughout the analysis, this resampling is not needed and a very high value of NCPC can be set to save some calculation time. For typical airbag applications, the default value is recommended.

2. **ICORR.** At any given time, the solver calculates the total volume of the domain, the total mass of gas inside the volume, and the total energy. From these

quantities, an expected average pressure can be calculated and used to correct the particle pressure field for a better thermodynamic balance.

*CONTROL_CPG_OUTPUT

Purpose: Control the content of the CPG output variables in the d3dat database. This keyword allows you to reduce the size of the output data by removing or adding variables.

Card Sets. The data cards for this keyword are one Card 1 followed by as many instances of Card 2 as desired. Card 1 specifies globally whether CPG variables are output, but this behavior is overwritten for individual variables using Card 2. For instance, setting OUTLV = -1 causes every CPG variable to not be output, except for those listed with Card 2 for which output is enabled (ION = 1).

This card is required.

Card 1	1	2	3	4	5	6	7	8
Variable	OUTLV							
Type	I							
Default	0							

Include as many instances of this card as needed. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	VNAME				ION			
Type	A40				I			
Default	none				0			

VARIABLE	DESCRIPTION
OUTLV	Global CPG data output level control: EQ.0: Every CPG variable is output (default). EQ.-1: No CPG variable is output.
VNAME	CPG variable name. This input is case insensitive. See Table 12-1 for a complete list.

VARIABLE	DESCRIPTION
ION	Output status: EQ.0: The behavior depends on OUTLV. EQ.1: This variable is output. EQ.2: This variable is not output.

Table 12-1. This table provides a list of the VNAMEs for each variable that can be output to d3dat.

VNAME	Description
adaptive_estimate	Estimated distanced to the wall for each particle. This is only available when *DEFINE_CPG_ADAPTIVE is used.
density	Density of the gas
gas_fractions	Mass fraction of each species in the gas
internal_energy	Internal energy of the gas in the airbag
mach_number	Mach number
neighbor_count	Number of neighbors for each particle
pressure_static	Static pressure
pressure_total	Total pressure
processor	Rank each particle belongs to
tau_wall	Shear stress on the wall of the airbag
temperature_static	Static temperature
temperature_total	Temperature a flowing fluid would reach if it stopped moving adiabatically
turbulence	Turbulence quantities. This VNAME is only available if ITURB ≠ 0 on *CONTROL_CPG. k, ε, and turbulent viscosity are output for ITURB = 1 (the standard k-ε model).
velocity	Velocity of the fluid
velocity_leakage	Component of the velocity for the leaked boundary particles normal to the boundary elements
velocity_mesh	Lagrangian velocity of the particles. It follows the mesh for boundary particles.
vorticity	Vorticity

*CONTROL_CPM

Purpose: Global control parameters for CPM (Corpuscular Particle Method).

Card 1	1	2	3	4	5	6	7	8
Variable	CPMOUT	NP2P	NCPMTS	CPMERR	SFFDC	BLKV	CPMMF	P2PMIX
Type	I	I	I	I	F	I	I	I
Default	11	5	0	0	1.0	0	0	0

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	PMIS							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
CPMOUT	Control CPM output database to the d3plot files (see Remark 1): EQ.11: Full CPM database in version 3 format (default) EQ.21: Full CPM database in version 4 format EQ.22: CPM coordinates only in version 4 format EQ.23: CPM summary only in version 4 format
NP2P	Number of cycles for repartition particle among processors. This option is only used in LS-DYNA/MPP. (Default = 5)
NCPMTS	Time step size estimation: EQ.0: Do not consider CPM (default) EQ.1: Use 1 microsecond as CPM time step size. This provides a better time step size if the model is made up of rigid bodies.

VARIABLE	DESCRIPTION
CPMERR	EQ.0: Disable checking and only output warning messages (Default) EQ.1: Enable error checking. If LS-DYNA detects any problem, it will either error terminate the job or try to fix the problem. Activated checks include: 1. *Airbag* integrity (see Remark 2) 2. *Chamber* integrity: this step applies the airbag integrity check to the chamber. 3. Inconsistent orientation between the shell reference geometry and FEM shell connectivity.
SFFDC	Scale factor for the force decay constant. The default value is 1.0 and allowable arrange is [0.01,100.0]. See Remark 3.
BLKV	Allocate additional memory for contact nodal forces, excluding force transducers and airbag single surface contacts using SOFT = 2. These nodal forces will be used to estimate blockage of external vents. BLKV does not affect the porosity and blockage of internal vents.
CPMMF	Flag to consider airbag system velocity based on the coordinates system defined by fields NID1, NID2, and NID3 on *AIRBAG_PARTICLE: EQ.0: No (default) EQ.1: Yes. The flow energy from the rigid body motion is fed back to the CPM particles.
P2PMIX	Control the energy transfer during particle-to-particle collision and change the thermalization of the particles (see Remark 4): EQ.0: Thermalization considered in all particle-to-particle collisions (default). EQ.1: Thermalization only considered within same gas species EQ.2: Same as 0 but treat temperature dependent $\xi(T)$ EQ.3: Same as 1 but treat temperature dependent $\xi(T)$
PMIS	Flag for choosing logic to use when a particle leaks out due to undetected contact (see Remark 5):

VARIABLE	DESCRIPTION
	EQ.0: Return particle to around inflator orifice node (default)
	EQ.1: Return particle to the nearest airbag fabric

Remarks:

1. **D3PLOT Version.** "Version 3" is an older format than "Version 4". Version 4 stores data more efficiently than version 3 and has options for what data is stored but may not be readable by old LS-PrePost executables.

2. **Airbag Integrity Checking.** The bag's volume is used to evaluate all bag state variables. If the volume is ill-defined or inaccurate, then the calculation will fail. Therefore, it is vital that that the volume be closed, and that all shell normal vectors point in the same direction.

   When CPMERR = 1, the calculation will error terminate if either the bag's volume is not closed or if one of its parts is not internally oriented (meaning that it contains elements that are not consistently oriented). Once it is verified that each part has a well-defined orientation, an additional check is performed to verify that all of bag's constituent parts are consistently oriented with respect to each other. If they are not, then the part orientations are flipped until the bag is consistently oriented with an *inward* pointing normal vector.

3. **Force Decay Constant.** Particle impact force is gradually applied to airbag segment by a special smoothing function with the following form.

$$F_{\text{apply}} = \left[1 - \exp\left(\frac{-dt}{\text{SFFDC} \times \tau}\right)\right](F_{\text{current}} + F_{\text{stored}}) \ ,$$

   where τ is the force decay constant stored in LS-DYNA.

4. **P2PMIX.** P2PMIX sets the energy transfer scheme during particle-to-particle collision**.**

   Based on the molecular kinetic theory, a fraction, ξ, of the total internal energy, e, is translational kinetic energy, w_k, for the CPM particle performing PV work:

$$p = \frac{2}{3} w_k = \frac{2}{3} \xi(T) e$$

   This pressure should be the same as the pressure given by the Ideal Gas Law:

$$p = (\gamma - 1) e$$

   In the above, ξ is a weak function of temperature like γ.

Each CPM particle represents a cloud of molecules, and CPM assumes those molecules are under thermal equilibrium with a proper Maxwell-Boltzmann distribution. When two particles collide, they will follow a perfectly elastic collision and the amount of energy transferred between them will keep both particles under the new Maxwell-Boltzmann distribution. By default, CPM assumes constant ζ during the process for better speed.

Since each gas species has its own Maxwell-Boltzmann distribution, the above calculation should only be applied to the same gas species. However, by default (P2PMIX = 0), thermalization is considered for all interactions. When P2PMIX equals 1, CPM particles collide perfectly elastically, and simple energy conservation is enforced when particles of different species interact. In this case the above calculation is only applied to particles of the same gas species.

5. **PMIS.** A particle may leak from a particle-to-surface contact due to numerical error. Once this particle is detected, it will be relocated to the gas inflator orifice node by default. If the bag is modeled with a multiple chamber definition, the default scheme may put the particle into a different chamber and, thus, unintentionally change the chamber pressure. If PMIS equals 1, this particle will be returned to the closest fabric segment.

*CONTROL_CPU

Purpose: Control CPU time.

Card 1	1	2	3	4	5	6	7	8
Variable	CPUTIM	IGLST						
Type	F	I						

VARIABLE	DESCRIPTION
CPUTIM	Seconds of CPU time: EQ.0.0: No CPU time limit set GT.0.0: Time limit for cumulative CPU of the entire simulation, including all restarts. LT.0.0: Absolute value is the CPU time limit in seconds for the first run and for each subsequent restart.
IGLST	Flag for outputting CPU and elapsed times in the glstat file: EQ.0: No EQ.1: Yes

Remarks:

The CPU limit is not checked until after the initialization stage of the calculation. Upon reaching the CPU limit, the code will output a restart dump file and terminate. The CPU limit can also be specified on the LS-DYNA execution line via "c=". The value specified on the execution line will override CPUTIM specified in this keyword.

*CONTROL_DEBUG

Purpose: Write supplemental information to the messag file(s). One effect of this command is that the sequence of subroutines called during initialization and memory allocation is printed. Aside from that, the extra information printed pertains only to a select few features, including:

1. Spot weld connections which use *MAT_100_DA and *DEFINE_CONNECTION_PROPERTIES.

2. The GISSMO damage model invoked using *MAT_ADD_EROSION. (Supplemental information about failed elements is written.)

*CONTROL_DISCRETE_ELEMENT

Purpose: Define global control parameters for discrete element spheres.

Card Summary:

Card 1. This card is required.

NDAMP	TDAMP	FRICS	FRICR	NORMK	SHEARK	CAP	VTK

Card 2. Additional card for CAP ≠ 0. If CAP = 0, include this card as a blank line if optional Card 3 is included.

GAMMA	VOL	ANG	GAP		IGNORE	NBUF	PARALLEL

Card 3. Card 3 and beyond are optional. If any of the following cards are included, then the preceding optional cards must be included, but they can be blank lines.

LNORM	LSHEAR		FRICD	DC	NCRB	BT	DT

Card 4. This card is optional.

				HT_COEF	ATM_TMP		

Card 5. This card is optional.

IDESOFT	SOFSCL		ISKIP	MAXNEI			

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NDAMP	TDAMP	FRICS	FRICR	NORMK	SHEARK	CAP	VTK
Type	F	F	F	F	F	F	I	I
Default	0.	0.	0.	0.	0.01	2/7	0	0

VARIABLE	DESCRIPTION
NDAMP	Normal damping coefficient
TDAMP	Tangential damping coefficient

VARIABLE	DESCRIPTION
FRICS	Static coefficient of friction (see Remark 4): EQ.0: 3 DOF NE.0: 6 DOF (consider rotational DOF)
FRICR	Rolling friction coefficient
NORMK	Optional: scale factor of normal spring constant. Norm contact stiffness is calculated as $K_n = \begin{cases} \frac{k_1 r_1 k_2 r_2}{k_1 r_1 + k_2 r_2}\text{NORMK} & \text{if NORMK} > 0 \\ \lvert\text{NORMK}\rvert & \text{if NORMK} < 0 \end{cases}$. NORMK is ignored if LNORM ≠ 0 or IDESOFT = 1.
SHEARK	Optional: ratio between SHEARK/NORMK. Tangential stiffness is calculated as $K_t = \text{SHEARK} \times K_n$. SHEARK is ignored if LSHEAR ≠ 0.
CAP	Capillary force flag: EQ.0: Dry particles NE.0: Wet particles. Capillary force is considered, and an additional input card is needed. See Remark 1.
VTK	Output DES in VTK format for ParaView: EQ.0: No EQ.1: Yes

Capillary Card. Additional card for CAP ≠ 0. If CAP = 0, include as a blank line if optional Card 3 is included.

Card 2	1	2	3	4	5	6	7	8
Variable	GAMMA	VOL	ANG	GAP		IGNORE	NBUF	PARALLEL
Type	F	F	F	F		I	I	I
Default	0.	0.	0.	0.		0	6	0

VARIABLE	DESCRIPTION
GAMMA	Liquid surface tension, γ
VOL	Volume fraction
ANG	Contact angle, θ
GAP	Optional parameter affecting the spatial limit of the liquid bridge. A liquid bridge exists when δ, as illustrated in Figure 12-15, is less than or equal to $\min(\text{GAP}, d_{\text{rup}})$ where d_{rup} is the rupture distance of the bridge automatically calculated by LS-DYNA.
IGNORE	Ignore initial penetration option: EQ.0: Calculate the contact force for DES with initial penetration. GT.0: Ignore the contact force calculation for DES with initial penetration.
NBUF	GE.0: Factor of memory use for asynchronous message buffer (default = 6). LT.0: Disable asynchronous scheme and use minimum memory for data transfer.
PARALLEL	Flag for calculating contact force between bonded DES: EQ.0: Skip the contact force calculation for bonded DES (default) EQ.1: Consider the contact force calculation for bonded DES

Card 3 is optional. If included, then Card 2 must be defined, even as a blank card.

Card 3	1	2	3	4	5	6	7	8
Variable	LNORM	LSHEAR		FRICD	DC	NCRB	BT	DT
Type	I	I		F	F	I	F	F
Default	0	0		FRICS	0	0	0.	10^{20}

VARIABLE	DESCRIPTION
LNORM	Load curve ID of a curve that defines a function for normal stiffness with respect to norm penetration ratio. See Remark 2.
LSHEAR	Load curve ID of a curve that defines a function for shear stiffness with respect to norm penetration ratio. See Remark 3.
FRICD	Dynamic coefficient of friction. See Remark 4.
DC	Exponential decay coefficient. See Remark 4.
NCRB	Rebalancing frequency, meaning the number of cycles between each rebalancing. This parameter only applies to MPP. EQ.0: No rebalancing is performed (default).
BT	Birth time
DT	Death time

Thermal Properties Card. Card 4 is optional.

Card 4	1	2	3	4	5	6	7	8
Variable					HT_COEF	ATM_TMP		
Type					F	F		
Default					0.0	0.0		

VARIABLE	DESCRIPTION
HT_COEF	Convective heat transfer coefficient in $W/(m^2K)$
ATM_TMP	Atmospheric temperature in K or °C

Card 5 is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	IDESOFT	SOFSCL		ISKIP	MAXNEI			
Type	I	F		I	I			
Default	0	0.1		0	20			

VARIABLE	DESCRIPTION
IDESOFT	Flag for soft constraint formulation: EQ.1: Soft constraint formulation. The contact stiffness is based on the nodal mass and the global time step size. This input provides a different way for calculating NORMK. NORMK is ignored if IDESOFT = 1. IDESOFT is ignored if LNORM ≠ 0.
SOFSCL	Scale factor applied to the contact stiffness in the soft constrain formulation
ISKIP	Flag for skipping the calculation of contact force between DES: EQ.0: Consider the particle-particle contact calculation (default). EQ.1: Skip the particle-particle contact calculation.
MAXNEI	Number of neighbors to be tracked for DES contact and capillary force calculation (default = 20). If particle sizes are very different, MAXNEI needs to be increased to capture more neighbors.

Background:

This method models all parts as being comprised of rigid spheres. These spheres interact with both conventional solids and other spheres. Sphere-sphere interactions are modeled in contact points using springs and dampers as illustrated in Figure 12-14. [Cundall & Strack 1979].

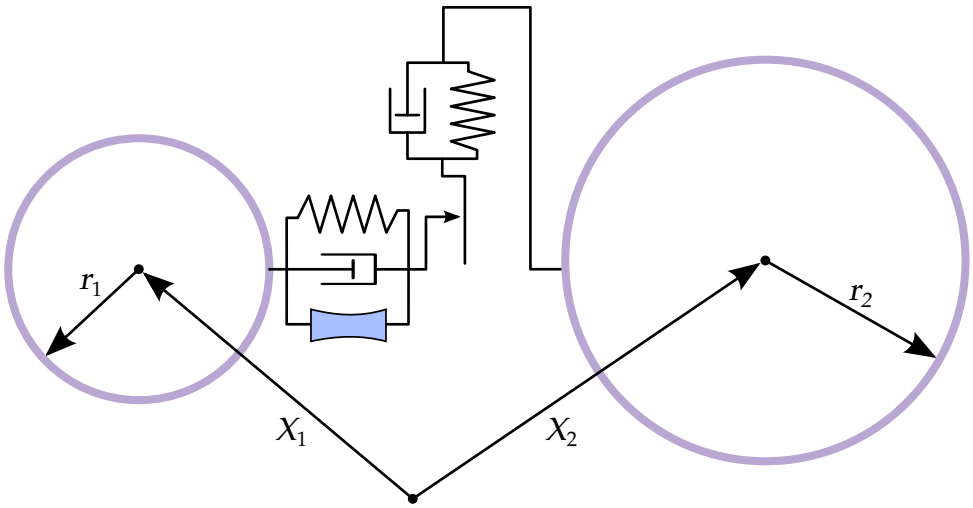


Figure 12-14. Schematic representation of sphere-sphere interaction

Remarks:

1. **Capillary forces to model cohesion.** This extension is enabled using the CAP field. Capillary force between wet particles is based on the following reference: "Capillary Forces between Two Spheres with a Fixed Volume Liquid Bridges: Theory and Experiment", Yakov I. Rabinovich et al. Langmuir 2005, 21, 10992-10997. See Figure 12-15.

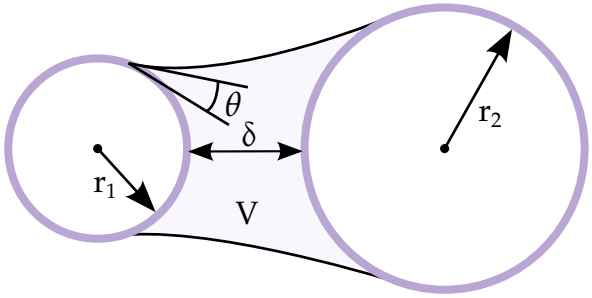


Figure 12-15. Schematic representation of capillary force model.

The capillary force is given by

$$F = -\frac{2\pi R\gamma\cos\theta}{1+\frac{\delta}{2d}},$$

where

$$d = \frac{\delta}{2}\left(-1+\sqrt{1+\frac{2V}{\pi R\delta^2}}\right),$$

and

$$R = \frac{2r_1 r_2}{r_1 + r_2} \ .$$

2. **User-defined norm stiffness.** Let $y = f(x)$ be a load curve specified with LNORM for the norm stiffness between two interacting discrete element spheres. x is the relative penetration, meaning $x = \delta/\min(r_1, r_2)$. δ is penetration. The normal spring force is calculated as

$$F_n = k_{\text{eff}} \times y \times \left(\min(r_1, r_2)\right)^2 ,$$

where k_{eff} is the effective bulk modulus of the two interacting DEM particles, that is, $k_{\text{eff}} = k_1 k_2/(k_1 + k_2)$. If the curve is defined as $y = cx$, the behavior is the same as NORMK = c.

3. **User-defined shear stiffness.** Let $y = f(x)$ be a load curve specified with LSHEAR for the shear stiffness between two interacting discrete element spheres. x is the relative penetration, meaning $x = \delta/\min(r_1, r_2)$. δ is penetration. The tangential stiffness is calculated as $K_t = y \times K_n$, where K_n is the norm stiffness defined by NORMK or a user-defined curve. If a curve is defined as $y = c$, the behavior is the same as SHEARK = c.

4. **Coefficient of friction.** The frictional coefficient is assumed to be dependent on the relative velocity v_{rel} of the two DEM in contact, that is,

$$\mu_c = \text{FRICD} + (\text{FRICS} - \text{FRICD})e^{-\text{DC} \times |v_{\text{rel}}|} \ .$$

*CONTROL_DYNAMIC_RELAXATION

Purpose: Initialize stresses and deformation in a model to simulate a preload. Examples of preload include load due to gravity, load due to a constant angular velocity, and load due to torquing of a bolt. After the preloaded state is achieved by one of the methods described below (see field IDRFLG), the time resets to zero and the normal phase of the solution automatically begins from the preloaded state.

Card Summary:

Card 1. This card is required.

NRCYCK	DRTOL	DRFCTR	DRTERM	TSSFDR	IRELAL	EDTTL	IDRFLG

Card 2a. This card is read if IDRFLG = 1. It is optional.

DRTMIN							

Card 2b. This card is included if IDRFLG = -3, 3, or 6.

DRPSET							

Card 2c. This card is included if IDRFLG = 2.

NC	NP						

Card 2c.1. Include NP of this card.

PSID	VECID						

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	NRCYCK	DRTOL	DRFCTR	DRTERM	TSSFDR	IRELAL	EDTTL	IDRFLG
Type	I	F	F	F	F	I	F	I
Default	250	0.001	0.995	infinity	TSSFAC	0	0.04	0

VARIABLE	DESCRIPTION
NRCYCK	Number of time steps between convergence checks for explicit dynamic relaxation.
DRTOL	Convergence tolerance for explicit dynamic relaxation (default = 0.001). See Remark 1.
DRFCTR	Dynamic relaxation factor for transient dynamic relaxation (default = .995). See Remark 1.
DRTERM	Optional termination time for dynamic relaxation. Termination occurs at this time or when convergence is attained (default = infinity). See Remarks 1 and 6.
TSSFDR	Scale factor for computed time step during explicit dynamic relaxation. If zero, the value is set to TSSFAC defined on *CONTROL_TIMESTEP. After converging, the scale factor is reset to TSSFAC.
IRELAL	Automatic control for dynamic relaxation option based on algorithm of Papadrakakis [1981]: EQ.0: Not active EQ.1: Active
EDTTL	Convergence tolerance on automatic control of dynamic relaxation.
IDRFLG	Dynamic relaxation flag which controls how the preloaded state is computed: EQ.-999: Dynamic relaxation not activated even if specified on a load curve, see *DEFINE_CURVE. See Remark 3. 2. EQ.-3: Dynamic relaxation is activated as with IDRFLG = 1, but the convergence check is made based only on the part set specified by DRPSET. All parts are active during the dynamic relaxation phase. EQ.-1: Dynamic relaxation is activated, and time history output is produced during dynamic relaxation; see Remarks 1 and 2. EQ.0: Not active. See Remark 3. EQ.1: Dynamic relaxation is activated. See Remark 1. EQ.2: Initialization to a prescribed geometry, see Remark 4,

VARIABLE	DESCRIPTION
	EQ.3: Dynamic relaxation is activated as with IDRFLG = 1, but only for the part set specified by DRPSET; see Remark 5.
	EQ.5: Initialize implicitly; see Remark 6.
	EQ.6: Initialize implicitly but only for the part set specified by DRPSET; see Remark 6.

Minimum Dynamic Relaxation Time Card. This card is read if IDRFLG = 1. It is optional.

Card 2a	1	2	3	4	5	6	7	8
Variable	DRTMIN							
Type	I							
Default	0.0							

VARIABLE	DESCRIPTION
DRTMIN	Minimum time for the dynamic relaxation phase. If set, dynamic relaxation will not terminate until drtmin is reach even if convergence has been reached.

Part Set Card. Additional card for IDRFLG = -3, 3 or 6.

Card 2b	1	2	3	4	5	6	7	8
Variable	DRPSET							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
DRPSET	Part set ID for IDRFLG = -3, 3 or 6.

Optional Card for IDRFLG = 2. This card is included if and only if IDRFLG = 2.

Card 2c	1	2	3	4	5	6	7	8
Variable	NC	NP						
Type	I	I						
Default	100	0						

VARIABLE	DESCRIPTION
NC	Number of time steps for initializing geometry when IDRFLG = 2. This variable applies to IDRFLG = 2 whether polar initialization is invoked or not.
NP	Number of cards defining parts to be initialized using the polar initialization approach and the associated polar coordinate system(s).

Optional Polar Initialization Cards. When IDRFLG = 2 and NP > 0, include NP additional cards which define the parts and polar coordinate systems used in polar initialization. This polar initialization interpolates from the initial to final state using a polar coordinate system. The objective is to avoid creating spurious stresses in parts subjected to large rigid body rotations and this option should not be used if that is not the case.

Card 2c.1	1	2	3	4	5	6	7	8
Variable	PSID	VECID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
PSID	Part set ID
VECID	Vector ID which defines the origin and axis of rotation of the polar coordinate system used in initializing parts of part set PSID. See Remark 7.

Remarks:

1. **Transient dynamic relaxation.** If IDRFLG is 1 or -1, a transient "dynamic relaxation" analysis is invoked in which an explicit analysis, damped by means of scaling nodal velocities by the factor DRFCTR each time step, is performed. When the ratio of current distortional kinetic energy to peak distortional kinetic energy (the convergence factor) falls below the convergence tolerance (DRTOL) or when the time reaches DRTERM, the dynamic relaxation analysis stops, and the current state becomes the initial state of the subsequent normal analysis.

   Distortional kinetic energy is defined as total kinetic energy less the kinetic energy due to rigid body motion. A history of the distortional kinetic energy computed during the dynamic relaxation phase is automatically written to a file called relax. This file can be read as an ASCII file by LS-PrePost and its data plotted. The relax file also includes a history of the convergence factor.

2. **Dynamic relaxation with time history data.** If IDRFLG is set to -1, the dynamic relaxation proceeds as normal, but time history data is written to the d3thdt file in addition to the normal data being written to the d3drlf file. At the end of dynamic relaxation, the problem time is reset to zero. However, information is written to the d3thdt file with an increment to the time value. The time increment used is reported at the end of dynamic relaxation.

3. **Dynamic relaxation from load curves.** Dynamic relaxation will be invoked if SIDR is set to 1 or 2 in any of the *DEFINE_CURVE commands, even if IDRFLG = 0 in *CONTROL_DYNAMIC_RELAXATION. Curves so tagged are applicable to the dynamic relaxation analysis phase. Curves with SIDR set to 0 or 2 are applicable to the normal phase of the solution. Dynamic relaxation will always be skipped if IDRFLAG is set to -999.

4. **Restarts and prescribed geometry file.** Following the dynamic relaxation phase and before the start of the normal solution phase, a binary dump file (d3dump01) and a "prescribed geometry" file (drdisp.sif) are written by LS-DYNA. Either of these files can be used in a subsequent analysis to quickly initialize to the preloaded state without having to repeat the dynamic relaxation run. The binary dump file is used with a restart analysis (see RESTART_INPUT_DATA). The drdisp.sif file is used by setting IDRFLG = 2. The IDRFLG = 2 approach does not initialize the location of rigid bodies in the model.

   When IDRFLG = 2, an ASCII file specified by "m=" on the execution line is read which describes the initialized state of deformable bodies. The ASCII file contains each node ID with prescribed values of nodal displacement (x, y, z), nodal rotation (x, y, z), and nodal temperature in (I8, 7E15.0) format. The preloaded state is quickly reached by linearly ramping nodal displacements, rotations, and temperatures of deformable bodies to prescribed values over a number of time steps as indicated by the variable NC.

5. **Part set for convergence checking.** When IDRFLG = 3, a dynamic relaxation is performed only for the part set specified by DRPSET. For example, if only the tires are being inflated on a vehicle, it may be sufficient in some cases to apply the dynamic relaxation on the part IDs in the tire and possibly the suspension system.

6. **Implicit preload.** If IDRFLG is set to 5 or 6, LS-DYNA performs an implicit analysis to obtain the preloaded state. Parameters for controlling the implicit preload solution are defined using appropriate *CONTROL_IMPLICIT keywords to specify solver type, implicit time step, etc. For example, *CONTROL_IMPLICIT_GENERAL specifies the implicit step size. The implicit analysis is, by default, static but can be made transient via the *CONTROL_IMPLICIT_DYNAMICS command.

   DRTERM indicates the termination "time" of the implicit preload analysis. When DRTERM is reached, the implicit preload phase terminates, and LS-DYNA begins the next phase of the analysis according to IMFLAG in *CONTROL_IMPLICIT_GENERAL. For example, if it is desired to run an implicit preload phase and switch to the explicit solver for the subsequent transient phase, IDRFLG should be set to 5 and IMFLAG should be set to 0.

   In contrast to IDRFLG = 5, IDRFLG = 6 performs an implicit analysis only for the part subset specified with DRPSET.

7. **Vector for prescribed geometry.** When the displacements for IDRFLG = 2 are associated with large rotations, the linear interpolation of the displacement field introduces spurious compression and tension into the part. If a part set is specified with a vector, the displacement is interpolated by using polar coordinates with the tail of the vector specifying the origin of the coordinate system and the direction specifying the normal to the polar coordinate plane. The run terminates with an error if VECID is not specified. If no large rotations are present, the optional cards may be omitted.

8. **Output database.** To create a binary output database having the same format as a d3plot database that pertains to the dynamic relaxation analysis, use *DATABASE_BINARY_D3DRLF. The output interval is given by this command as an integer representing the number of convergence checks between output states. The frequency of the convergence checks is controlled by the parameter NRCYCK.

*CONTROL_EFG

Purpose: Define controls for the mesh-free computation.

Card 1	1	2	3	4	5	6	7	8
Variable	ISPLINE	IDILA	ININT					
Type	I	I	I					
Default	0	0	12					
Remarks			1					

Card 2	1	2	3	4	5	6	7	8
Variable	IMLM	ETOL	IDEB	HSORT	SSORT			
Type	I	F	I	I	I			
Default	0	10^{-4}	0	not used	0			

VARIABLE	DESCRIPTION
ISPLINE	Optional choice for the mesh-free kernel functions: EQ.0: Cubic spline function (default) EQ.1: Quadratic spline function EQ.2: Cubic spline function with circular disk (see Remark 2)
IDILA	Optional choice for the normalized dilation parameter: EQ.0: Maximum distance based on the background element EQ.1: Maximum distance based on surrounding nodes
ININT	Factor needed for the estimation of maximum workspace (MWSPAC) that can be used during the initialization phase.

VARIABLE	DESCRIPTION
IMLM	Optional choice for the matrix operation, linear solving and memory usage (see Remark 3): EQ.1: Original BCSLIB-EXT solvers EQ.2: EFGPACK (recommended). When this option is used, ININT in Card 1 becomes redundant.
ETOL	Error tolerance for the IMLM option.
IDEB	Output internal debug message
HSORT	Not used
SSORT	Automatic sorting of background triangular shell elements to FEM #2 when EFG shell type 41 is used: EQ.0: no sorting EQ.1: full sorting

Remarks:

1. **Maximum Workspace.** The mesh-free computation requires calls to use BCSLIB-EXT solvers during the initialization phase. The maximum workspace (MWSPAC) that can be used during the call is calculated as

$$\text{MWSPAC} = \text{ININT}^3 \times \text{NUMNEFG},$$

where NUMNEFG is the total number of mesh-free nodes. ININT, which is the number of nodes that a node influences along each cardinal direction, defaults to 12. When the normalized dilation parameters (DX,DY,DZ) in *SECTION_SOLID_EFG are increased, ININT must likewise increase.

2. **Cubic Spline Function with Circular Disk.** When ISPLINE = 2 is used, the input of the normalized dilation parameters (DX,DY,DZ) for the kernel function in *SECTION_SOILD_EFG and SECTION_SHELL_EFG only requires the DX value.

3. **Solvers.** EFGPACK was added to automatically compute the required maximum workspace in the initialization phase and to improve efficiency in the matrix operations, linear solving, and memory usage. The original BCSLIB-EXT solver requires an explicit workspace (ININT) for the initialization.

*CONTROL_ENERGY

Purpose: Provide controls for energy dissipation options.

Card 1	1	2	3	4	5	6	7	8
Variable	HGEN	RWEN	SLNTEN	RYLEN	IRGEN	MATEN	DRLEN	DISEN
Type	I	I	I	I	I	I	I	I
Default	1	2	1	1	2	1	1	1

VARIABLE	DESCRIPTION
HGEN	Hourglass energy calculation option. This option requires significant additional storage and increases cost by ten percent: EQ.1: Hourglass energy is not computed (default). EQ.2: Hourglass energy is computed and included in the energy balance. The hourglass energies are reported in the ASCII files glstat and matsum, see *DATABASE_*OPTION*. For implicit, or if the DRCPSID is active on *CONTROL_SHELL, the drilling energy is included here.
RWEN	Rigidwall energy (a.k.a. stonewall energy) dissipation option: EQ.1: Energy dissipation is not computed. EQ.2: Energy dissipation is computed and included in the energy balance (default). The rigidwall energy dissipation is reported in the ASCII file glstat; see *DATABASE_*OPTION*.
SLNTEN	Sliding interface energy dissipation option (This parameter is always set to 2 if contact is active. The option SLNTEN = 1 is not available.): EQ.1: Energy dissipation is not computed. EQ.2: Energy dissipation is computed and included in the energy balance. The sliding interface energy is reported in ASCII files glstat and sleout; see *DATABASE_*OPTION*.
RYLEN	Rayleigh energy dissipation option (damping energy dissipation):

VARIABLE	DESCRIPTION
	EQ.1: Energy dissipation is not computed (default).
	EQ.2: Energy dissipation is computed and included in the energy balance. The damping energy is reported in ASCII file glstat and matsum; see *DATABASE_*OPTION*.
IRGEN	Initial reference geometry energy option (included in internal energy, resulting from *INITIAL_FOAM_REFERENCE_GEOMETRY):
	EQ.1: Initial reference geometry energy is not computed.
	EQ.2: Initial reference geometry energy is computed and included in the energy balance as part of the internal energy (default).
MATEN	Detailed material energies option. For a choice of material models (currently supported are 3, 4, 15, 19, 24, 63, 81, 82, 98, 104, 105, 106, 107, 123, 124, 188, 224, 225, 240, and 251 for shell and solid elements), internal energy is additionally split into elastic, plastic, and damage portions:
	EQ.1: Detailed material energies are not computed (default).
	EQ.2: Detailed material energies are computed and reported as *mat_energy_elastic*, *mat_energy_plastic*, and *mat_energy_damage* in the ASCII files glstat and matsum.
DRLEN	Drilling energy calculation option, for implicit and with use of DRCPSID/DRCPRM on *CONTROL_SHELL:
	EQ.1: Drilling energy is not computed (default).
	EQ.2: Drilling energy is computed and included in the energy balance. The drilling energies are reported in the ASCII file glstat, see *DATABASE_*OPTION*.
DISEN	Dissipation energy calculation option, for implicit:
	EQ.1: Dissipated energy is not computed (default).
	EQ.2: Dissipated kinetic and internal energy is computed and included in the energy balance. The dissipation energies are reported in the ASCII file glstat, see *DATABASE_*OPTION*.

*CONTROL_EOS_USER_LIBRARY_{OPTION}

Available options include:

<BLANK>

ASCII

Purpose: Provide a path to a user-defined library called seslib and scaling factors to convert the units of the input deck into the units of the seslib data.

Note that the units of the seslib data are not in a consistent unit system. The units are g, cm, μs, and GPa. If the units were consistent, the pressure would be in Mbars (= 100 GPa) and would not need an additional conversion factor.

The ASCII keyword option enables specifying the name of an ASCII file that contains data tables. The library seslib is generated during initialization. These tables should have the format of the Sesame ASCII File (LA-UR-19-24891).

Card 1	1	2	3	4	5	6	7	8
Variable	PATH							
Type	A80							
Default	Current working directory							

Optional Unit Conversion Card. Optional card for converting the units of the input deck into the units for seslib data.

Card 2	1	2	3	4	5	6	7	8
Variable	CONM	CONL	CONT	CONP				
Type	F	F	F	F				
Default	1.0	1.0	1.0	100.0				

Optional Plot Card. Optional card for outputting 5 keyword files (*.k files with *NODE and *ELEMENT_SHELL) during initialization to facilitate plotting 3D surface graphs based on seslib data in LS-PrePost. When data differ by several orders of magnitude, the scale factors on this card improve displaying the graphs in LS-PrePost, such as by preventing these surfaces from being stretched.

Card 3	1	2	3	4	5	6	7	8
Variable	SCLR	SCLT	SCLE	SCLP				
Type	F	F	F	F				
Default	1.0	1.0	1.0	1.0				

VARIABLE	DESCRIPTION
PATH	Path to the library seslib. The default path is the current working directory. For the ASCII keyword option, the path is replaced by the name of an ASCII file containing data tables. The filename should include the path if the file is not in the current directory.
CONM	Scaling factor for mass conversion from the input deck units to the library units
CONL	Scaling factor for length conversion from the input deck units to the library units
CONT	Scaling factor for time conversion from the input deck units to the library units
CONP	Scaling factor for pressure conversion from the input deck units to the library units
SCLR	Plot scaling factor for densities
SCLT	Plot scaling factor for temperatures
SCLE	Plot scaling factor for energies
SCLP	Plot scaling factor for pressures

*CONTROL_EXPLICIT_THERMAL

The *CONTROL_EXPLICIT_THERMAL_SOLVER keyword activates an explicit finite volume code to solve heat transfer by conduction. Enthalpies and temperatures are element centered. The elements supported by the thermal solver are beams, shells, solids, and multi-material 3D ALE elements. The *CONTROL_EXPLICIT_THERMAL_PROPERTIES keyword defines the heat capacities and conductivities by parts. These 2 keywords are mandatory to properly run the solver. Other keywords can be used to set the initial and boundary conditions and control the outputs. They are all listed below in alphabetical order:

*CONTROL_EXPLICIT_THERMAL_ALE_COUPLING

*CONTROL_EXPLICIT_THERMAL_BOUNDARY

*CONTROL_EXPLICIT_THERMAL_CONTACT

*CONTROL_EXPLICIT_THERMAL_INITIAL

*CONTROL_EXPLICIT_THERMAL_OUTPUT

*CONTROL_EXPLICIT_THERMAL_PROPERTIES

*CONTROL_EXPLICIT_THERMAL_SOLVER

*CONTROL_EXPLICIT_THERMAL_ALE_COUPLING

Purpose: Define the shell and solid parts involved in an explicit finite volume thermal coupling with multi-material ALE groups. This keyword requires *CONSTRAINED_-LAGRANGE_IN_SOLID with CTYPE = 4.

Card 1	1	2	3	4	5	6	7	8
Variable	PARTSET	MMGSET						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
PARTSET	Part set ID (See *SET_PART)
MMGSET	Multi-material set ID (see *SET_MULTI-MATERIAL_GROUP_-LIST)

*CONTROL_EXPLICIT_THERMAL_BOUNDARY

Purpose: Set temperature boundaries with segment sets for an explicit finite volume thermal analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	SEGSET	LCID						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
SEGSET	Segment set ID (See *SET_SEGMENT)
LCID	*DEFINE_CURVE ID defining the temperature as a function of time

Remarks:

1. **Boundary Elements.** The boundary temperatures are set at segment centers. If shells or beams have all their nodes in the segment set, these elements will be considered boundary elements; the temperatures at their centers will be controlled by the curve LCID.

*CONTROL_EXPLICIT_THERMAL_CONTACT

Purpose: Define the beam, shell and solid parts involved in an explicit finite volume thermal contact.

Card 1	1	2	3	4	5	6	7	8
Variable	PARTSET	NCYCLE						
Type	I	F						
Default	none	1						

VARIABLE	DESCRIPTION
PARTSET	Part set ID (See *SET_PART)
NCYCLE	Number of cycle between checks of new contact

*CONTROL_EXPLICIT_THERMAL_INITIAL

Purpose: Initialize the temperature centered in beams, shells or solids involved in an explicit finite volume thermal analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDTYP	TEMPINI					
Type	I	F	F					
Default	none	none	0.0					

VARIABLE	DESCRIPTION
ID	Flag to determine if ID is an element or set ID: GT.0: ID is a set LT.0: |ID| is an element
IDTYP	Type of ID: EQ.1: solid EQ.2: shell EQ.3: beam EQ.4: thick shell
TEMPINI	Initial temperature (see Remark 1)

Remarks:

1. **Material with *EOS.** The volumetric enthalpy is the sum of the pressure and volumetric internal energy (as defined in *EOS). If the material has an equation of state, the enthalpy should not be initialized by the temperature but by the initial volumetric internal energy and pressure set in *EOS.

*CONTROL_EXPLICIT_THERMAL_OUTPUT

Purpose: Output temperatures and enthalpies for an explicit finite volume thermal analysis. See Remark 1.

Card 1	1	2	3	4	5	6	7	8
Variable	DTOUT	DTOUTYP	SETID	SETYP				
Type	F	I	I	I				
Default	none	0	0	0				

VARIABLE	DESCRIPTION
DTOUT	Time interval between outputs
DTOUTYP	Type of DTOUT: EQ.0: DTOUT is a constant EQ.1: DTOUT is the ID of *DEFINE_CURVE defining a table of time as function of DTOUT
SETID	Set ID. If SETID = 0, then temperatures and enthalpies are output for the whole model. See Remark 1.
SETYP	Type of set: EQ.1: solid set (see *SET_SOLID) EQ.2: shell set (see *SET_SHELL) EQ.3: beam set (see *SET_BEAM) EQ.4: thick shell set (see *SET_TSHELL)

Remarks:

1. **Output Files.** The temperatures and enthalpies are calculated at the element center. If SETID is the default 0, then the temperature and enthalpy data is output in d3plot format files with the following basename: xplcth_output. If SETID is not 0, then the temperatures and enthalpies are calculated for a subset of elements defined as specified in the SETYP field. For this case the temperature and enthalpy histories are output by element in a .xy format. The file names are temperature_{beam,shell,solid}ID.xy and enthalpy_{beam,shell,solid}ID.xy.

2. **Viewing Results with LS-Prepost.** When SETID = 0, the output can be viewed with LS-Prepost using a fringe plot. To view the output for solid and shell elements select *FriComp* → *X-stress and Y-stress* (select Mid for the integration points in shells); for beams select *FriComp* → *beam* → *axial force resultant* and *s-force resultant*.

*CONTROL_EXPLICIT_THERMAL_PROPERTIES

Purpose: Define the thermal properties of beam, shell and solid parts involved in an explicit finite volume thermal analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	PARTSET	CP	CPTYP	VECID1	VECID2	LOCAL		
Type	I	F	I	I	I	I		
Default	none	none	0	0	0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	Kxx	Kxy	Kxz	KxxTYP	KxyTYP	KxzTYP		
Type	F	F	F	I	I	I		
Default	0.0	0.0	0.0	0	0	0		

Card 3	1	2	3	4	5	6	7	8
Variable	Kyx	Kyy	Kyz	KyxTYP	KyyTYP	KyzTYP		
Type	F	F	F	I	I	I		
Default	0.0	0.0	0.0	0	0	0		

Card 4	1	2	3	4	5	6	7	8
Variable	Kzx	Kzy	Kzz	KzxTYP	KzyTYP	KzzTYP		
Type	F	F	F	I	I	I		
Default	0.0	0.0	0.0	0	0	0		

VARIABLE	DESCRIPTION
PARTSET	Part set ID (See *SET_PART)
CP	Heat capacity
CPTYP	Type of CP: EQ.0: CP is a constant EQ.1: CP is the ID of a *DEFINE_CURVE defining a table of temperature as a function of heat capacity
VECID1, VECID2	*DEFINE_VECTOR IDs to define a specific coordinate system. VECID1 and VECID2 give the x- and y-direction, respectively. The z-vector is a cross product of VECID1 and VECID2. If VECID2 is not orthogonal to VECID1, its direction will be corrected with a cross-product of the z- and x-vectors. The conductivity matrix Kij is applied in this coordinate system.
LOCAL	Flag to activate an element coordinate system: EQ.0: The vectors VECIDj are considered in a global coordinate system. EQ.1: The vectors VECIDj are considered in a local system attached to the element. For shells and solids, the system is the same as DIREC = 1 and CTYPE = 12 in *CONSTRAINED_LAGRANGE_IN_SOLID. For shells, the edge centers replace the face centers. For beams, the x-direction is aligned with the first 2 nodes in *ELEMENT_BEAM and there should be a 3rd node for the y-direction.
Kij	Heat conductivity matrix

VARIABLE	DESCRIPTION
KijTYP	Type of Kij: EQ.0: Kij is a constant EQ.1: Kij is the ID of a *DEFINE_CURVE defining a table of temperature as a function of heat conductivity

*CONTROL_EXPLICIT_THERMAL_SOLVER

Purpose: Define the beam, shell, and solid parts involved in a finite volume thermal analysis. The enthalpies and temperatures are explicitly updated in time.

Card 1	1	2	3	4	5	6	7	8
Variable	PARTSET	DTFAC						
Type	I	F						
Default	none	1.0						

VARIABLE	DESCRIPTION
PARTSET	Part set ID (See *SET_PART)
DTFAC	Time step factor (see Remark 1)

Remarks:

1. **Time Step.** The time step is the minimum of the mechanical and thermal time steps. The thermal time step is the minimum of the element thermal time steps, which are half the enthalpies divided by the right hand side of the heat equation (conductivity × temperature Laplacian). The thermal time step is scaled by DTFAC.

*CONTROL_EXPLOSIVE_SHADOW_{OPTION}

The available options are:

<BLANK>

SET

Purpose: Compute detonation times for explosive elements for which there is no direct line of sight. If this command is not included in the input, the lighting time for an explosive element is computed using the distance from the center of the element to the nearest detonation point, L_d; the detonation velocity, D; and the lighting time for the detonator, t_d:

$$t_L = t_d + \frac{L_d}{D}$$

The detonation velocity for this option is taken from the element whose lighting time is computed and does not account for the possibilities that the detonation wave may travel through other explosives with different detonation velocities or that the line of sight may pass outside of the explosive material.

If this command is present, the lighting time of each explosive element is based on the shortest path through the explosive material from the associated detonation point(s) to the explosive element. If inert obstacles exist within the explosive material, the lighting time will account for the extra time required for the detonation wave to travel around the obstacles. The lighting times also automatically accounts for variations in the detonation velocity if different explosives are used.

The SET option requires input of a set ID of two-dimensional shell elements or three-dimensional solid elements for which explosive shadowing is active. If the SET option is not used, Card 1 should be omitted and shadowing is active for all explosive elements.

See also *INITIAL_DETONATION and *MAT_HIGH_EXPLOSIVE.

Card 1. Card for SET keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	SETID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SETID	Set ID of a *SET_SHELL or *SET_SOLID. If the SET option is active, the lighting times are computed for a set of shells (*SET_SHELL in two dimensions) or solids (*SET_SOLID in three dimensions).

*CONTROL_FORMING

Purpose: Set parameters for metal forming related features.

*CONTROL_FORMING_AUTO_NET

*CONTROL_FORMING_AUTOCHECK

*CONTROL_FORMING_AUTOPOSITION_PARAMETER

*CONTROL_FORMING_BESTFIT

*CONTROL_FORMING_DRAWBEAD_TRANSITION

*CONTROL_FORMING_HOME_GAP

*CONTROL_FORMING_INITIAL_THICKNESS

*CONTROL_FORMING_MAXID

*CONTROL_FORMING_ONESTEP

*CONTROL_FORMING_OUTPUT

*CONTROL_FORMING_PARAMETER_READ

*CONTROL_FORMING_POSITION

*CONTROL_FORMING_PRE_BENDING

*CONTROL_FORMING_PROJECTION

*CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS

*CONTROL_FORMING_SCALE

*CONTROL_FORMING_SCRAP_FALL

*CONTROL_FORMING_SHELL_TO_TSHELL

*CONTROL_FORMING_STONING

*CONTROL_FORMING_STRAIN_RATIO_SMOOTH

*CONTROL_FORMING_TEMPLATE

*CONTROL_FORMING_TIPPING

*CONTROL_FORMING_TRAVEL

*CONTROL_FORMING_TRIM_MERGE

*CONTROL_FORMING_TRIM_SOLID_REFINEMENT

*CONTROL_FORMING_TRIMMING

*CONTROL_FORMING_UNFLANGING

*CONTROL_FORMING_USER

*CONTROL_FORMING_AUTO_NET

Purpose: Automatically generate rectangular nets with specified dimensions and positions for use with springback. This keyword is used for simulating springback when the stamping panel is resting on the nets of a checking fixture.

Card Sets. Add to the deck as many pairs of Cards 1 and 2 as needed. This section is terminated by the next keyword ("*") card. In general, for N nets add 2N cards.

Card 1	1	2	3	4	5	6	7	8
Variable	IDNET	ITYPE	IDV	IDP	X	Y	Z	
Type	I		I	I	F	F	F	
Default	none		0	0	0.0	0.0	0.0	

Card 2	1	2	3	4	5	6	7	8
Variable	SX	SY	OFFSET					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
IDNET	ID of the net; must be unique. See Remark 1.
ITYPE	Not used at this time.
IDV	Vector ID for surface normal of the net. See *DEFINE_VECTOR. If not defined, the normal vector will default to the global z-axis.
IDP	Part ID of the panel undergoing springback simulation
X	The x-coordinate of a reference point for the net to be generated
Y	The y-coordinate of a reference point for the net to be generated
Z	The z-coordinate of a reference point for the net to be generated

VARIABLE	DESCRIPTION
SX	Length of the net along the first tangential direction. (The *x*-axis when the normal is aligned along the global *z*-axis).
SY	Length of the net along the second tangential direction. (The *y*-axis when the normal is aligned along the global *z*-axis).
OFFSET	The net center will be offset a distance of OFFSET in the direction of its surface normal. For positive values, the offset is parallel to the normal; for negative values, antiparallel.

Remarks:

1. **Net ID.** The IDNET field of card 1 sets the "net ID," which is distinct from the part ID of the net; the net ID serves distinguishes *this* net from *other* nets.

2. **Properties of the Net.** The part ID assigned to the net is generated by incrementing the largest part ID value in the model. Other properties such as section, material, and contact interfaces between the panel and nets are likewise automatically generated.

3. **Contact Type.** The auto nets use contact type *CONTACT_FORMING_ONE_-WAY_SURFACE_TO_SURFACE.

Examples:

The partial input file (see below) specifies four auto nets having IDs 1 through 4. The vector with ID = 89 is normal to the net. The nets are offset 4 mm *below* their reference points; the direction is *below* because the normal vector (ID = 89) is parallel to the *z*-axis and the offset is negative. This example input can be readily adapted to a typical gravity-loaded springback simulation obviating the need for SPC constraints (see *CON-STRAINED_COORDINATE).

```
*CONTROL_FORMING_AUTO_NET
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$    IDNET     ITYPE       IDV       IDP         X         Y         Z
         1                   89         5   2209.82  -33.6332   1782.48
$       SX        SY    OFFSET
      15.0      15.0      -4.0
$    IDNET     ITYPE       IDV       IDP         X         Y         Z
         2                   89         5   3060.23  -33.6335   1782.48
$       SX        SY    OFFSET
      15.0      15.0      -4.0
$    IDNET     ITYPE       IDV       IDP         X         Y         Z
         3                   89         5   3061.21   31.4167   1784.87
$       SX        SY    OFFSET
      15.0      15.0      -4.0
$    IDNET     ITYPE       IDV       IDP         X         Y         Z
         4                   89         5   2208.84   31.4114   1784.87
```

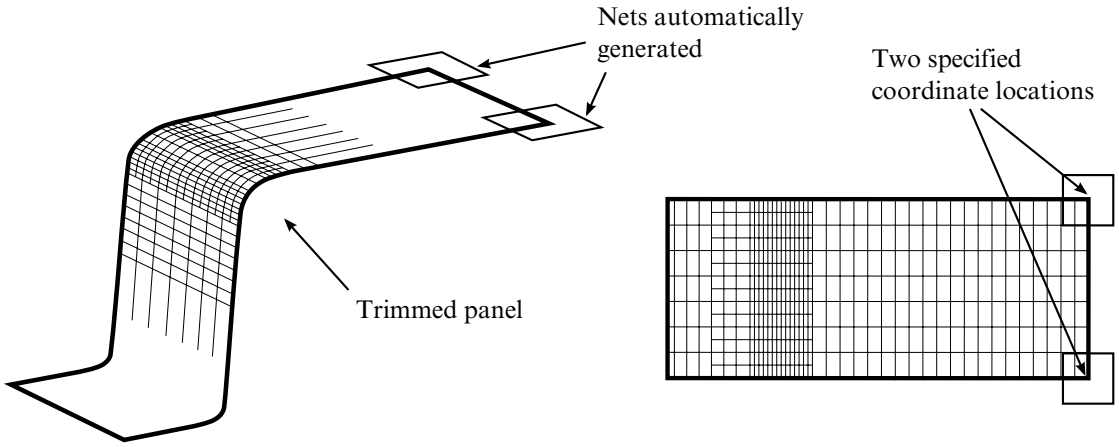


Figure 12-16. An example problem.

```
$       SX         SY     OFFSET
      15.0       15.0       -4.0
*DEFINE_VECTOR
$ VID, Tail X, Y, Z, Head X, Y, Z
89,0.0,0.0,0.0,0.0,0.0,100.0
```

Discussion of Figures:

Figure 12-16 shows a formed and trimmed panel of a hat-shaped channel with an auto net at two corners. The nets are offset 4 mm away from the panel. When gravity loading is downward the nets must be below the panel (Figure 12-17 left) so that the panel comes into contact with the nets after springback as expected (Figure 12-17 right). As shown in Figure 12-18 the situation must be reversed when gravity loading points upward.

Revision Information:

This feature is now available starting in implicit static in double precision LS-DYNA Revision 62781.

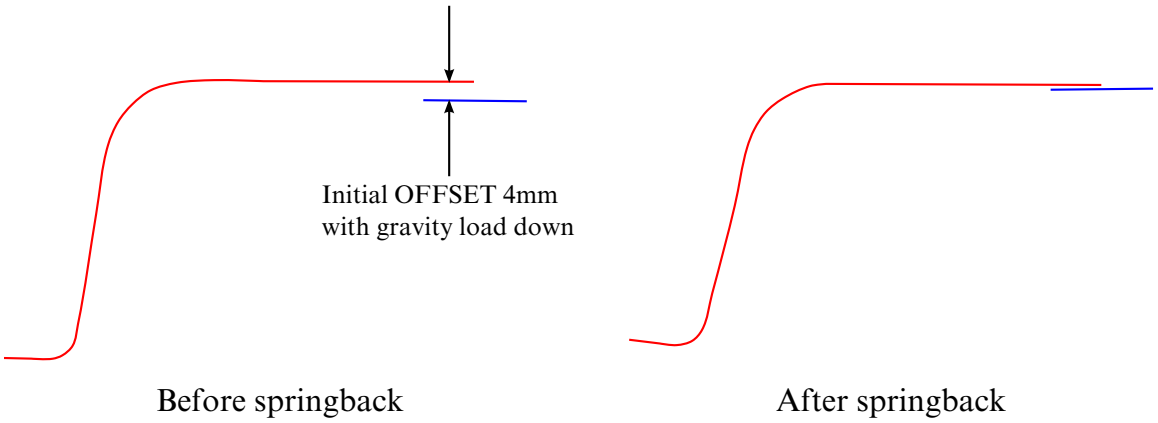


Figure 12-17. Springback and contact with nets - gravity down.

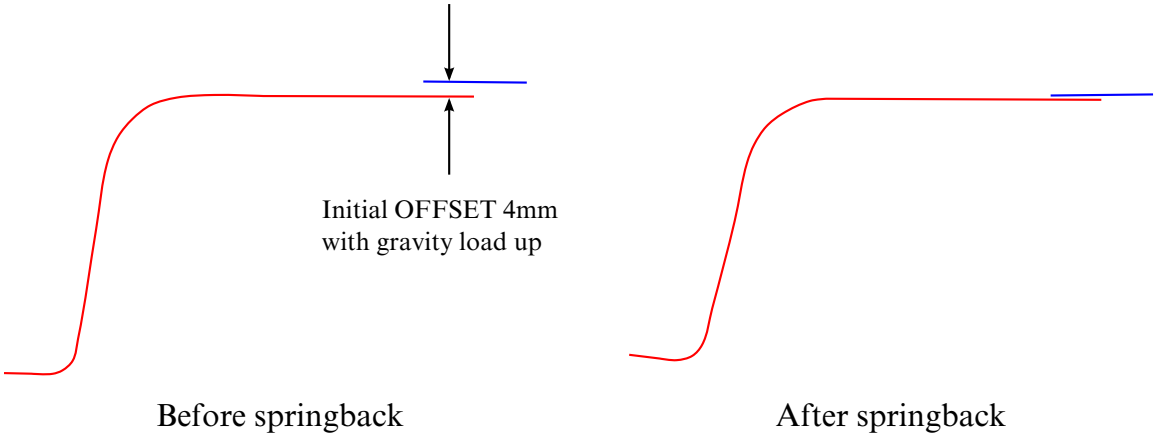


Figure 12-18. Springback and contact with nets – gravity up.

*CONTROL_FORMING_AUTOCHECK

Purpose: This keyword detects and corrects flaws in the mesh for the *rigid body* that models the tooling. Among its diagnostics are checks for duplicated elements, overlapping elements, skinny/long elements, degenerated elements, disconnected elements, and inconsistent element normal vectors.

This feature also automatically orients each tool's element normal vectors so that they face the blank. Additionally, an offset can be specified to create another tool (tool physical offset) based on the corrected tool meshes. Note that this keyword is distinct from *CONTROL_CHECK_SHELL, which checks and corrects mesh quality problems after trimming, to prepare the trimmed mesh for the next stamping process. This keyword only applies to shell elements.

The tool offset feature is now available in LS-PrePost 4.2 under Application → Metal-Forming → Easy Setup. The offset from Die button under Binder can be used to create offset tools.

Card 1	1	2	3	4	5	6	7	8
Variable	ICHECK	IGD	IOFFSET	IOUTPUT	IFSHARP			
Type	I	I	I	I	I			
Default	0	blank	0	blank	0			

VARIABLE	DESCRIPTION
ICHECK	Tool mesh checking/correcting flag: EQ.0: Do not activate mesh checking/correcting feature. EQ.1: Activate comprehensive mesh check and correct problematic tool meshes (see Figure 12-20). This option reduces the likelihood of unreasonable forming results and/or error termination. This is only for regular forming simulations. The calculation will continue after the tool mesh checking/correcting phase is completed. See Example 1. The corrected tool meshes can be viewed and recovered from the resulting d3plot files. If the termination time is set to "0.0" or the keyword *CONTROL_TERMINATION is absent all together, the simulation will terminate as soon

VARIABLE	DESCRIPTION
	as checking/correcting is completed, and corrected tool meshes can be extracted from the d3plot files.
IGD	Not used.
IOFFSET	Tool mesh offset flag. This variable works only when IOUTPUT is defined, and ICHECK is set to "1": EQ.0: Do not offset rigid tool mesh. The sheet blank does not need to be present. In this case the output files rigid_offset.inc and rigid_offset_before.inc will be identical. See Example 2. EQ.1: Perform rigid tool mesh offset using the variable SBST (see Figure 12-19) as specified on a *CONTACT_FORMING_… card. The blank must be defined and positioned completely above or below the rigid tool to be offset. Both part ID and part SID (SURFBTYP) can be used in defining SURFB. IOUTPUT must also be defined.
IOUTPUT	Output option flag: EQ.1: Output offset rigid tool meshes into a keyword file rigid_offset.inc, and terminate the simulation. EQ.2: Output offset rigid tool meshes as well as nodes used to define draw beads into a keyword file rigid_offset.inc, and terminate the simulation. See Example 4. EQ.3: Output checked/corrected tool as well as offset rigid tool meshes into two separate keyword files, rigid_offset_before.inc, and rigid_offset.inc, respectively, and terminate the simulation. See Example 3. EQ.4: Output checked/corrected tool meshes, offset rigid tool meshes as well as the nodes used to define draw beads into two separate keyword files, rigid_offset_before.inc, and rigid_offset.inc, respectively, and terminate the simulation. If not defined and ICHECK = 1, corrected and reoriented tool meshes can be viewed and recovered from d3plot files. See Example 1.
IFSHARP	Sharp edge checking option for those tool area without fillet radii: EQ.0: Check any sharp edges and delete the elements.

VARIABLE	DESCRIPTION
	EQ.1: Skip checking sharp edges.

Remarks:

In sheet metal forming, tools are typically modelled as rigid bodies, and their meshes are prepared from CAD (IGES or STEP) files according to the following procedure:

1. The user imports the CAD data into a preprocessor, such as LS-PrePost.

2. The preprocessor automatically generates a mesh. LS-PrePost features a streamlined GUI for this application.

3. Export the generated mesh to LS-DYNA input files. The LS-PrePost eZ-Setup user interface provides quick access to generate the necessary input files for metal forming applications.

Ideally, this process should produce a good mesh requiring no manual intervention. Often, though, such meshes that have been automatically generated from CAD data have flaws severe enough to prevent an accurate or complete calculation. This feature, *CONTROL_FORMING_AUTOCHECK, is intended to make LS-DYNA more robust with respect to tooling mesh quality.

This keyword *requires* that the tooling meshes represent rigid bodies. Also, when this keyword is used, a part ID or a part set ID, corresponding to SURFBTYP = 2 or 3 on the *CONTACT_FORMING_… card, may be used to define the SURFB side. Segment set ID input, SURFBTYP = 0, is not supported.

Some cases of incoming bad tooling meshes which can be corrected by this keyword are shown in Figure 12-20. This keyword can be inserted anywhere in the input deck. To include the corrected tooling mesh into the d3plot the ICHECK field must be defined. The corrected mesh is written to rigid_offset_before.inc file if IOFFSET and IOUTPUT are defined.

When IOFFSET = 1 and IOUTPUT is defined, the tool meshes will first be checked, corrected, and reoriented correctly towards the blank. Then the tool is offset by an amount of 0.5|SBST| either on the same or opposite side of the blank, depending on the signs of the SBST field on the *CONTACT_FORMING_… card (Figure 12-19). A new keyword file, "rigid_offset.inc" file, will be output as containing the corrected, reoriented, and offset tooling mesh.

Example 1 - Mesh checking/correction in a regular forming simulation:

The keyword can be inserted anywhere in a regular forming simulation input deck. A partial input example of checking, correcting the tool meshes and reorienting all tools' normals is provided below. Note that although SBST is defined between blank and die contact interface, die meshes will not be offset, since IOFFSET is not defined. Simulation will continue if "&endtime" is not zero but will terminate as soon as the checking and correcting are done if "&endtime" is set to "0.0", or *CONTROL_TERMINATION is absent all together. Corrected and reoriented tool meshes can be viewed and recovered from d3plot files.

```
*KEYWORD
*INCLUDE
Tool_blank.k
*CONTROL_FORMING_AUTOCHECK
$   ICHECK       IGD   IOFFSET   IOUTPUT
         1
*CONTROL_TERMINATION
&endtime
⋮
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         1  blank to punch
         1         2         2         2                             1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.0000E+00 0.100E+21
     0.000     0.000                 0.0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         2  blank to die
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
         1         3         2         2                             1         1
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.000E+00 0.100E+21
$     SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
     0.000     0.000               -1.600
$     SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         3  blank to binder
         1         4         2         2                             1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.000E+00 0.100E+21
     0.000     0.000                 0.0
⋮
*END
```

Example 2 - Mesh checking/correction only for rigid tool mesh (sheet blank not required):

A much shorter but complete input example of checking, correcting the tool meshes and reorienting all tools' normal vectors is shown below. Note the sheet blank does not need to be present, and both rigid_offset.inc and rigid_offset_before.inc will be the same, representing the checked, corrected, and reoriented tool mesh file, since IOFFSET is undefined (no tool offset will be done).

```
*KEYWORD
*INCLUDE
toolmesh.k
*CONTROL_FORMING_AUTOCHECK
```

```
$   ICHECK       IGD   IOFFSET   IOUTPUT
         1                             1
*PARAMETER_EXPRESSION
I toolpid          3
*PART

$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
  &toolpid         2         2
*MAT_RIGID
$      MID        RO         E        PR         N    COUPLE         M     ALIAS
         2  7.83E-09  2.07E+05      0.28
$      CMO      CON1      CON2
         1         4         7
$LCO or A1        A2        A3        V1        V2        V3

*SECTION_SHELL
$    SECID    ELFORM      SHRF       NIP     PROPT   QR/IRID     ICOMP     SETYP
         2         2       1.0       3.0       0.0
$       T1        T2        T3        T4      NLOC
       1.0       1.0       1.0       1.0
*END
```

Example 3 - Mesh checking/correction and tool offset (sheet blank required):

In addition to checking, correcting, and reorienting all tools' normals, the following partial input will offset the die meshes in toolmesh.k by 0.88 mm (using the SBST value defined for the die) on the opposite side of the blank, and output the offset tool meshes in the file rigid_offset.inc. The checked/corrected original die meshes will be written to rigid_offset_before.inc. The simulation will terminate as soon as the files are written, regardless of what the "&endtime" value is. In fact, the keyword *CONTROL_TERMINATION can be omitted all together.

```
*KEYWORD
*INCLUDE
Tool_blank.k
*PARAMETER_EXPRESSION
R blankt          0.8
R offset         -1.1
R sbst           blankt*offset*2.0
*CONTROL_FORMING_AUTOCHECK
$   ICHECK       IGD   IOFFSET   IOUTOUT
         1                   1         3
*CONTROL_TERMINATION
&endtime
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         1  blank to punch
         1         2         2         2                             1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.0000E+00 0.100E+21
     0.000     0.000                 0.0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         2  blank to die
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
         1         3         2         2                             1         1
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.000E+00 0.100E+21
$     SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
     0.000     0.000                &sbst
$     SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
```

```
         3  blank to binder
         1         4         2         2                                 1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02           0.000E+00 0.100E+21
     0.000     0.000                 0.0
*END
```

Example 4 - Mesh checking/correction and tool offset, bead nodes output (sheet blank required):

In addition to checking, correcting, and reorienting all tools' normals, the following partial input will create an offset tool in the file rigid_offset.inc on the same side of the blank; the file will also contain the nodes used to define the contact draw beads #1 and #2.

```
*KEYWORD
*INCLUDE
Tool_blank.k
R blankt          0.8
R offset          1.1
R sbst           blankt*offset*2.0
*CONTROL_FORMING_AUTOCHECK
$   ICHECK       IGD   IOFFSET   IOUTPUT
         1                     1         2
*CONTROL_TERMINATION
&endtime
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         1  blank to punch
         1         2         2         2                                 1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.0000E+00 0.100E+21
     0.000     0.000                 0.0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         2  blank to die
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
         1         3         2         2                               1         1
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02          0.000E+00 0.100E+21
$     SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
     0.000     0.000               &sbst
$     SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
         3  blank to binder
         1         4         2         2                                 1         1
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+02           0.000E+00 0.100E+21
     0.000     0.000                 0.0
⋮
*CONTACT_DRAWBEAD_ID
     10001  Draw bead #1
         1         1         4         2                   0         0         0
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+00              0.0615 0.100E+21
     0.200     0.200
$   LCIDRF    LCIDNF     DBDTH     DFSCL    NUMINT
        10         9 0.100E+02 0.700E+00
*CONSTRAINED_EXTRA_NODES_SET
        40         1
*SET_NODE_LIST
         1
    915110    915111    915112    915113    915114    915115    915116    915117
    915118    915119    915120    915121    915122
*CONTACT_DRAWBEAD_ID
     10002  Draw bead #2
         2         1         4         2                   0         0         0
 0.110E+00 0.000E+00 0.000E+00 0.000E+00 0.200E+00              0.0615 0.100E+21
```

```
     0.200     0.200
$   LCIDRF    LCIDNF     DBDTH     DFSCL    NUMINT
        11         9 0.100E+02 0.400E+00
*CONSTRAINED_EXTRA_NODES_SET
$       PID      NSID
         40         2
*SET_NODE_LIST
         2
    915123    915124    915125    915126    915127    915128    915129    915130
    915131    915132    915133    915134    915135    915136    915137
⋮
*END
```

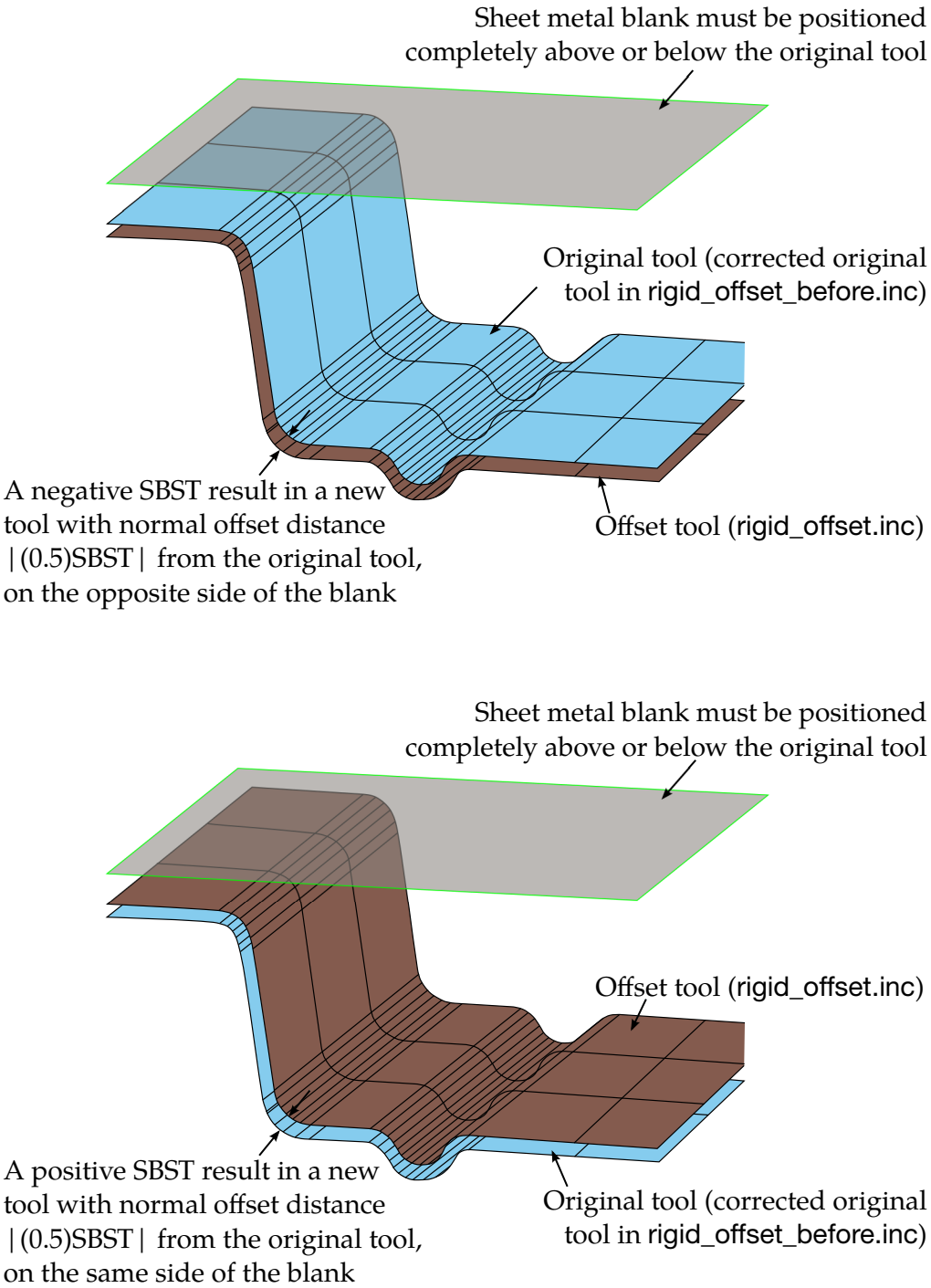
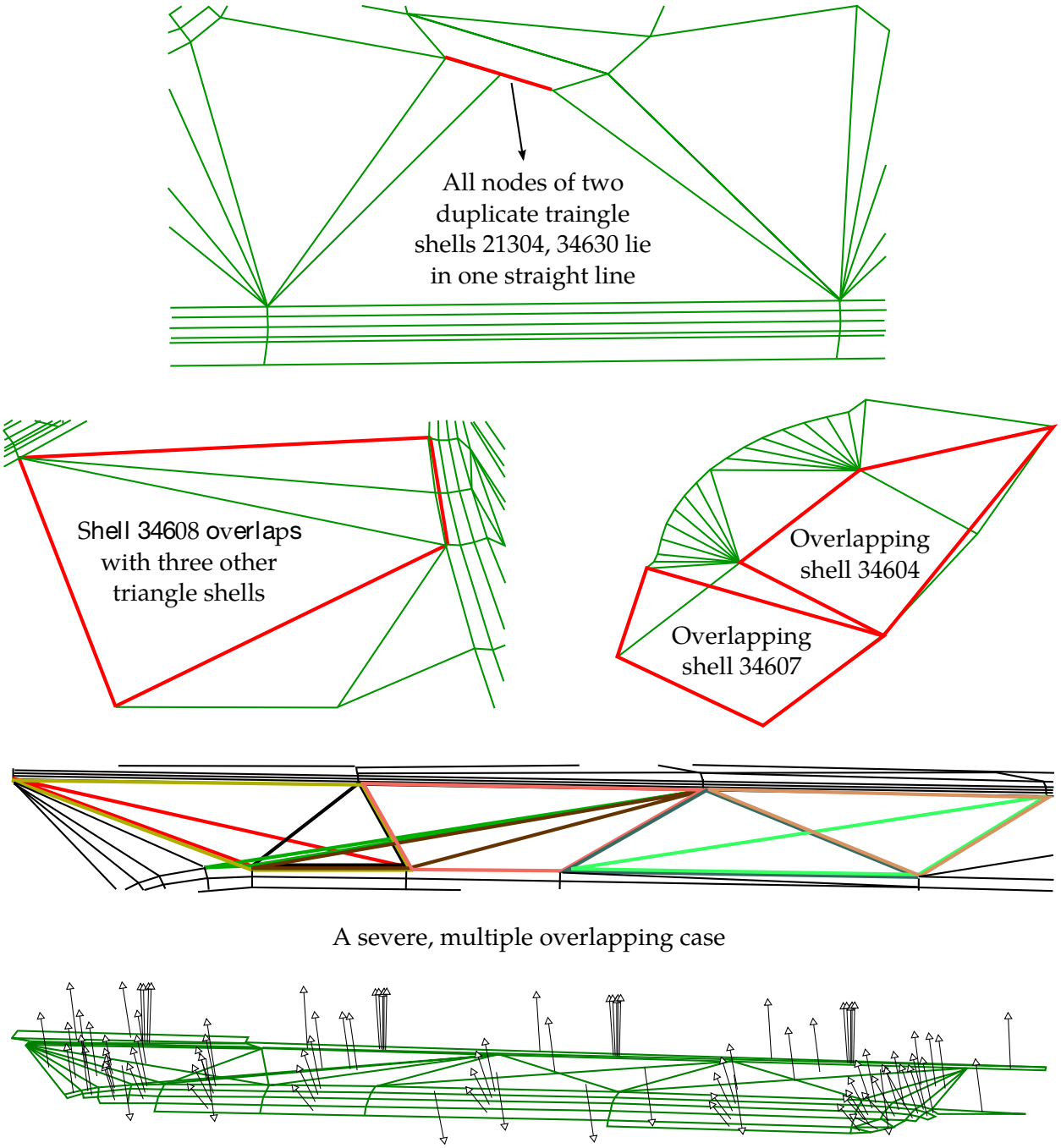


Figure 12-19. Offset using the SBST value defined in *CONTACT_FORMING_…



A severe, multiple overlapping case

A case of typical incoming, inconsistent shell normals

Figure 12-20. A few cases of the tooling mesh problems handled by this keyword.

*CONTROL_FORMING_AUTOPOSITION_PARAMETER_{OPTION1}_{OPTION2}

Available options for *OPTION1* include:

<BLANK>

SET

With SET, a set of parts will be repositioned together.

Available options for *OPTION2* include:

<BLANK>

POSITIVE

When *OPTION2* is set to POSITIVE, the calculated distance to reposition will always be a positive value.

Purpose: *Calculate* the minimum required separation distances among forming tools for initial tool and blank positioning in metal forming simulation. It is applicable to sheet blanks with shell and solid elements. It does not, actually, move the part; for that, see *PART_MOVE.

> **NOTE:** This keyword requires that the parts start in their home positions. While processing this card, LS-DYNA moves the parts to match the auto-position results so that auto-position operations are correctly composed. Upon completion of the auto-positioning phase, the parts are returned to their home positions.

Auto-Position Part Cards. Add one card for each part to be auto-positioned. The next keyword ("*") card terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	CID	DIR	MPID	POSITION	PREMOVE	THICK	PORDER
Type	I	I	I	I	I	F	F	I/A
Default	none	global	none	none	0	0.0	0.0	none

VARIABLE	DESCRIPTION
PID	Part ID. This part will be moved based on the following controlling parameters. When the option SET is activated, PID becomes a part set ID, defined by *SET_PART_LIST. This is useful in defining tailor-welded blanks, where two pieces of the blank must be moved simultaneously.
CID	Coordinate ID set with *DEFINE_COORDINATE_SYSTEM or *DEFINE_COORDINATE_VECTOR. The default is the global coordinate system. LT.0: |CID| is vector ID giving the direction the part will be moved.
DIR	Direction in which the part will be moved: EQ.1: x-direction, EQ.2: y-direction, EQ.3: z-direction.
MPID	Master part ID, whose position is to be referenced by PID for positioning. When the option SET is activated, MPID becomes part set ID, defined by *SET_PART_LIST.
POSITION	Definition of relative position between PID and MPID: EQ.1: PID is above MPID; EQ.-1: PID is below MPID. The definition of "above" is determined by the defined coordinate system or vector. If PID is above MPID, it means PID has a larger z-coordinate. In the case of CID < 0, the local z-direction is taken as the direction defined with |CID|. This definition is helpful in line-die simulations where the local coordinate system may be used.
PREMOVE	Move PID through distance PREMOVE *prior* to processing the other *CONTROL_FORMING_AUTOPOSITION cards. See Remark 4.
THICK	One half of this distance value (THICK) will be used to separate the tools and the blank. The same value must be used in *all* defined move operations under this keyword. The calculation error of the separation distance is one half of THICK.

VARIABLE	DESCRIPTION
PORDER	The name of the parameter without the ampersand "&," as defined in *PARAMETER, or the position or order of the parameter defined in the *PARAMETER list.

Background:

In a line-die (multi-stage) simulation, initial positioning of the tools and blank is one of the major issues preventing several die processes from being run automatically from a single job submission. The most basic method for running a line-die simulation is to chain a series of calculations together using the previous calculation's partially formed blank, written to a dynain file, as a part of the input for the next calculation.

Since the partial results are unknown until the preceding calculation completes, the tools need to be repositioned before the next calculation. Without this card the repositioning step must be done by hand using a preprocessor. With the combination of this card and the LS-DYNA case driver (see *CASE card) or a continuous run script (works for both Linux and Windows), the repositioning can be fully automated, enabling a complete line-die simulation to be performed with a single job submission.

Workflow:

This card requires that all parts start in their home (tool closed) position. It calculates how far the parts need to be moved to prevent initial penetration. The results are stored into the parameter listed in the PORDER field to be used for a part move operation.

1. For each defined move operation, a *PARAMETER card *must* initialize the parameter referred to in the PORDER field.

2. All tools must start in the home position, including the *desired final gaps*.

3. The required distance between each contact pair is calculated and stored in the initialized parameter named in the PORDER field. Starting with Revision 124103, if the separation distance cannot be found, such as when MPID is not found or is out of position, or DIR is not input correctly, the value of PREMOVE will be returned instead of a very large number.

4. The parts are repositioned through a distance based on the value written to the parameter PORDER using the *PART_MOVE card.

5. The *PARAMETER_EXPRESSION can be used to evaluate expressions depending on the move distances, such as times and tool move speeds.

6. The *CASE feature can be used to chain together the sub-processes in a line die (process chain) simulation. Alternatively, a continuous run script (for both Linux and Windows environments) can be generated for an entire process chain simulation using the Metal Forming GUI starting in LS-PrePost 4.3.

Remarks:

1. **Order dependence.** Input associated with this keyword is order sensitive. The following order should be observed:

   a) All model information, *including* all elements and nodes

   b) Part definitions (see *PART)

   c) Part set definitions (see *SET_PART_LIST)

   d) *PARAMETER initialization

   e) This keyword

   f) *PARAMETER_EXPRESSION

   g) *PART_MOVE

2. **New keyword input with positioned model.** This keyword can also be used to generate a new keyword input (dynain) containing the fully positioned model (without actually running the entire simulation). This procedure is identical to a full calculation except that the *PARAMETER_EXPRESSION keyword, the *CONTROL_TERMINATION keyword, and tool kinematic definitions are omitted.

3. **Local coordinate systems and computed parameters.** When working in local coordinate systems, the sign of the computed parameter may not necessarily correspond to its intended use. In this case, the absolute value function, ABS, for the *PARAMETER_EXPRESSION keyword is especially useful.

4. **PREMOVE.** Cards with the PREMOVE field set are processed before *all* other *CONTROL_FORMING_AUTOPOSITION_... cards, regardless of their location in the input deck. The PREMOVE field serves to modify the initial state on which the calculations of the other AUTOPOSITION cards are based.

   For instance, when a binder is moved downward with the PREMOVE feature, it will be in its post-PREMOVE position for *all other* AUTOPOSITION calculations. But, as is the case with the other AUTOPOSITION cards, the model will be returned to its home position upon completion of the AUTOPOSITION phase. Note that the master part, MPID, and the POSITION fields are *ignored* when the

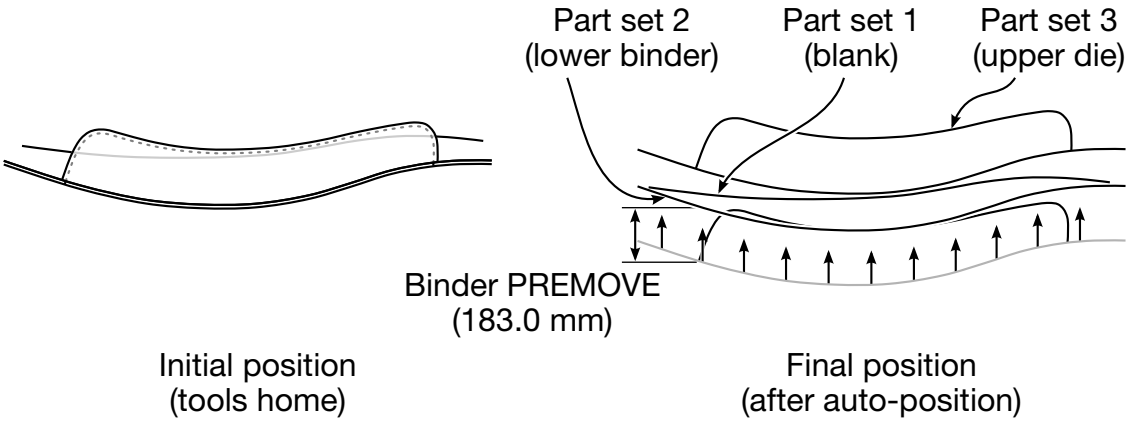


Figure 12-21. An example of using the variable PREMOVE

PREMOVE field is set, and that the PREMOVE value is copied into the PORDER parameter.

5. **Calculated move distance.** Calculated move distance for each part will be displayed in the beginning of the messag file. All parameters (defined and calculated) can be found in the d3hsp file.

Example 1:

An air draw process like the one shown in Figure 12-21 provides a clear illustration of how this card, and, in particular, the PREMOVE field is used to specify the lower binder's travel distance.

1. The card with the PREMOVE field set, the *third* AUTOPOSITON card, is processed first. It moves lower binder 183 mm upward from its home position, and it will form the base configuration for other AUTOPOSITION cards. It will also store this move into &bindmv. Note that although the POSITION and MPID fields are set, they are ignored.

2. The *first* autoposition card, which will be the *second* one processed, calculates the minimum offset distance (&blankmv) necessary for the blank (part set 1) to clear part set 9999, which consists of the lower binder (PID = 2), which is in its post-PREMOVE location, and of the lower punch (PID = &lpunid).

3. The next card determines the minimum offset (&updiemv) necessary to bring the upper die (part set 3) as close to the blank as possible without penetrating. *This calculation proceeds under the assumption that the blank part set has been moved through &blankmv.*

```
*SET_PART_LIST
9999
&lpunpid,2
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

```
*CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$      PID       CID       DIR      MPID  POSITION   PREMOVE     THICK    PORDER
$ blank move
         1                   3      9999         1             &bthick   blankmv
$ upper die move
         3                   3         1         1             &bthick   updiemv
$ lower binder move
         2                   3         1        -1     183.0   &bthick    bindmv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PART_MOVE
$    SID             XMOV             YMOV             ZMOV     CID   IFSET
$ blank move
       1              0.0              0.0         &blankmv               1
$ upper die move
       3              0.0              0.0         &updiemv               1
$ lower binder move
       2              0.0              0.0          &bindmv               1
```

Example 2:

The following examples demonstrates the *PARAMETER_EXPRESSION card, which is used to derive new parameters from the value calculated during auto-positioning. In this example, the auto-positioned distance for binder, which is stored in the parameter, &bindmv, is used to define an additional parameter,

$$\&bindmv1 = \&bindmv - 30 \text{ mm}$$

The *PART_MOVE step uses &bindmv1 rather than &bindmv, to move both the lower binder and the draw beads.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$      PID       CID       DIR      MPID  POSITION   PREMOVE     THICK  PORDER
$ blank move
   &blksid                 3      9999         1             &bthick   blankmv
$ upper die move
  &udiesid                 3   &blksid         1             &bthick   updiemv
$ lower binder move
  &bindsid                 3   &blksid        -1             &bthick   bindmv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PARAMETER_EXPRESSION
bindmv1   bindmv-30.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PART_MOVE
$    SID             XMOV             YMOV             ZMOV     CID   IFSET
$ blank move
 &blksid              0.0              0.0         &blankmv               1
$ upper die move
&udiesid              0.0              0.0         &updiemv               1
$ lower binder move
&bindsid              0.0              0.0         &bindmv1               1
$ draw beads move
     909              0.0              0.0         &bindmv1               1
```

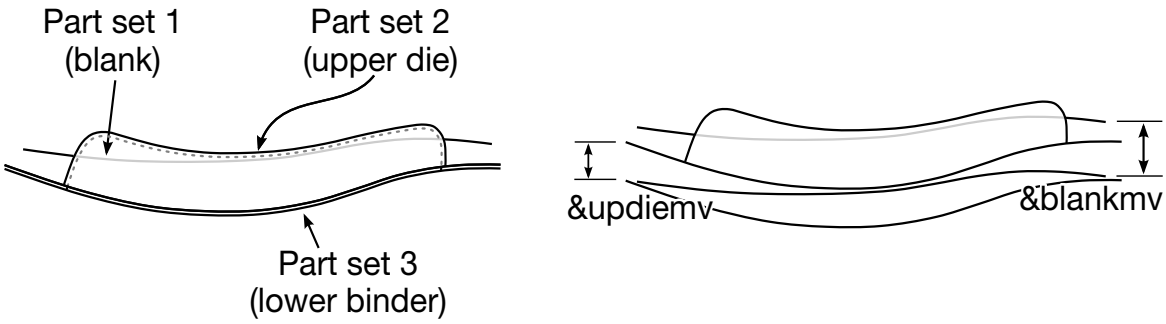


Figure 12-22. An example of binder closing in air draw

Example 3:

Figure 12-22 schematically shows the binder closing in the global *Z*-direction. A partial keyword details follow.

```
*INCLUDE
$blank from previous case
case5.dynain
*INCLUDE
closing_tool.k
*INCLUDE
beads_home.k
*SET_PART_LIST
$ blank
1
1
*SET_PART_LIST
$ upper die
2
2
*SET_PART_LIST
$ lower binder
3
3
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*parameter
$$$$$$$$$$$$$$$$$$$$$$$$$ Tool move variables
R blankmv        0.0
R updiemv        0.0
R bindmv         0.0
$$$$$$$$$$$$$$$$$$$$$$$$$ Tool speed and ramp up definition
R tclsup       0.001
R vcls        1000.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_AUTOPOSION_PARAMETER_SET
$      PID       CID       DIR      MPID  POSITION    PREMOVE     THICK  PORDER
$ positioning blank on top of lower binder
         1                   3         3         1                 0.7   blankmv
$ positioning upper die on top of blank
         2                   3         1         1                 0.7   updiemv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PARAMETER_EXPRESSION
$    PRMR1    EXPRESSION
R clstime     (abs(updiemv)-vcls*tclsup)/vcls+2.0*tclsup
R endtime     &clstime
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PART_MOVE
$    PID            XMOV            YMOV            ZMOV     CID
```

```
       1             0.0             0.0        &blankmv
       2             0.0             0.0        &updiemv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_TERMINATION
&endtime
```

*CONTROL_FORMING_BESTFIT_{OPTION}

Available options include:

<BLANK>

VECTOR

Purpose: Rigidly move a part to the target using an iterative least-squares method so that they maximally coincide. This feature can be used to assess the accuracy of the spring-back prediction in sheet metal forming by translating and rotating a sprung part (source) to a scanned part (target). This keyword applies to shell elements only. The VECTOR option causes vector components of the normal distance from the target to the part node to be included in the output file, bestfit.out, under the keyword *NODE_TO_TARGET_-VECTOR.

This feature is available as of LS-PrePost 4.5 in *Metal Forming Application/eZ Setup*.

Card 1	1	2	3	4	5	6	7	8
Variable	IFIT	NSKIP	GAPONLY	IFAST	IFSET	NSETS	NSETT	
Type	I	I	I	I	I	I	I	
Default	0	-3	0	1	0	none	none	

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
IFIT	Best fit program activation flag: EQ.0: Do not perform best-fit. EQ.1: Activate the best-fit feature

VARIABLE	DESCRIPTION
NSKIP	Optional skipping scheme during bucket searching to aid the computational speed (zero is no skipping): GT.0: Number of nodes to skip in bucket searching. NSKIP set to 1 does not skip any nodes in searching, resulting in the slowest computing speed but the highest accuracy. Larger values of NSKIP speed up the calculation time with slightly deteriorating accuracies. Based on studies, a value of 5 is recommended with IFAST = 1, which balances the speed and accuracy. LT.0: Absolute value is the distance to skip in bucket searching. This scheme is faster compared to the previous method and therefore is recommended for computational efficiency and accuracy. A value of -5 is suggested.
GAPONLY	Separation distance calculation flag: EQ.0: After performing best-fit, calculate the separation between the two parts. EQ.1: No best fit, just calculate separation between the two existing parts. EQ.2: You are responsible for moving the parts closer together in both distance and orientation for the situation where the target and source are not similar in shape. Also see NSETS and NSETT (recommended method).
IFAST	Computing performance optimization flag: EQ.0: No computing speed optimization. EQ.1: Activate computing speed optimization (default and recommended).
IFSET	Optional flag to define a node set to be included or excluded in the source mesh file for best fitting. The node set can be defined in a file together with the source mesh. EQ.0: All nodes in the source mesh file will be best fitted. GT.0: The input value is a node set ID; only the nodes in the set will be best fitted. LT.0: The absolute value is a node set ID; all nodes excluding those in the set will be best fitted.

VARIABLE	DESCRIPTION
NSETS	An optional node set ID of three nodes from the source mesh to help align the source to the target. See Remark 4. The nodes should be selected based on distinctive geometry features, such as, the center of an arc, the center of a dart, or the end node of a take-up bead (see Figure 12-23). The three nodes must not be aligned in one straight line. Define NSETS if the orientations of the source mesh and the target have a large deviation (> ~30 degrees in any direction). This method is recommended.
NSETT	An optional node set ID of three nodes from the target mesh that consists of nodes corresponding to the same geometry features as the source mesh to help align the source to the target. The three nodes should be input in the same order as those from the source mesh. Approximate locations are acceptable. See Remark 4. Define NSETT only if NSETS is defined. See Figure 12-23 for details. This method is recommended.
FILENAME	Target mesh file in keyword format, in which only *NODE and *ELEMENT_SHELL should be included. The target mesh is typically the scanned part.

Remarks:

1. **Source mesh.** The source mesh can be included in the input file using *INCLUDE.

2. **Target mesh coarseness.** To reduce the computing time, the scan file (STL) mesh can be coarsened in a scan-processing software from a typically very dense mesh to a more reasonably sized mesh.

3. **Fit evaluation.** The distances between corresponding portions of the two parts are calculated after they are fitted. The distance is given as a positive or negative value based on the target's normal directions. The distance vector points from the target to the corresponding portion on the source. If the projection of the distance vector onto the target's normal vector at this portion is in the direction of the target's normal direction, then the distance is "positive", otherwise it is "negative." For areas where no corresponding meshes can be found between the two parts, the distances are set to nearly zero. The fitting accuracy is within 0.02 mm.

   The distances are stored as thickness values in bestfit.out. They can be plotted in LS-PrePost using *COMP → Thickness*. By importing both bestfit.out and the

target mesh to LS-PrePost, the deviation on cut-sections can be evaluated using the *SPLANE* feature.

4. **Initial orientation.** If the orientation of the source and target exceeds 30 degrees, the fit process becomes computationally costly. NSETS and NSETT can be used to initially align the source mesh to the target mesh before a full fitting is performed.

5. **Output.** The following information will be output to the results file bestfit.out. The summary gives a percentage of the source nodes that are between 0.0 and 6.0 mm of the target, with a 0.5 mm interval range, along with the maximum deviation value of the source from the target. A transformation matrix as well as the transformation given as a set of euler angles (radians) and a translation vector (in mm) are also output starting in Revision 140609. The transformation matrix can be used directly in the keyword *INCLUDE_STAMPED_PART_MATRIX to perform a validation or for other needs. The euler angle and translation vector form allows you to transform (note it is order sensitive) the source to the target in a more direct method, namely rotate in RZ first, followed by RY and RX, followed by translations in DX, DY, and DZ.

```
$    Summary:
$  between 0.00 to 0.50:   100.000   100.000
$  between 0.50 to 1.00:     0.000   100.000
$  between 1.00 to 1.50:     0.000   100.000
$  between 1.50 to 2.00:     0.000   100.000
$  between 2.00 to 2.50:     0.000   100.000
$  between 2.50 to 3.00:     0.000   100.000
$  between 3.00 to 3.50:     0.000   100.000
$  between 3.50 to 4.00:     0.000   100.000
$  between 4.00 to 4.50:     0.000   100.000
$  between 4.50 to 5.50:     0.000   100.000
$  between 5.00 to 5.50:     0.000   100.000
$  between 5.50 to 6.00:     0.000   100.000
$ The maximum deviation is:     0.004
$*TRANSFORMATION_MATRIX
$    0.649517E+00    0.433010E+00    0.625003E+00   -0.365657E+03
$    0.125007E+00    0.749999E+00   -0.649519E+00   -0.216987E+02
$   -0.750000E+00    0.500003E+00    0.433008E+00    0.252244E+02
$
$ --OR--
$       RZ:   -0.588001E+00
$       Ry:    0.675136E+00
$       RX:    0.982798E+00
$ DX,DY,DZ:   -0.365657E+03   -0.216987E+02    0.252244E+02
```

Example 1 – fitting with NSETS and NSETT (recommended):

In the following partial keyword example (meshes and results shown in Figure 12-23) a source mesh sourcemesh.k is being best-fitted to a target mesh targetmesh.k.

Node sets 1 and 2 are defined from nodes in the source and target meshes, respectively, to help with alignement. Node ID 1001 and 1 are both located at the center of a dart on

the top surface of the hat-shaped part. Node ID 1002 and 2 are selected at the center of an arc of a cutout hole. Lastly, node ID 1003 and 3 are at the center of a tangent line of a radius.

In this example, since the source and target meshes are exactly the same, the normal distance, as displayed by *thickness* is nearly zero everywhere.

```
*CONTROL_FORMING_BESTFIT
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$#    IFIT     NSKIP   GAPONLY     IFAST     IFSET     NSETS     NSETT
         1        -5         0         1         0         1         2
$# FILENAME
targetmesh.k
*INCLUDE
sourcemesh.k
*SET_NODE_LIST
1
1,2,3
*SET_NODE_LIST
2
1001,1002,1003
```

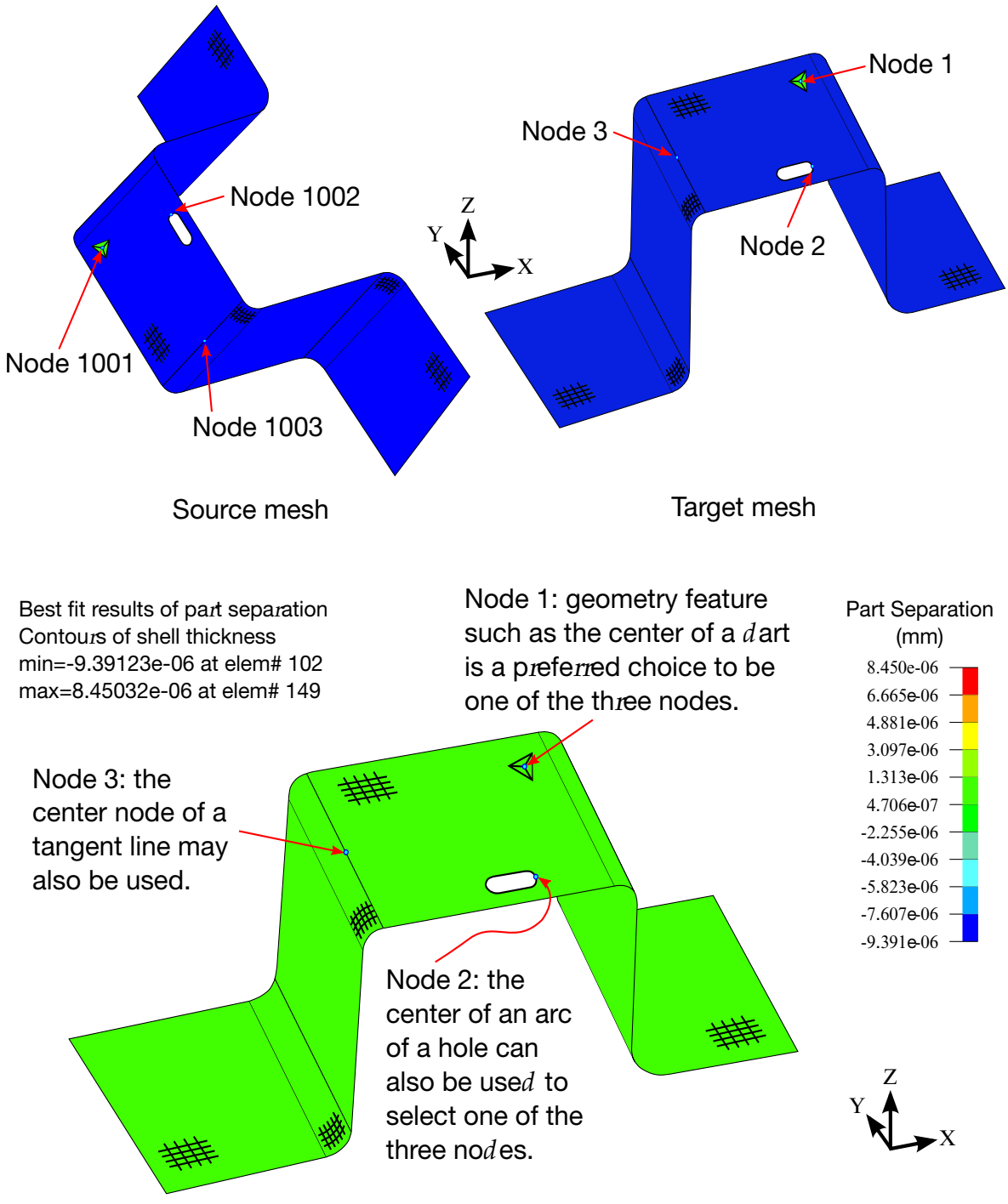


Figure 12-23. Best fit of two meshes with an orientation difference of greater than 30 degrees.

*CONTROL_FORMING_DRAWBEAD_TRANSITION

Purpose: Include a transitional draw bead for obtaining the transitional bead force. In a stamping simulation, a line bead is commonly used. The line bead has a constant bead force in the numerical model. This keyword enables obtaining the transitional bead force. The transitional bead force changes linearly from one end to the other. If the end of the transitional draw bead has a neighboring draw bead, then the force on that end takes the value from its neighboring draw bead. If the end is open, the bead force is zero on that end. See Figures 12-24 and 12-25.

Card 1	1	2	3	4	5	6	7	8
Variable	LENGTH							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
LENGTH	Length of the transitional draw bead

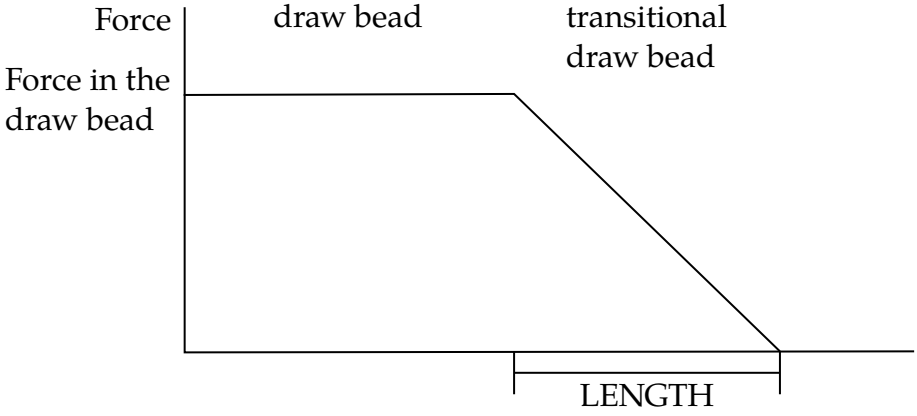


Figure 12-24. Force in a transitional draw bead if one end is open

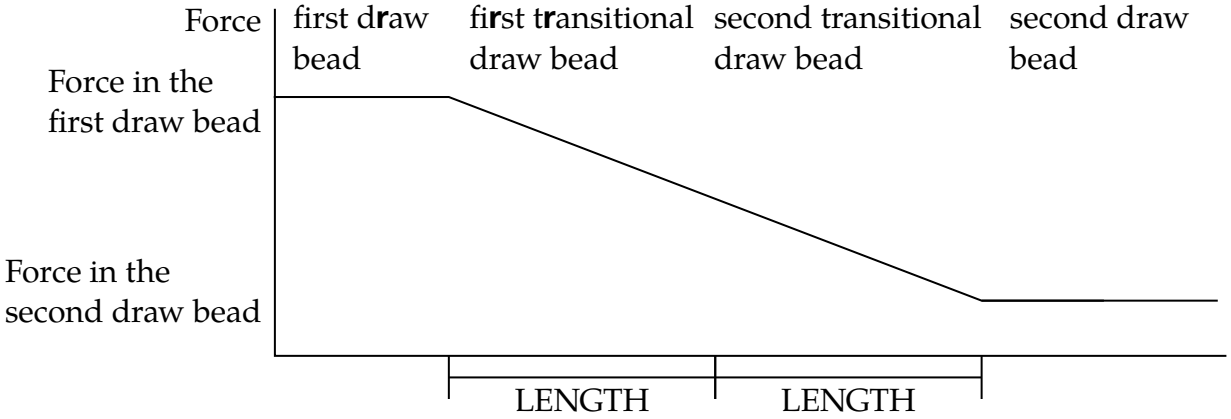


Figure 12-25. Force in transitional drawbeads if both ends have neighbors

*CONTROL_FORMING_HOME_GAP

Purpose: Calculate the minimum gap between the upper and lower tools. Gaps smaller than the blank thickness can lead to problems with the simulation. The gap is initially measured from the home position. Then the tools are incrementally moved to the starting position where the gap is again measured.

Card 1	1	2	3	4	5	6	7	8
Variable	PSIDU	PSIDL	GAP	MVINC	ISTOP			
Type	I	I	F	F	I			
Default	none	none	none	none	0			

VARIABLE	DESCRIPTION
PSIDU	Part set ID of the tools above the blank (upper tools)
PSIDL	Part set ID of the tools below the blank (lower tools)
GAP	Minimum gap allowed between the upper and lower tools
MVINC	Incremental movement of tools from home position to starting position to check the gap
ISTOP	How to proceed if the minimum gap found is less than GAP: EQ.0: Output a warning message. Job continues EQ.1: Terminate the job.

*CONTROL_FORMING_INITIAL_THICKNESS

Purpose: Specify a varying thickness field along a specific direction on a sheet blank (shell elements only) as a result of a metal forming process, such as a tailor-rolling, that will be used for an additional metal forming simulation. A related keyword is *ELEMENT_SHELL_THICKNESS.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCID	X0	Y0	Z0	VX	VY	VZ
Type	I	I	F	F	F/I	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
PID	Part ID of the sheet blank to be defined with varying thickness, as in *PART. Currently only 1 PID is allowed.
LCID	Load curve ID defining thickness (ordinate) as a function of distance (abscissa) starting from position coordinates (X0,Y0,Z0) and in the direction of the vector (VX,VY,VZ). See *DEFINE_CURVE.
X0, Y0, Z0	Starting position coordinates
VX, VY, VZ	Vector components defining the direction of the distance in the load curve

Background:

Tailor-rolling is a process used to vary the thickness of the blank. A judiciously designed and manufactured tailor-rolled blank will reduce the number of parts (reinforcements) involved in the stamping process, as well as the number tools needed to make them. By reducing the number of spot welds, tailor-rolled pieces also possess superior structural integrity.

Remarks:

1. **Thickness Curve.** Beyond the last data point LS-DYNA extrapolates the load curve specified in LCID as being constant.

2. **Overriding Set Thickness.** This card overrides thicknesses set with the *SECTION_SHELL keyword.

Application Example:

A reduced input deck containing a characteristic example of this keyword's application is given below. In this example the blank is part ID 1. The axis of the load curve starts at position $(-295, -607, -43)$ and the direction along which the load curve sets the thickness is given by $(524, 607, 0)$. For each of the load curve's abscissa values, t, the corresponding geometrical coordinate is given by:

$$\boldsymbol{r} = \begin{bmatrix} -295 \\ -607 \\ -43 \end{bmatrix} + \begin{bmatrix} 524 \\ 607 \\ 0 \end{bmatrix} t$$

For negative values along the load curve, $t < 0$, and values of $t > 101.0$, the thickness is extrapolated as a constant value of 0.8, and 0.9, respectively.

```
*CONTROL_FORMING_INITIAL_THICKNESS
$      PID       LCID         X0         Y0         Z0         VX         VY         VZ
         1       1012    -295.0     -607.0     -43.0      524.0      607.0        0.0
*DEFINE_CURVE
1012
0.0, 0.8
21.0, 0.9
43.0, 1.0
65.0, 1.1
82.0, 1.0
101.0, 0.9
```

In Figure 12-26, a sheet blank is defined with a varying thickness across its surface in a vector direction pointed from the start to end point. The thickness variation as a function of the distance from starting point in section A-A is shown in Figure 12-27.

Revision information:

This feature is available in LS-DYNA starting in Revision 82990.

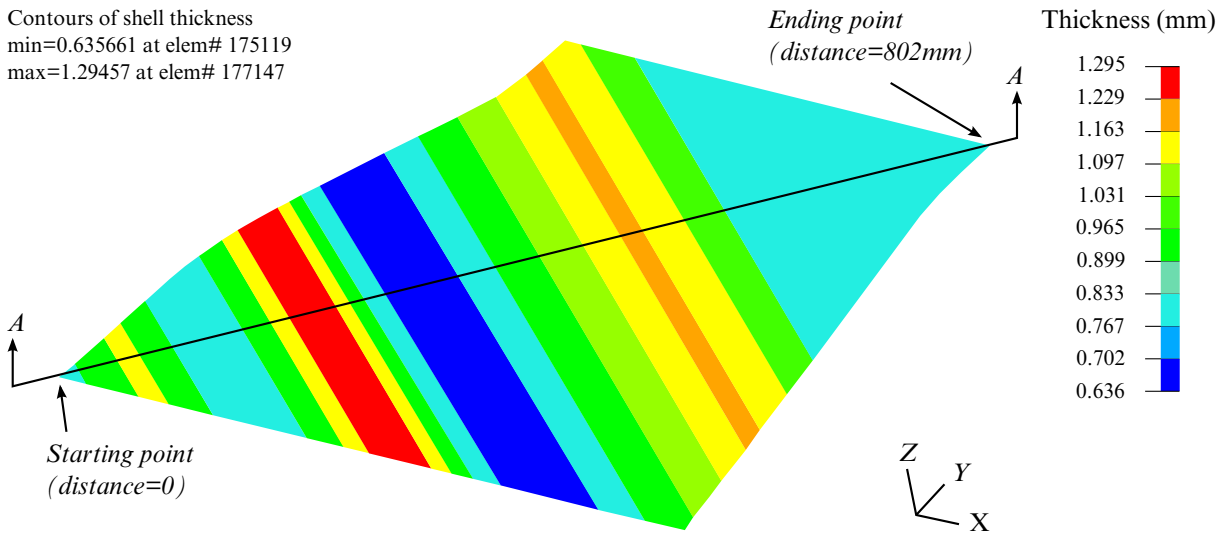


Figure 12-26. Define a varying thickness field across the sheet blank.

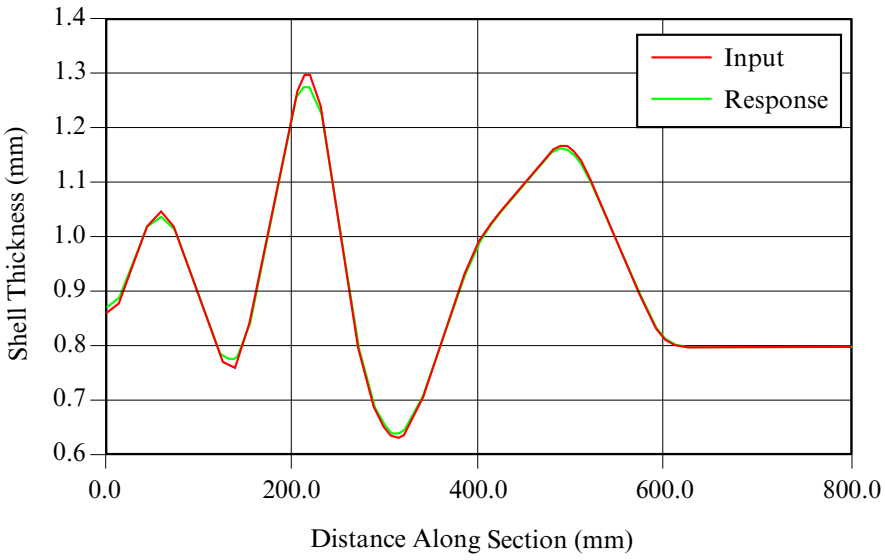


Figure 12-27. Thickness variation across section A-A

*CONTROL_FORMING_MAXID

Purpose: This card sets the node and element ID numbers for an adaptive sheet blank. The new node and element number of the adaptive mesh will start at the values specified on this card, typically greater than the last node and element number of all tools and blanks in the model. This keyword is often used in multi-stage sheet metal forming simulation. The *INCLUDE_AUTO_OFFSET keyword is related.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	MAXIDN	MAXIDE		I2DYNAIN			
Type	I	I	I		I			
Default	none	none	none		0			

VARIABLE	DESCRIPTION
PID	Part ID of the sheet blank, as in *PART.
MAXIDN	Node ID number from which adaptive node ID numbers will be created.
MAXIDE	Element ID number from which adaptive element ID numbers will be created.
I2DYNAIN	Setting I2DYNAIN to 1 will cause this keyword to be output to a dynain file with the updated maximum node and element IDs. This output simplifies post-processing for multi-step processes since it ensures that element and node IDs generated in subsequent steps are larger than those in previous steps. By default, this keyword is not output to a dynain file.

Remarks:

In a multi-stage automatic line die simulation the adaptivity feature may generate node and element IDs that collide with those of the tools used in the later stages of the process. Before the calculation begins, the set of IDs used by the tools is known. By setting MAXIDN to a value greater than the largest tool node ID and MAXIDE to a value greater than the largest tool element ID, it is guaranteed that refinement during the early stages will not lead to conflicts with tool IDs in the later stages.

The following example shows this feature applied in a 2D trimming simulation. Nodes and elements ID numbers generated from an adaptive trim simulation will be larger than the specified ID numbers of 5921980 and 8790292, respectively, for a sheet blank with part ID of 4.

```
*KEYWORD
*INCLUDE_TRIM
sim_trimming.dynain
⋮
*CONTROL_ADAPTIVE_CURVE
$     IDSET     ITYPE         N      SMIN
   &blksid         2         2       0.6
*CONTROL_CHECK_SHELL
$      PSID    IFAUTO    CONVEX      ADPT    ARATIO     ANGLE      SMIN
  &blksid1         1         1         1  0.250000150.000000  0.000000
*INCLUDE
EZtrim.k
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE_TRIM_NEW
$#    tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
     90914         2         0         1  1.250000  1.000000         0         0
sim_trimming_trimline_01.igs
*DEFINE_VECTOR
$#     vid        xt        yt        zt        xh        yh        zh       cid
         1     0.000     0.000     0.000     0.000     0.000  1.000000         0
*CONTROL_FORMING_MAXID
$       pid    maxidn    maxide
         4   5921980   8790292
*END
```

*CONTROL_FORMING_ONESTEP_{OPTION}

Purpose: Activate a one-step solution using the *total strain theory* approximation to plasticity (also known as deformation theory) to implement an inverse method. Given the *final* geometry, the one-step method uses the implicit statics solver to compute an approximate solution for (1) the stresses and strains in the formed part, (2) the thickness of the formed part, and (3) the size of the initial blank (unfolded flat blank). This method is useful for estimating the initial blank size with attendant material costs, and for augmenting crashworthiness models to account for metal-forming effects, such as plastic strains and blank thickness in crash simulations. It is implemented only for SMP double precision.

NOTE: The entire input deck must contain a single *fully connected* surface consisting entirely of shells (nothing more, nothing less). This surface specifies the *final* geometry. Isogeometric shells *are* supported.

The input may contain more than one *PART, provided the parts share common boundary nodes (for instance, to model a tailor-welded blank).

Adaptive meshes are not supported. Use *CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS if the part is a formed blank with an adaptive mesh.

The keyword *DEFINE_FORMING_ONESTEP_PRIMARY can be used to model "patch-welded" blanks.

Keywords associated with *CONTROL_FORMING_ONESTEP are:

*CONTROL_FORMING_UNFLANGING

*INTERFACE_BLANKSIZE_DEVELOPMENT

*DEFINE_FORMING_ONESTEP_PRIMARY

Available options include:

<BLANK>

AUTO_CONSTRAINT

DRAWBEAD

FRICTION

TRIA

QUAD

QUAD2 (default)

ORTHO

Summary of keyword options:

1. The AUTO_CONSTRAINT option excludes rigid body motion from the implicit solution by automatically adding nodal constraints. A deck with a *CONTROL_FORMING_ONESTEP card should contain at most one *CONTROL_FORMING_ONESTEP_AUTO_CONSTRAINT card. In addition, three nodes can be specified on the final part to position the unfolded blank for easier blank nesting and for blank alignment in forming simulations.

2. The DRAWBEAD option is used to apply draw bead forces in addition to those provided by the AUTOBD field in Card 1a.1. A deck containing a *CONTROL_FORMING_ONESTEP card may contain as many *CONTROL_FORMING_ONESTEP_DRAWBEAD cards as there are draw beads to be defined. See Remark 6.

3. The FRICTION option applies friction along the edge of the part based on the binder tonnage input by the user in the BDTON field of Card 1. A deck containing a *CONTROL_FORMING_ONESTEP card may contain as many *CONTROL_FORMING_ONESTEP_FRICTION cards as there are friction node sets to be defined.

4. When the one-step forming method was first introduced back in 2011, all quadrilateral elements in the model were split into two triangular elements internally for calculation. As of R10, this original formulation (with no option) is set as option TRIA. QUAD supports quadrilateral elements and uses an improved algorithm, leading to better results. In addition, this option greatly improves calculation speed under multiple CPUs in SMP. QUAD2 improves upon option QUAD with an enhanced element formulation, further improving thinning and plastic strain results while slightly increasing CPU time. Remark 16 provides calculation speed comparisons of the three options. The option QUAD2 is set as a default as of R10 and is the recommended option.

Card Summary:

Card 1a.1. Include this card if the keyword option is unset (<BLANK>), TRIA, QUAD, or QUAD2.

OPTION	TSCLMAX	AUTOBD	TSCLMIN	EPSMAX		LCSDG	DMGEXP

Card 1a.2. Include this card if the keyword option is unset (<BLANK>), TRIA, QUAD, or QUAD2.

FLATNAME

Card 1b. Include this card if the AUTO_CONSTRAINT keyword option is used.

ICON	NODE1	NODE2	NODE3	PADID		IDRAW	

Card 1c. Include this card if the DRAWBEAD keyword option is used.

NDSET	LCID	TH	PERCNT				

Card 1d. Include this card if the FRICTION keyword option is used.

NDSET	BDTON	FRICT					

Card 1e. Include this card if the ORTHO keyword option is used.

PID	NODE1	NODE2					

Data Card Definitions:

Card 1 for no option (<BLANK>), TRIA, QUAD, and QUAD2.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	OPTION	TSCLMAX	AUTOBD	TSCLMIN	EPSMAX		LCSDG	DMGEXP
Type	I	F	F	F	F		I	F
Default	6	0.0	0.0	0.0	1.0		none	none

Card 2 for no option (<BLANK>), TRIA, QUAD, and QUAD2.

Card 1a.2	1	2	3	4	5	6	7	8
Variable	FLATNAME							
Type	A							
Default	none							

VARIABLE	DESCRIPTION
OPTION	Options to invoke the one-step solution methods which account for undercut conditions in the formed part: EQ.6: One-step solution with unfolded blank (flat) provided by LS-PrePost (see Remark 3). Input in Card 1a.2 is required. EQ.7: One-step solution with blank automatically unfolded in LS-DYNA. Card 1a.2 must be included as a blank line. This option is recommended. L.T.0: If a negative sign precedes any of the above OPTIONs, the stress and strain output in the file onestepresult will be in a large format (E20.0), which leads to more accurate stress results. Card 1a.2 must be included as a blank line.
TSCLMAX	If nonzero, it defines a thickness scale factor limiting the maximum thickness in the part. See Remark 12. For example, if the maximum thickness allowed is 0.8 mm for a blank with an initial thickness of 0.75 mm, TSCLMAX can be set to 1.0667. All thicknesses that are computed as more than 0.8 mm in the sheet blank will be reset to 0.8 mm. The scale factor is useful in advance feasibility analysis, where part design and stamping process have not been finalized, and could potentially cause large splits or severe wrinkles during unfolding, rendering the forming results unusable for crash/safety simulation.
AUTOBD	Apply a fraction of a fully locked bead force along the entire periphery of the blank. The fully locked bead force is automatically calculated based on a material hardening curve input. AUTOBD can be increased to easily introduce more thinning and effective plastic strain in the part. See Remark 6. LT.0.0: Turns off the auto-bead feature.

VARIABLE	DESCRIPTION
	EQ.0.0: Automatically applies 30% of fully locked force. GT.0.0: Fraction input will be used to scale the fully locked force.
TSCLMIN	If not zero, defines a thickness scale factor limiting the maximum thickness reduction. See Remark 12. For example, if the minimum thickness allowed is 0.6 mm for a blank with an initial thickness of 0.75 mm, TSCLMIN can be set to 0.8. All thicknesses that are computed as less than 0.6 mm in the sheet blank will be reset to 0.6 mm. The scale factor is useful in advance feasibility analysis, where part design and stamping process have not been finalized, and could potentially cause large splits or severe wrinkles during unfolding, rendering the forming results unusable for crash/safety simulation.
EPSMAX	If not zero, it defines the maximum effective plastic strain allowed. All computed effective plastic strains that are greater than this value in the blank will be set to this value. See Remark 12.
LCSDG	Load curve ID defining equivalent plastic strain to failure as a function of stress triaxiality; see *MAT_ADD_EROSION.
DMGEXP	Exponent for nonlinear damage accumulation; see *MAT_ADD_EROSION. Damage accumulation is written as history variable #6 in the file onestepresult.
FLATNAME	File name of the initial unfolded blank by LS-PrePost (see Remark 3). This is needed only for the OPTION = 6. Leave a blank line for OPTION = 7.

Card 1 for option AUTO_CONSTRAINT.

Card 1b	1	2	3	4	5	6	7	8
Variable	ICON	NODE1	NODE2	NODE3	PADID		IDRAW	
Type	I	I	I	I	I		I	
Default	none	none	none	none	0		0	

VARIABLE	DESCRIPTION
ICON	Automatic nodal constraining option to eliminate the rigid body motion (see Remark 8): EQ.1: Apply
NODE[1,2,3]	Node IDs for which the position is fixed during the unfolding. The position of these nodes in the calculated unfolded piece will coincide with the corresponding nodes in the input. The transformed and unfolded blank is written to keyword file repositioned.k. *When these fields are undefined, the orientation of the unfolded blank is arbitrary.* As of R11, the file repositioned.k is always output regardless of whether these fields are defined.
PADID	Optional shell set ID defining the binder region. If the pad is defined, the final flat blank is positioned using the pad as reference, such that the (x,y) mass centers match. Note that the pad region must be flat and the 3D meshes must be close to each other.
IDRAW	Draw orientation: EQ.-1: Reorient the part based on an approximate draw orientation if possible. This option may improve the consistency of the one-step prediction. EQ.0: Do not reorient the part based on an approximate draw orientation GT.0: Vector ID defining the drawing direction

Card 1 for option DRAWBEAD.

Card 1c	1	2	3	4	5	6	7	8
Variable	NDSET	LCID	TH	PERCNT				
Type	I	I	F	F				
Default	none	none	0.0	0.0				

VARIABLE	DESCRIPTION
NDSET	Node set ID along the periphery of the part; see *SET_NODE_LIST.
LCID	Load curve ID that defines the material hardening curve.

VARIABLE	DESCRIPTION
TH	Thickness of the unformed sheet blank.
PERCNT	Draw bead lock force fraction of the fully locked bead force.

Card 1 for option FRICTION.

Card 1d	1	2	3	4	5	6	7	8
Variable	NDSET	BDTON	FRICT					
Type	I	F	F					
Default	none	0.0	0.12					

VARIABLE	DESCRIPTION
NDSET	Node set ID along the periphery of the part, as defined by keyword *SET_NODE_LIST.
BDTON	Binder tonnage used to calculate the friction force. See Remark 7.
FRICT	Coefficient of friction

Card 1 for option ORTHO, for anisotropic material.

Card 1e	1	2	3	4	5	6	7	8
Variable	PID	NODE1	NODE2					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID of the final formed blank mesh
NODE1	First node in the part to define the metal rolling direction
NODE2	Second node in the part to define the metal rolling direction

About the One-Step Forming Solution:

The one-step solution employs the total strain (or deformation) theory of plasticity in place of the more realistic incremental strain (or flow) theory. The total deformation theory expresses stress as a function of total strain; whereas the incremental strain theory requires that LS-DYNA compute a stress update at each time step (strain increment) from the deformation that occurred *during that time step*. Deformation theory results, therefore, do not depend on strain path, forming history, or the details of the stamping process.

When this card is included, the input must contain the *final geometry* from which the one-step algorithm calculates the initial flat state using the inverse method. The one-step solution results can get close to the incremental results only when the forming process follows a linear strain path, where the deformation is either monotonically increasing or decreasing. In most cases total strain theory does not match incremental forming.

Path independence leads to several key simplifications:

1. Binder and addendum geometry are not required. There is no need to measure or model these geometries.

2. The solution is independent of stamping die processes (including part tipping).

3. There is no need for contact treatment since there are no tools and dies involved.

The one-step solution is most commonly used for advanced formability studies, where you need to quickly compare a wide range of design alternatives. With this method, you can evaluate blank size, estimate material cost, and generate a first guess for blank size development (see also *CONTROL_FORMING_UNFLANGING and *INTERFACE_BLANKSIZE_DEVELOPMENT). This method is also widely used to initialize forming stresses and strains in crash and occupant safety analyses.

Example:

The following example provides a partial input file with typical control cards. It will iterate for four steps, with auto beads of 0% lock force applied around the part boundary and automatic nodal constraints.

```
*CONTROL_TERMINATION
$    ENDTIM
        1.0
*CONTROL_IMPLICIT_GENERAL
$    IMFLAG        DT0
           1     0.25
*CONTROL_FORMING_ONESTEP
$    OPTION              AUTODB
         7

*CONTROL_FORMING_ONESTEP_AUTO_CONSTRAINT
$      ICON
         1
```

```
*CONTROL_IMPLICIT_TERMINATION
$    DELTAU
      0.001
*CONTROL_IMPLICIT_SOLUTION
$  NSLOLVR     ILIMIT     MAXREF      DCTOL      ECTOL
         2         11       1200       0.01       1.00
*CONTROL_IMPLICIT_SOLVER
$    LSOLVR
         5
*CONTROL_IMPLICIT_AUTO
$     IAUTO     ITEOPT     ITEWIN      DTMIN      DTMAX
         0          0          0        0.0        0.0
```

Additional cards below specify extra bead forces of 45% and 30% applied to node sets 22 and 23 along the part periphery, respectively. Also, the friction forces are applied to the same node sets with a friction coefficient of 0.1 and a binder tonnage of 10000.0 N.

```
*CONTROL_FORMING_ONESTEP_DRAWBEAD
$    NDSET       LCID         TH     PERCNT
        22        200        1.6       0.45
*CONTROL_FORMING_ONESTEP_DRAWBEAD
        23        200        1.6       0.30
*CONTROL_FORMING_ONESTEP_FRICTION
$    NDSET      BDTON     FRICT
        22    10000.0       0.1
*CONTROL_FORMING_ONESTEP_FRICTION
$    NDSET      BDTON     FRICT
        23    10000.0       0.1
```

The one-step forming results for the NCAC Taurus model's firewall are shown in Figure 12-28. The average element size across the blank is 8 mm, and the trimmed part (with holes filled) consists of 15490 elements. *MAT_24 was used with BH210 material properties. On a 1 CPU Xeon E5520 Linux machine, it took 4 minutes to complete the run with a total of four steps. The thickness, the plastic strain, and the blank size prediction were reasonable, as shown in Figures 12-29, 12-30 and 12-31.

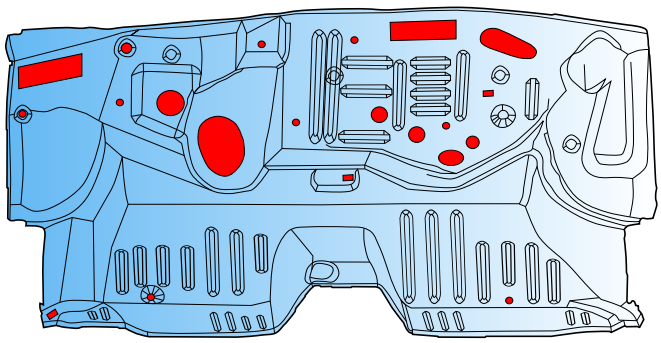


Figure 12-28. A trimmed dash panel (firewall) with holes auto-filled using LS-PrePost 4.0 (original model courtesy of NCAC Taurus crash model).

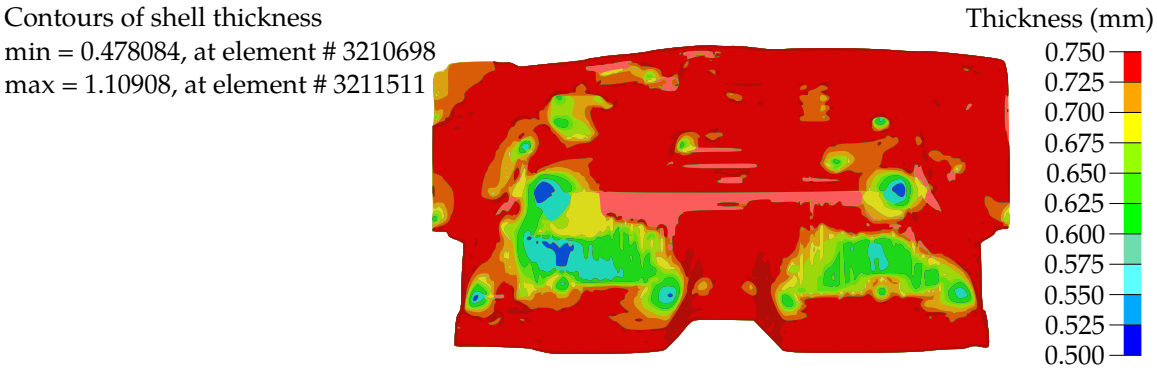


Figure 12-29. Shell thickness prediction (t_0 = 0.75 mm).

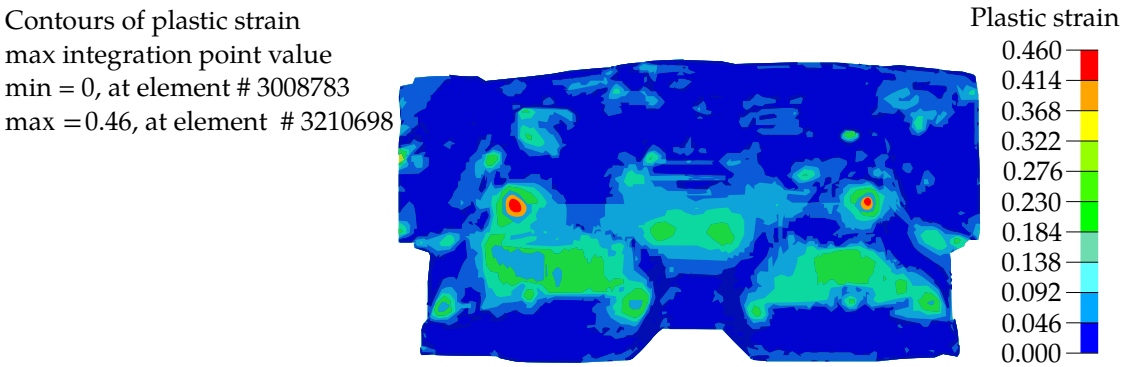


Figure 12-30. Effective plastic strain prediction.

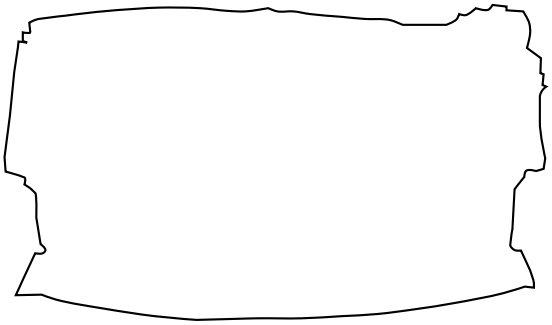


Figure 12-31. Initial blank size prediction (flat, not to scale).

Remarks:

1. **Mesh.** In addition to the usual material and physical property definitions, this method requires that the final part be fully meshed using shell elements. *No adaptive mesh is supported.* This mesh should look quite different from a formed blank undergoing incremental forming. For example, along the part bend radius, there is no need to build six elements along the arc length as one would do for the punch/die radius in an incremental forming; two elements may be

enough. A mesh consisting of uniformly distributed quadrilateral shell elements is ideal. *All elements in the mesh must also have normal consistency*.

With LS-PrePost 4.0, this kind of mesh can be generated using *Mesh* → *AutoM* → *Size*. Since this method uses an implicit static solution scheme, the number of elements controls the computational cost; element size has no effect. Furthermore, to obtain forming results that are similar to the incremental forming results, the part in the one-step input should be similar in size to the final formed blank shape in the incremental forming (before trimming).

Note that dynain files with *ELEMENT_SHELL_THICKNESS, *INITIAL_STRESS_SHELL, and *INITIAL_STRAIN_SHELL are not supported.

2. **Holes.** Any trimmed-out holes can be filled (but not necessary). The filling can be done semi-automatically using LS-PrePost 4.0 by selecting *Mesh* → *EleGen* → *Shell* → *Shell by Fill_Holes* → *Auto Fill*. The filled area of the part can be saved in a different part, as multiple parts (PID) are allowed. The forming results will be more realistic when holes are filled or left alone, depending on the actual stamping process intent.

3. **Unfolding.** For OPTION = 6, the unfolded blank can be obtained from LS-PrePost using *EleTol* → *Morph* → *Type* = *Mesh_Unfolding* → *Unfold*. The unfolded mesh can be saved as a keyword file and used as input (see the FLATNAME field in Card 1a.2). With OPTION = 7, LS-DYNA unfolds the mesh itself.

4. **Element formulation.** Shell element types 2 and 16 are supported. Since this feature uses the implicit method, type 16 is more convergent and computationally efficient than type 2; therefore, it is strongly recommended. Results are output on all integration points, as seen in the ELFORM and NIP variables in *SECTION_SHELL.

5. **Supported materials.** *MAT_024, *MAT_036, *MAT_037, *MAT_112, *MAT_114, *MAT_123, *MAT_133, and *MAT_238 are supported. The user *must* provide a material hardening curve. For *MAT_024, the hardening curve must be specified with the LCSS field. It can be input as a table. Strain rate, however, is ignored for this material, even if the variables C and P are provided. For MAT_036, the hardening curve must be a load curve (HR = 3), and for *MAT_037, it must be specified with HLCID.

   For an anisotropic material, the option ORTHO can be used to define the material rolling direction. Since there is no flat blank existing as an input, the only way to define the rolling direction is to use two existing nodes in the 3D formed part to meet the requirement.

6. **Boundary conditions.** The primary boundary/loading condition for the one-step solution is the draw bead forces, which are set with the AUTOBD field or with the DRAWBEAD keyword option.

   a) With the DRAWBEAD option, draw bead forces are applied to a user-defined node set (see NDSET). A fraction of the full lock force, determined by the tensile strength and sheet thickness, can be specified. The larger the fraction, the less the metal will flow into the die, resulting in more stretching and thinning.

   b) Boundary conditions may also be set using the auto-beads feature (see the AUTOBD field), with which draw bead forces are automatically applied to all nodes along the part boundary. The users must specify the fraction of the fully locked bead force to be applied. The default value of 30% is sufficient for crash/occupant safety applications.

   The last important, but often overlooked, boundary condition is the part's shape. For example, an oil pan with a larger flange area will experience greater thinning in the part wall, whereas a smaller flange area leads to the opposite effect. To obtain results closer to incremental strain theory, additional material may need to be added to the final part geometry in cases where the sheet blank is not fully developed, meaning no trimming is required to finish the part.

7. **Friction.** Friction effects can be included with the FRICTION option. The frictional force is based on an expected binder tonnage and is a percentage of the input force. Note that the binder tonnage value (see BDTON) is used *exclusively* in calculating friction forces. The binder tonnage is not actually applied to the binder as a boundary condition.

8. **Rigid body motion.** To automatically add nodal constraints to prevent rigid body motion, use the AUTO_CONSTRAINT keyword option with ICON set to 1.

9. **Implicit solver options.** All other implicit cards, such as *CONTROL_IMPLICIT_GENERAL, *CONTROL_IMPLICIT_SOLUTION, *CONTROL_IMPLICIT_SOLVER, *CONTROL_IMPLICIT_AUTO, *CONTROL_IMPLICIT_TERMINATION, etc., are used to set the convergence tolerance, termination criterion, etc. The two most important variables controlling the solution convergence are DELTAU from *CONTROL_IMPLICIT_TERMINATION and DCTOL from *CONTROL_IMPLICIT_SOLUTION. Experience has shown that they should be set to 0.001 and 0.01, respectively, to obtain the most efficient solution with the best results. Typically, four implicit steps are sufficient, and DT0 in *CONTROL_IMPLICIT_GENERAL and ENDTIM in *CONTROL_TERMINATION should be set accordingly. For difficult parts, more steps may be needed. For some parts, ILIMIT in *CONTROL_IMPLICIT_SOLUTION may need to be set to 1 for the full Newton iteration.

10. **Results.** Results are stored in an ASCII file named onestepresult using the dynain format. This file contains the forming thickness, stress field, and strain field on the final part. It can be plotted with LS-PrePost. One quick and useful LS-PrePost plotting feature is the formability contour map, which colors the model to highlight various forming characteristics, including cracks, severe thinning, wrinkles, and good surfaces. The formability map feature is in *Post → FLD → Formability*.

    The variable PSID in *INTERFACE_SPRINGBACK_LSDYNA can be used to control the output to onestepresult and dynain files, starting in R12.

11. **Final estimated blank size.** Additionally, the final estimated blank size in its initial, flat state is stored in the d3plot files. The d3plot files also contain intermediate shapes from each implicit step. The final blank mesh in its flat state can be written to a keyword file using LS-PrePost by the following steps:

    a) Go to *Post → Output → Keyword*,

    b) Check the box to include *Element* and *Nodal Coordinates*

    c) Move the animation bar to the last state, and

    d) Click on *Curr* and *Write.*

    In addition, blank outlines can be created by:

    e) Menu option *Curve → Spline → From Mesh (Method)*,

    f) Checking *Piecewise → byPart*,

    g) Select the blank,

    h) Click on *Apply*, and

    i) Finally, save the curves in IGES format using the *File* menu at the upper left corner.

    The unformed blank mesh will be output into the file repositioned.k.

12. **Effects of TSCLMAX, TSCLMIN, and EPSMAX.** During the early stages of product design, the initial product specifications may lead to large strains and excessive thinning or thickening on the formed panel. The ensuing one-step results may not be suitable to be used in a crashworthiness simulation. However, these kinds of forming issues are certain to be fixed as a natural part of the design and stamping engineering process. The variables TSCLMAX, TSCLMIN, and EPSMAX thus impose artificial limits on excessive thickening (indication of wrinkles), excessive thinning, and plastic strains, respectively. These fields

provide a convenient way to run a crash simulation with approximate and reasonable forming effects before the design is finalized. In the keyword below (which is a part of the firewall model with original thickness of 0.75 mm), TSCLMIN and EPSMAX are set to 0.8 and 0.3, respectively.

```
*CONTROL_FORMING_ONESTEP
$    OPTION    TSCLMAX     AUTODB    TSCLMIN     EPSMAX
         7                    0.5        0.8        0.3
```

The thickness and effective plastic strain plots for the firewall model are shown in Figures 12-32 and 12-33, respectively. The minimum value in the thickness contour plot and the maximum value in the plastic strain contour plot, as shown in the upper left corner, correspond to the values specified in TSCLMIN and EPSMAX, respectively.

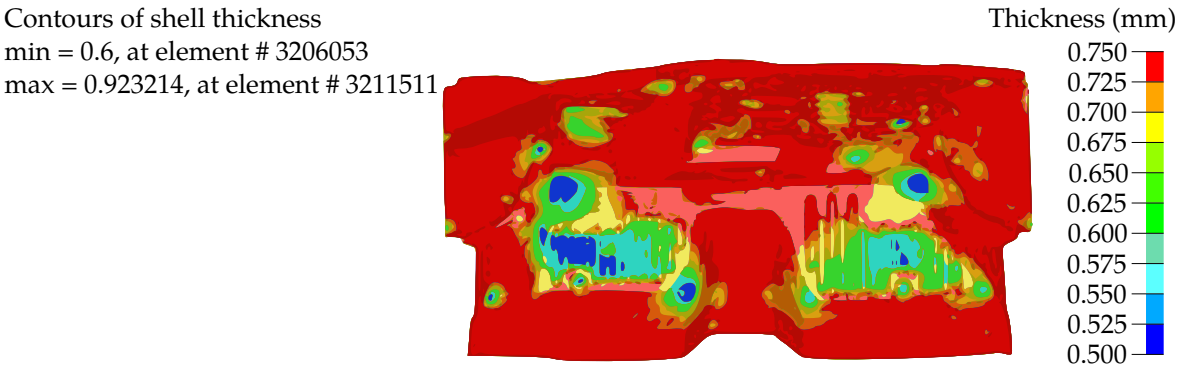


Figure 12-32. Blank thickness prediction with TSCLMIN = 0.8.

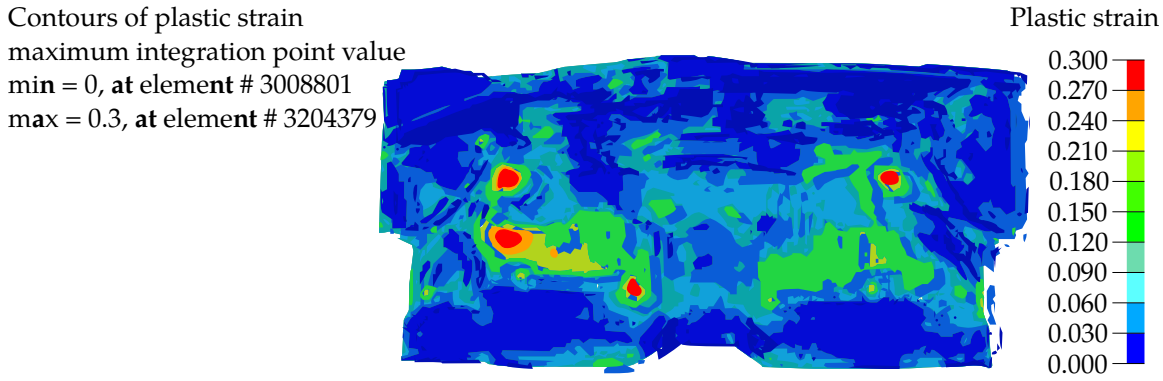


Figure 12-33. Effective plastic strain with EPSMAX = 0.3.

Similarly, TSCLMAX can be set to 1.0667 to limit the max thickening in the part to 0.8 mm:

```
*CONTROL_FORMING_ONESTEP
$    OPTION    TSCLMAX     AUTODB    TSCLMIN     EPSMAX
         7     1.0667        0.5        0.8        0.3
```

13. **Repositioning the unfolded flat blank.** Often, the input to a one-step simulation is the final product part in a coordinate system that is useful for subsequent

simulations. However, after the one-step simulation, the unfolded flat blank will be in a different orientation and position, requiring manual repositioning of the blank to its desired orientation and position. The variables NODE1, NODE2, and NODE3 allow users to specify three nodes so that the blank is transformed onto the final part (the input), superimposing the exact same three nodes in both parts. In the example shown in Figure 12-34, the three nodes (Nodes 197, 210, and 171) are defined near the edges of two holes. The transformed and unfolded flat blank (written to a keyword file repositioned.k) is seen superimposed onto the final part according to the three nodes specified (Figure 12-34 bottom). If these nodes are not defined, the simulation will result in the unfolded flat blank in a state shown in Figure 12-34 (top), which is undesirable to most users.

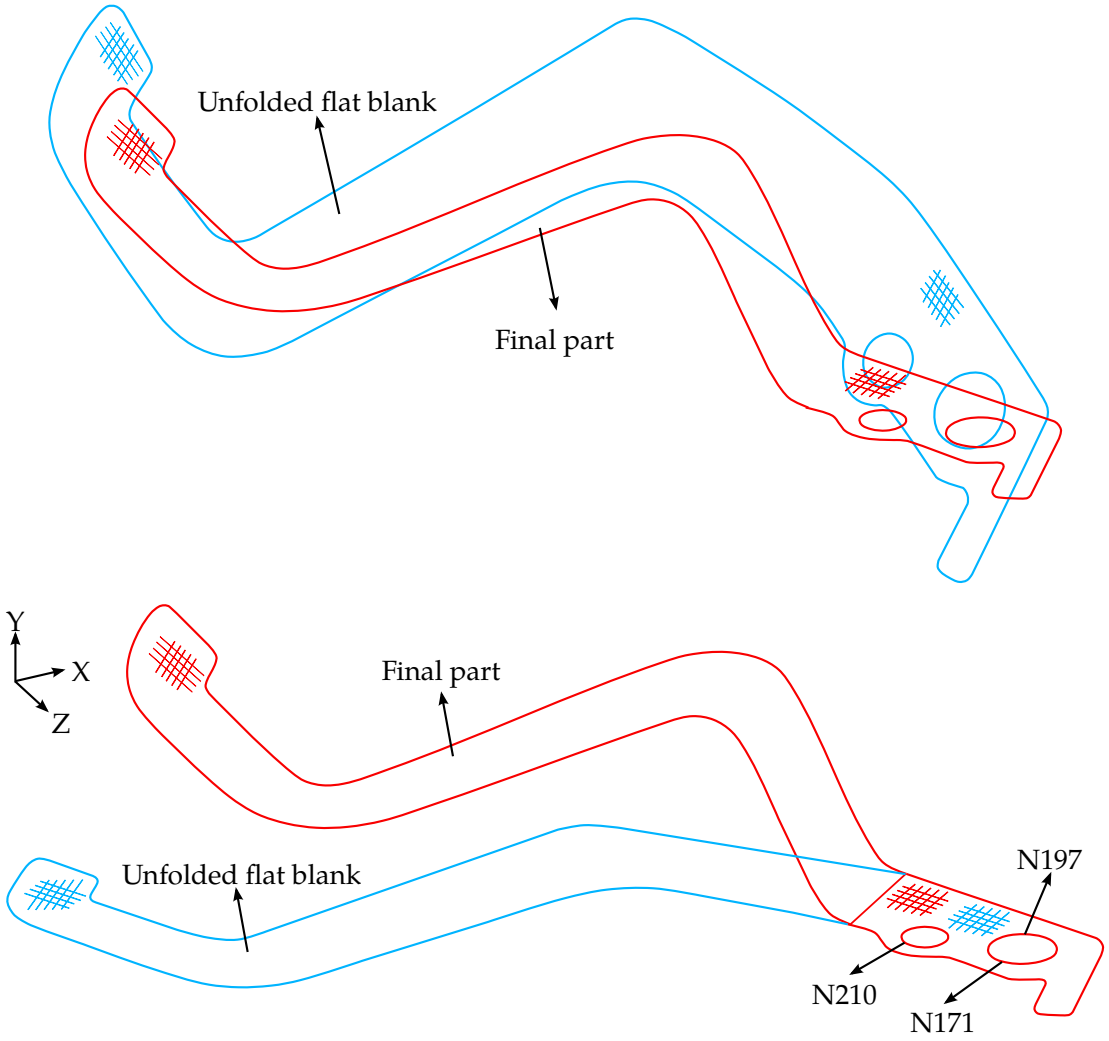


Figure 12-34. An example of the results when using the NODE1, NODE2, and NODE3 feature (bottom) and without using the feature (top), courtesy of Kai-zenet Technologies Pvt Ltd, India.

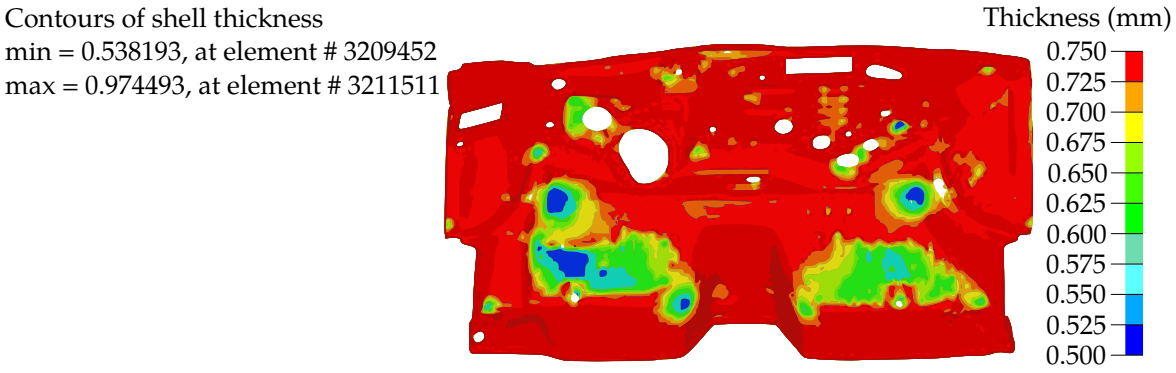


Figure 12-35. Blank thickness with trimmed holes (t_0 = 0.75 mm).

14. **Damage accumulation.** Damage accumulation D is calculated based on (see *MAT_ADD_EROSION):

$$D = \left(\frac{\varepsilon_p}{\varepsilon_f}\right)^{\text{DMGEXP}},$$

where ε_p is the equivalent plastic strain and ε_f is the equivalent plastic strain to failure.

In the example below, load curve 500 (LCSDG) gives plastic failure strain as a function of stress triaxiality, and DMGEXP is assumed to be 1.254. Since the damage accumulation is written into the file onestepresult as history variable #6, the variable NEIP in *DATABASE_EXTENT_BINARY should be set to at least 6.

```
*CONTROL_FORMING_ONESTEP
$    OPTION              AUTODB    TSCLMIN     EPSMAX               LCSDG     DMGEXP
         7                              0.8        0.3                 500      1.254
*DEFINE_CURVE
500
-0.3,0.6
-0.2,0.3
0.0,0.2
0.2,0.25
0.4,0.46
0.65,0.28
0.9,0.18
*DATABASE_EXTENT_BINARY
$     NEIPH      NEIPS     MAXINT     STRFLG     SIGFLG     EPSFLG     RLTFLG     ENGFLG
                     6          7          1
$    CMPFLG     IEVERP     BEAMIP      DCOMP       SHGE      STSSZ
                     1                     2
```

The damage-accumulation contour map from onestepresult can be plotted in LS-PrePost.

15. **Effects of unfilled holes on forming results.** In Figure 12-35, a thickness contour plot of a one-step calculation on the NCAC Taurus firewall model with its holes unfilled is shown. The unfilled case will undergo slightly less thinning, since the holes will expand as material flows outward away from the hole.

However, the thicknesses with holes filled are likely closer to reality, since the holes are mostly filled during forming on the draw panel and then trimmed off afterwards in a trim process. On the other hand, it is important to realize that not all the holes are filled in a draw panel. Some holes are cut inside the part in the scrap area (but not all the way to the trim line) during the draw process to allow material to flow into areas that are difficult to form, which avoids splitting.

16. **Performance comparison of keyword options TRIA, QUAD, and QUAD2.** The following partial keyword input is an example of using the option QUAD. Note the draw bead force parameter AUTOBD is set at 0.5. Calculation speed comparison among options QUAD, QUAD2, and TRIA can be found in Table 12-2.

```
*KEYWORD
*include
model.k
*CONTROL_TERMINATION
1.0
*CONTROL_FORMING_ONESTEP_QUAD
$#  option  maxthick    autobd   thinmin    epsmax
         7                0.5

*CONTROL_FORMING_ONESTEP_AUTO_CONSTRAINT
         1
*CONTROL_IMPLICIT_GENERAL
$#  imflag       dt0    imform      nsbs       igs     cnstn      form    zero_v
         1    0.2500         2         1         0         0         0         0
*CONTROL_IMPLICIT_TERMINATION
$#  deltau    delta1     ketol     ietol     tetol     nstep
  0.001000     0.000     0.000     0.000     0.000         0
*CONTROL_IMPLICIT_NONLINEAR
$#  nsolvr    ilimit    maxref     dctol     ectol  not used     lstol      rssf
        12        11       200  0.010000  0.100000     0.000     0.000     0.000
$#   dnorm    diverg     istif   nlprint
         0         0         0         2
$#  arcctl    arcdir    arclen    arcmth    arcdmp
         0         0     0.000         1         2
*CONTROL_IMPLICIT_SOLVER
5
*PART

   5000000   5000000   5000000
*SECTION_SHELL
   5000000        16        1.        5.        1.
      0.72      0.72      0.72      0.72
```

Table 12-2. Calculation speed for various options. These results are from using a double-precision SMP executable (revision 112720 in December of 2016) with 8 CPUs.

		Calculation speed		
	# of elements	Option TRIA	Option QUAD	Option QUAD2
A hat-shaped part	71000	21.0 min	14.1 min	16.6 min
An upper dash panel	61700	24.5 min	11.5 min	17.2 min

***CONTROL_FORMING_OUTPUT**_*{OPTION}*

Available options include:

<BLANK>

INTFOR

Purpose: This card defines the times at which states are written to the d3plot and intfor files based on the tooling's distances from the home (final) position. When the INTFOR option is set, this keyword card controls when states are written to the intfor file, otherwise it controls the d3plot file. This feature may be combined with parameterized input and/or automatic positioning of the stamping tools using the *CONTROL_FORMING_-AUTOPOSITION_PARAMETER card.

> **NOTE:** When this card is present, no states are written except for those specified on this card. This card supersedes the *DATABASE_BINARY_D3PLOT card.

Forming Output Cards. Repeat as many times as needed to define additional outputs in separate tooling kinematics curves. The next keyword ("*") card terminates the input.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	NOUT	TBEG	TEND	Y1/LCID	Y2/CIDT	Y3	Y4
Type	I	I	F	F	F/I	F/I	F	F
Default	none	0	0.0	none	Rem 3	Rem 3	Rem 3	Rem 3

VARIABLE	DESCRIPTION
CID	ID of a tooling kinematics curve. This curve is integrated so that the specified output distances can be mapped to times. For correct distance-to-time mapping CID must be applied to the tool of interest using a *BOUNDARY_PRESCRIBED_MOTION_-RIGID card. The ordinate scale factor SFO in the *DEFINE_-CURVE is supported in this keyword starting from Dev Revision 82755.
NOUT	Total number of states written to the d3plot or intfor databases for the tooling kinematics curve, CID, excluding the beginning and final states. If NOUT is larger than the number of states specified by either LCID or Y*i* fields (5 through 8), the remaining states are

VARIABLE	DESCRIPTION
	evenly distributed between TBEG and the time corresponding to the biggest Y*i* from the home position, as shown in Figure 12-36. If NOUT is left as blank or as "0", the total number of output states will be determined by either LCID or Y*i*'s. Starting in Dev Revision 124051, NOUT works alone, without the need to define LCID or Y*i*'s.
TBEG	Start time of the curve. This time should be consistent with the BIRTH in *BOUNDARY_PRESCRIBED_MOTION_RIGID.
TEND	End time of the curve. This time should be consistent with the DEATH in *BOUNDARY_PRESCRIBED_MOTION_RIGID. This time is automatically reset backward removing any idling time if the tool finishes traveling early, so output distances can start from the reset time. A state is written at TEND.
Y1/LCID, Y2, Y3, Y4	Y1/LCID.GT.0: All four variables (Y1, Y2, Y3, Y4) are taken to be the distances from the punch home, where d3plot files will be output. Y1/LCID.LT.0: The absolute value of Y1/LCID (must be an integer) is taken as a load curve ID (see *DEFINE_CURVE). Only the abscissas in the load curve, which are the distances to punch home, are used. These distances specify the states that are written to the d3plot files. Ordinates of the curve are ignored. This case accommodates more states than is possible with the four variables Y1, Y2, Y3, Y4. Furthermore, when Y1/LCID < 0, Y2, Y3, and Y4 are ignored. Available starting from Dev Revision 112604, the output will be skipped for any negative abscissa in the load curve. Note a curve with *only* negative abscissas is not allowed.
Y2/CIDT	Y2/CIDT.GT.0: The input is taken as the distance from the punch home, where a d3plot file will be output. Y2/CIDT.LT.0: The absolute value of Y2/CIDT (must be an integer) is taken as a load curve ID (see *DEFINE_CURVE). Only the abscissas in the load curve, which are the simulation times, are used. These times specify the states that are written to the d3plot files. Ordinates of the curve are ignored.

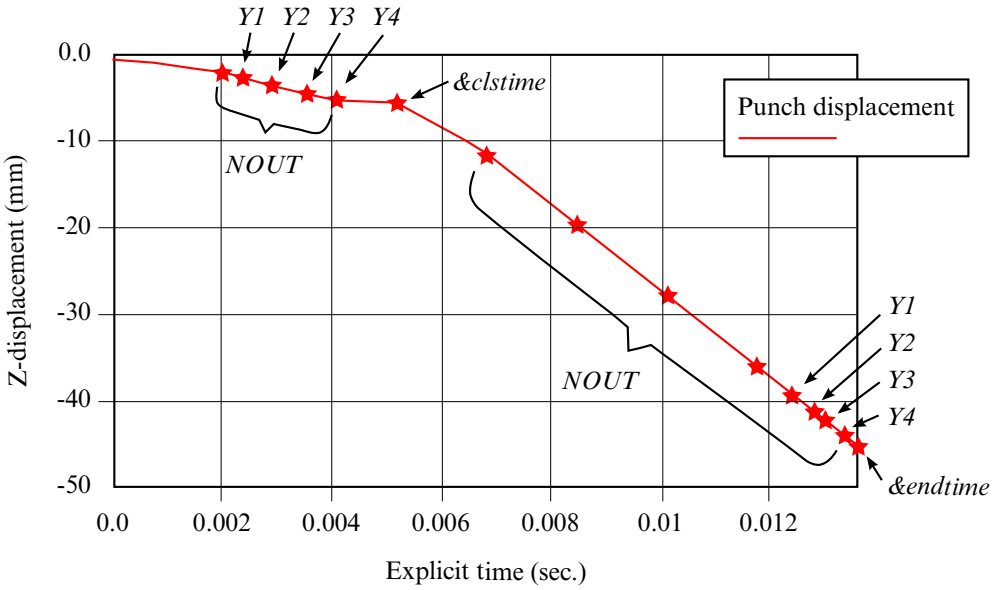


Figure 12-36. An output example for closing and drawing. See the example provided at the end of this section.

VARIABLE	DESCRIPTION
	Note this time-dependent load curve will output additional d3plot files on top of the d3plot files already written in case Y1/LCID < 0 (if specified). Furthermore, when Y2/CIDT < 0, Y3 and Y4 are ignored. See the example Using CIDT below.

Motivation:

In stamping simulations not all time steps are of equal interest. This feature allows you to save special states, usually those for which wrinkling and thinning conditions arise as the punch approaches its home position.

Remarks:

1. **Related Keywords.** Keywords *DATABASE_BINARY_D3PLOT and *DATABASE_BINARY_INTFOR are not required (ignored if present) to output d3plot and intfor files when this keyword is present.

2. **CID.** *CONTROL_FORMING_OUTPUT and *CONTROL_FORMING_OUTPUT_INTFOR can share the same CIDs.

3. **Output Distribution.** The following are rules for output distribution:

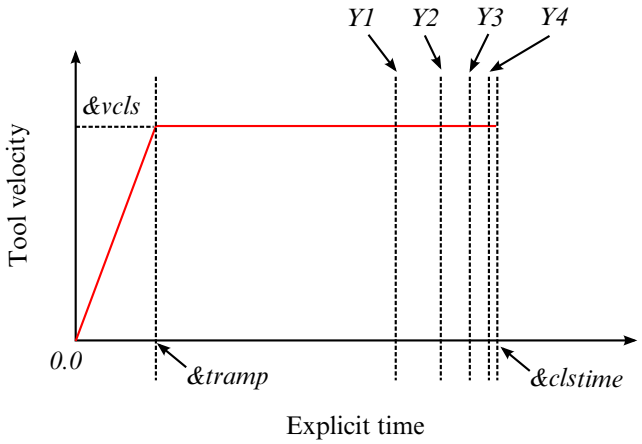


Figure 12-37. Specifying d3plot/intfor output at specific distances to punch home.

a) If columns 5 through 8 are left blank, output (NOUT) will be evenly distributed through the travel.

b) The variable NOUT has priority over the number of points on the LCID.

c) Distances input (in LCID) that are greater than the actual tool travel will be ignored.

d) Distance input (in LCID) does not necessarily have to be in a descending or ascending order.

Applicability:

Velocity must be the prescribed motion in *BOUNDARY_PRESCRIBED_MOTION_-RIGID (VAD = 0) for this keyword to apply. This keyword only works for explicit dynamics. Tooling kinematics profiles of various trapezoids (including right trapezoid) are all supported. Local coordinate systems are supported.

Air Draw Example:

In the sample keyword input below (air draw, referring to Figures 12-36 and 12-38), a total of five states will be output during a binder closing. The kinematics are specified by the curve of ID 1113, which defines tooling kinematics starting time 0.0 and ending at time `&clstime`.

Curve 1113 is used to associate the specified distances to the appropriate time step. In this example NOUT is set to 5. Of these five outputs states the last four will be output at upper die distance to closing of 3.0, 2.0, 1.0, and 0.5 mm according to the values specified in the Y1, Y2, Y3, and Y4 fields.

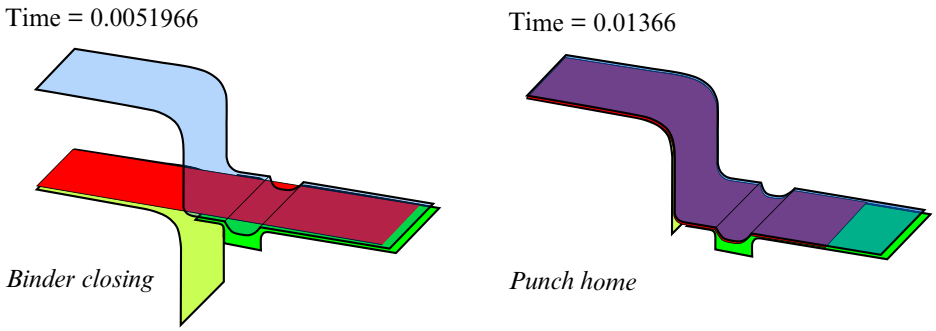


Figure 12-38. An air draw example with closing and drawing.

Similarly, a total of eight states will be written to the d3plot file made during draw forming according curve ID 1115, which defines tooling kinematics starting at time `&clstime`, and ending at time `&endtime`. Of the eight states the last four will be output at punch distance to draw home of 6.0, 4.0, 3.0, and 1.0 mm; the remaining four outputs will be evenly distributed between starting punch distance to home and punch distance of 6.0mm to home.

Likewise, for intfor, 15 states will be written before closing and 18 states after the closing. The d3plot and intfor files will always be output for the first and last states as a default; and at where the two curves meet at &clstime, only one d3plot and intfor will be output.

To output intfor, "S=*filename*" needs to be specified on the command line, and SAPR and SBPR need to be set to "1" on the *CONTACT_… cards.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_OUTPUT
$      CID      NOUT      TBEG      TEND        y1        y2        y3        y4
      1113         5             &clstime       3.0       2.0       1.0       0.5
      1115         8  &clstime   &endtime       6.0       4.0       3.0       1.0
*CONTROL_FORMING_OUTPUT_INTFOR
$      CID      NOUT      TBEG      TEND        y1        y2        y3        y4
      1113        15             &clstime       3.5       2.1       1.3       0.7
      1115        18  &clstime   &endtime      16.0       4.4       2.1       1.3
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
&udiepid           3         0      1113      -1.0         0  &clstime       0.0
&bindpid           3         0      1114       1.0         0  &clstime       0.0
&udiepid           3         0      1115      -1.0         0  &endtime  &clstime
&bindpid           3         0      1115      -1.0         0  &endtime  &clstime
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE
1113
0.0,0.0
&clsramp,&vcls
&clstime,0.0
*DEFINE_CURVE
1114
0.0,0.0
10.0,0.0
*DEFINE_CURVE
1115
0.0,0.0
&drwramp,&vdraw
```

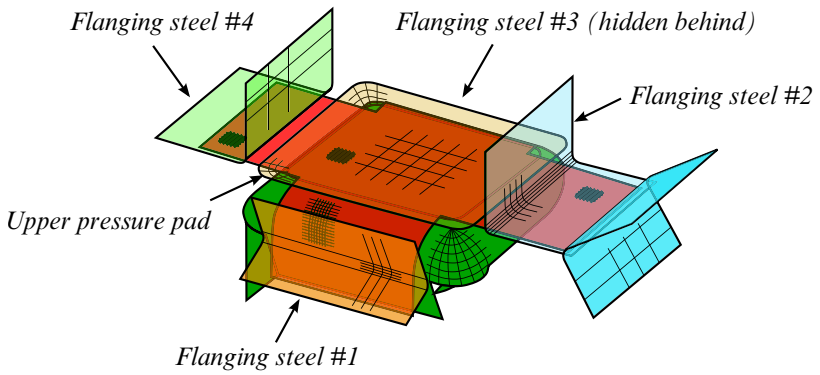


Figure 12-39. An example of multiple flanging process.

```
&drwtime,&vdraw
```

The keyword example below illustrates the use of load curves 3213 and 3124 to specify the states written to the d3plot and intfor files respectively. In addition to the eight states specified by curve 3213, five additional outputs will be generated. Similarly, in addition to the 10 intfor states defined by curve 3214, eight additional states will be output.

```
*CONTROL_FORMING_OUTPUT
$       CID       NOUT       TBEG       TEND         y1         y2         y3         y4
       1113         13             &clstime      -3213
*CONTROL_FORMING_OUTPUT_INTFOR
$       CID       NOUT       TBEG       TEND         y1         y2         y3         y4
       1113         18             &clstime      -3214
*DEFINE_CURVE
3213
88.0
63.0
42.0
21.5
9.8
5.2
3.1
1.0
*DEFINE_CURVE
3214
74.0
68.0
53.0
32.0
25.5
7.8
4.2
2.1
1.4
0.7
```

Multiple Flanging Process Example:

Referring to Figure 12-39 and a partial keyword example listed below, flanging steels #1 through #4 are defined as parameters `&flg1pid` through `&flg4pid`, respectively, which are moving in their own local coordinate systems. The termination time `&endtime` is defined as pad closing time, `&clstime`, plus the maximum travel time of all four flanging

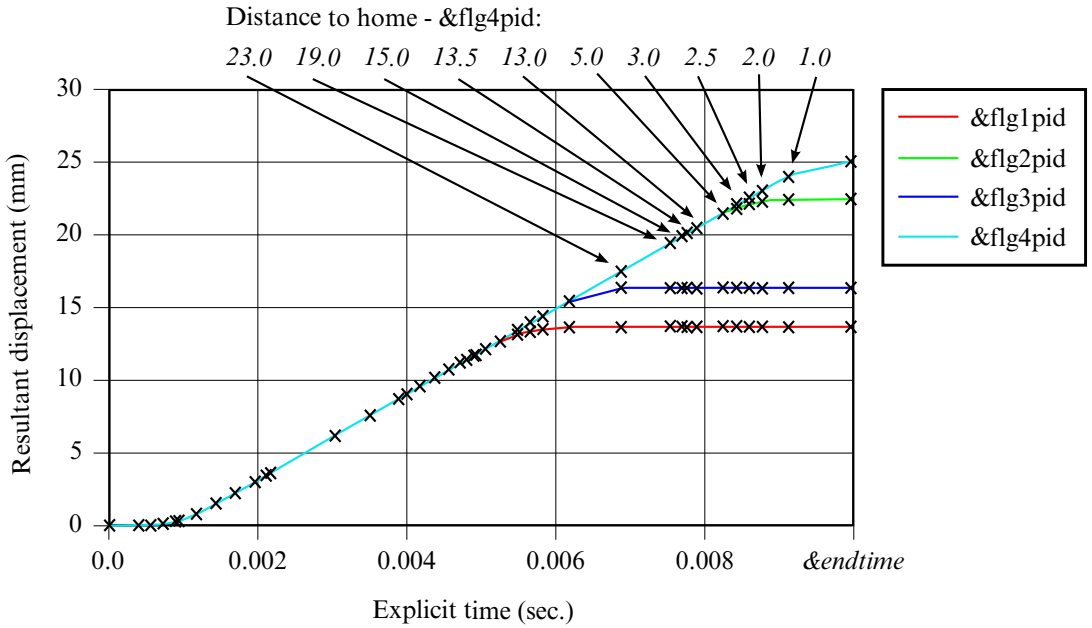


Figure 12-40. D3PLOT/INTFOR output in case of multiple flanging process.

steels. A total of ten d3plot states and ten intfor states are defined for each flanging steel using curve IDs 980 and 981, respectively. Curve values outside of the last 10 states (distances) are ignored; and reversed points are automatically adjusted.

In Figure 12-40, locations of d3plot states are indicated by "x" markers for each flanging steel move. Note that for flanging steels with longer travel distances, there may be additional d3plot states between the defined points, controlled by distance output defined for other flanging steels with shorter travels. The total number of d3plot (and intfor) states is the sum of all nout defined for each flanging steel so care should be taken to limit the total d3plot (and intfor) states, especially if large number of flanging steels are present.

```
*KEYWORD
$  -------------------------closing
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
    &upid1         3         0      1113  &padvdir         0  &clstime
*BOUNDARY_PRESCRIBED_MOTION_RIGID_local
  &flg1pid         3         0      1114       1.0         0  &clstime
  &flg2pid         3         0      1114       1.0         0  &clstime
  &flg3pid         3         0      1114       1.0         0  &clstime
  &flg4pid         3         0      1114       1.0         0  &clstime
$  -------------------------flanging
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
    &upid1         3         0      1115  &padvdir         0            &clstime
*BOUNDARY_PRESCRIBED_MOTION_RIGID_local
  &flg1pid         3         0      1116       1.0         0            &clstime
  &flg2pid         3         0      1117       1.0         0            &clstime
  &flg3pid         3         0      1118       1.0         0            &clstime
  &flg4pid         3         0      1119       1.0         0            &clstime
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE
      1116
                0.0                0.0
```

```
             &tdrwup               &vdrw
             &tdown1               &vdrw
            &drw1tim                 0.0
             1.0E+20                 0.0
*DEFINE_CURVE
      1117
                 0.0                 0.0
             &tdrwup               &vdrw
             &tdown2               &vdrw
            &drw2tim                 0.0
             1.0E+20                 0.0
*DEFINE_CURVE
      1118
                 0.0                 0.0
             &tdrwup               &vdrw
             &tdown3               &vdrw
            &drw3tim                 0.0
             1.0E+20                 0.0
*DEFINE_CURVE
      1119
                 0.0                 0.0
             &tdrwup               &vdrw
             &tdown4               &vdrw
            &drw4tim                 0.0
             1.0E+20                 0.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE
980
60.0
55.0
42.0
40.0
38.0
31.0
23.0
19.0
15.0
13.0
13.5
5.0
3.0
2.0
2.5
1.0
*DEFINE_CURVE
981
23.0
19.0
15.0
13.0
13.5
  ⋮
*CONTROL_FORMING_OUTPUT
$ -------1---------2---------3---------4---------5---------6---------7---------8
$       CID      NOUT      TBEG      TEND   Y1/LCID
      1116        10  &clstime  &endtime      -980
      1117        10  &clstime  &endtime      -980
      1118        10  &clstime  &endtime      -980
      1119        10  &clstime  &endtime      -980
*CONTROL_FORMING_OUTPUT_INTFOR
$ -------1---------2---------3---------4---------5---------6---------7---------8
$       CID      NOUT      TBEG      TEND   Y1/LCID
      1116        10  &clstime  &endtime      -981
      1117        10  &clstime  &endtime      -981
      1118        10  &clstime  &endtime      -981
      1119        10  &clstime  &endtime      -981
```

⋮ ⋮ ⋮ ⋮ ⋮

Using CIDT Example:

The example below shows in addition to the 7 states output based on various distances from punch home, defined by load curve 980, 4 more states are output based on simulation time, defined by load curve 999.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE
999
1.0e-03
2.0e-03
3.0e-03
4.0e-03
*DEFINE_CURVE
980
13.5,0.0
13.0,0.0
5.0,0.0
3.0,0.0
2.5,0.0
2.0,0.0
1.0,0.0
*CONTROL_FORMING_OUTPUT
$ -------1---------2---------3---------4---------5---------6---------7---------8
$      CID      NOUT      TBEG      TEND   Y1/LCID   Y2/CIDT
      1116         0  &clstime  &endtime      -980     -999
      1117         0  &clstime  &endtime      -980     -999
      1118         0  &clstime  &endtime      -980     -999
      1119         0  &clstime  &endtime      -980     -999
```

*CONTROL_FORMING_PARAMETER_READ

Purpose: This feature allows for reading of a numerical number from an existing file to store in a defined parameter. The parameter can be used and referenced in the current simulation. The file to be read may be a result from a previous simulation. The file may also simply contain a list of numbers defined beforehand and to be used for the current simulation.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

Parameter Cards. Include one card for each parameter. The next keyword ("*") card terminates the input.

Card 2	1	2	3	4	5	6	7	8
Variable	PARNAME	METHOD	LINE #	BEGIN	END			
Type	C	I	I	I	I			
Default	none	0	0	0	0			

VARIABLE	DESCRIPTION
FILENAME	Name of the file to be read.
PARNAME	Parameter name. Maximum character length is 7.
METHOD	Read instruction: EQ.1: read, follow definition by LINE #, BEGIN and END definition
LINE #	Line number in the file.
BEGIN	Beginning column number in the line number defined above.
END	Ending column number in the line number defined above.

Remarks:

1. **Input Order.** Keyword input order is sensitive. Recommended order is to define variables in *PARAMETER first, followed with this keyword, using the defined variables.

2. **Multiple Variables.** Multiple variables can be defined with one such keyword, with the file name needed to be defined only once. If there are variables located in multiple files, the keyword needs to be repeated for each file.

Examples:

An example provided below shows that multiple PIDs for individual tools and blank are defined in files data.k and data1.k. In the main input file, sim.dyn, used for LS-DYNA execution, variables (integer) are first initialized for PIDS of all tools and blank with *PARAMETER. These variables are updated with integers read from files data.k and data1.k from respective line number and column number through the use of this keyword. In the *SET_PART_LIST definition, these PIDs are used to define the part set.

Below is file data.k, to be read into sim.dyn:

```
$$$$$$$$$$$$$$$$$$$$$$$$
$$$ define PIDs
$$$$$$$$$$$$$$$$$$$$$$$$
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
upper die  pid:              3
lower post pid:              2
Below is file "data1.k", also to be read into "sim.dyn":
$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$
$$$ define PIDs
$$$$$$$$$$$$$$$$$$$$$$$$
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
lower binder pid:            4
blank pid:                   1
```

Below is partial input for the main input file sim.dyn:

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*INCLUDE
blank.k
*INCLUDE
tool.k
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*PARAMETER
Iblankp,0
Iupdiep,0
Ipunchp,0
Ilbindp,0
Rblankmv,0.0
Rpunchmv,0.0
Rupdiemv,0.0
```

```
Rbindmv,0.0
Rbthick,1.6
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*CONTROL_FORMING_PARAMETER_READ
data.k
updiep,1,5,30,30
punchp,1,6,30,30
*CONTROL_FORMING_PARAMETER_READ
data1.k
lbindp,1,7,30,30
blankp,1,8,30,30
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*SET_PART_LIST
1
&blankp
*SET_PART_LIST
2
&punchp
*SET_PART_LIST
3
&updiep
*SET_PART_LIST
4
&lbindp
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET
$#    psid       cid       dir     mpsid  position   premove    thick parname
         1         0         3         2         1     0.000  &bthick blankmv
         3         0         3         1         1     0.000          updiemv
         4         0         3         1        -1     0.000           bindmv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*PART_MOVE
$pid,xmov,ymov,zmov,cid,ifset
1,0.0,0.0,&blankmv,,1
3,0.0,0.0,&updiemv,,1
4,0.0,0.0,&bindmv,,1
```

Revision Information:

This feature is available in LS-DYNA R5 Revision 55035 and later releases.

*CONTROL_FORMING_POSITION

Purpose: This keyword allows user to position tools and a blank in setting up a stamping process simulation. All tools must be pre-positioned at their home positions. For tools that are positioned above the sheet blank (or below the blank) and ready for forming, *CONTROL_FORMING_TRAVEL should be used. This keyword is used together with *CONTROL_FORMING_USER. One *CONTROL_FORMING_POSITION card may be needed for each part.

> **NOTE:** This option has been deprecated in favor of *CONTROL_FORMING_AUTOPOSITION_PARAMETER).

Positioning Cards. For each part to be positioned include an additional card. The next "*" card terminates the input.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PREMOVE	TARGET					
Type	I	F	I					
Default	none	none	I					

VARIABLE	DESCRIPTION
PID	Part ID of a tool to be moved, as in *PART
PREMOV	The distance to pre-move the tool, in the reverse direction of forming.
TARGET	Target tool PID, as in *PART. The tool (PID) will be moved in the reverse direction of the forming and positioned to clear the interference with the blank, then traveled to its home position with a distance GAP (*CONTROL_FORMING_USER) away from the TARGET tool to complete the forming.

Remarks:

When this keyword is used, all stamping tools must be in their respective home positions, which is also the position of each tool at its maximum stroke. From the home position each tool will be moved to its start position, clearing interference between the blank and

tool yet maintaining the minimum separation needed to avoid initial penetration. Currently the tools can only be moved and travels in the direction of the global Z-axis.

A partial keyword example is provided in manual pages under *CONTROL_FORMING_USER.

Revision information:

This feature is available starting in Revision 24641.

*CONTROL_FORMING_PRE_BENDING_{OPTION}

Available options include:

<BLANK>

LOCAL

Purpose: Bend an initially flat sheet metal blank with a user-specified radius to control the blank's gravity loaded shape during sheet metal forming.

Card 1	1	2	3	4	5	6	7	8
Variable	PSET	RADIUS	VX	VY	VZ	XC	YC	ZC
Type	I	F	F	F	F	F	F	F
Default	none	none	none	none	none	↓	↓	↓

Local Coordinate System Card. This card is required if the option LOCAL is used.

Card 2	1	2	3	4	5	6	7	8
Variable	CID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PSET	Part set ID to be included in the pre-bending
RADIUS	Radius of the pre-bending: GT.0.0: bending center is on the same side as the element normal. LT.0.0: bending center is on the reverse side of the element normals. See Figure 12-41.

VARIABLE	DESCRIPTION
VX, VY, VZ	Vector components of an axis about which the flat blank will be bent
XC, YC, ZC	(X, Y, Z) coordinates of the center of most-bent location. If undefined, center of gravity of the blank will be used as a default.
CID	ID for a local coordinate system in which the blank will be bent

About Pre-Bending due to Gravity:

In some situations, gravity loading upon a flat blank will result in a "concave" shape in a die. This mostly happens in cases where there is little or no punch support in the middle of the die cavity and in large stamping dies. Although the gravity loaded blank shape is correct, the end result is undesirable. For example, buckles may result during the ensuing closing and forming simulations. In practice, a true flat blank rarely exists. Typically, the blank is either manipulated (shaking or bending) by die makers in the tryout stage or by suction cups in a stamping press to get an initial convex shape prior to the binder closing and punch forming. This keyword allows this bending to be performed.

Example 1:

A partial keyword example (NUMISHEET2002 fender outer) is provided below, where blank part set ID variable &BLKSID defined previously is to be bent with a radius value of -10000.0 mm with the Z-axis as the bending axis on the reverse side of the blank positive normal (see Figure 12-41). The bending is off the center of gravity at $(234.0, 161.0, 81.6)$ (to the right along positive X-axis). Only a slight pre-bending on the blank is needed to ensure a convex gravity-loaded shape. Note this keyword is input order sensitive and requires all model information to be placed ahead, therefore it is best to place the keyword at the end of an input deck.

```
*KEYWORD
⋮
*CONTROL_IMPLICIT_FORMING
1
*CONTROL_FORMING_PRE_BENDING
$     PSET    RADIUS        VX        VY        VZ        XC        YC        ZC
   &BLKSID   -10000.      0.00      0.00       1.0   234.000   161.000     81.60
...
*END
```

In Figures 12-42, initial blank shape without pre-bending is shown. Without pre-bending, the gravity loaded blank sags in the middle of the die cavity (see Figure 12-43) which is likely unrealistic and would lead to predictions of surface quality issues. With pre-bending applied (see Figure 12-44), the blank bends slightly into a convex shape before loading. This shape results in an overall convex shape after gravity completes loading

(see Figure 12-45), leading to a much shorter binder closing distance, and a more realistic surface quality assessment.

Example 2 - bending in a local coordinate system:

A partial keyword example below bends the blank (part set ID 5) in local coordinate system #8, about an axis 45 degrees from both the local *X*- and *Y*- axes, and perpendicular to the local *Z*-axis, in a radius of 1000 mm. The bending center is located in the negative surface normal side of the blank.

```
*KEYWORD
⋮
*DEFINE_COORDINATE_SYSTEM
$#     cid         xo         yo         zo         xl         yl         zl       cidl
         8    490.902    1.26663    98.7325      500.0    1.26663    98.7325          0
$#      xp         yp         zp
       0.0    71.9769    167.651
*CONTROL_FORMING_PRE_BENDING_LOCAL
$      PSET     RADIUS         VX         VY         VZ         XC         YC         ZC
         5    -1000.0       1.00       1.00
$       CID
         8
⋮
*END
```

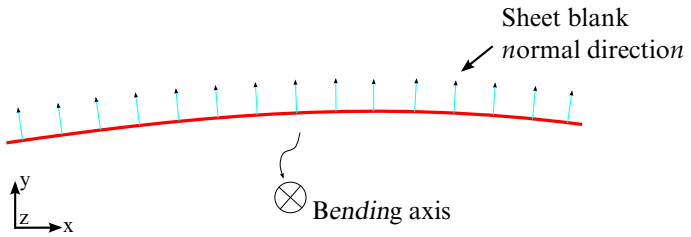


Figure 12-41. Negative "R" puts center of bending on the opposite side of the positive blank normal.

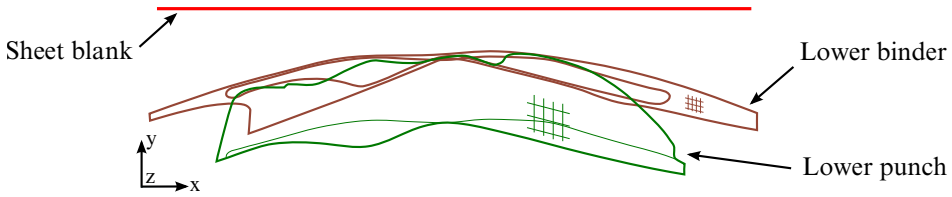


Figure 12-42. Initial model before auto-positioning.

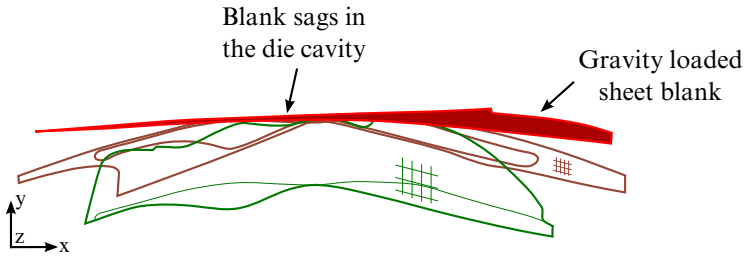


Figure 12-43. Gravity loaded blank without using this keyword.

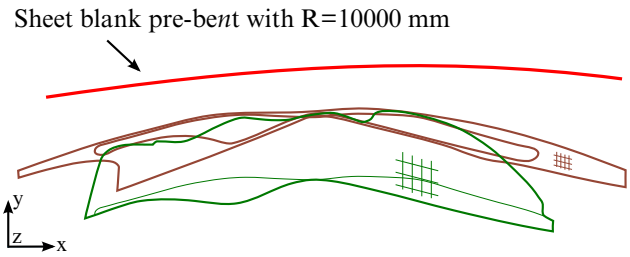


Figure 12-44. Pre-bending using this keyword (1st state in d3plot).

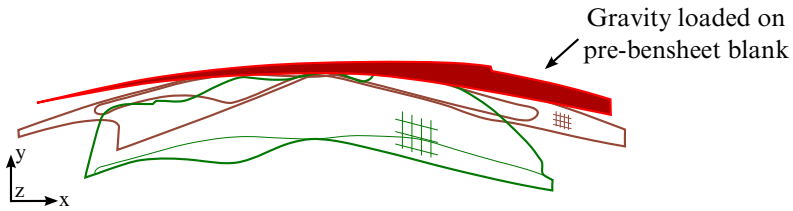


Figure 12-45. Gravity loaded shape (final state in d3plot) with convex shape.

*CONTROL_FORMING_PROJECTION

Purpose: Remove initial penetrations between the blank and the tooling (shell elements only) by projecting the penetrated blank nodes along a normal direction to the surface of the blank or to the surface of the tool with the specified gap between the node and the tooling surface. This is useful for line die simulations of a previously formed panel to reduce tool travel, therefore saving simulation time.

Define Projection Card. This card may not be repeated.

Card 1	1	2	3	4	5	6	7	8
Variable	PIDB	PIDT	GAP	NRBST	NRTST			
Type	I	I	F	I	I			

VARIABLE	DESCRIPTION
PIDB	Part ID of the blank
PIDT	Part ID for the tool
GAP	A distance, which defines the minimum gap required
NRBST	Specify whether the blank will move along its normal direction. If its moves along the normal of blank, then this flag also specifies the direction the normal is pointing with respect to the tool. EQ.0: Move the blank's nodes along the blank's normal. The normal to the surface of the blank is pointing towards the tool. EQ.1: Move the blank's nodes along the blank's normal. The normal to the surface of the blank is pointing away from the tool. EQ.2: Move the blank nodes along the tool's normal direction. This case is useful for contact between a guide pin and blank.
NRTST	Normal direction of the tool: EQ.0: The normal to the surface of the tool is pointing towards the blank. EQ.1: The normal to the surface of the tool is pointing away from blank.

Remarks:

This feature requires consistent normal vectors for both the rigid tooling surface and the blank surface.

*CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS

Purpose: Convert an adaptive mesh into a fully connected mesh. This feature removes adaptive constraints and connects the mesh with triangular elements.

Card 1	1	2	3	4	5	6	7	8
Variable	PID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PID	Part ID (see *PART) of the part whose adaptive mesh constraints are to be removed and its mesh converted into connected meshes.

Remarks:

In some applications of sheet metal forming, such as stoning or springback simulations, adaptive refinement of the sheet blank may affect the accuracy of the calculation. To avoid this problem, a non-adapted mesh is required. However, an adaptively refined mesh has the optimal mesh density that is tailored to the tooling geometry; the resulting mesh, in its initial shape, either flat or deformed, has fewer elements than a blank with non-adapted and uniformly-sized elements and thus is the most efficient for simulation. If the parameter IOFLAG in *CONTROL_ADAPTIVE is turned on, such a mesh adapt.msh will be generated at the end of each simulation, with its shape conforming to the initial input blank shape.

This keyword takes the adapted mesh, removes the adaptive constraints, and use triangular elements to connect the otherwise disconnected mesh. The resulting mesh is a fully connected mesh, with the optimal mesh density, to be used to re-run the simulation (without mesh adaptivity) for better accuracy.

Note that the original adapt.msh file from an LS-DYNA run will include not only the blank but the tooling mesh as well. For this keyword to be used, the original file can be read into LS-PrePost with the blank shown in active display only; the menu option *File* → *Save As* → *Save Active Keyword As* can be used to write out the adapted blank mesh only.

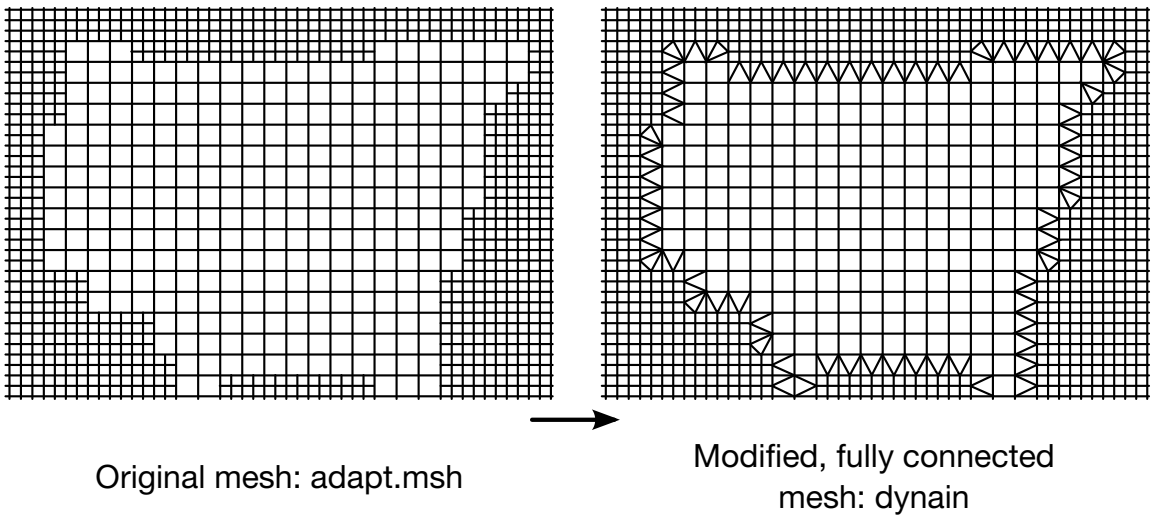


Figure 12-46. Converting an adaptive mesh to a fully connected mesh.

Example:

The following complete input file converts an adaptive mesh file blankadaptmsh.k (Figure 12-46 left) with the PID of 1 into a connected mesh (Figure 12-46 right). The resulting mesh will be in the dynain file.

```
*KEYWORD
*INCLUDE
blankadaptmsh.k
*PARAMETER
I blkpid           1
$--------1---------2-------
*CONTROL_TERMINATION
0.0
*CONTROL_FORMING_REMOVE_ADAPTIVE_CONSTRAINTS
$    PID
&blkpid
*set_part_list
1
&blkpid
*INTERFACE_SPRINGBACK_LSDYNA_NOTHICKNESS
1
*INTERFACE_SPRINGBACK_EXCLUDE
INITIAL_STRAIN_SHELL
INITIAL_STRESS_SHELL
*PART

$       PID       SID       MID
   &blkpid         1         1
*MAT_037
...
*SECTION_SHELL
$    SECID    ELFORM      SHRF       NIP
         1         2 0.000E+00         3
1.0,1.0,1.0,1.0
*END
```

*CONTROL_FORMING_SCALE

Purpose: Scale the blank in each direction. In the draw die design process, the draw die usually needs to be scaled so that the draw shell can fit on the tools in the subsequent process. This keyword provides a more efficient way to obtain an optimized scaling through scaling the draw shell for the following gravity simulations instead of scaling the draw die directly. Thus, performing the draw simulation ahead of time is not necessary, and you can try many different scales to find the optimized scale.

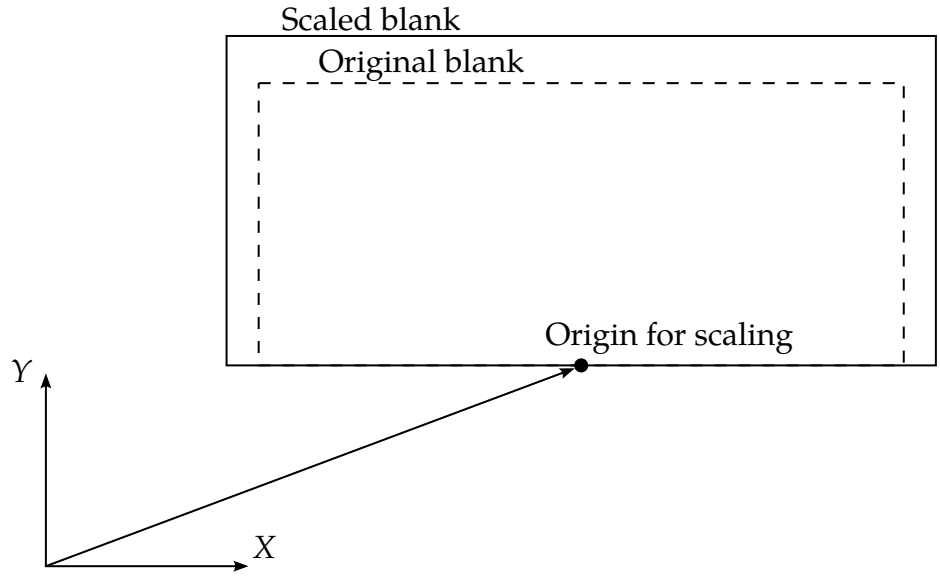


Figure 12-47. Example of a scaling.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	SCPSX	SCPSY	SCPSZ	SCPOX	SCPOY	SCPOZ	
Type	I	F	F	F	F	F	F	
Default	none	1.0	1.0	1.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
PSID	Part set ID of the blank
SCPSX	Scale factor in the *x*-direction. We recommend a value between 1.001 and 1.01.

VARIABLE	DESCRIPTION
SCPSY	Scale factor in the y-direction. We recommend a value between 1.001 and 1.01.
SCPSZ	Scale factor in the y-direction. We recommend a value between 1.001 and 1.01.
SCPOX	x-component of the origin for scaling the blank. The x-component of all physical points with this x-component remain fixed.
SCPOY	y-component of the origin for scaling the blank. The y-component of all physical points with this y-component remain fixed.
SCPOZ	z-component of the origin for scaling the blank.The z-component of all physical points with this z-component remain fixed.

*CONTROL_FORMING_SCRAP_FALL

Purpose: Direct and aerial trimming of a sheet metal part by trim steels in a trim die. According to the trim steels and trim vectors defined, the sheet metal part will be trimmed into a parent piece and multiple scrap pieces. The parent piece is defined as a fixed rigid body. Trimmed scraps (deformable shells) are constrained along trim edges until they come into contact with the trim steel; the edge constraints are gradually released as the trim steel's edge contacts the scrap piece, allowing for simulating contact-based scrap fall. This keyword applies to shell elements only.

Card Sets. Include Card 1 columns 1-6 only per each scrap piece for the *constraint release method* (see Remarks). For the *scrap trimming method* include one set of Cards 1, 2, and 3 per trim steel. The next keyword ("*") card terminates the input.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	VECTID	NDSET	LCID	DEPTH	DIST	IDRGD	IFSEED
Type	I	I	I	I	F	F	I	I
Default	none	global z	none	none	none	none	none	none

Card 2	1	2	3	4	5	6	7	8
Variable	NDBEAD	SEEDX	SEEDY	SEEDZ	EFFSET	GAP	IPSET	EXTEND
Type	I	F	F	F	F	F	I	F
Default	none	↓	↓	↓	none	none	none	none

Card 3	1	2	3	4	5	6	7	8
Variable	NEWID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PID	Part ID of a scrap piece. This part ID becomes a dummy ID if all trimmed scrap pieces are defined by NEWID. See definition for NEWID and Figure 12-50.
VECTID	Vector ID for a trim steel movement, as defined by *DEFINE_VECTOR. If left undefined (blank), global *z*-direction is assumed.
NDSET	A node set consisting of all nodes along the cutting edge of the trim steel. Note that prior to Revision 90339 the nodes in the set must be defined in consecutive order. See Remarks (LS-PrePost) below on how to define a node set along a path in LS-PrePost. This node set, together with VECTID, is projected to the sheet metal to form a trim curve. To trim a scrap out of a parent piece involving a neighboring trim steel, which also serves as a scrap cutter, the node set needs to be defined for the scrap cutter portion only for the scrap, see Figure 12-50.
LCID	Load curve ID governing the trim steel kinematics, as defined by *DEFINE_CURVE. GT.0: velocity-controlled kinematics LT.0: displacement-controlled kinematics Example input decks are provided below.
DEPTH	A small penetrating distance between the cutting edge of the trim steel and the scrap piece, as shown in Figure 12-49. Nodes along the scrap edge are released from automatically added constraints at the simulation start and are free to move after this distance is reached.
DIST	A distance tolerance measured in the plane normal to the trim steel moving direction, between nodes along the cutting edge of the trim steel defined by NDSET and nodes along an edge of the scrap, as shown in Figure 12-48. This tolerance is used to determine if the constraints need to be added at the simulation start to the nodes along the trim edge of the scrap piece.
IDRGD	Part ID of a parent piece, which is the remaining sheet metal after the scrap is successfully trimmed out of a large sheet metal. Note the usual *PART needs to be defined somewhere in the input deck, along with *MAT_20 and totally fixed translational and rotational DOFs. See Figure 12-50.

VARIABLE	DESCRIPTION
IFSEED	A flag to indicate the location of the scrap piece. EQ.0: automatically determined. The trim steel defined will be responsible to trim as well as to push (have contact with) the scrap piece. EQ.1: automatically determined; however, the trim steel in definition will only be used to trim out the scrap, not to push (have contact with) the scrap piece. EQ.-1: user specified by defining SEEDX, SEEDY, and SEEDZ
NDBEAD	A node set to be excluded from initially imposed constraints after trimming. This node set typically consists of nodes in the scrap draw bead region where due to modeling problems the beads on the scrap initially interfere with the beads on the rigid tooling; it causes scrap to get stuck later in the simulation if left as is. See Figure 12-51.
SEEDX, SEEDY, SEEDZ	x, y, z coordinates of the seed node on the scrap side; define only when IFSEED is set to "-1". See Figure 12-50.
EFFSET	Scrap edge offset amount away from the trim steel edge, towards the scrap seed node side. This is useful to remove initial interference between the trimmed scrap (because of poorly modeled trim steel) and coarsely modeled lower trim post. See Figure 12-50.
GAP	Scrap piece offset amount from the part set defined by IPSET (e.g. top surfaces of the scrap cutters), in the direction of the element normals of the IPSET. This parameter makes it easier to remove initial interference between the scrap and other die components. See Figure 12-53.
IPSET	A part set ID from which the scrap will be offset to remove the initial interference; works together only with GAP. The part set ID should only include portions of tool parts that are directly underneath the scrap (top surface portion of the tools). The normals of the IPSET must point toward the scrap. The parts that should belong to IPSET are typically of those elements on the top surface of the scrap cutter; see Figure 12-53.
EXTEND	An amount to extend a trim steel's edge based on the NDSET defined, so it can form a continuous trim line together with a

VARIABLE	DESCRIPTION
	neighboring trim steel, whose edge may also be extended, to trim out the scrap piece. See Figure 12-50.
NEWID	New part ID of a scrap piece for the scrap area defined by the seed location. If this is not defined (left blank) or input as "0", the scrap piece will retain original PID as its part ID. See Figure 12-50. This is useful when one original scrap is trimmed into multiple smaller pieces, and contacts between these smaller pieces need to be defined.

Background:

Sheet metal trimming and the resulting scrap fall are top factors in affecting the efficiency of stamping plants worldwide. Difficult trimming conditions, such as those multiple direct trims, a mixture of direct and cam trims, and multiple cam trims involving bypass condition, can cause trimmed scraps to get stuck around and never separate from the trim edge of the upper trim steels or lower trim post. Inappropriate design of die structure and scrap chute can slow down or prevent scraps from tumbling out to the scrap collectors. Smaller scrap pieces (especially aluminum) can sometimes shoot straight up, and get stuck and gather in areas of the die structure. All these problems result in shutdowns of stamping presses, reducing stroke-per-minute (SPM) and causing hundreds of thousands of dollars in lost productivity.

With this keyword, engineers can consider the trimming details, manage the scrap trim and the drop energy, study different trimming sequences, explore better die structure and scrap chutes design and layout before a trim die is even built. This feature is developed in conjunction with the *Ford Motor Company*.

The constraint release method:

Prior to Revision 91471 (see Revision), simulating the scrap trim and fall uses the "constraint release" method, where only the scrap piece is modeled and defined.

As shown in Figure 12-48, the scrap piece is modeled as a deformable body and the trim steel and trim post as rigid shell elements, while the parent piece does not need to be modeled at all. Between the trim edge of the scrap piece and the post there should be a gap (indicated by GAP in the figure). The gap ensures that the contact interface (to be explained later) correctly accounts for the shell thickness along the edge. A gap that is too small may cause initial penetration between the scrap and the post which may manifest as unphysical adhesion between the scrap and the post.

The edge of the scrap piece should initially be flush with that of the trim post (perpendicular to the trim direction), just as exactly what happens in the production environment.

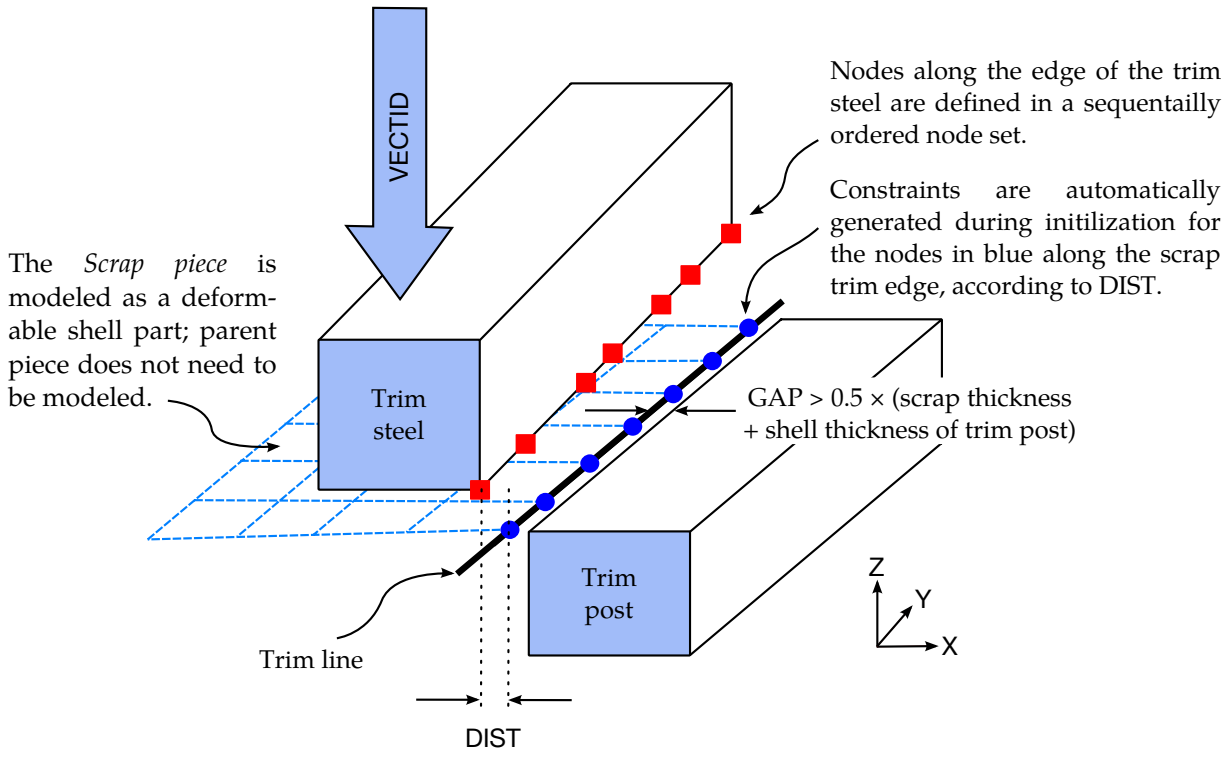


Figure 12-48. Modeling details of the constraint release method. *Drawing modified from the original sketches courtesy of the Ford Motor Company.*

If the scrap is unrealistically positioned above the trim post edge, the scrap may be permanently caught between the trim steel and the post under a combination of uncertain trimming forces as the trim steel moves down.

During initialization, constraints are added automatically on the nodes along the scrap trim edge corresponding to the node set (NDSET) along the trim steel, based on the supplied tolerance variable DIST and trim vector VECTID. Although the direction of the path is not important, prior to Revision 90339, the NDSET must be arranged so that the nodes are in a sequential order (*LS-PrePost 4.0* creating node set by *path*). As the edge of the trim steel comes within DEPTH distance of the trim line, the constraints are removed. The contact interfaces serve to project the motion of the trim steel onto the scrap piece, see Figure 12-49.

The scrap trimming method:

The original simplified method has the following drawbacks:

1. No scrap trimming – the scrap piece cannot be trimmed directly from a parent piece; an exact scrap piece after trimming must be modeled.

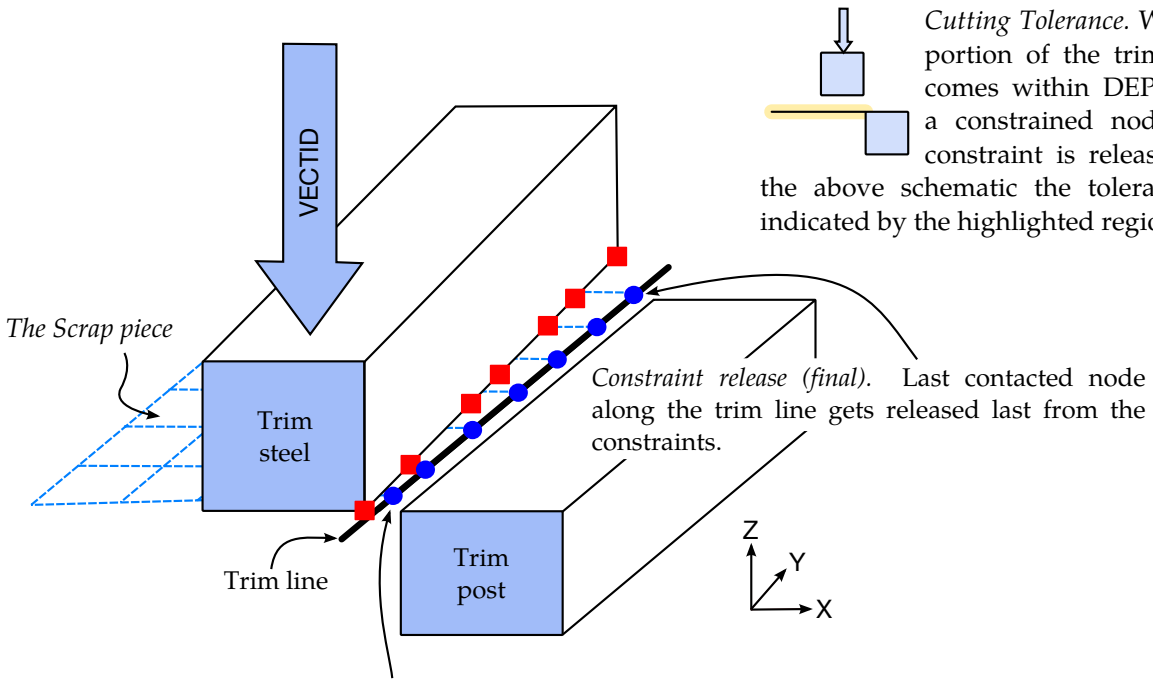


Figure 12-49. Contact-based separation and contact-driven kinematics and dynamics in the constraint release method. *Drawing modified from the original sketches courtesy of the Ford Motor Company.*

2. Poorly (or coarsely) modeled draw beads in the scrap piece do not fit properly in badly modeled draw beads on the tooling, resulting in initial interferences between the two and therefore affecting the simulation results.

3. For poorly (or coarsely) modeled scrap edges and trim posts, users have to manually modify the scrap trim edges to clear the initial interference with the trim posts.

4. Users must clear all other initial interferences (e.g. between scrap and scrap cutter) manually.

Based on users' feedback, a new method "scrap trimming" (after Revision 91471) has been developed to address the above issues and to, furthermore, reduce the effort involved in preparing the model. The new method (Figure 12-50) involves trimming scrap from an initially large piece of sheet metal, leaving the parent piece as a new fixed rigid body, defined by user (along with *SECTION_SHELL and *MAT_20). If one large piece of sheet metal is trimmed in many different places to form several scrap pieces, a new PID can be defined for each of the scrap pieces (NEWID), along with *SECTION_SHELL and *MAT_037, for example. The trim lines are obtained from the trim steel edge node set NDSET and the trim vector VECTID.

Note: The trim steel cannot have walls that are parallel to the trim direction; otherwise the scrap will not be trimmed properly from the parent piece. This is illustrated in Figure 12-52.

Parameters related to the constraint release method:

1. The value of DEPTH is typically set to one-half of the scrap thickness.

2. The initial gap separating the scrap from the post must be greater than the average of the scrap and post thickness values; see Figure 12-48.

3. The input parameter DIST should be set larger than the maximum distance between nodes along the trim steel edge and scrap edge in the view along the trim direction; see Figure 12-48.

Parameters related to the scrap trimming method:

1. Similar to DEPTH, EFFSET should be typically set to one-half of the scrap thickness, although it may be larger for some poorly modeled trim steels and trim posts.

Contact:

Only *CONTACT_FORMING contact interfaces are allowed for contact between the scrap piece and the trim steel. In particular, *CONTACT_FORMING_SURFACE_TO_SURFACE is recommended. A negative contact offset must be used; this is done typically by setting the variable SBST in *CONTACT_FORMING_SURFACE_TO_SURFACE to the negative thickness value of the scrap piece.

For contact between the scrap piece and the shell elements in all the other die structures, *CONTACT_AUTOMATIC_GENERAL should be used for the edge-to-edge contact frequently encountered during the fall of the scrap piece. All friction coefficients should be small. The explicit time integrator is recommended for the modeling of scrap trim and fall. Mass scaling is not recommended.

LS-PrePost:

The node set (NDSET) defined along the trim steel edge can be created with *LS-PrePost 4.0*, via *Model/CreEnt/Cre, Set Data, *SET_NODE, ByPath*, then select nodes along the trim edge continuously until finish and then hit *Apply*.

Keyword examples – the constraint release method:

A partial example of using the keyword below includes a node set ID 9991 along the trim steel (PID 2) edge used to release the constraints between the scrap piece with PID 1, and the parent piece. The LCID for the trim steel kinematics is (+)33 (load curve is controlled by velocity) moving in the $-z$ direction. The trimming velocity is defined as 1000 mm/s, and the retracting velocity is 4000 mm/s. The variables DEPTH and DIST are set to 0.01 and 2.5, respectively. The contact interface between the trim steel and scrap piece is defined using *CONTACT_FORMING_SURFACE_TO_SURFACE and contact between the scrap and all other die structures are defined using *CONTACT_AUTOMATIC_GENERAL.

```
*KEYWORD
*CONTROL_TERMINATION
&endtime
*CONTROL_FORMING_SCRAP_FALL
$       PID    VECTID     NDSET      LCID     DEPTH      DIST
          1                 9991        33      0.75      2.0
*SET_NODE_LIST
       9991
     24592     24591     24590     24589     24593     24594     24595     24596
*BOUNDARY_PRESCRIBED_MOTION_rigid
$pid,dof,vad,lcid,sf,vid,dt,bt
2,3,0,33,-1.0
*DEFINE_CURVE
33
0.0,0.0
0.216,1000.0
0.31,-4000.0
0.32,0.0
0.5,0.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTACT_forming_surface_to_surface_ID
          1
          1          2          3          3          0          0          0          0
       0.02        0.0        0.0        0.0       20.0          0        0.01.0000E+20
$#     sfsa       sfsb       sast       sbst      sfsat     sfsbt       fsf        vsf
        0.0        0.0        0.0      &sbst        1.0       1.0        1.0        1.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTACT_AUTOMATIC_GENERAL_ID
          2

*END
```

For the negative option of LCID, displacement will be used as input to control the tool kinematics. A partial example is provided below, where LCID is defined as a negative integer of a load curve, controlling the trim steel kinematics. The trim steel is moving down for 27.6075 mm in 0.2 sec to trim, and moving up for the same distance to its original position in 0.3 sec to retract. Although this option is easier to use, the corresponding velocity from the input time and displacement must be realistic for a realistic simulation.

```
*CONTROL_FORMING_SCRAP_FALL
$ LCID<0: trimming steel kinematics is controlled by displacement.
$       PID    VECTID     NDSET      LCID     DEPTH      DIST
          1         44          1    -33332      0.70      2.00
```

```
*DEFINE_VECTOR
44,587.5,422.093,733.083,471.104,380.456,681.412
*BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL
$pid,dof,vad,lcid,sf,vid,dt,bt
11,3,2,33332,1.0,44
*DEFINE_CURVE
33332
0.0,0.0
0.2,-27.6075
0.5,0.0
```

A keyword example – the scrap trimming method:

The keyword example below shows three scrap pieces, with part ID &spid1 (PID, also is the original blank to be trimmed) and new part IDs 1001 and 1002 (NEWID), being trimmed out of the original blank &spid1 (PID); the remaining parent piece is defined as a fixed rigid body with part ID 110 (IDRGD). Each scrap piece is indicated by a point location defined by SEEDX, SEEDY and SEEDZ. Each scrap is also to be trimmed by three different trim steels. For example, scrap piece &spid1 is to be trimmed by trim steels with part IDs 22, 23 and 24 (as referenced by *BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL) in trim directions defined respectively by vector IDs (from VID in *DEFINE_VECTOR) &cord1, &cord2 and &cord3 (VECTID) and with trim kinematics defined respectively by load curve IDs 1800, 1801 and 1802 (LCID). The scrap pieces &spid1, 1001 and 1002 are each offset by 0.60mm (GAP), in the direction normal to the elements defined by part set IDs 887, 888, and 889 (IPSET), respectively. The trim edge offset is 0.90mm (EFFSET) away from the trim edge for all scraps. The draw bead node sets to be released are, 987, 988 and 989 for each scrap &spid1, 1001, and 1002, respectively.

```
*CONTROL_FORMING_SCRAP_FALL
$ The next three cards define scrap piece &spid1 being trimming simultaneously
$  with three trim steels with VLCIDs of 1800, 1801 and 1802:
$       PID     VECTID      NDSET       LCID      DEPTH       DIST      IDRGD     IFSEED
     &spid1     &cord1     &nset1       1800     &depth1       2.00        110         -1
$    NDBEAD      SEEDX      SEEDY      SEEDZ     EFFSET        GAP      IPSET     EXTEND
        987   -528.046    373.40    710.000       0.90       0.60        887        8.0
$     NEWID
          0
     &spid1     &cord2     &nset2       1801     &depth1       2.00        110         -1
        987   -528.046    373.40    710.000       0.90       0.60        887        8.0
          0
     &spid1     &cord3     &nset3       1802     &depth1       2.00        110         -1
        987   -528.046    373.40    710.000       0.90       0.60        887        8.0
          0
$ The next three cards define scrap piece 1001 being trimming simultaneously
$  with three trim steels with VLCIDs of 1802, 1803 and 1804:
     &spid1     &cord3    &nset33       1802     &depth1       2.00        110         -1
        988   -252.452    383.322    799.974       0.90       0.60        888        8.0
       1001
     &spid1     &cord4     &nset4       1803     &depth1       2.00        110         -1
        988   -252.452    383.322    799.974       0.90       0.60        888        8.0
       1001
     &spid1     &cord5     &nset5       1804     &depth1       2.00        110         -1
        988   -252.452    383.322    799.974       0.90       0.60        888        8.0
       1001
$ The next three cards define scrap piece 1002 being trimming simultaneously
$  with three trim steels with VLCIDs of 1804, 1805 and 1806:
```

```
    &spid1    &cord5   &nset55      1804   &depth1      2.00       110        -1
       989    74.452   404.522   857.974      0.90      0.60       889       8.0
      1002
    &spid1    &cord6    &nset6      1805   &depth1      2.00       110        -1
       989    74.452   404.522   857.974      0.90      0.60       889       8.0
      1002
    &spid1    &cord7    &nset7      1806   &depth1      2.00       110        -1
       989    74.452   404.522   857.974      0.90      0.60       889       8.0
      1002
*BOUNDARY_PRESCRIBED_MOTION_rigid
$      PID       DOF       VAD      LCID        SF       VID        DT        BT
        22         3         0      1800      -1.0    &cord1
*DEFINE_CURVE
1800
0.0,0.0
0.216,1000.0
0.31,-4000.0
0.32,0.0
1.0,0.0
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH
    &cord1     587.5   422.093   733.083   471.104   380.456   681.412
*BOUNDARY_PRESCRIBED_MOTION_rigid
$      PID       DOF       VAD      LCID        SF       VID        DT        BT
        23         3         0      1801      -1.0    &cord2
*DEFINE_CURVE
1801
0.0,0.0
0.326,1000.0
0.44,-4000.0
0.60,0.0
1.0,0.0
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH
    &cord2     587.5     522.0     763.0    10.104      80.6       1.1
*BOUNDARY_PRESCRIBED_MOTION_rigid
$      PID       DOF       VAD      LCID        SF       VID        DT        BT
        24         3         0      1802      -1.0    &cord3
*DEFINE_CURVE
1802
0.0,0.0
0.36,1000.0
0.42,-4000.0
0.64,0.0
1.0,0.0
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH
    &cord3      85.5     42.93      3.93       7.4      30.4      41.2
```

Revision/Other information:

A graphical user interface capable of setting up a complete input deck for the original simplified method is now available in *LS-PrePost* under *APPLICATION/Scrap Trim*. A reference paper regarding the development and application of this keyword for the constraint release method can be found in the *proceedings of the 12th International LS-DYNA User's Conference*. The following provides a list of revision history for the keyword:

1. The *constraint release method* is available between LS-DYNA Revision 63618 and 91471.

2. The *scrap trimming method* is available starting in Revision 91471.

3. The parameter NEWID is available starting in Revision 92578.

The restriction that NDSET must be defined in a consecutive order is lifted starting in Revision 90339.

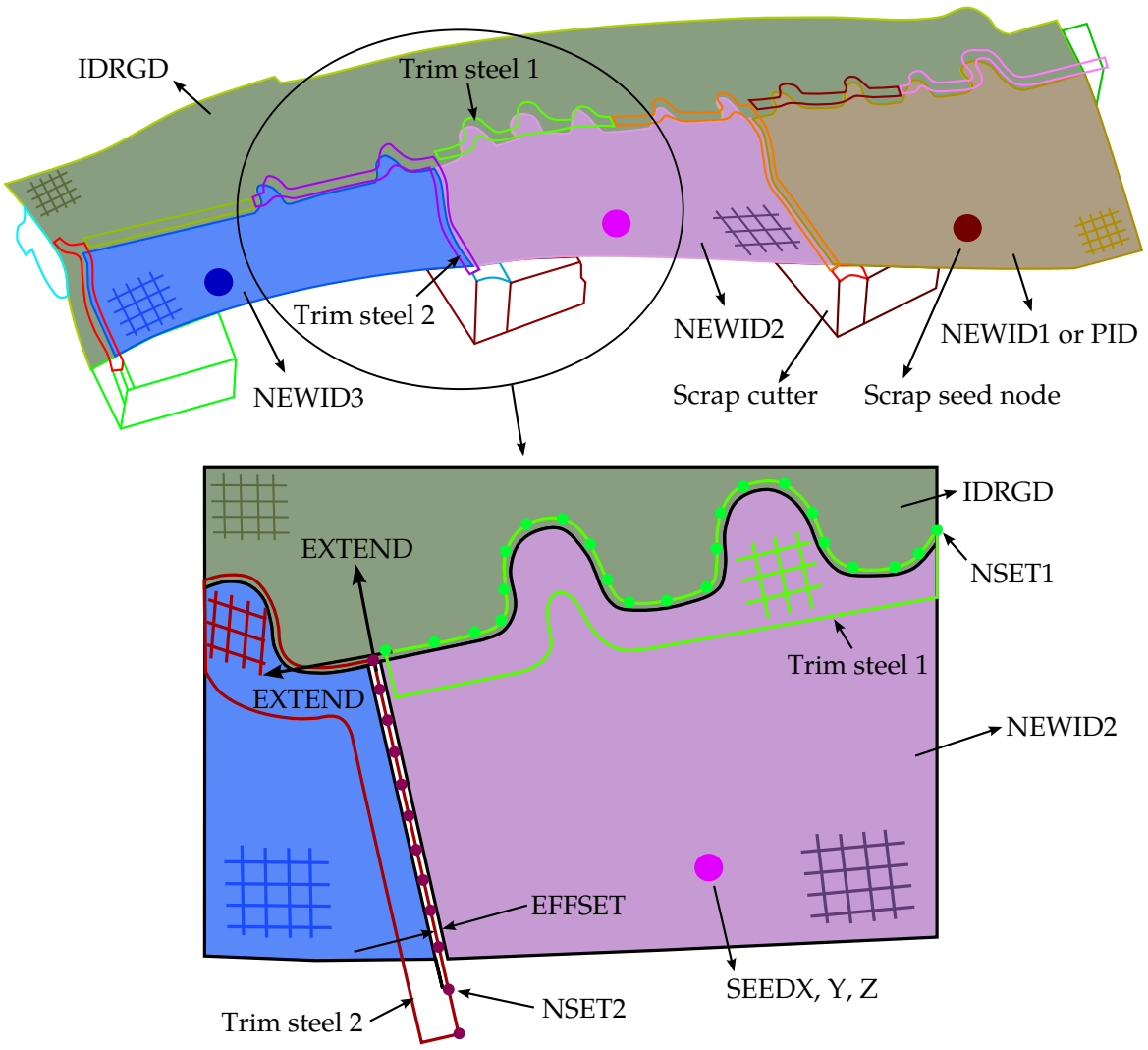


Figure 12-50. Trimming of multiple scraps and parameter definitions in the scrap trimming method. *Model courtesy of the Ford Motor Company.*

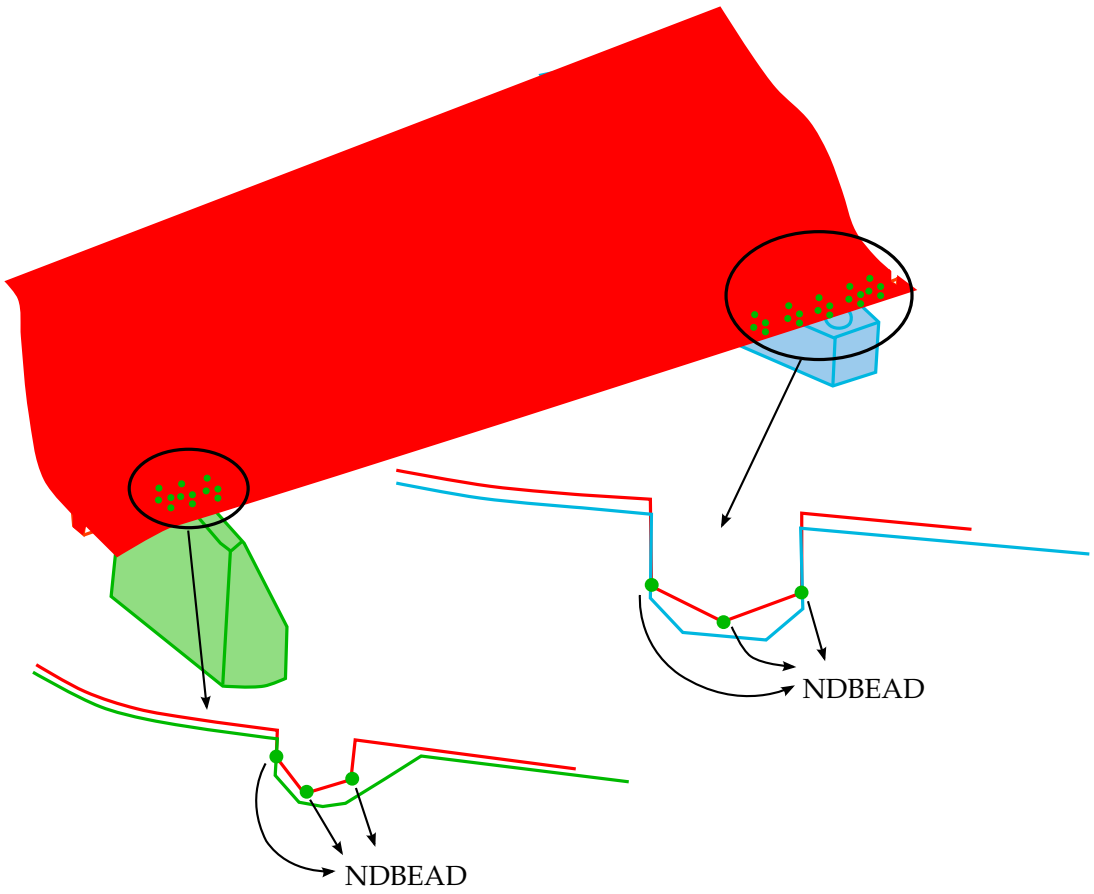


Figure 12-51. Definition of NDBEAD in the scrap trimming method. *Model courtesy of the Ford Motor Company.*

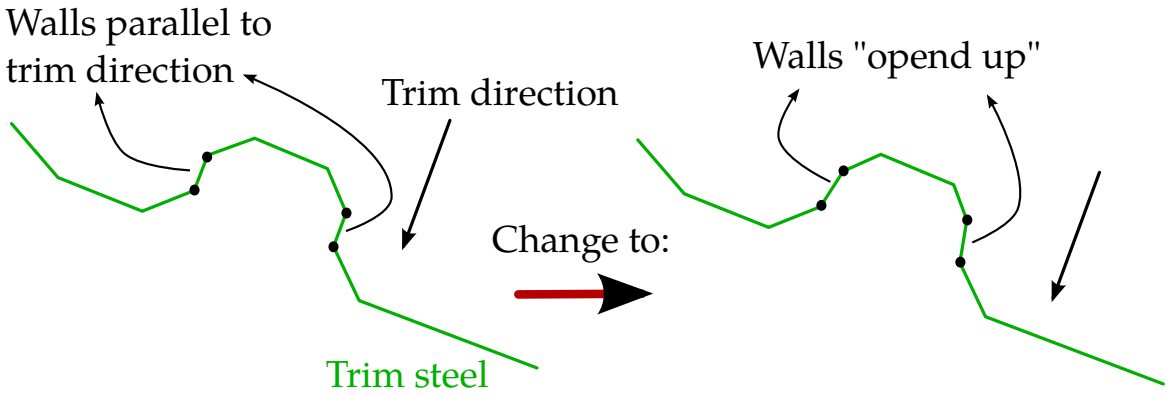


Figure 12-52. Walls that are parallel to trim direction anywhere in a trim steel (left) are not allowed. Wall needs to be opened up in relation to the trim direction for a successful simulation (right).

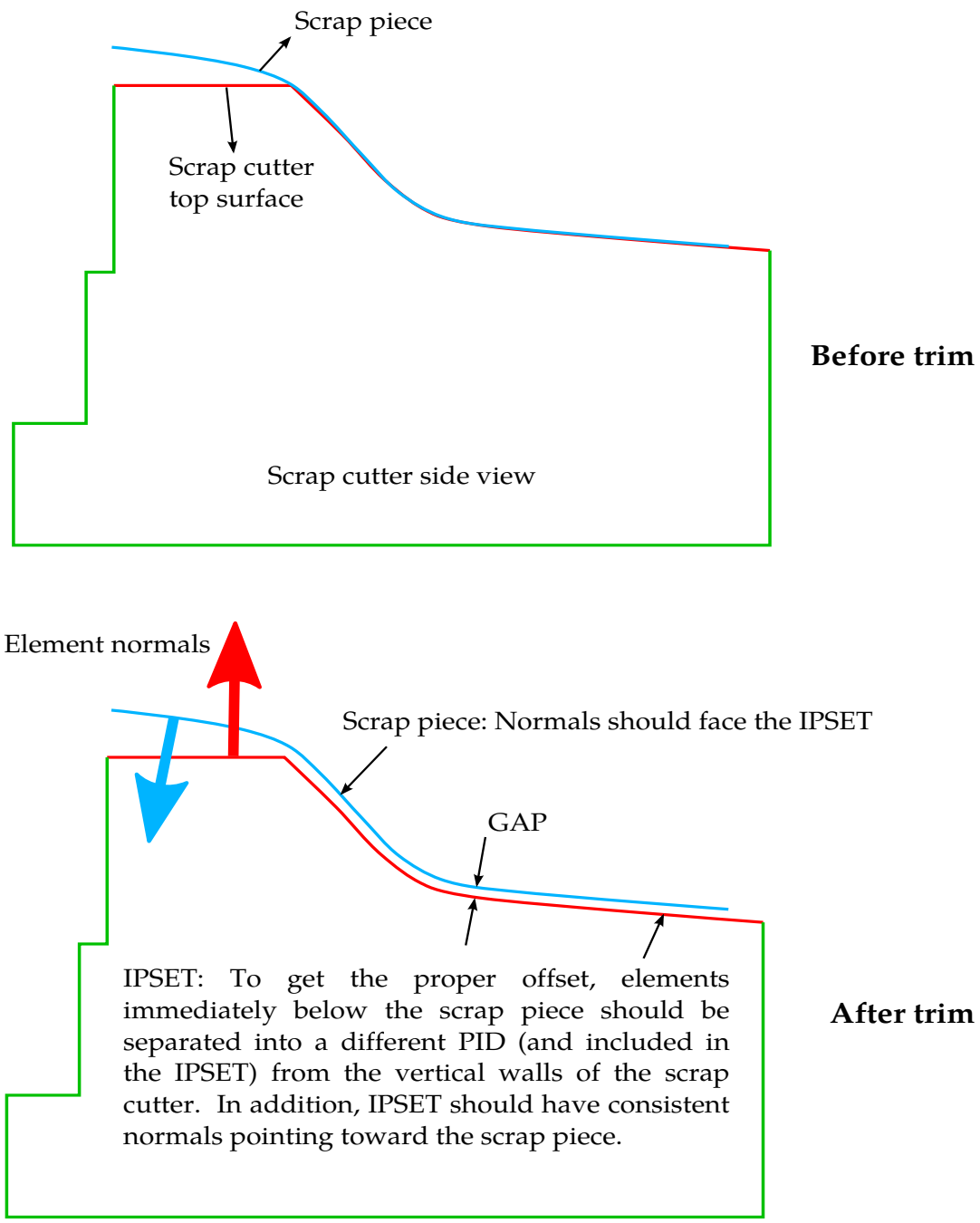


Figure 12-53. Element normal of the IPSET in the scrap trimming method. *Model courtesy of the Ford Motor Company.*

*CONTROL_FORMING_SHELL_TO_TSHELL

Purpose: This keyword converts thin shell elements (*SECTION_SHELL) to thick shell elements (*SECTION_TSHELL). It generates segments on both the top and bottom sides of the thick shells for contact. Mesh adaptivity is also supported.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	THICK	MIDSF	IDSEGB	IDSEGT			
Type	I	I	I	I	I			
Default	none	none	0	Rem 4	Rem 4			

VARIABLE	DESCRIPTION
PID	Part ID of the thin shell elements.
THICK	Thickness of the thick shell elements (TSHELL).
MIDSF	TSHELL's mid-plane position definition (see Figure 12-54 and Remark 4): EQ.0: Mid-plane is at thin shell surface. EQ.1: Mid-plane is at one half of THICK above thin shell surface. EQ.-1: Mid-plane is at one half of THICK below thin shell surface.
IDSEGB	Set ID of the segments to be generated at the bottom layer of the TSHELLs, which can be used for defining a contact surface with segments. The bottom layer of the TSHELLs has an outward normal that points in the opposite direction to the positive normal side of thin shells; see Figure 12-54. Note the default normal of the generated segments are consistent with the thin shells' normal. To reverse this default normal, set the IDSEGB to a negative number. See Remark 2.
IDSEGT	Set ID of the segments to be generated at the top layer of the TSHELLs, which can be used for defining a contact surface with segments. The top side of a TSHELL has an outward normal that points in the same direction as the positive normal side of the thin shells; see Figure 12-54.

VARIABLE	DESCRIPTION
	The normal of the generated segments are consistent with the thin shells' normal. To reverse this default normal, set the IDSEGT to a negative number. See Remark 2.

Remarks:

1. **Node and Element Numbering**. Node IDs of the thick shell elements will be the same as those for the thin shells. Element IDs of the thick shell elements will start at 2 (so renumber element IDs of other parts accordingly). Only one layer of thick shells will be created.

2. **Contact.** For FORMING contacts, if the generated segments are used as a SURFA member in contact with a SURFB member of a rigid body, the normal vectors for the rigid body must be consistent and facing the SURFA segments. The normal vectors for the SURFA segments are not required to point at the rigid bodies, although they should be made consistent.

3. **Adaptivity.** New nodes generated adaptively from their parent nodes with *BOUNDARY_SPC are automatically constrained accordingly.

4. **Contact and Segment Sets.** If no contact for the TSHELLs is defined in the input deck, then the fields IDSEGB and IDSEGT are ignored and the field MIDSF will be set to 0, regardless of the user set value. See the first example.

Examples:

1. A standalone part of thin shell elements can be changed to thick shell elements with a simplified small input deck. The following will convert shell elements with PID 100 of thickness 1.5 mm to thick shell elements of PID 100 with thickness of 2.0 mm, with thick shell meshes stored in the file dynain.geo. Note that MIDSF, IDSEGB and IDSEGT are ignored in this case and do not need to be set.

```
*KEYWORD
*CONTROL_TERMINATION
0.0
*INCLUDE
shellupr.k
*SET_PART_LIST
1,
100
*PART
Sheet blank
100,100,100
*SECTION_SHELL
$      SID    ELFORM      SHRF       NIP     PROPT
       100         2     0.833         3       1.0
$       T1        T2        T3        T4      NLOC
1.5,1.5,1.5,1.5
```

```
*MAT_024
$      MID        RO         E        PR      SIGY      ETAN      FAIL      TDEL
       100  7.85E-09  2.07E+05      0.28     382.8       0.0       0.0       0.0
$        C         P      LCSS      LCSR        VP
       0.0       0.0        92         0       0.0
$     EPS1      EPS2      EPS3      EPS4      EPS5      EPS6      EPS7      EPS8

$      ES1       ES2       ES3       ES4       ES5       ES6       ES7       ES8

*DEFINE_CURVE
92,,,0.5
    0.0000000000E+00    3.8276000000E+02
    4.0000000000E-03    3.9616000000E+02
    8.0000000000E-03    4.0695000000E+02
          ⋮                   ⋮
*DEFINE_CURVE
     90905
     0.000000000E+00     0.380000000E+03
     0.300000003E-02     0.392489226E+03
     0.600000005E-02     0.403294737E+03
     0.899999961E-02     0.412847886E+03
     0.120000001E-01     0.421429900E+03
     0.150000006E-01     0.429234916E+03
     0.179999992E-01     0.436402911E+03
     0.209999997E-01     0.443038343E+03
*INTERFACE_SPRINGBACK_LSDYNA
1
OPTCARD,,,1
*CONTROL_FORMING_SHELL_TO_TSHELL
$      PID     THICK
       100       2.0
*END
```

2. The conversion can also be done in an input deck set up for a complete metal forming simulation with thin shell elements that are part of a sheet blank. The conversion happens at the beginning of the simulation, as shown in an example below. This partial deck is of a thin shell sheet blank with PID 1 that is to be converted to a thick shell sheet blank with thickness of 1.6 mm. The commented out keywords were originally for the thin shells but have been changed due to the conversion. For example, the *SECTION_TSHELL is defined instead of *SECTION_SHELL for the sheet blank. The corresponding material type for the sheet blank (*MAT_037) is also changed to a type that supports solid element simulations (*MAT_024). The mid-plane of the thick shells is one half of 1.6 mm below the thin shells' surface. Segment IDs 10 (IDSEGB) and 11 (IDSEGT) are created at the bottom and top side of the thick shells, respectively, as shown in Figure 12-54. Segment set 10 is defined to contact with the lower punch (part set ID 2), while segment set 11 is used for contact with the upper die cavity (part set ID 3).

```
*KEYWORD
...
*PART
Sheet blank
1,1,1
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$                    Blank property
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$*SECTION_SHELL
```

```
$       SID     elform      SHRF       nip     PROPT    QR/IRID     ICOMP     SETYP
$         1         2     0.833         3       1.0
$        T1        T2        T3        T4      NLOC
$&b1thick,&b1thick,&b1thick,&b1thick
*SECTION_TSHELL
$       SID     elform      SHRF       nip     PROPT    QR/IRID     ICOMP     SETYP
          1         1     0.833      &nip       1.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*MAT_024
$       MID        RO         E        PR      SIGY      ETAN      FAIL      TDEL
          1  7.85E-09  2.07E+05      0.28     382.8       0.0       0.0       0.0
$         C         P      LCSS      LCSR        VP
        0.0       0.0        92         0       0.0
$      EPS1      EPS2      EPS3      EPS4      EPS5      EPS6      EPS7      EPS8

$       ES1       ES2       ES3       ES4       ES5       ES6       ES7       ES8

*DEFINE_CURVE
92,,,0.5
    0.0000000000E+00    3.8276000000E+02
    4.0000000000E-03    3.9616000000E+02
    8.0000000000E-03    4.0695000000E+02
           ⋮                   ⋮
*INTERFACE_SPRINGBACK_LSDYNA
1
OPTCARD,,,1
*CONTROL_FORMING_SHELL_TO_TSHELL
$       PID     THICK     MIDSF    IDSEGB    IDSEGT
        100       1.6        -1        10        11
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE
$     SURFA     SURFB  SURFATYP  SURFBTYP
         11         2         0         2
          ⋮         ⋮         ⋮
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE
$     SURFA     SURFB  SURFATYP  SURFBTYP
         10         3         0         2
          ⋮         ⋮         ⋮
...
*END
```

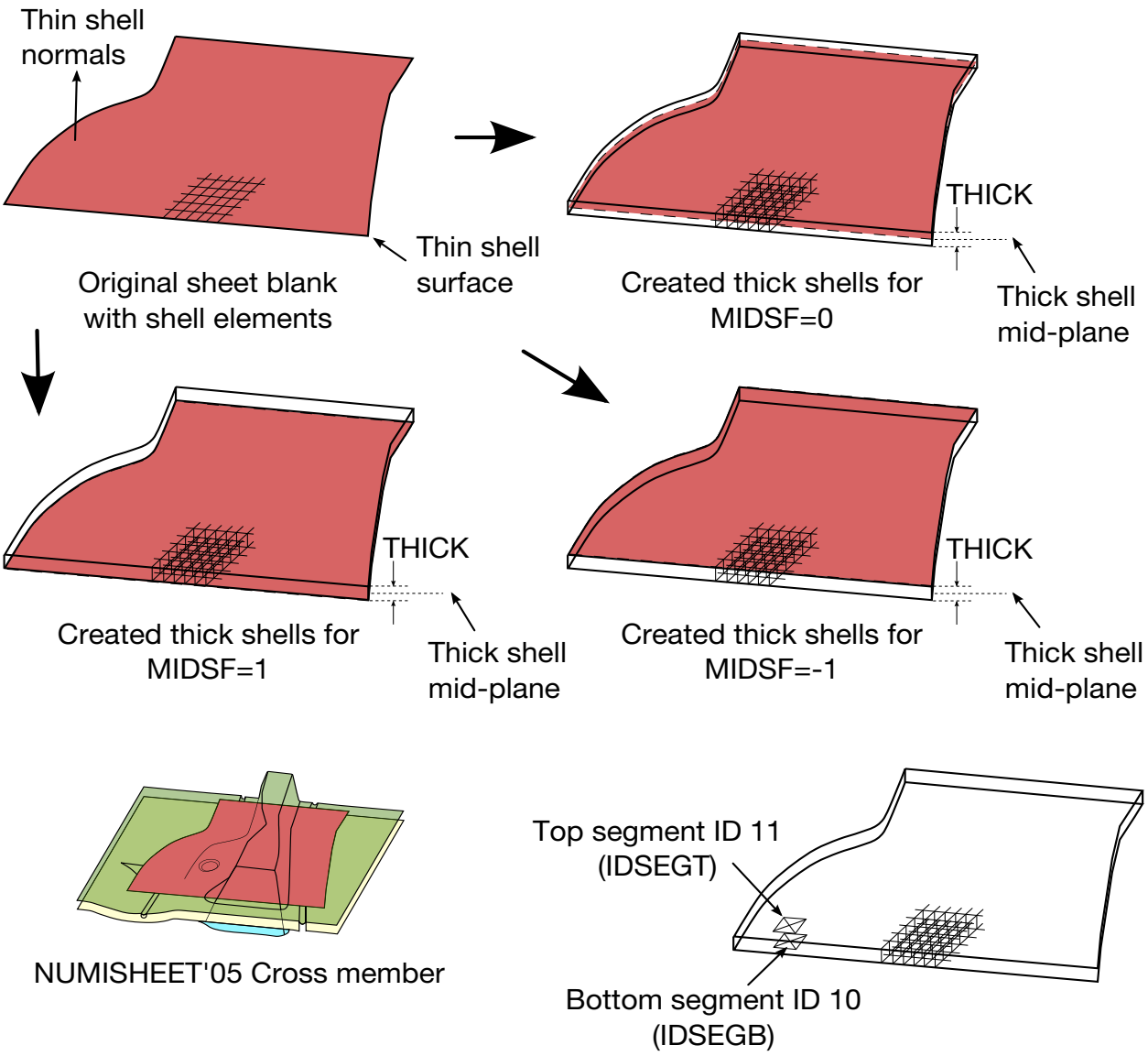


Figure 12-54. Converting thin shells to thick shells in sheet metal forming

*CONTROL_FORMING_STONING

Purpose: Detect surface lows or surface defects formed during metal stamping. This calculation is typically performed after a springback simulation. A curvature-based method is implemented with the feature. Users have the option to check an entire part or just a few local areas, defined by node set or shell element set. In each area, direction of the stoning action can be specified or the program can automatically determine the stoning direction.

Card 1	1	2	3	4	5	6	7	8
Variable	ISTONE	LENGTH	WIDTH	STEP	DIRECT	REVERSE	METHOD	
Type	I	F	F	F	F	I	I	
Default	none	0.0	0.0	0.0	0.0	0	0	

Card 2	1	2	3	4	5	6	7	8
Variable	NODE1	NODE2	SID	ITYPE	V1	V2	V3	
Type	I	I	I	I	F	F	F	
Default	0	0	Rem 4	Rem 4	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
ISTONE	Stoning calculation option. EQ.1: calculate panel surface quality using stoning method.
LENGTH	Length of the stone. See Remark 1.
WIDTH	Width of the stone. See Remark 1.
STEP	Stepping size of the moving stone. See Remark 1.
DIRECT	Number of automatically determined stoning direction(s). See Remark 2.

VARIABLE	DESCRIPTION
REVERSE	Surface normal reversing option (see Remark 3): EQ.0: do not reverse surface normals. EQ.1: reverse surface normals.
METHOD	Stoning method. EQ.0: curvature-based method.
NODE1	Tail node defining stoning moving direction. See Remark 2.
NODE2	Head node defining stoning moving direction. See Remark 2.
SID	Node or shell set ID.
ITYPE	Set type designation: EQ.1: node set EQ.2: element set
V1, V2, V3	Vector components defining stoning direction (optional). See Remark 2.

About Stoning:

Stoning is a quality checking process on class-A exterior stamping panels. Typically the long and wider surfaces of an oil stone of a brick shape are used to slide and scratch in a given direction against a localized area of concern on a stamped panel. Surface "lows" are shown where scratch marks are not visible and "highs" are shown in a form of scratch marks. This keyword is capable of predicting both the surface "lows" and "highs". Since stoning process is carried out after the stamping (either drawn or trimmed) panels are removed from the stamping dies, a springback simulation needs to be performed prior to conducting a stoning analysis.

Remarks:

1. **Stone Reference Sizes.** As a reference, typical stone length and width can be set at 150.0 and 30.0 mm, respectively. The step size of the moving stone is typically set about the same order of magnitude of the element length. The smallest element length can be selected as the step size.

2. **Stoning Direction.** Stoning direction can be set in three different ways:

a) The variables NODE1 and NODE2 are used to define a specific stoning direction. The stone is moved in the direction defined by NODE1 to NODE2.

b) Alternatively, one can leave NODE1 and NODE2 blank and define the number of automatically determined stoning directions by using the variable DIRECT. Any number can be selected but typically 2 is used. Although CPU time required for the stoning calculation is trivial, a larger DIRECT consumes more CPU time.

c) Furthermore, stoning direction can also be defined using a vector by defining the variables V1, V2, and V3.

3. **Element Normal Vectors.** Stoning is performed on the outward normal side of the mesh. Element normals must be consistent and oriented accordingly. Element normal can be automatically made consistent in LS-PrePost4.0 under *EleTol/Normal* menu. Alternatively, the variable REVERSE provides in the solver an easy way to reverse a part with consistent element normals before the computation.

4. **Region for Stoning Analysis.** The blank model intended for analysis can be included using keyword *INCLUDE. If nothing is defined for SID and ITYPE, then the entire blank model included will be used for stoning analysis.

   A large area mesh can be included in the input file. An ELSET may also be included, which defines a local area that requires stoning computation. Alternatively, an ELSET can define several local areas to be used for the computation. Furthermore, an ELSET should not include meshes that have reversed curvatures. An ELSET can be easily generated using LS-PrePost4.0, under *Model* → *CreEnt* → *Cre* → *Set_Data* → **SET_SHELL*.

5. **Modeling Guidelines.** Since stoning requires high level of accuracy in springback prediction, it is recommended that the SMOOTH option in keyword *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE to be used during the draw forming simulation. Not all areas require SMOOTH contact, only areas of interest may apply. In addition, meshes in the areas of concern need to be very fine, with average element size of 1 to 2 mm. Mesh adaptivity is not recommended in the SMOOTH/stoning areas. Also, mass scaling with DT2MS needs to be sufficiently small to reduce the dynamic effect during forming. For binder closing of large exterior panels, implicit static method using *CONTROL_IMPLICIT_FORMING type 2 is recommended, to further reduce potential buckles caused by the inertia effect.

6. **Stoning Output.** It is recommended that double precision version of LS-DYNA be used for this application. The output of the stoning simulation results is in a file named filename.output, where "filename" is the name of the LS-DYNA stoning input file containing this keyword, without the file extension. The stoning

results can be viewed using LS-PrePost4.0, under *MFPost* → *FCOMP* → *Shell_Thickness*.

Example:

An example of a stoning analysis on a Ford Econoline door outer panel is provided for reference. The original part model comes from National Crash Analysis Center at The George Washington University. The original part was modified heavily in LS-PrePost4.0 for demonstration purposes. Binder and addendum were created and sheet blank size was assumed. The blank is assigned 0.65 mm thickness and a BH210 properties with *MAT_037. Shell thickness contour plots for the drawn and trimmed panels are shown in Figures 12-55 and 12-56, respectively. Springback amount in the Z-direction is plotted in Figure 12-57. The complete input deck used for the stoning simulation is provided below for reference; where, a local area mesh of the door handle after springback simulation Doorhandle.k and an element set elset1.k are included in the deck. Locations of the ELSETs are defined for the upper right (Figure 12-58 left) and lower right corners (Figure 12-59 left) of the door handle, where "mouse ears" are expected.

```
*KEYWORD
*TITLE
Stoning Analysis
*INCLUDE
Doorhandle.k
*INCLUDE
elset1.k
*CONTROL_FORMING_STONING
$    ISTONE     LENGTH      WIDTH       STEP     DIRECT    REVERSE     METHOD
         1.0      150.0        4.0        1.0          9          0          0
$     NODE1      NODE2        SID      ITYPE
                                 1          2
*END
```

Stoning results are shown in Figures 12-58 (right) and 12-59 (right) for the upper right and lower right corners, respectively. "Mouse ears" are predicted where anticipated.

Revision Information:

The stoning feature is available in LS-DYNA Revision 54398 and later releases. Vector component option is available in Revision 60829 and later releases.

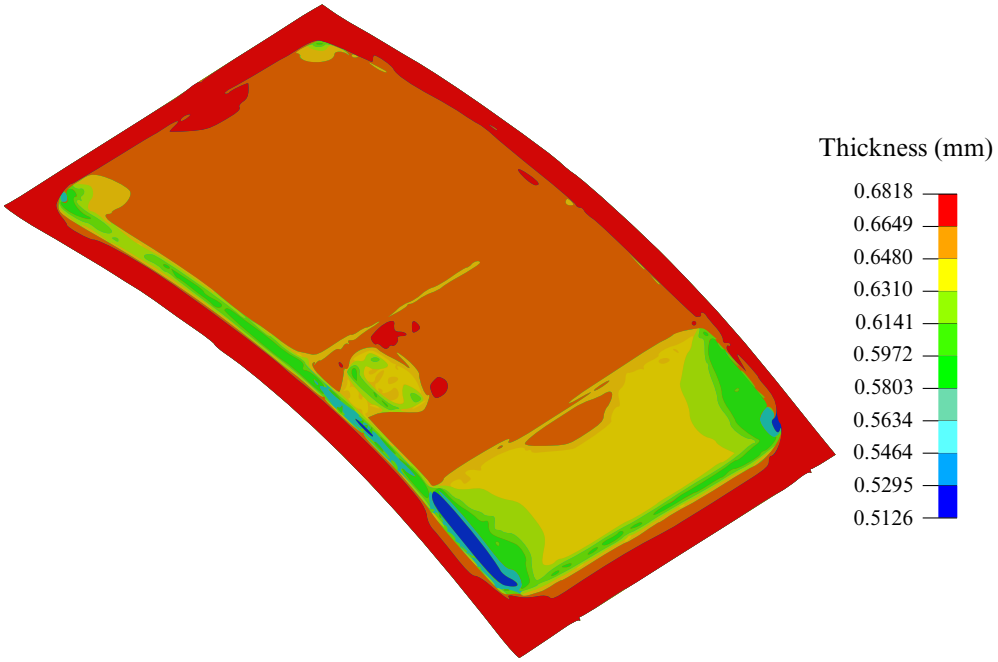


Figure 12-55. Thickness contour of the panel after draw simulation.

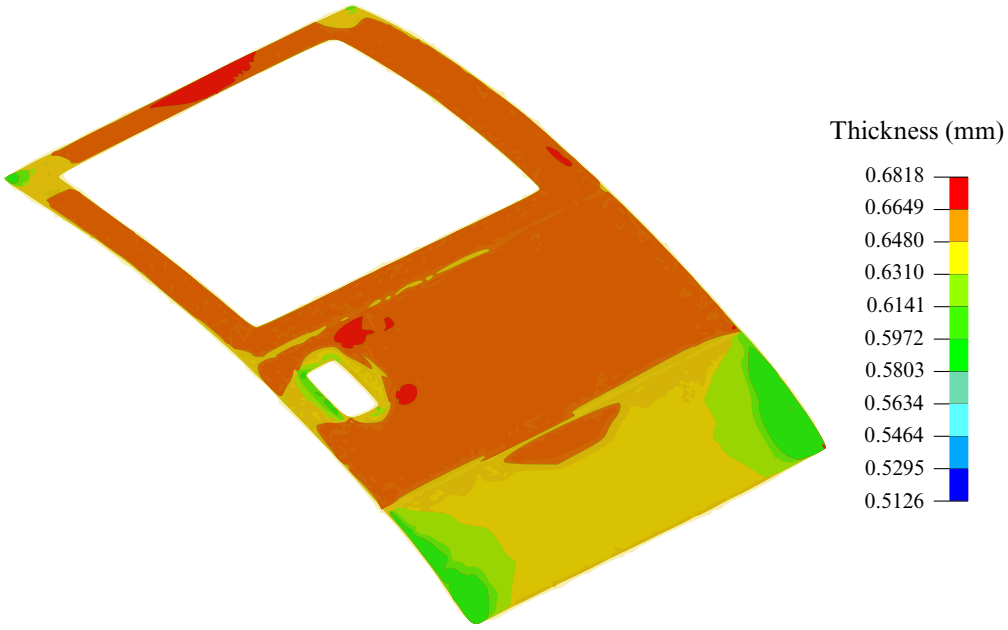


Figure 12-56. Thickness contour of the panel after trimming.

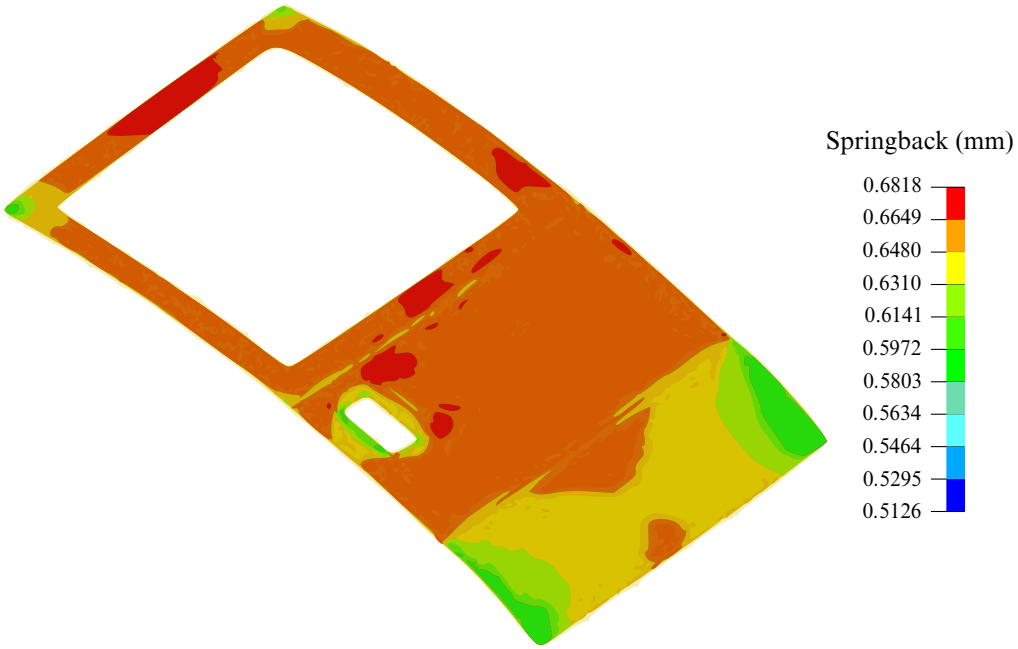


Figure 12-57. Springback amount (mm).

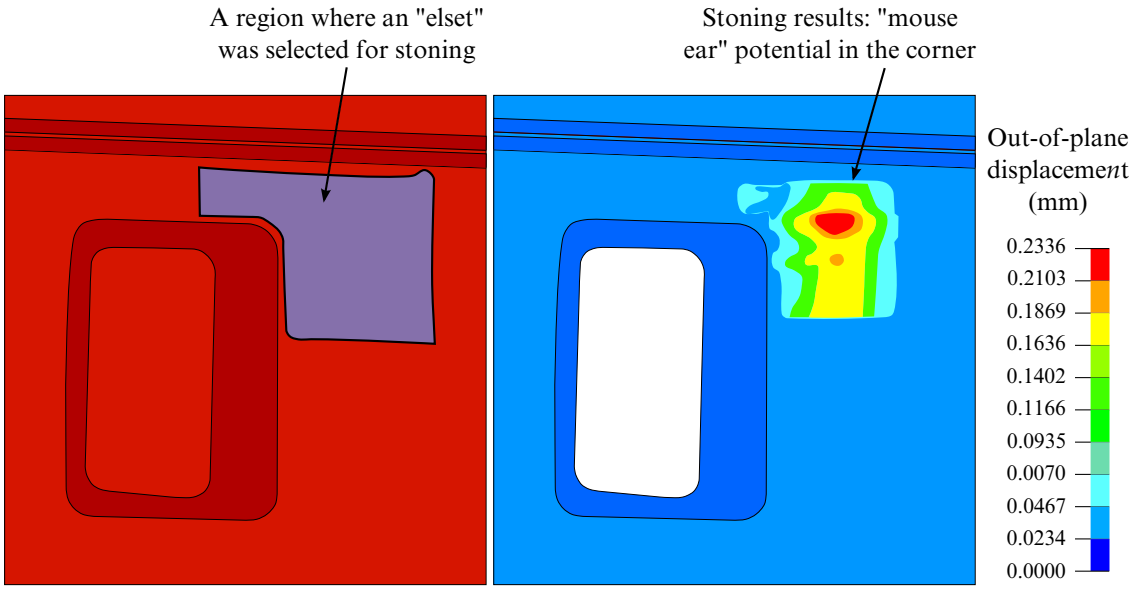


Figure 12-58. Stoning simulation for the upper right door corner.

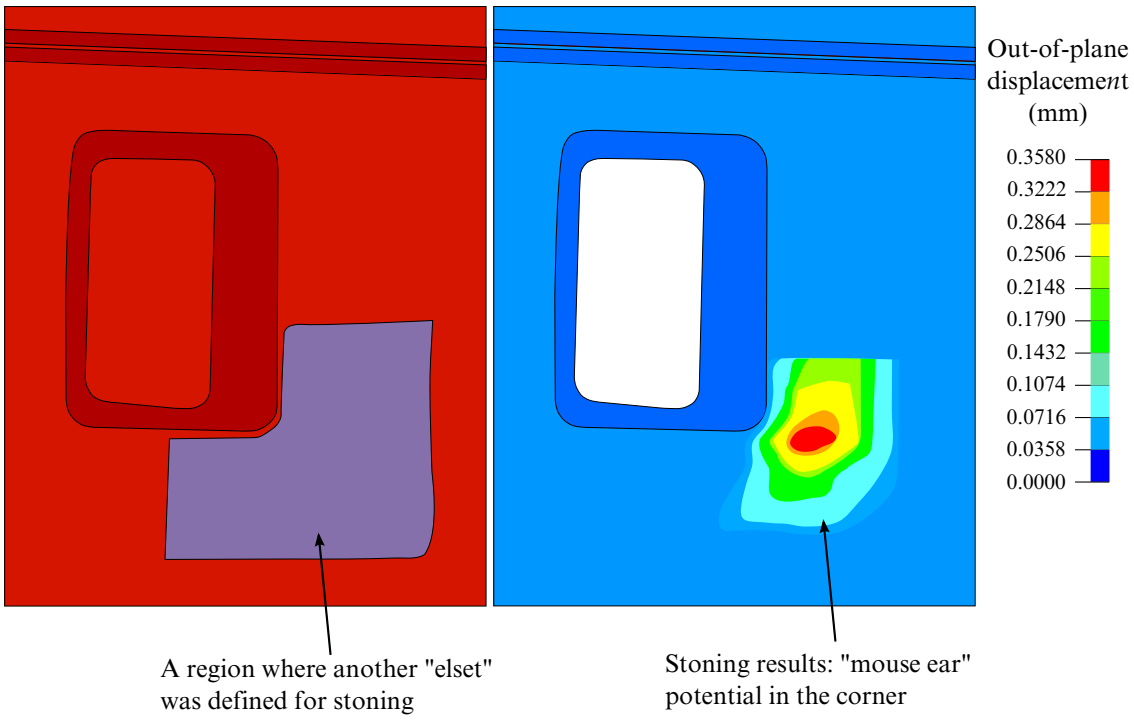


Figure 12-59. Stoning simulation for the lower right door corner.

*CONTROL_FORMING_STRAIN_RATIO_SMOOTH

Purpose: In explicit methods, the output strain path looks smooth, but the strain incremental ratio, β ($= d\epsilon 2/d\epsilon 1$) can oscillate significantly during each step and will result in significant error in formability prediction. This keyword uses a smoothing algorithm to smooth the beta value when calculating the Formability Index, which is designed to better calculate formability by considering a non-linear strain path.

This keyword must be used with the NLP option in *MAT_36, *MAT_37, *MAT_125, or *MAT_226 for shell elements only. It is jointly developed with Ford Motor Company.

Card 1	1	2	3	4	5	6	7	8
Variable	DT/CYCLE	WEIGHT						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
DT/CYCLE	Flag for output option (time interval or cycle number). LT.0: the absolute value is the time interval between outputs. GT.0: number of cycles between outputs
WEIGHT	Coefficient α in equation below.

Remarks:

The incremental change of in-plane major and minor strains are smoothed according to the following formula:

$$\Delta\epsilon 1_{(n-1)} \times (1-\alpha) + d\epsilon 1_{(n)} \times \alpha$$
$$\Delta\epsilon 2_{(n-1)} \times (1-\alpha) + d\epsilon 2_{(n)} \times \alpha$$

where, $d\epsilon 1_{(n)}$ and $d\epsilon 2_{(n)}$ are incremental changes of $\epsilon 1$ and $\epsilon 2$ in the current time step n, $\Delta\epsilon 1_{(n-1)}$ and $\Delta\epsilon 2_{(n-1)}$ are incremental changes of $\epsilon 1$ and $\epsilon 2$ in the previous time step $n-1$. The weighting coefficient α regulates the smoothness of the incremental changes in $\epsilon 1$ and $\epsilon 2$.

With the smoothed incremental major and minor strains, β is:

$$\beta = \frac{\Delta\epsilon 2_{(n-1)} \times (1-\alpha) + d\epsilon 2_{(n)} \times \alpha}{\Delta\epsilon 1_{(n-1)} \times (1-\alpha) + d\epsilon 1_{(n)} \times \alpha} \ .$$

The lower limit of β is the minimum strain ratio defined in the FLD curve.

Note that β is stored in history variable #2.

A partial keyword deck that illustrates using this keyword is shown below. The material model is *MAT_36. Note NEIPS is set to 3 so three history variables that include formability index (F.I.), strain ratio β and effective plastic strain $\bar{\varepsilon}$ are output.

```
*DATABASE_EXTENT_BINARY
$    NEIPH     NEIPS    MAXINT    STRFLG    SIGFLG    EPSFLG    RLTFLG    ENGFLG
                   3      &nip         1
*CONTROL_FORMING_STRAIN_RATIO_SMOOTH
$ DT/CYCLE    WEIGHT    OUTPUT
    -0.001      0.35         1
*MAT_3-PARAMETER_BARLAT_NLP
$      MID        RO         E        PR        HR
        13  7.8E-09  2.07E+05       0.30     3.000
$        M       R00       R45       R90      LCID
     6.000     1.200     1.450     1.090        99
$     AOPT         C         P     VLCID                          NLP
     2.000                                                       200
$                                      A1        A2        A3
                                    1.000     0.000     0.000
$       V1        V2        V3        D1        D2        D3
                                    0.000     1.000     0.000
*DEFINE_CURVE
      200
$FLD
             -0.7000             0.8309
             -0.4500             0.6805
             -0.2500             0.5081
              0.0000             0.2479
              0.2000             0.3487
              0.4000             0.3845
```

*CONTROL_FORMING_TEMPLATE

Purpose: Simplify the required input for sheet metal stamping simulations. With this keyword, five templates are given: three-piece air draw, three-piece toggle draw, four-piece stretch draw, trimming, and springback.

> **NOTE:** This option has been deprecated in favor of *CONTROL_FORMING_AUTOPOSITION_PARAMETER.

Card 1	1	2	3	4	5	6	7	8
Variable	IDTEMP	BLKID	DIEID	PNCH	BNDU	BNDL	TYPE	PREBD
Type	I	I	I	I	I	I	I	F
Default	none	none	none	none	none	none	0	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	LCSS	AL/FE	R00	R45	R90	E	DENSITY	PR
Type	I	C	F	F	F	F	F	F
Default	none	F	1.0	R00	R00	none	none	none

Card 3	1	2	3	4	5	6	7	8
Variable	K	N	MTYP	UNIT	THICK	GAP	FS	
Type	F	F	I	I	F	F	F	
Default	none	none	37	1	none	1.1t	0.1	

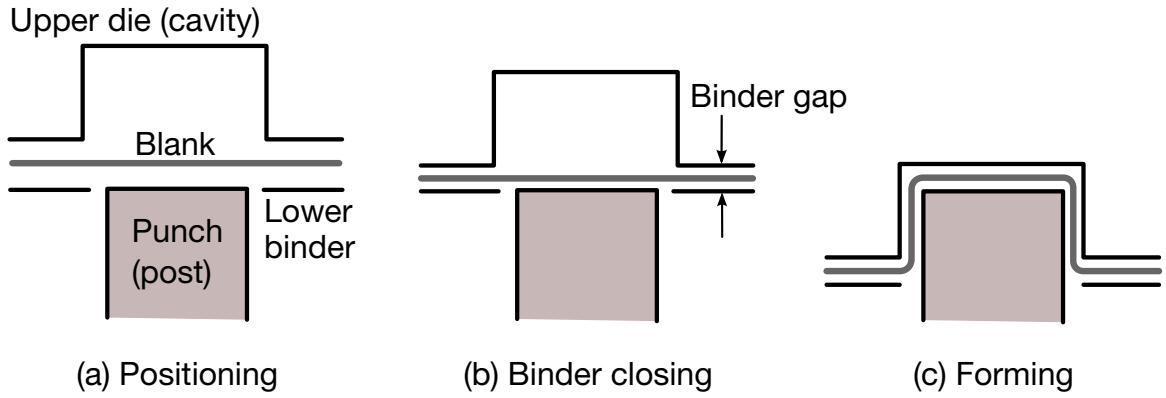


Figure 12-60. IDTEMP = 1: forming in 3-piece air draw.

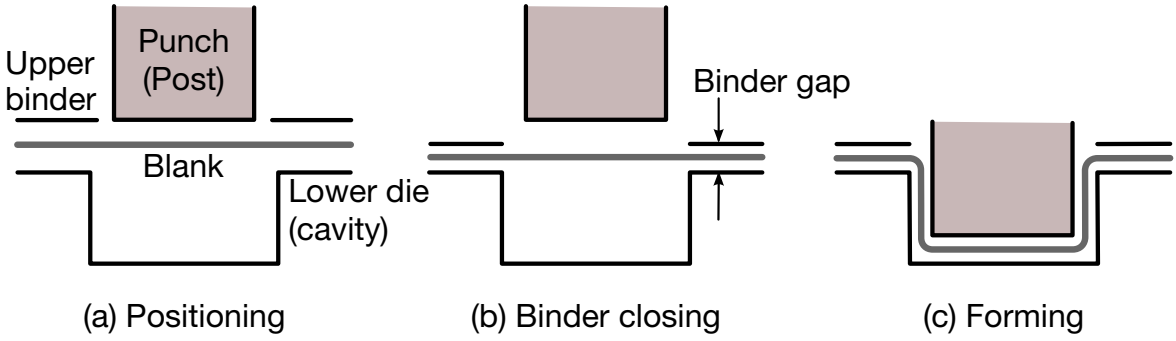


Figure 12-61. IDTEMP = 2: forming in 3-piece toggle draw.

Card 4	1	2	3	4	5	6	7	8
Variable	PATERN	VMAX	VX	VY	VZ	VID	AMAX	
Type	I	F	F	F	F	I	F	
Default	1	1000	0	0	-1	—	10^6	

Card 5	1	2	3	4	5	6	7	8
Variable	LVLADA	SIZEADA	TIMSADA	D3PLT				
Type	I	F	I	I				
Default	1	none	20	10				

VARIABLE	DESCRIPTION
IDTEMP	Type of forming process (see About IDTEMP below): EQ.1: 3-piece air-draw (Figure 12-60) EQ.2: 3-piece Toggle-draw (Figure 12-61) EQ.3: 4-piece stretch draw (Figure 12-62) EQ.4: springback EQ.5: trimming
BLKID	Part or part set ID (see TYPE) that defines the blank.
DIEID	Part or part set ID that defines the die. See Figures 12-60, 12-61 and 12-62 for more information
PNCH	Part or part set ID that defines the punch
BNDU	Part or part set ID that defines the upper binder
BNDL	Part or part set ID that defines the lower binder
TYPE	Flag for part or part set ID used in the definitions of BLKID, DIEID, PNCH, BNDU, and BNDL: EQ.0: Part ID EQ.1: Part set ID
PREBD	"Pull-over" distance, for 4 piece stretch draw only. This is the travel distance of both upper and lower binder together after they are fully closed. Typically, this distance is below 50 mm. See Figure 12-62 for more information.
LCSS	If the material (*MAT_XXX) for the blank is not defined, this curve ID will define the stress-strain relationship; otherwise, this curve is ignored.
AL/FE	This parameter is used to define the Young's Modulus and density of the blank. If this parameter is defined, E and DENSITY will be defined in the units given by Table 12-3. EQ.A: the blank is aluminum EQ.F: the blank is steel (default)
R00, R45, R90	Material anisotropic parameters. For transversely anisotropy the R value is set to the average value of R00, R45, and R90.

VARIABLE	DESCRIPTION
E	Young's Modulus. If AL/FE is user defined, E is unnecessary.
DENSITY	Material density of blank. If AL/FE is user defined, this parameter is unnecessary.
PR	Poisson's ratio.
K	Strength coefficient for exponential hardening ($\overline{\sigma} = k\bar{\varepsilon}^n$). If LCSS is defined, or if a blank material is user defined by *MAT_XXX, this parameter is ignored.
N	Exponent for exponential hardening ($\overline{\sigma} = k\bar{\varepsilon}^n$). If LCSS is defined, or if a blank material user defined, this parameter is ignored.
MTYP	Only material models *MAT_036 and *MAT_037 are supported.
UNIT	Define a number between 1 and 10 (Table 12-3) to indicate the units used in this simulation. This unit is used to obtain proper punch velocity, acceleration, time step, and material properties.
THICK	Blank thickness. If the blank thickness is already defined with *SECTION_SHELL, this parameter is ignored.
GAP	The gap between rigid tools at their home position. If *BOUNDARY_PRESCRIBED_MOTION_RIGID is defined, this parameter is ignored. The default is 1.1 x blank thickness.
FS	Friction coefficient (default is 0.10). If the contact (*CONTACT) is user defined, this parameter is ignored.
PATERN	Velocity profile of moving tool. If the velocity defined using *BOUNDARY_PRESCRIBED_MOTION_RIGID, PATERN is ignored. EQ.1: Ramped velocity profile EQ.2: Smooth velocity curve
VX, VY, VZ	Vector components defining the direction of the punch movement. The default direction is defined by VID.
VID	Vector ID defining the direction of the punch movement. This variable overrides the vector components (VX,VY,VZ). If VID and (VX,VY,VZ) are undefined, the punch is assumed to move in the negative z-direction.

VARIABLE	DESCRIPTION
AMAX	The maximum allowable acceleration.
LVLADA	Maximum adaptive level.
SIZEADA	Minimum element size permitted in the adaptive mesh.
TIMSADA	Total number of adaptive steps during the forming simulation.
D3PLT	The total number of output states in the D3PLOT database.

Applicable units:

Table 12-3. Available units for metal stamping simulation.

UNIT	1	2	3	4	5	6	7	8	9	10
Mass	Ton	Gm	Gm	Gm	Gm	Kg	Kg	Kg	Kg	Kg
Length	Mm	Mm	Mm	Cm	Cm	Mm	Cm	Cm	Cm	m
Time	S	Ms	S	Us	S	Ms	Us	Ms	S	S
Force	N	N	μN	1e7N	Dyne	KN	1e10N	1e4N	1e-2N	N

About IDTEMP:

When the variable IDTEMP is set to 1, it represents a 3-piece draw in air, as shown in Figure 12-60. When IDTEMP is set to 2, a 3-piece toggle draw is assumed, as in Figure 12-61. For IDTEMP of 1 or 2, the solver automatically positions the tools and minimizes the punch travel (step a), calculates the binder and punch travel based on the blank thickness and the home gap (step b), sets the termination time based on steps (a) and (b), defines the rigid body motion of the tooling, establishes all the contacts between the blank and rigid tools, and selects all necessary control parameters.

When IDTEMP is set to 3, a 4-piece stretch draw shown in Figure 12-62 will be followed. The die action goes as follows: after upper binder moves down to fully close with lower binder, both pieces move together down a certain distance (usually ~50 mm) to "pull" the blank "over" the lower die; then the upper punch closes with the lower die; and finally the binders move down together to their home position.

Both toggle draw and 4-piece stretch draw are called "double action" processes which suffer from a slower stamping speed. As the metric of "hits per minute" (or "parts per minute") becomes a stamping industry benchmark for efficiency, these types of draw are

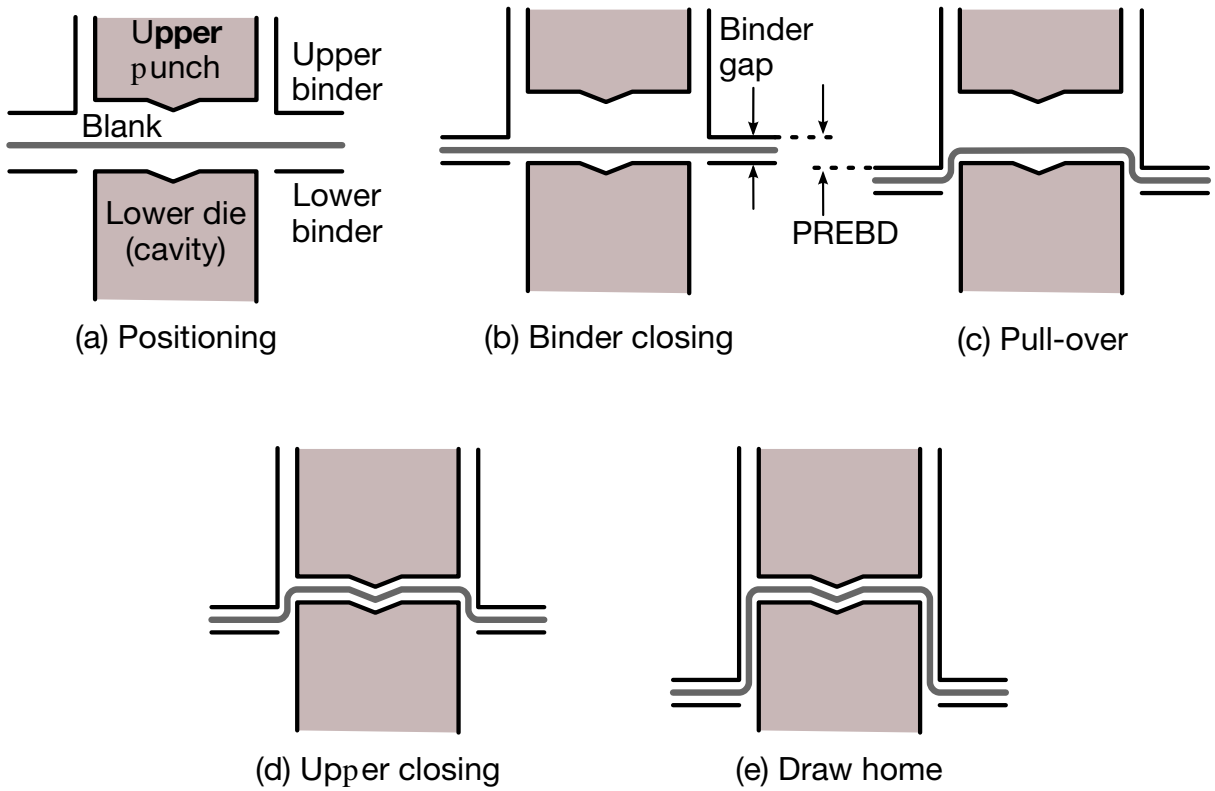


Figure 12-62. IDTEMP = 3: forming in 4-piece stretch draw.

becoming less popular (especially the 4-piece stretch draw). Nevertheless, they remain important stamping processes for controlling wrinkles in difficult-to-form panels such as lift gate inners, door inners and floor pans. These two processes are also used in situation where deep drawn panels require draw depth of over 250 mm, the usual limit for automatic transfer presses.

For all the above IDTEMP values, you do not need to define additional keywords, such as *PART, *CONTROL, *SECTION, *MAT_…, *CONTACT_… (drawbead definition is an exception), and *BOUNDARY_PRESCRIBED_MOTION_RIGID. If any such keyword is defined, automatic default settings will be overridden.

When IDTEMP is set to 4, a springback simulation will be conducted. The only additional keyword required is *BOUNDARY_SPC_… to specify the constraints in the input deck.

When IDTEMP is set to 5, a trimming operation will be performed. The only additional keyword required is *DEFINE_CURVE_TRIM to specify the trim curves in the input deck.

*CONTROL_FORMING_TIPPING

Purpose: Reorient or reposition a part between the stamping dies. In a stamping line die simulation, panel tipping and translation between the die stations are frequently required. Typically, such transformations involve only a small rotation (less than 15 degrees) but a large translation. For example, a part may need to be tipped by an angle of 10 degrees along the Y-axis and translated 2000 mm along the X-axis between the current trimming die and the next flanging die. This feature is available for shell and solid elements.

Card Summary:

Card Sets. For each rotated or translated part or set, add one Card 1 (Tipping Card). If NMOVE ≠ -6, include NMOVE of Cards 2a and 2b combined. If NMOVE = -6, include Cards 2c.1 through 2c.6. The data set for this keyword ends at the next keyword ("*") card.

Card 1. This card is required.

PID/SID	ITYPE	ISTRAIN	IFSTRSS	NMOVE			

Card 2a. If NMOVE ≠ -6, this card may be included. It is recognized when the first field (ROT/TRAN) is 1. It defines a rotation transformation. Include a combination of this card and Card 2b to get NMOVE transformation cards.

ROT/TRAN	V11	V12	V13	X01	Y01	Z01	DISTA1

Card 2b. If NMOVE ≠ -6, this card may be included. It is recognized when the first field (ROT/TRAN) is 2. It defines a translation transformation. Include a combination of this card and Card 2a to get NMOVE transformation cards.

ROT/TRAN	DX	DY	DZ				

Card 2c.1. Include this card if NMOVE = -6.

P1	X1	Y1	Z1			

Card 2c.2. Include this card if NMOVE = -6.

P2	X2	Y2	Z2			

Card 2c.3. Include this card if NMOVE = -6.

P3	X3	Y3	Z3			

Card 2c.4. Include this card if NMOVE = -6.

P4	X4	Y4	Z4			

Card 2c.5. Include this card if NMOVE = -6.

P5	X5	Y5	Z5			

Card 2c.6. Include this card if NMOVE = -6.

P6	X6	Y6	Z6			

Data Card Definitions:

Tipping Card. Specify a part or set ID to be tipped.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/SID	ITYPE	ISTRAIN	IFSTRSS	NMOVE			
Type	I	I	I	I	I			
Default	none	none	0	0	0			

VARIABLE	DESCRIPTION
PID/SID	Part ID or part set ID of part(s) that requires tipping and/or translation.
ITYPE	Part ID or part set ID indicator: EQ.1: PID means part ID, EQ.2: PID/SID means part set ID.
ISTRAIN	Strain tensors inclusion option: EQ.1: include in tipping/translation.
ISTRESS	Stress tensors inclusion option: EQ.1: include in tipping/translation.
NMOVE	Total number of tipping and translation transformations intended for this part/part set. However, when NMOVE: EQ.-6: Transformation is done based on user-specified coordinates of 3 points on the source, and the corresponding point coordinates on the target.

Move Card 2a (Rotation). Include when first entry, ROT/TRAN, is set to 1.

Card 2a	1	2	3	4	5	6	7	8
Variable	ROT/TRAN	V11	V12	V13	X01	Y01	Z01	DISTA1
Type	I	F	F	F	F	F	F	F
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Move Card 2b (Translation). Include when first entry, ROT/TRAN, is set to 2.

Card 2b	1	2	3	4	5	6	7	8
Variable	ROT/TRAN	DX	DY	DZ				
Type	I	F	F	F				
Default	none	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
ROT/TRAN	Transformation type: EQ.1: rotation, EQ.2: translation.
V11, V12, V13	Vector components of an axis about which tipping is performed.
X01, Y01, Z01	*X*, *Y*, and *Z* coordinates of a point through which the tipping axis passes.
DSITA	Tipping angle in degrees. See Remark 2.
DX, DY, DZ	Translation distances along global *X*-axis, *Y*-axis and *Z*-axis.

Move Card 2c.1 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.1	1	2	3	4	5	6	7	8	9	10
Variable	P1	X1		Y1		Z1				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

Move Card 2c.2 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.2	1	2	3	4	5	6	7	8	9	10
Variable	P2	X2		Y2		Z2				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

Move Card 2c.3 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.3	1	2	3	4	5	6	7	8	9	10
Variable	P3	X3		Y3		Z3				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

Move Card 2c.4 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.4	1	2	3	4	5	6	7	8	9	10
Variable	P4	X4		Y4		Z4				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

Move Card 2c.5 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.5	1	2	3	4	5	6	7	8	9	10
Variable	P5	X5		Y5		Z5				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

Move Card 2c.6 (3-point Transformation). Include when NMOVE is set to -6. (I8, 3E16.0)

Card 2c.6	1	2	3	4	5	6	7	8	9	10
Variable	P6	X6		Y6		Z6				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

VARIABLE	DESCRIPTION
P1, P2, P3	Unique point IDs from the target, where it will be transformed to.
X1, Y1, Z1 X2, Y2, Z2 X3, Y3, Z3	X, Y, and X coordinates of the points P_1, P_2, and P_3 from the target, respectively.

VARIABLE	DESCRIPTION
P4, P5, P6	Unique point IDs from the source, where it will be transformed from.
X4, Y4, Z4 X5, Y5, Z5 X6, Y6, Z6	X, Y, and Z coordinates of the points P_4, P_5, and P_6 from the source, respectively. Note that points P_4, P_5, and P_6 correspond to points P_1, P_2, and P_3, respectively.

Remarks:

1. **Include keyword.** Keyword *INCLUDE can be used to include the file to be tipped or translated.

2. **Tipping angle.** Tipping angle DISTA1 is defined in degrees. Signs of the tipping angles follow the right-hand rule.

3. **NMOVE ≠ -6 example.** The keyword input below tips a part by +23.0°, -31.0°, and +8.0° about the X-, Y-, and Z-axes that pass through the origin, respectively. The part also translates 12.0 mm, -6.0 mm and 91.0 mm along X-, Y-, and Z-axes, respectively.

```
*INCLUDE
trimmedpart.dynain
*CONTROL_FORMING_TIPPING
$ PID/PSID     ITYPE   ISTRAIN    ISTRSS     NMOVE
         1         0         1         1         4
$ ROT/TRAN       V11       V12       V13       X01       y01       z01    DSITA1
         1     1.000  0.000000     0.000     0.000     0.000     0.000      23.0
$ ROT/TRAN       V21       V22       V23       X21       y21       z21    DSITA2
         1     0.000  1.000000     0.000     0.000     0.000     0.000     -31.0
$ ROT/TRAN       V31       V32       V33       X31       y31       z31    DSITA3
         1  0.000000     0.000     1.000     0.000     0.000     0.000       8.0
$ ROT/TRAN        DX        DY        DZ
         2      12.0      -6.0      91.0
```

4. **NMOVE = -6 example.** An example of the keyword input for NMOVE = -6 is included below. The part tips based on 3-points from the source and the corresponding 3-points from the target.

```
*CONTROL_FORMING_TIPPING
$ PID/PSID     ITYPE   ISTRAIN    ISTRSS     NMOVE
         2         1         1         1        -6
$ target coordinates (I8, 3E16.0):
       4         20.1878        -55.4753        -30.5631
       5      0.00092581        -55.4871        -30.5094
       6      7.61563E-4        -76.1135        -33.6276
$ source coordinates (I8, 3E16.0):
       1         20.1878        -55.4753        -30.5631
       2      0.00925064        -55.5290        -30.5749
       3     0.000757217        -52.4108        -51.2013
```

*CONTROL_FORMING_TRAVEL

Purpose: This keyword allows user to define tool travel for each phase in a stamping process simulation. The entire simulation process can be divided into multiple phases corresponding to the steps of an actual metal forming process. This keyword is to be used for tools that are pre-positioned above the sheet blank (or below the blank) and ready for forming. For tools that are pre-positioned at their home positions, *CONTROL_FORMING_TRAVEL should be used. This keyword is used together with *CONTROL_FORMING_USER.

> **NOTE:** This option has been deprecated in favor of *CONTROL_FORMING_AUTOPOSITION_PARAMETER).

Define Travel Cards. Repeat Card as many times as needed to define travels in multiple phases. The next "*" card terminates the input.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	VID	TRAVEL	TARGET	GAP	PHASE	FOLLOW	
Type	I	I	F	I	F	I	I	
Default	none	none	none	none	1.0t	none	none	

VARIABLE	DESCRIPTION
PID	Part ID of a stamping tool, as defined in *PART.
VID	Vector ID defining the direction of travel for the tool defined by the PID.
TRAVEL	The distance in which the tool will be traveled to complete forming in the direction specified by the VID. If TRAVEL is defined, it is unnecessary to define TARGET.
TARGET	Target tool PID, as defined in *PART. The tool (defined by PID) will be traveled to where the TARGET tool is to complete forming.
GAP	The minimum distance between the tool (PID) and TARGET tool at the home position (forming complete). The GAP is by default the sheet blank thickness "t".

VARIABLE	DESCRIPTION
PHASE	Phase number, starting sequentially from 1. For example, phase 1 is the binder closing, and phase 2 is the drawing operation.
FOLLOW	Part ID of a stamping tool to be followed by the tool (PID). When this variable is defined, the distance between the tool (PID) and part ID defined by FOLLOW, will remain constant during the phase.

Remarks:

FOLLOW can be used to reduce total simulation time. For example, in a toggle draw, the upper punch travels together with the upper binder during binder closing phase, thus reducing the upper travel distance during the draw, shortening the overall termination time.

An example is provided in manual pages under *CONTROL_FORMING_USER.

*CONTROL_FORMING_TRIM_MERGE

Purpose: This feature allows for automatic close of any open trim loop curve for a successful trimming simulation. Previously, sheet metal trimming would fail if a trim curve does not form a closed loop. This keyword is used together with *DEFINE_CURVE_TRIM, *ELEMENT_TRIM, *DEFINE_VECTOR, *CONTROL_ADAPTIVE_CURVE, and *CONTROL_CHECK_SHELL. It applies to shell elements only.

Card 1	1	2	3	4	5	6	7	8
Variable	IMERGE	GAPM						
Type	I	F						
Default	1	0.0						

VARIABLE	DESCRIPTION
IMERGE	Activation flag. Set to '1' (default) to activate this feature.
GAPM	Gap distance between two open ends of a trim loop curve in the model. If the gap is smaller than GAPM, the two open ends of a trim curve will be closed automatically.

Remarks:

If multiple open trim loop curves exist, GAPM should be set to a value larger than any of the gap distances of any trim curves in the trim model.

Example:

The example provided below includes two trim curves, a three-dimensional curve (#90905) with an open gap of 2.3 mm and a two-dimensional trim curve (#90907) with an open gap 2.38 mm. An automatic merge operation is being performed with the GAPM set at 2.39 mm. Since this set value is larger than both gaps in the model, trimming will automatically close the gap for both curves to form two closed-loop curves for a successful trim. In Figure 12-63, two different trimming results are illustrated with GAPM of 2.39 (successful) as well as 2.37 (fail).

```
*KEYWORD
*INCLUDE_TRIM
drawn2.dynain
⋮
*CONTROL_ADAPTIVE_CURVE
```

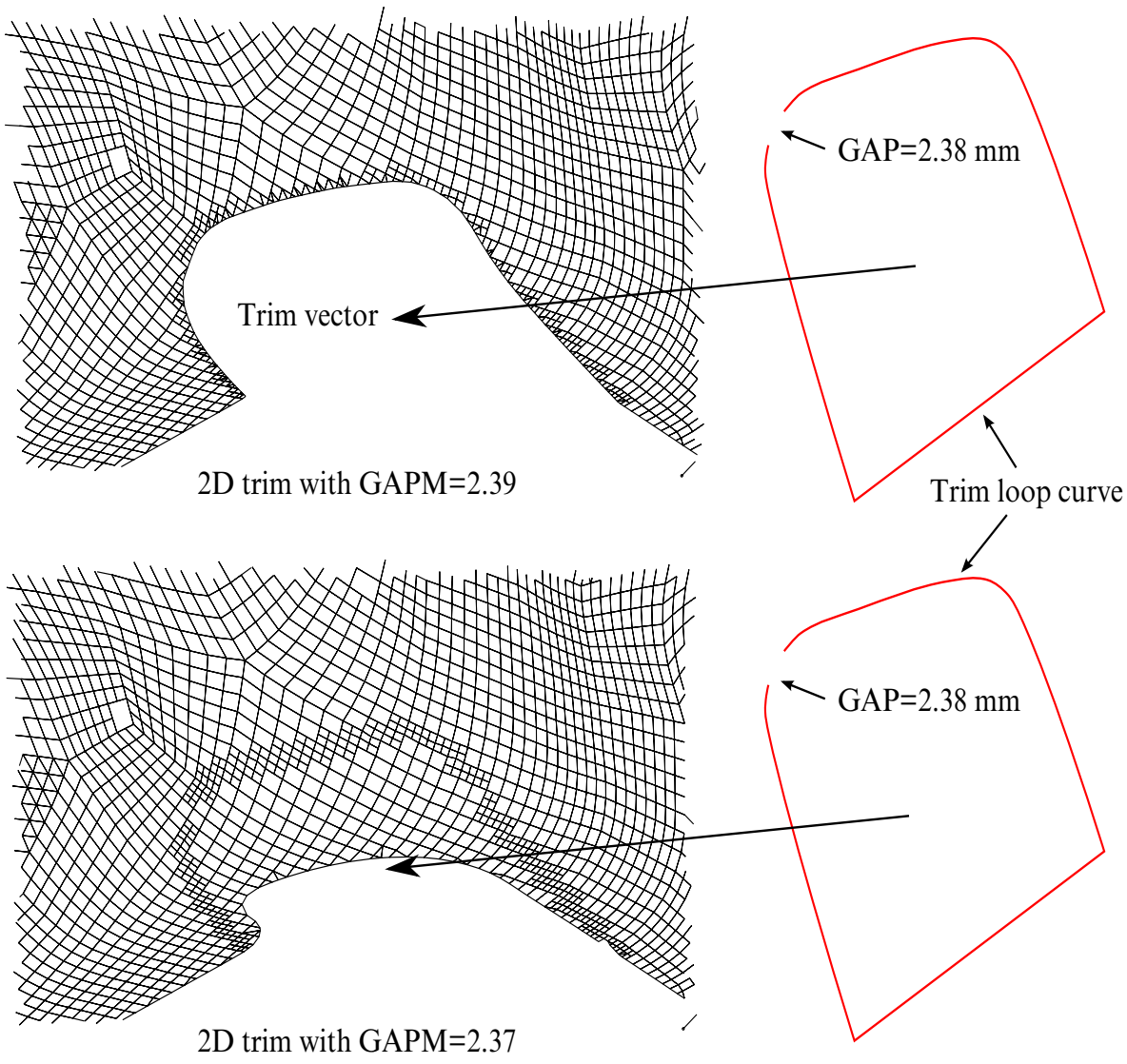


Figure 12-63. A 2D trimming with different GAPM values.

```
$    IDSET     ITYPE         N      SMIN
   &blksid         2         3       0.6
*CONTROL_CHECK_SHELL
$     PSID    IFAUTO    CONVEX      ADPT    ARATIO     ANGLE      SMIN
  &blksid1         1         1         1  0.250000150.000000  0.000000
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
     90907         2         1         0  1.250000  2.500000         0         0
sim_trimline_03.igs
*DEFINE_CURVE_TRIM_NEW
$#    tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
     90905         2         0         2  1.250000  1.000000         0         0
$# filename
sim_trimline_02.igs
*DEFINE_VECTOR_TITLE
vector for Trim curve 90905
$#     vid        xt        yt        zt        xh        yh        zh       cid
         2     0.000     0.000     0.000 -0.170000  0.950000 -0.260000         0
*ELEMENT_TRIM
&blksid
*DEFINE_TRIM_SEED_POINT_COORDINATES
```

```
$ NSEED,X1,Y1,Z1,X2,Y2,Z2
1,&seedx,&seedy,&seedz
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_TRIM_MERGE
$   IMERGE      GAPM
         1      2.39
$ Note that the 3D trim curve has a gap of 2.3 and the 2D trim curve has a gap of 2.38
*END
```

Revisions:

This feature is available starting in LS-DYNA Revision 84098.

*CONTROL_FORMING_TRIM_SOLID_REFINEMENT

Purpose: Activate adaptive refinement along the trim curve for trimming a multi-layered sandwiched part.

Card 1	1	2	3	4	5	6	7	8
Variable	IREFINE	ILEVEL						
Type	I	I						
Default	1	0						

VARIABLE	DESCRIPTION
IREFINE	A flag to activate the adaptive trimming of a multi-layer sandwiched part. Currently setting this to either 0 or 1 will turn on the adaptive trimming. EQ.1: Activate the adaptive trimming.
ILEVEL	Adaptive refinement level. Currently setting this variable to any integer other than 0 will refine the mesh one level down along the trim curve. EQ.0: No refinement EQ.1: Refine one level down.

Example:

A partial keyword input example is shown below where the included file incoming.dynain is a sandwiched part (ITYP = 1). The part is to be trimmed with adaptive refinement one level down (IREFINE = 1, ILEVEL = 2) along a 2D trim curve in the global *X*-direction that is defined in trimcurves2d.iges.

```
*Keyword
*Include_trim
incoming.dynain
*CONTROL_FORMING_TRIMMING
$     PSID                ITYP
        11                   1
*CONTROL_FORMING_TRIM_SOLID_REFINEMENT
$  irefine    ilevel
         1         2
*SET_PART_LIST
11
100,101,102
```

```
*DEFINE_CURVE_TRIM_NEW
$#    tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
         2         2         0         1   0.10000  1.000000         0         0
$# filename
trimcurves2d.iges
*DEFINE_VECTOR_TITLE
vector for Trim 1
$#     vid        xt        yt        zt        xh        yh        zh       cid
         1     0.000     0.000     0.000     1.000     0.000  0.000000         0
*DEFINE_TRIM_SEED_POINT_COORDINATES
$ NSEED,X1,Y1,Z1,X2,Y2,Z2
1,-69.5309,114.833,49.55
```

Note that for sandwiched part trimming, the keyword *INCLUDE_TRIM (not *INCLUDE) *must* be used to include the dynain file to be trimmed.

Revision information:

1. This feature is available starting in Dev 134513, in SMP and MPP.

*CONTROL_FORMING_TRIMMING

Purpose: Define a part subset to be trimmed by *DEFINE_CURVE_TRIM. This feature is intended for metal forming simulations. Currently trimming is enabled for shell elements, solid elements, adaptive sandwiched parts (one layer of solid elements with top and bottom layers of shell elements), non-adaptive sandwiched parts (multiple layers of solid elements with top and bottom layers of shell elements), thick shell elements (TSHELL, 2D trimming only) and isogeometric shell elements (*ELEMENT_SHELL_-NURBS_PATCH, 3D trimming only). Note it is not applicable for axisymmetric solids or 2D plane strain/stress elements. For details, see *DEFINE_CURVE_TRIM.

NOTE: Before revision 87566 this card was called ELEMENT_TRIM.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID		ITYP					
Type	I		I					
Default	none		0					

VARIABLE	DESCRIPTION
PSID	Part set ID for trimming, see *SET_PART.
ITYP	Activation flag for sandwiched parts (laminates) trimming: EQ.0: Trimming for solid elements. EQ.1: Trimming for laminates.

Remarks:

This keyword is used together with *DEFINE_CURVE_TRIM to trim the parts defined in PSID at time zero, that is, before any stamping process simulation begins. Elements in the part set will be automatically trimmed in the defined direction if they intersect the trim curves. See examples in keyword section *DEFINE_CURVE_TRIM.

Note for solid element trimming, the keyword *INCLUDE_TRIM (not *INCLUDE) *must* be used to include the dynain blank to be trimmed.

Revision information:

1. Revision 87566: *ELEMENT_TRIM was changed to the current name *CONTROL_FORMING_TRIMMING.

2. Revision 95745: *CONTROL_FORMING_TRIMING was changed to *CONTROL_FORMING_TRIMMING.

3. Revision 92088: 2D trimming of solid elements is implemented.

4. Revision 92289: 2D and 3D trimming of laminates (ITYP) is added.

5. Revision 93467: 3D trimming of solid elements is added.

6. Latter Revisions may incorporate more improvements and are suggested to be used for trimming.

*CONTROL_FORMING_UNFLANGING_{OPTION}

Available options include:

<BLANK>

OUTPUT

Purpose: Unfold flanges of a deformable blank onto a rigid tooling mesh using an implicit statics solver. This is typically used in trim line unfolding during a stamping die development process. The option OUTPUT must be used together with *CONTROL_FORMING_UNFLANGING to obtain the modified trim curves. Other keywords related to blank size development are *CONTROL_FORMING_ONESTEP and *INTERFACE_BLANKSIZE_DEVELOPMENT. The feature is available in double precision for SMP.

Card Summary:

Card 1a.1. Include this card if no keyword option is used.

NOPTION	DVID	NUNBEND	STFBEND	STFCNT	IFLIMIT	DIST	ILINEAR

Card 1a.2. Include this card if no keyword option is used.

NB1	NB2	NB3	CHARLEN	NDOUTER			

Card 1b. Include this card if the keyword option OUTPUT is used.

THMX	THMN	EPSMX					

Data Cards:

Card 1a.1	1	2	3	4	5	6	7	8
Variable	NOPTION	DVID	NUNBEND	STFBEND	STFCNT	IFLIMIT	DIST	ILINEAR
Type	I	I	I	F	F	I	F	I
Default	none	N/A	none	none	none	none	↓	2

VARIABLE	DESCRIPTION
NOPTION	Flag to turn on an unfolding simulation: EQ.1: Activate the unfolding simulation program.

VARIABLE	DESCRIPTION
DVID	This variable is currently not being used.
NUNBEND	Estimated number of unbending, ranging from 10 to 100.
STFBEND	Unflanging stiffness, ranging from 0.1 to 10.0.
STFCNT	Normal stiffness, ranging from 0.1 to 10.0.
IFLIMIT	Iteration limit for the first phase of unfolding, typically ranging from 11 to 400.
DIST	Distance tolerance for auto-SPC along flange root. DIST (Figure 12-65) is usually slightly more than ½ of the flange thickness. *This field must be left blank for ILINEAR = 2.* Also, nodes along the root can be directly positioned on the rigid body surface (addendum), leaving a DIST of zero (Figure 12-65).
ILINEAR	Unfolding algorithm selection flag: EQ.0: Nonlinear unfolding. EQ.1: Linear unfolding. EQ.2: A hybrid unfolding method (Revision 87100 and later). The curved 3D meshes of the flange will first be mapped onto the tooling surface to be used as a starting porting for nonlinear iterations; unfolding completes when force balance is reached. (recommended).

Card 1a.2	1	2	3	4	5	6	7	8
Variable	NB1	NB2	NB3	CHARLEN	NDOUTER			
Type	I	I	I	F	I			
Default	none	↓	↓	150.0	none			

VARIABLE	DESCRIPTION
NB1	The start node ID (Figure 12-66) on a flange root boundary (fixed end of the flange, see Figures 12-65 and 12-66). For closed-loop flange root boundary, only this parameter needs to be defined; for open-loop flange root boundary, define this parameter as well as

VARIABLE	DESCRIPTION
	NB2 and NB3. The solver will automatically identify and automatically impose the necessary boundary constraints on all the nodes along the entire three-dimensional flange root boundary.
NB2	The ID of a node in the middle of the flange root boundary, see Figure 12-66. Define this parameter for open-loop flange root boundary only.
NB3	The end node ID on a flange root boundary. Define this parameter for open-loop flange root boundary only. The "path" formed by NB1, NB2 and NB3 can be in any direction, meaning NB1 and NB3 (Figure 12-66) can be interchangeable.
CHARLEN	Maximum flange height (Figure 12-66) to limit the search region for the boundary nodes along the flange root. This value should be set bigger than the longest width (height) of the flange; and it is needed in some cases. This parameter is now automatically calculated as of Revision 92860.
NDOUTER	A node ID on the outer flange (free end of the flange) boundary. This node helps search of nodes along the flange root, especially when holes are present in the flange area, see Figure 12-66.

Card 1b	1	2	3	4	5	6	7	8
Variable	THMX	THMN	EPSMX					
Type	I	I	I					
Default	10^{20}	0.0	10^{20}					

VARIABLE	DESCRIPTION
THMX	Maximum thickness beyond which elements are deleted; this is useful in removing wrinkling areas of the flange (shrink flange). Modified, unfolded flange outlines based on this parameter are stored in a file called trimcurve_upd.k, written using the *DEFINE_CURVE_TRIM_3D keyword. The modified flanges (before unfolding) are in a keyword file called mdfiedflangedpart.k, and the unmodified flange (unfolded) is in trimcurve_nmd.k, also written using keyword *DEFINE_CURVE_TRIM_3D. See Figure 12-66 for an explanation. Currently, the modified flange and curves are

VARIABLE	DESCRIPTION
	not smooth, which will be improved in the future. To convert between *DEFINE_CURVE_TRIM_3D and IGES format, refer to the figures in *INTERFACE_BLANKSIZE.
THMN	Minimum thickness below which elements are deleted; this is useful in removing overly thinned areas of the flange (stretch flange). Updated flange information based on this parameter is stored in the files listed above.
EPSMX	Maximum effective plastic strain beyond which elements are deleted; this is useful in removing flange areas with high effective plastic strains (stretch flange). Updated flange information based on this parameter is stored in the files listed above.

Introduction:

Unfolding of flanges is one of the first steps in a stamping die development process. Immediately after tipping, the binder and addenda are built to unfold the flanges. According to process considerations (trim conditions, draw depth, and material utilization, etc.), the addenda are built either parallel or perpendicular to the draw die axis, tangentially off the main surface of the breakline (see Appendix T), or in any combination of these three scenarios. Trim lines are developed by unfolding the flanges in the finished (hemmed or flanged) position onto these addenda. In some areas, the addendum length may need to be adjusted to accommodate the unfolded trim lines. Trim line development is very critical in hard tool development. Inaccurate trim lines lead to trim die rework.

Inputs and Outputs:

The inputs for the keyword are:

1. blank or flanges in the finished configuration, and,
2. the draw die surface in mesh.0.

Meshes for flanges should be of a quality similar to the blank mesh used for a forming simulation. In LS-PrePost 4.0, this kind of mesh can be created using *Mesh* → *Automesh* → *Size*. Element formulation 16 with NIP set to 5 is recommended for the blank. The output results, in terms of unfolding steps and final unfolded flanges, are stored in the d3plot files. The unfolded flange/trim curves can be created from the unfolded flanges using *Curve* → *Spline* → *From Mesh* → *By part*. Since the program uses an implicit statics solver, the double precision version of LS-DYNA must be used.

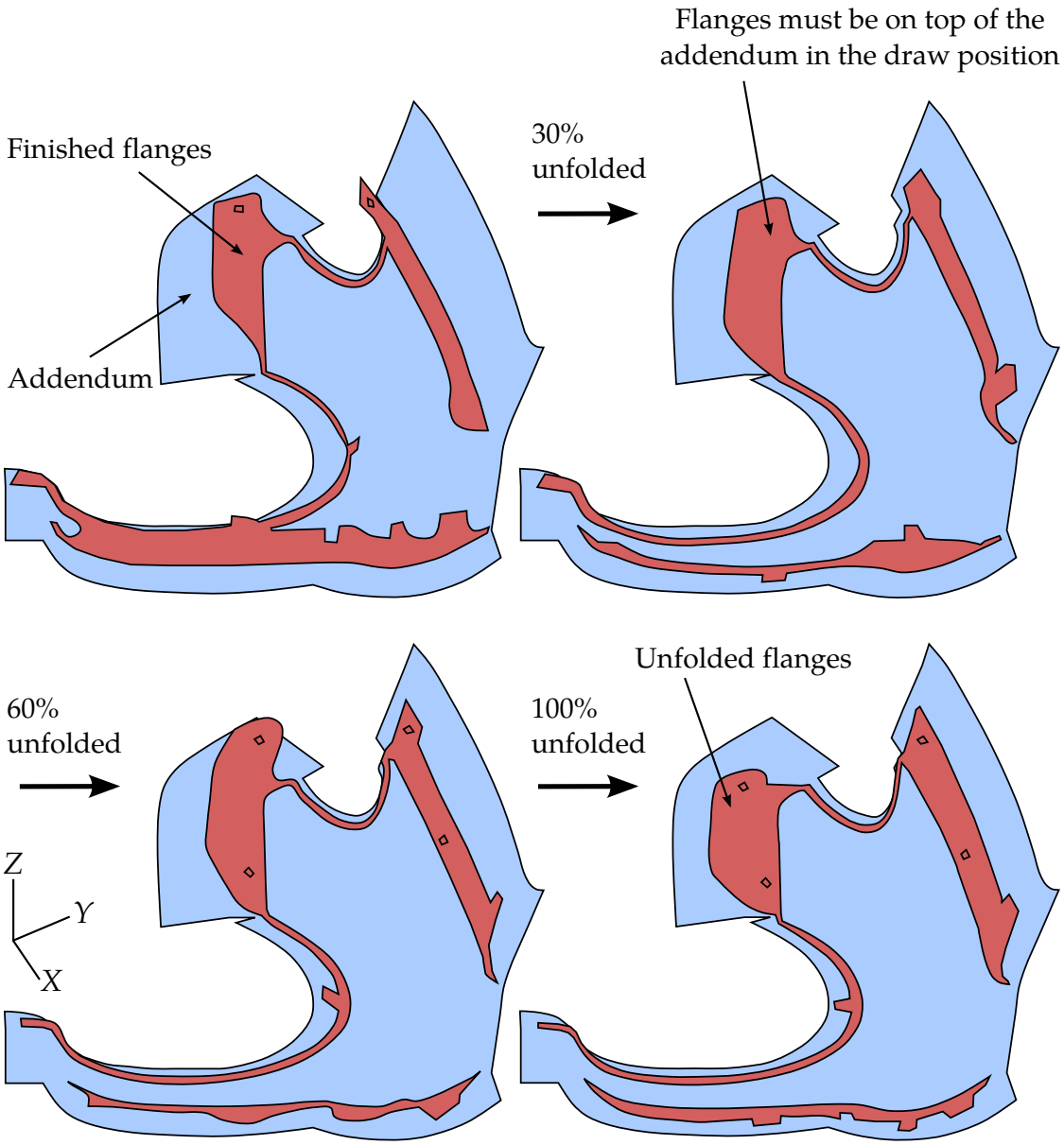


Figure 12-64. Flange unfolding progression of a fender outer (original model courtesy of NCAC at George Washington University).

Other Modeling Guidelines:

1. All addenda and flanges need to be oriented as if they are in a draw position, with the drawing axis parallel to the global Z-axis; specifically, the flanges need to be on top of the addendum, as noted in Figures 12-64, 12-65 and 12-66.

2. Normals of the to-be-unfolded flange side and tool surface side must be consistent and must face toward each other when the flange is unfolded, see Figure 12-66.

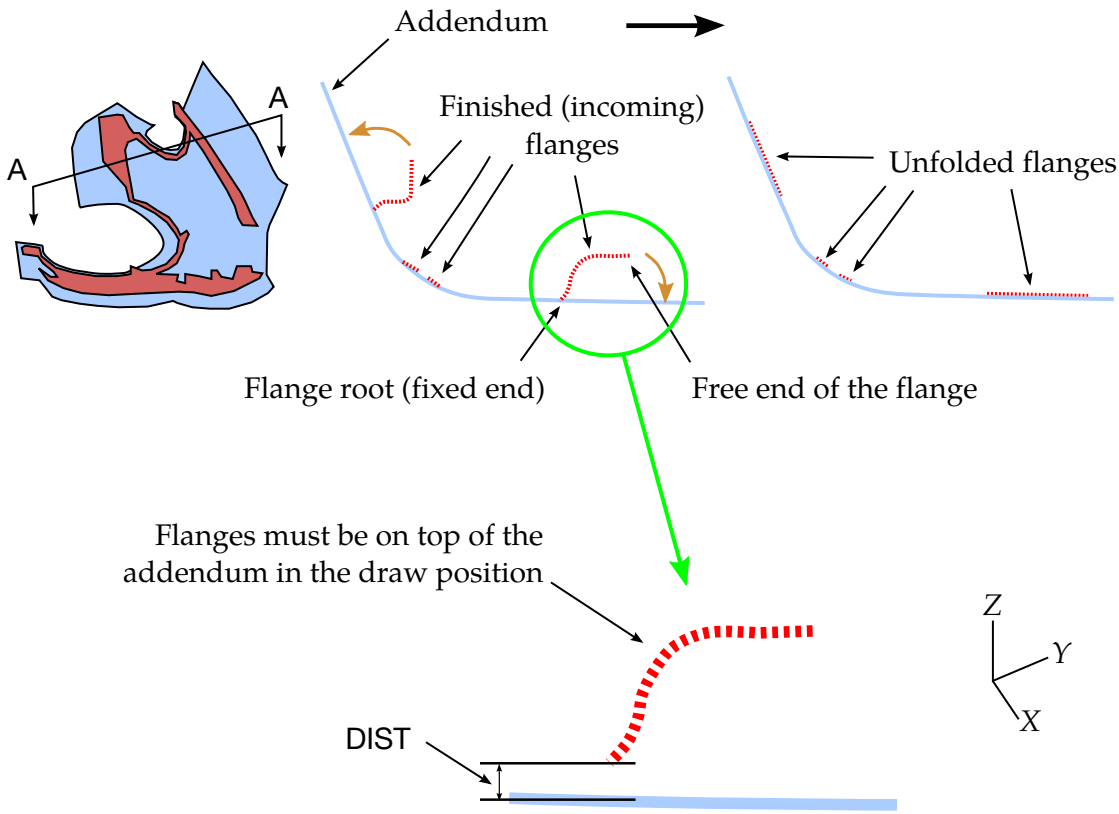


Figure 12-65. A section view showing flange unfolding before and after.

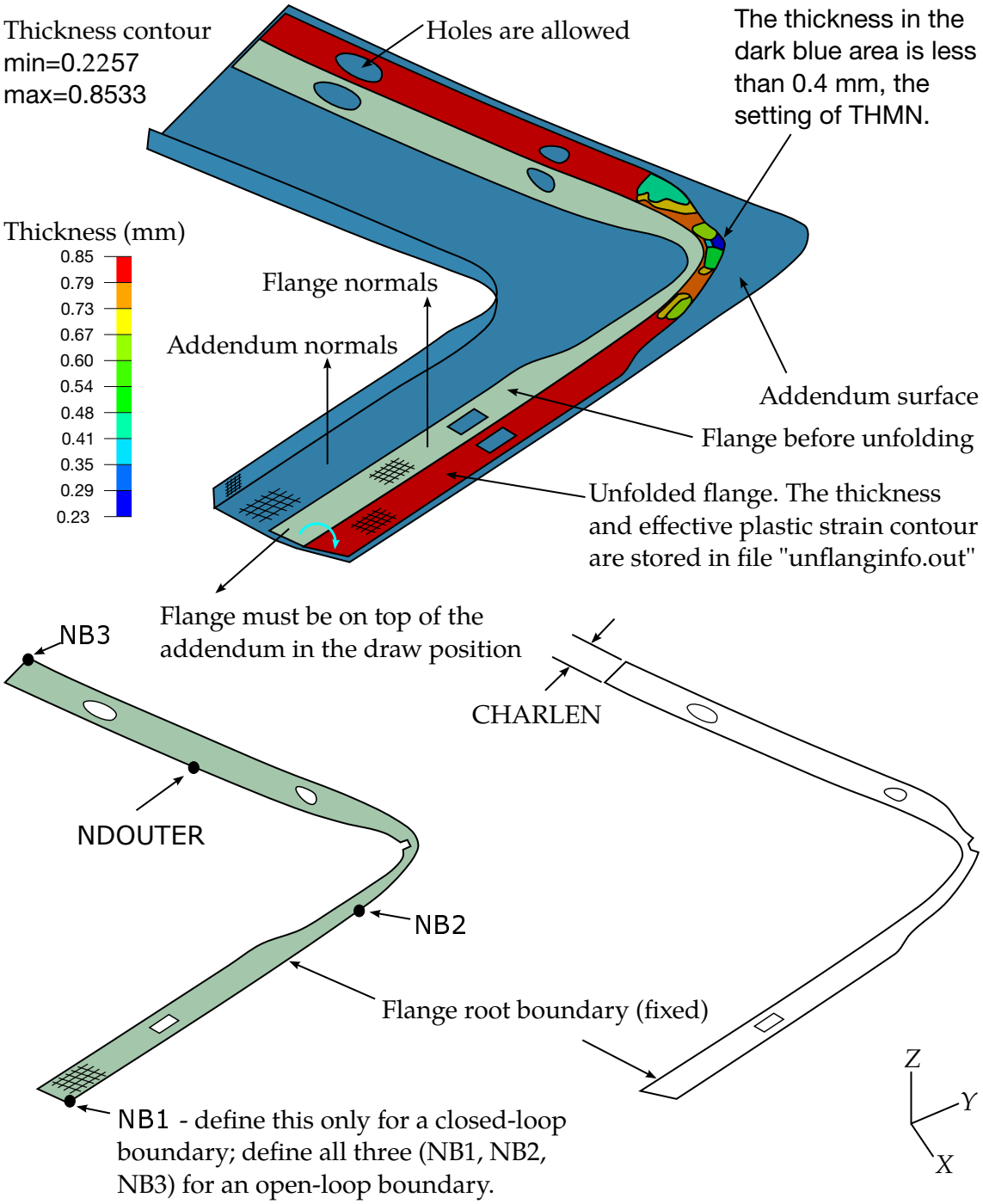
3. Holes in the blank are allowed only for ILINEAR = 2.

4. Adaptive re-meshing is not supported.

5. To-be-unfolded flange and tool meshes must not share the same nodes. This can be easily done using the mesh detaching feature under *EleTol* → *DetEle* in LS-PrePost.

6. Meshes of the flange part and rigid tool can slightly overlap each other, but *large amounts of overlap* (area of flange already on addendum surface) *is not allowed*. In LS-PrePost, the *EleTol* → *PtTrim* feature can trim off the overlapped flange portion. The curves used for the trimming can be obtained from the flange tangent curves on the addendum (which has a more regulated mesh pattern) using LS-PrePost's *Curve* → *Spline* → *Method From Mesh* → *By Edge* → *Prop* feature with appropriate angle definition. Furthermore, no holes are allowed in the overlapping area.

7. *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE should be used for the contact between the blank and tool. Negative tool offsets on the *CONTACT_… keyword is *not* supported.



The original flange is modified based on THMN = 0.4, and the mesh is stored in file "mdfiedflangedpart.k". Boundary curves can be created using LSPP under Curve/Spline/From mesh/by part.

Modified boundary curves on unfolded flange are stored in file "trimcurve_upd.k"; original boundary curves (without the corner cut out) are in "trimcurve_nmd.k".

Figure 12-66. Unfolding details and output files

8. The rigid tool (total fixed in *MAT_020) must be larger than the unfolded flanges, especially along symmetric lines. This may be obvious; nevertheless, it is sometimes overlooked.

9. Nodes along the flange root are automatically fixed by defining NB1, NB2, and NB3, as shown in Figure 12-66.

10. No "zigzag" along the flange root boundary, meaning that the boundary along the flange root must be smooth. This restriction is removed as of R10.0.

11. Symmetric boundary conditions are supported.

12. Thickness and effective plastic strain are stored in a file unflanginfo.out, which can be plotted in LS-PrePost 4.0; see Figure 12-66.

Application Examples:

A partial input deck is provided below for flange unfolding of a fender outer, modified from the original NCAC Taurus model. Shown in Figure 12-64 are the progressions of the unfolding process, where the finished flanges are to be unfolded onto the addendum (rigid body). A section view of the same unfolding before and after is shown in Figure 12-65. ILINEAR is set at 2 while DIST is left blank. The total number of elements is 1251 on the blank and 6600 on the tooling. It took less than 3 minutes on an 8-CPU (SMP) machine. Note that additional keywords, such as *CONTROL_IMPLICIT_FORMING, etc., are used. Termination criterion is set using the variable DELTAU in *CONTROL_-IMPLICIT_TERMINATION. Termination is reached when the relative displacement ratio criterion is met, as indicated in the messag file. A termination time of 10.0 (steps) is sufficient for most cases, but may need to be extended to satisfy DELTAU in some cases.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*KEYWORD
*INCLUDE
toolblankmesh.k
*CONTROL_FORMING_UNFLANGING
$  NOPTION      DVID   NUNBEND   STFBEND    STFCNT   IFLIMIT      DIST   ILINEAR
         1                 100       0.2      15.0       400                 2
$       NB1       NB2       NB3   CHARLEN   NDOUTER
        321       451       322      60.0      6245
*CONTROL_IMPLICIT_FORMING
1
*CONTROL_IMPLICIT_TERMINATION
$   DELTAU
$ set between 0.0005~0.001
    0.0005
*CONTROL_IMPLICIT_GENERAL
$   IMFLAG       DT0
         1     .1000
*CONTROL_IMPLICIT_SOLUTION
$  NSLOLVR    ILIMIT    MAXREF     DCTOL     ECTOL     RCTOL     LSTOL
         2         2      1100     0.100     1.e20     1.e20
$    dnorm   divflag   inistif
         0         2         0         1                 1
*PARAMETER
R ENDTIME       10.0
```

```
I elform          16
I nip              5
R bthick         1.0
*PARAMETER_EXPRESSION
R D3PLOTS    ENDTIME/60.0
*CONTROL_TERMINATION
&ENDTIME
*DATABASE_BINARY_D3PLOT
&D3PLOTS
*CONTROL_RIGID...
*CONTROL_HOURGLASS...
*CONTROL_BULK_VISCOSITY...
*CONTROL_SHELL...
*CONTROL_CONTACT
$    SLSFAC    RWPNAL    ISLCHK    SHLTHK    PENOPT    THKCHG     ORIEN
       0.01       0.0         2         1         4         0         4
$    USRSTR    USRFAC     NSBCS    INTERM     XPENE     SSTHK      ECDT   TIEDPRJ
          0         0        10         0       2.0         0
*CONTROL_ENERGY...
*CONTROL_ACCURACY...
*DATABASE_EXTENT_BINARY...
*SECTION_SHELL_TITLE
BLANK/FLANGE thickness and elform/nip specs.
&blksec      &elform     0.833      &nip       1.0
&bthick,&bthick,&bthick,&bthick
*PART...
*MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC...
*MAT_RIGID...
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE
$     SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBP
   &blkpid   &diepid         3         3
$        FS        FD        DC        VC       VDC    PENCHK        BT        DT
     0.125       0.0       0.0       0.0      20.0         0       0.0 1.000E+20
$      SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
        1.0       1.0       0.0
$      SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
          0
$    PENMAX    THKOPT    SHLTHK     SNLOG      ISYM     I2D3D    SLDTHK    SLDSTF

$      IGAP    IGNORE    DPRFAC    DTSTIF                       FLANGL
         2
*END
```

In Figure 12-66 (top), with THMN set at 0.4 mm, the stretch flange area of the corner, which has a thickness less than 0.4 mm, is removed; and the modified flange outlines are created accordingly (bottom). The partial input used is listed below.

```
*CONTROL_FORMING_UNFLANGING
$  NOPTION      DVID   NUNBEND   STFBEND    STFCNT   IFLIMIT      DIST   ILINEAR
         1                 100       0.2      15.0       400                 2
$       NB1       NB2       NB3   CHARLEN   NDOUTER
        321       451       322      60.0      6245
*CONTROL_FORMING_UNFLANGING_OUTPUT
$      THMX      THMN     EPSMX
                0.4
```

*CONTROL_FORMING_USER

Purpose: This keyword, along with *CONTROL_FORMING_POSITION, or *CONTROL_FORMING_TRAVEL, allow user to set up a stamping process simulation. From this card various model parameters may be specified:

- material properties,
- material model,
- tooling kinematics,
- mesh adaptivity
- D3PLOT generation

NOTE: This option has been deprecated in favor of *CONTROL_FORMING_AUTOPOSITION_PARAMETER).

Card 1	1	2	3	4	5	6	7	8
Variable	BLANK	TYPE	THICK	R00	R45	R90	AL/FE	UNIT
Type	I	I	F	F	F	F	A	I
Default	none	0	none	1.0	R00	R00	F	1

Card 2	1	2	3	4	5	6	7	8
Variable	LCSS	K	N	E	DENSITY	PR	FS	MTYPE
Type	I	F	F	F	F	F	F	I
Default	none	none	none	none	none	none	0.1	37

Card 3	1	2	3	4	5	6	7	8
Variable	PATERN	VMAX	AMAX	LVLADA	SIZEADA	ADATIMS	D3PLT	GAP
Type	I	F	F	I	F	I	I	F
Default	1	1000.0	500000.	0	0	0	10	1.1t

VARIABLE	DESCRIPTION
BLANK	PID of a sheet blank, as in *PART.
TYPE	Flag of part or part set ID for the blank: EQ.0: Part ID. EQ.1: Part set ID.
THICK	Thickness of the blank. This variable is ignored if the thickness is already defined in *SECTION_SHELL.
R00, R45, R90	Material anisotropic parameters. For transverse anisotropy the R value is set to the average value of R00, R45, and R90.
AL/FE	This parameter is used to define the Young's Modulus, E, and density, ρ, for the sheet blank. If this variable is defined, E and ρ will be found by using the proper unit, as listed in Table 8.1, under *CONTROL_FORMING_TEMPLATE. EQ.A: the blank is aluminum EQ.F: the blank is steel (default)
UNIT	Units adopted in this simulation. Define a number between 1 and 10. Table 8.1 is used to determine the value for UNIT. This unit is used to obtain proper values for punch velocity, acceleration, time step, and physical and material properties.
LCSS	If the material for the blank has not been defined, this curve will be used to define the stress-strain relation. Otherwise, this variable is ignored.
PREBD	"Pull-over" distance for the upper and lower binders after closing in a 4-piece stretch draw, as shown in Figure 12-62.

VARIABLE	DESCRIPTION
K	Strength coefficient for exponential hardening ($\overline{\sigma} = k\bar{\varepsilon}^n$). If LCSS is defined, or if a blank material is defined with *MAT_036 or *MAT_-037, this variable is ignored.
N	Exponent for exponential hardening ($\overline{\sigma} = k\bar{\varepsilon}^n$). If LCSS is defined, or if a blank material is defined with *MAT_036 or *MAT_037, this variable is ignored.
E	Young's Modulus. If AL/FE is user defined, E is unnecessary.
DENSITY	Material density of the blank. If AL/FE is defined, this variable is unnecessary.
PR	Poisson's ratio. If AL/FE is user defined, this variable is unnecessary.
FS	Coulomb friction coefficient. If contact is defined with *CONTACT_FORMING_..., this variable is ignored.
MTYP	Material model identification number, for example, 36 for *MAT_-036 and 37 for *MAT_037. Currently only material models 36 and 37 are supported.
PATERN	Velocity profile of the moving tool. If the velocity and the profile are defined by *BOUNDARY_PRESCRIBED_MOTION_RIGID, and *DEFINE_CURVE, this variable is ignored. EQ.1: Ramped velocity profile. EQ.2: Smooth velocity curve.
VMAX	The maximum allowable tool travel velocity.
AMAX	The maximum allowable tool acceleration.
LVLADA	Maximum mesh adaptive level.
SIZEADA	Minimum element size permitted during mesh adaptivity.
ADATIMS	Total number of adaptive steps during the simulation.
D3PLT	The total number of output states in the D3PLOT database.
GAP	Minimum gap between two closing tools at home position, in the travel direction of the moving tool. This variable will be used for *CONTROL_FORMING_POSITION.

Keyword examples:

A partial keyword example provided below is for tools in their home positions in a simple 2-piece crash forming die. A steel sheet blank PID 1, is assigned with a thickness of 0.76mm (UNIT = 1) and *MAT_037 with anisotropic values indicated, to follow hardening curve of 90903, form in a 'ramped' type of velocity profile with maximum velocity of 5000mm/s and acceleration of 500000.0 mm/s2, adapt mesh 5 levels with smallest adapted element size of 0.9 for a total of 20 adaptive steps, create a total of 15 post-processing states, and to finish forming with a final gap of 1.1mm between the tools (PID3 and 5) at home position. The upper tool with PID 3 is to be moved back in Z axis to clear the interference with the blank before close toward the lower tool of target PID 5.

```
*CONTROL_FORMING_USER
$    BLANK      TYPE     THICK       R00       R45       R90     AL/FE      UNIT
         1         0      0.76       1.5       1.6       1.4         F         1
$     LCSS         K         N         E   DENSITY        PR        FS     MTYPE
     90903                                                                   37
$  PATTERN      VMAX      AMAX    LVLADA   SIZEADA   ADATIMS     D3PLT       GAP
         1    5000.0  500000.0         5       0.9      20.0      15.0       1.1
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_POSITION
$ This is for tools in home position.
$      PID   PREMOVE    TARGET
         3                   5
```

The following partial keyword example is for tools already positioned in relationship to the blank and ready to close. All assigned properties for the blank remain the same. Here the upper tool PID3 is not going to be moved back, but instead it will move forward to close with the lower tool of target PID 5 in the direction specified by the vector ID 999.

```
*CONTROL_FORMING_USER
         1         0       1.0       1.5       1.6       1.4         F         1
     90903                                                                   37
         1    5000.0  500000.0         5       0.9      20.0      15.0       1.1
*CONTROL_FORMING_TRAVEL
$      PID       VID    TRAVEL    TARGET       GAP     PHASE    FOLLOW
         3       999                   5       1.1         1
```

Revision information:

This keyword is available starting in LS-DYNA Revision 48319.

*CONTROL_FREQUENCY_DOMAIN

Purpose: Set global control flags and parameters for frequency domain analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	REFGEO	MPN	MCF					
Type	I	F	I					
Default	0	0.0	0					

VARIABLE	DESCRIPTION
REFGEO	Flag for reference geometry in acoustic eigenvalue analysis: EQ.0: use original geometry ($t = 0$), EQ.1: use deformed geometry at the end of transient analysis.
MPN	Large mass added per node to be used in large mass method for enforced motion.
MCF	Flag for writing out MCF (Modal Coefficient File) from SSD analysis, to be used in fatigue analysis with nCode Designlife: EQ.0: Don't write out MCF. EQ.1: Write out MCF.

Remarks:

1. **Deformed Geometry Eigenvalue Analysis.** For acoustic eigenvalue analysis (see keyword *FREQUENCY_DOMAIN_ACOUSTIC_ FEM_EIGENVALUE), sometimes it is desired to extract the eigenvalues at the end of transient analysis, based on the deformed geometry. This is useful to study the effect of loading history on acoustic eigenvalues. In this case, one can set REFGEO = 1 to use the deformed geometry at the end of transient analysis.

2. **Large Mass Method.** For enforced motion excitation (e.g. nodal acceleration, velocity, or displacement) in FRF, SSD, or random vibration analysis, one can use the large mass method to compute the response. With the large mass method, the user attaches a large mass to the nodes under excitation. LS-DYNA converts the enforced motion excitation to nodal force on the same nodes in the same direction to produce the desired enforced motion. MPN is the large mass

attached to each node under excitation (usually it is in the range of 10^5-10^7 times of the original mass of the entire structure). The large mass still needs to be applied to the nodes using the keyword *ELEMENT_MASS_*{OPTION}*.

The large nodal force p is computed as shown in the following table:

Nodal Motion	Nodal Force
Acceleration	$p = m_L \ddot{u}$
Velocity	$p = i\omega m_L \dot{u}$
Displacement	$p = -\omega^2 m_L u$

where ω is the round frequency, m_L is the large mass attached to each node (MPN), and $\ddot{u}$, $\dot{u}$ and u are the enforced acceleration, velocity and displacement, respectively.

***CONTROL_HOURGLASS**_*{OPTION}*

Available options include:

<BLANK>

936

The "936" option switches the hourglass formulation for shells so that it is identical to that used in LS-DYNA version 936. The modification in the hourglass control from version 936 was to ensure that all components of the hourglass force vector are orthogonal to rigid body rotations. However, problems that run under version 936 sometimes lead to different results in versions 940 and later. This difference in results is primarily due to the modifications in the hourglass force vector. Versions released after 936 should be more accurate.

Purpose: Redefine the default values of hourglass control type and coefficient.

Card 1	1	2	3	4	5	6	7	8
Variable	IHQ	QH						
Type	I	F						
Default	Rem 1	0.1						
Remarks	1,2	3,4						

VARIABLE	DESCRIPTION
IHQ	Default hourglass control type: EQ.0: See Remark 1, EQ.1: Standard viscous form (may inhibit body rotation if solid element shapes are skewed), EQ.2: Viscous form, Flanagan-Belytschko integration for solid elements, EQ.3: Viscous form, Flanagan-Belytschko with exact volume integration for solid elements, EQ.4: Stiffness form of type 2 (Flanagan-Belytschko),

VARIABLE	DESCRIPTION
	EQ.5: Stiffness form of type 3 (Flanagan-Belytschko) for solid elements, EQ.6: Belytschko-Bindeman [1993] assumed strain co-rotational stiffness form for 2D and 3D solid elements, EQ.7: Linear total strain form of type 6 hourglass control. EQ.8: Activates full projection warping stiffness for shell formulations 9, 16 and -16. A speed penalty of 25% is common for this option. EQ.9: Puso [2000] enhanced assumed strain stiffness form for 3D hexahedral elements, EQ.10: Cosserat Point Element (CPE) developed by Jabareen and Rubin [2008] for 3D hexahedral elements and Jabareen et.al [2013] for 10-noded tetrahedral elements. See Remark 6.
QH	Default hourglass coefficient.

Remarks:

1. **Hourglass control types.** Hourglass control is viscosity or stiffness that is added to quadrilateral shell elements and hexahedral solid elements that use reduced integration. It also applies to formulation 1 tshells. Without hourglass control, these elements would have zero energy deformation modes which could grow large and destroy the solution. *CONTROL_HOURGLASS can be used to redefine the default values of the hourglass control type and coefficient. If omitted or if IHQ = 0, the default hourglass control types are as follows:

   a) For shells: viscous type for explicit; stiffness type for implicit.

   b) For solids: type 2 for explicit; type 6 for implicit.

   c) For tshell formulation 1: type 2.

   These default values are used unless HGID on *PART is used to point to *HOURGLASS data which overrides the default values for that part.

   For explicit analysis, shell elements can be used with viscous hourglass control, (IHQ = 1 = 2 = 3) or stiffness hourglass control (IHQ = 4 = 5). Only shell forms 9, 16 and -16 use the warping stiffness invoked by IHQ = 8. For implicit analysis, the viscous form is unavailable.

For explicit analysis, hexahedral elements can be used with any of the hourglass control types except IHQ = 8. For implicit analysis, only IHQ = 6, 7, 9, and 10 are available.

If IHQ is set to a value that is invalid for some elements in a model, then the hourglass control type for those elements is automatically reset to a valid value. For explicit analysis, if IHQ = 6, 7, 9, or 10, then shell elements will be switched to type 4 except for forms 16 and -16 shells that are switched to type 8. If IHQ = 8, then solid elements and shell elements that are not form 16 will be switched to type 4. For implicit analysis, if IHQ = 1-5, then solid elements will be switched to type 6, and if IHQ = 1, 2, 3, 6, 7, 9, or 10, then shell elements will be switched to type 4.

2. **Viscous hourglass control**. Viscous hourglass control has been used successfully with shell elements when the response with stiffness based hourglass control was overly stiff. As models have grown more detailed and are better able to capture deformation modes, there is less need for viscous forms. To maintain back compatibility, viscous hourglass control remains the default for explicit analysis, but there may be better choices, particularly the newer forms for bricks (6, 7, 9, and 10).

3. **Hourglass coefficient stability.** QH is a coefficient that scales the hourglass viscosity or stiffness. With IHQ = 1 through 5 and IHQ = 8, values of QH that exceed 0.15 may cause instabilities. Hourglass types 6, 7, 9, and 10 will remain stable with larger QH and can work well with QH = 1.0 for many materials. However, for plasticity models, a smaller value such as QH = 0.1 may work better since the hourglass stiffness is based on elastic properties.

4. **Hourglass control for hexahedral elements.** Hourglass types 6, 7, 9, and 10 for hexahedral elements are based on physical stabilization using an enhanced assumed strain method. When element meshes are not particularly skewed or distorted, their behavior may be very similar and all can produce accurate coarse mesh bending results for elastic material with QH = 1.0. However, form 9 gives more accurate results for distorted or skewed elements. In addition, for materials 3, 18 and 24 there is the option to use a negative value of QH. With this option, the hourglass stiffness is based on the current material properties, i.e., the plastic tangent modulus, and scaled by |QH|.

5. **IHQ type 7.** Hourglass type 7 is a variation on form 6. Instead of updating the hourglass forces incrementally using the current stiffness and an increment of deformations, the total hourglass deformation is evaluated each cycle. This ensures that elements always spring back to their initial geometry if the load is removed and the material has not undergone inelastic deformation. Hourglass type 7 is recommended for foams that employ *INITIAL_FOAM_REFER-

ENCE_GEOMETRY. However, the CPU time for type 7 is roughly double that for type 6, so it is only recommended when needed.

6. **IHQ type 10.** Hourglass type 10 for 1-point solid elements or 10-noded tetrahedron of type 16 are strucural elements based on Cosserat point theory that allows for accurate representation of elementary deformation modes (stretching, bending and torsion) for general element shapes and hyperelastic materials. To this end, the theory in Jabareen and Rubin [2008] and Jabareen et.al [2013] has been generalized in the implementation to account for any material response. The deformation is separated into a homogenous and an inhomogeneous part where the former is treated by the constitutive law and the latter by a hyperelastic formulation that is set up to match analytical results for the deformation modes mentioned above. Tests have shown that the element is giving more accurate results than other hexahedral elements for small deformation problems and more realistic behavior in general.

*CONTROL_IMPLICIT

LS-DYNA's implicit mode may be activated in two ways. Using the *CONTROL_IMPLICIT_GENERAL keyword, a simulation may be flagged to run entirely in implicit mode. Alternatively, an explicit simulation may be seamlessly switched into implicit mode at the termination time using the *INTERFACE_SPRINGBACK_SEAMLESS keyword. The seamless switching feature is intended to simplify metal forming springback calculations, where the forming phase can be run in explicit mode, followed immediately by an implicit static springback simulation. In case of difficulty, restart capability is supported.

Several control cards are available to support implicit analysis. Default values are carefully selected to minimize input necessary for most simulations. The implicit control cards follow:

*CONTROL_IMPLICIT_AUTO

*CONTROL_IMPLICIT_BUCKLE

*CONTROL_IMPLICIT_CONSISTENT_MASS

*CONTROL_IMPLICIT_DYNAMICS

*CONTROL_IMPLICIT_EIGENVALUE

*CONTROL_IMPLICIT_FORMING

*CONTROL_IMPLICIT_GENERAL

*CONTROL_IMPLICIT_INERTIA_RELIEF

*CONTROL_IMPLICIT_JOINTS

*CONTROL_IMPLICIT_MODAL_DYNAMIC

*CONTROL_IMPLICIT_MODAL_DYNAMIC_DAMPING

*CONTROL_IMPLICIT_MODAL_DYNAMIC_MODE

*CONTROL_IMPLICIT_MODES

*CONTROL_IMPLICIT_ORDERING

*CONTROL_IMPLICIT_RESIDUAL_VECTOR

*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS

*CONTROL_IMPLICIT_SOLUTION

*CONTROL_IMPLICIT_SOLVER

*CONTROL_IMPLICIT_SSD_DIRECT

*CONTROL_IMPLICIT_STABILIZATION

*CONTROL_IMPLICIT_STATIC_CONDENSATION

*CONTROL_IMPLICIT_TERMINATION

***CONTROL_IMPLICIT_AUTO**_{OPTION}

Available options for *OPTION* include:

<BLANK>

DYN

SPR

Purpose: Define parameters for automatic time step control during implicit analysis (see also *CONTROL_IMPLICIT_GENERAL). The DYN option allows setting controls specifically for the dynamic relaxation phase. The SPR option allows setting controls specifically for the springback phase.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_AUTO. For example, if IAUTO = 1 for *CONTROL_IMPLICIT_AUTO and the value of IAUTO is left blank for *CONTROL_IMPLICIT_AUTO_DYN, then the value of IAUTO for *CONTROL_IMPLICIT_AUTO_DYN will also be 1.

Card 1	1	2	3	4	5	6	7	8
Variable	IAUTO	ITEOPT	ITEWIN	DTMIN	DTMAX	DTEXP	KFAIL	KCYCLE
Type	I	I	I	F	F	F	I	I
Default	0	11	5	DT/1000.	DT×10.	none	1	none

Mid-Step Residual Optional Card. Define this card if and only if IAUTO.EQ.3

Card 2	1	2	3	4	5	6	7	8
Variable	HCMIN	HCMAX	HMMIN	HMMAX	HNTMAX	HNRMAX	HRTMAX	HRRMAX
Type	F	F	F	F	F	F	F	F
Default	2× RCTOL	10× RCTOL	2× RMTOL	10× RMTOL	Infinity	Infinity	Infinity	Infinity

VARIABLE	DESCRIPTION
IAUTO	Automatic time step control flag EQ.0: Constant time step size EQ.1: Automatically adjust time step size EQ.2: Automatically adjust time step size and synchronize with thermal mechanical time step. EQ.3: Same as 1, but accounting for mid step residual values with respect to parameters on card 2 and according to the Remark for IAUTO. LT.0: Curve ID = (-IAUTO) gives time step size as a function of time. If specified, DTMIN and DTMAX will still be applied.
ITEOPT	Optimum equilibrium iteration count per time step. See Figure 12-67.
ITEWIN	Allowable iteration window. If iteration count is within ITEWIN iterations of ITEOPT, step size will not be adjusted for the next step.
DTMIN	Minimum allowable time step size. Simulation stops with error termination if time step falls below DTMIN. LT.0: Enable automatic key point generation. Minimum allowable time step is |DTMIN|.
DTMAX	Maximum allowable time step size. LT.0: Curve ID = (-DTMAX) gives max step size as a function of time. Also, the step size is adjusted automatically so that the time value of each point in the curve is reached exactly

VARIABLE	DESCRIPTION
	(see Figures 12-68 and 0-3).
DTEXP	Time interval to run in explicit mode before returning to implicit mode. Applies only when automatic implicit-explicit switching is active (IMFLAG = 4 or 5 on *CONTROL_IMPLICIT_GENERAL). Also, see KCYCLE. EQ.0: Defaults to the current implicit time step size. LT.0: Curve ID = (-DTEXP) gives the time interval as a function of time.
KFAIL	Number of failed attempts to converge implicitly for the current time step before automatically switching to explicit time integration. Applies only when automatic implicit-explicit switching is active. The default is one attempt. If IAUTO = 0, any input value is reset to unity.
KCYCLE	Number of explicit cycles to run in explicit mode before returning to the implicit mode. The actual time interval that is used will be the maximum between DTEXP and KCYCLE × (latest estimate of the explicit time step size).
HCMIN, HCMAX	Mid-point relative Euclidian residual norm min and max tolerance, to be seen as a confidence interval, see Remark for IAUTO = 3. Only active if RCTOL on *CONTROL_IMPLICIT_SOLUTION is set.
HMMIN, HMMAX	Mid-point relative maximum residual norm min and max tolerance, to be seen as a confidence interval, see Remark for IAUTO = 3. Only active if RMTOL on *CONTROL_IMPLICIT_SOLUTION is set.
HNTMAX	Mid-point absolute nodal translational norm tolerance; see Remark for IAUTO = 3. Only active if NTTOL on *CONTROL_IMPLICIT_SOLUTION is set.
HNRMAX	Mid-point absolute nodal rotational norm tolerance; see Remark for IAUTO = 3. Only active if NRTOL on *CONTROL_IMPLICIT_SOLUTION is set.
HRTMAX	Mid-point absolute rigid body translational norm tolerance; see Remark for IAUTO = 3. Only active if RTTOL on *CONTROL_IMPLICIT_SOLUTION is set.

VARIABLE	DESCRIPTION
HRRMAX	Mid-point absolute rigid body rotational norm tolerance; see Remark for IAUTO = 3. Only active if RRTOL on *CONTROL_IMPLICIT_SOLUTION is set.

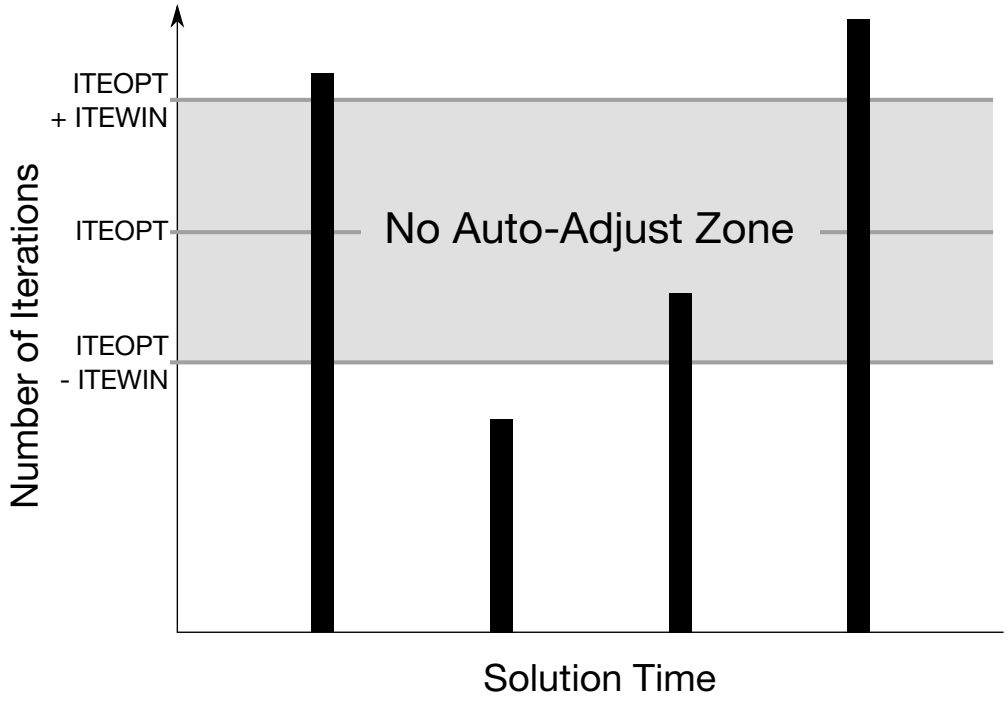


Figure 12-67. Iteration Window as defined by ITEOPT and ITEWIN.

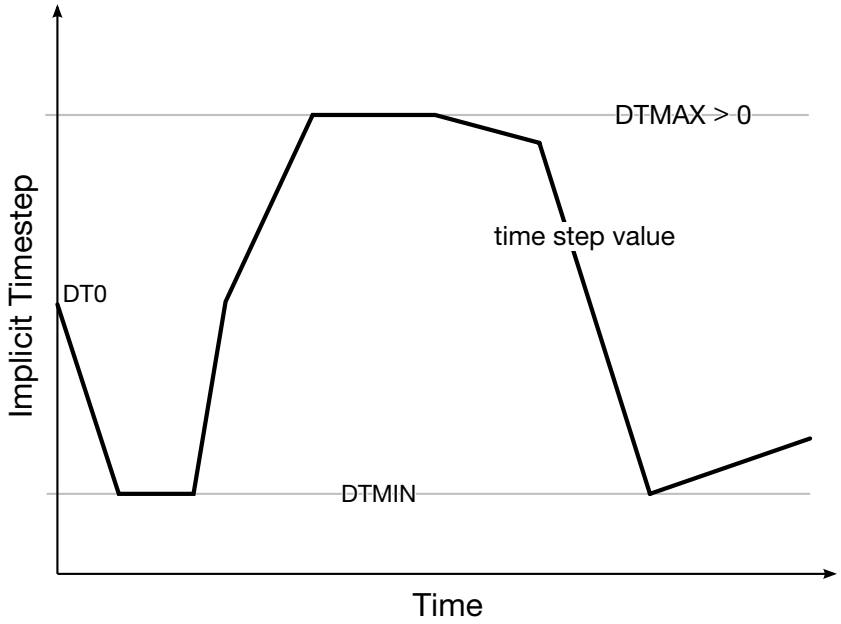


Figure 12-68. The implicit time step size changes continuously as a function of convergence within the bounds set by DTMIN and DTMAX

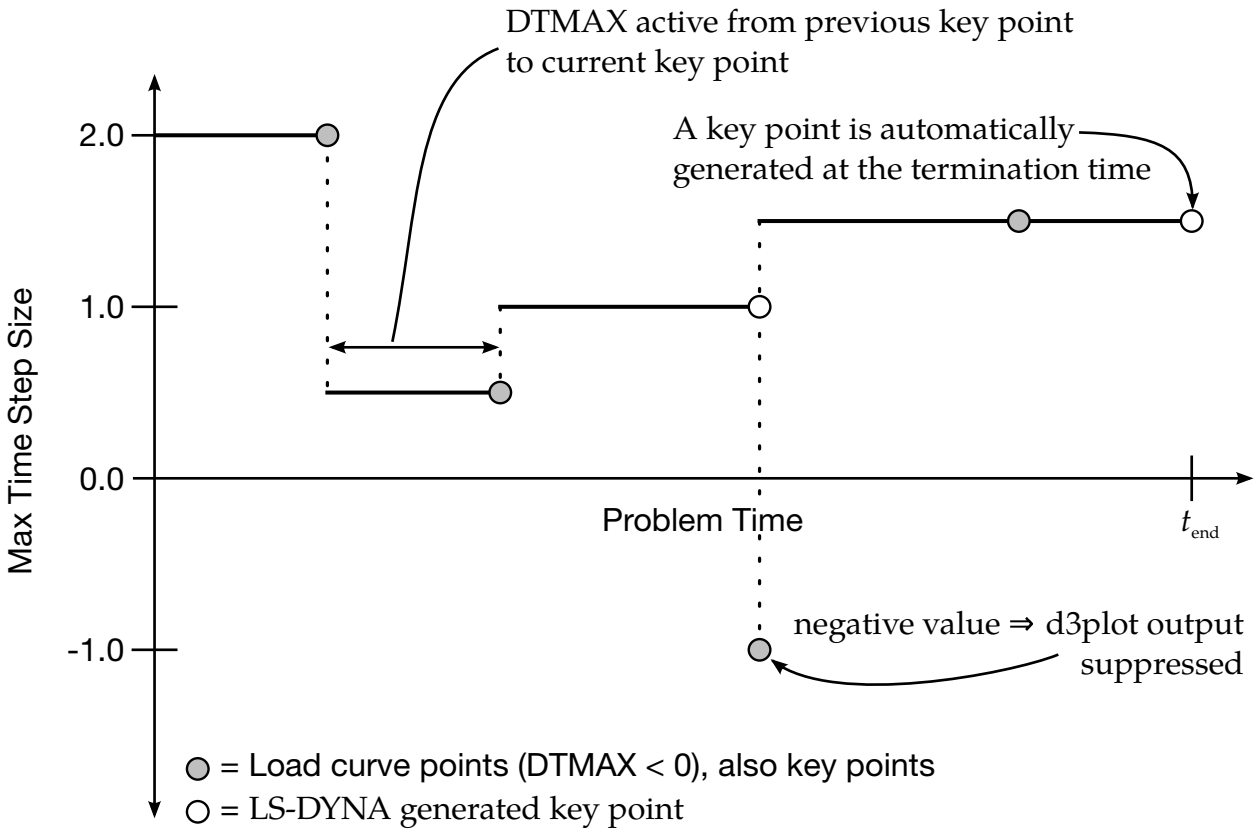


Figure 0-3. *DTMAX < 0.* The maximum time step is set by a load curve of LCID = −DTMAX interpolated using piecewise constants. The abscissa values of the load curve determine the set of *key points*. The *absolute value* of the ordinate values set the maximum time step size. *Key points* are special time values for which the integrator will adjust the time step so as to reach *exactly*. For each key point with a positive function value, LS-DYNA will write the state to the binary database.

Remarks:

VARIABLE	REMARK
IAUTO	The default for IAUTO depends on the analysis type. For "spring-back" analysis, automatic time step control and artificial stabilization are activated by default. IAUTO = 3 accounts for the residual norm at the mid-point geometry between converged steps. We have included below a numbered set of conditions. These conditions are used to adjust the time step according to the following algorithm:

VARIABLE	REMARK

1. The time step is *increased* if either of the following criteria is met simultaneously:
   - 1b, 2b, 3, 4a, or
   - 1a, 2a, 3, 4b.
2. If the time step is not increased by these conditions, then it is decreased if any of the following above criteria is *violated*
   - 1b or
   - 2b or
   - 3 or
   - 4b.

The Criteria Mentioned Above

1. The relative Euclidian residual norm at the mid-point is less than
   a) HCMIN
   b) HCMAX.
2. The relative maximum residual norm at the mid-point is less than
   a) HMMIN
   b) HMMAX.
3. At least one of the following conditions is satisfied at the midpoint
   a) The max norm of nodal translation < HNTMAX
   b) The max norm of nodal rotation < HNRMAX
   c) The max norm of rigid body translation < HRTMAX
   d) The max norm of rigid body rotation < HRRMAX
4. The number of iterations for convergence is
   a) less than ITEOPT − ITEWIN
   b) or equal to ITEOPT + ITEWIN.

Note

IAUTO = 1 is obtained as a special case for

$$HCMIN = HMMIN = 0$$

VARIABLE	REMARK
	and $$HCMAX = HMMAX = HNTMAX = HNRMAX = HRTMAX = HRRMAX = \infty$$ The remaining parameters below works the same way as for IAUTO = 1.
ITEOPT	With IAUTO = 1 or 2, the time step size is adjusted if convergence is reached in a number of iterations that falls outside the specified "iteration window", increasing after "easy" steps, and decreasing after "difficult" but successful steps. ITEOPT defines the midpoint of the iteration window. A value of ITEOPT = 30 or more can be more efficient for highly nonlinear simulations by allowing more iterations in each step, hence fewer total steps.
DTMIN	Often specific simulation times that are of special interest, such as when a peak load value occurs, require the simulation to be solved at exactly that time. The automatic key point generation feature attempts to automatically find such times by analyzing the load curves for stationary points in time. The keywords whose load curves are analyzed along with what type of curve value are shown below. • *INITIAL_AXIAL_FORCE_BEAM - global max • *INITIAL_STRESS_SECTION - global max • *LOAD_NODE_POINT - local min/max • *LOAD_NODE_SET - local min/max • *LOAD_SEGMENT - local min/max • *LOAD_SEGMENT_SET - local min/max • *LOAD_SHELL - local min/max • *LOAD_SHELL_SET - local min/max • *BOUNDARY_PRESCRIBED_MOTION_NODE - local min/max • *BOUNDARY_PRESCRIBED_MOTION_SET - local min/max For *CONTROL_IMPLICIT_DYNAMICS, the birth, death, and burial time are key points. For *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR, MPAR1 is a key point when IGNORE = 3. The start time and end time of the simulation are also

VARIABLE	REMARK
	key points. To enable automatic key point generation, enter DTMIN negated.
DTMAX	To strike a particular simulation time exactly, create a key point curve (Figure 0-3) and enter DTMAX = -(curve ID). This is useful to guarantee that important simulation times, such as when peak load values occur, are reached exactly. A user defined key point curve can be used together with automatic key point generation. In that case all key points are merged into a compound list of key points.
DTEXP	When the automatic implicit-explicit switching option is activated (IMFLAG = 4 or 5 on *CONTROL_IMPLICIT_GENERAL), the solution method will begin as implicit, and if convergence of the equilibrium iterations fails, automatically switch to explicit for a time interval of DTEXP. A small value of DTEXP should be chosen so that significant dynamic effects do not develop during the explicit phase, since these can make recovery of nonlinear equilibrium difficult during the next implicit time step. A reasonable starting value of DTEXP may equal several hundred explicit time steps.

*CONTROL_IMPLICIT_BUCKLE

Purpose: Activate implicit buckling analysis when termination time is reached (see also *CONTROL_IMPLICIT_GENERAL). Optionally, buckling analyses are performed at intermittent times.

Card 1	1	2	3	4	5	6	7	8
Variable	NMODE	BCKMTH						
Type	I	I						
Default	0	↓						

VARIABLE	DESCRIPTION
NMODE	Number of buckling modes to compute: EQ.0: none (default) GT.0: compute n lowest buckling modes LT.0: curve ID = (-NEIG) used for intermittent buckling analysis
BCKMTH	Method used to extract buckling modes: EQ.1: Use Block Shift and Invert Lanczos. Default of all problems not using *CONTROL_IMPLICIT_INERTIA_RELIEF. EQ.2: Use Power Method. Only valid option for problems using *CONTROL_IMPLICIT_INERTIA_RELIEF. Optional for other problems. See Remarks.

Remarks:

1. **Buckling Eigenproblem.** Buckling analysis is performed at the end of a static implicit simulation or at specified times during the simulation. The simulation may be linear or nonlinear but must be implicit. After loads have been applied to the model, the buckling eigenproblem is solved:

$$[\mathbf{K}_M + \lambda \mathbf{K}_G]\{u\} = 0$$

where $\mathbf{K}_M$ is the material tangent stiffness matrix, and the geometric or initial stress stiffness matrix $\mathbf{K}_G$ is a function of internal stress in the model. The lowest

n eigenvalues and eigenvectors are computed. The eigenvalues, written to text file eigout, represent multipliers to the applied loads which give buckling loads. The eigenvectors, written to binary database d3eigv, represent buckling mode shapes. These modes can be viewed and animated using LS-PrePost.

2. **NMODE.** When NMODE > 0, eigenvalues will be computed at the termination time and LS-DYNA will terminate.

   When NMODE < 0, an intermittent buckling analysis will be performed. This is a transient simulation during which loads are applied, with buckling modes computed periodically during the simulation. Changes in geometry, stress, material, and contact conditions will affect the buckling modes. The transient simulation must be implicit. The curve ID = -NMODE indicates when to extract the buckling modes, and how many to extract. Define one curve point at each desired extraction time, with a function value equal to the number of buckling modes desired at that time. A d3plot database will be produced for the transient solution results. Consecutively numbered d3eigv and eigout databases will be produced for each intermittent extraction. The extraction time is indicated in each database's analysis title.

3. **Solver.** The buckling modes can be computed using either Block Shift and Invert Lanczos or the Power Method. It is strongly recommended that the Block Shift and Invert Lanczos method is used as it is a more powerful and robust algorithm. For problems using *CONTROL_IMPLICIT_INERTIA_RELIEF the Power Method must be used and any input value for BCKMTH will be overridden with the required value of 2. There may be some problems, which are not using *CONTROL_IMPLICIT_INERTIA_RELIEF, where the Power Method may be more efficient than Block Shift and Invert Lanczos. But the Power Method is not as robust and reliable as Lanczos and results should be verified. Furthermore convergence of the Power Method is better for buckling problems where the expected buckling mode is close to one in magnitude and the dominant mode is separated from the secondary modes. The number of modes extracted via the Power Method should be kept in the range of 1 to 5.

4. **Geometric Stiffness.** The geometric stiffness terms needed for buckling analysis will be automatically computed when the buckling analysis time is reached, regardless of the value of the geometric stiffness flag IGS on *CONTROL_IMPLICIT_GENERAL.

5. **Executable Precision.** A double precision executable should be used for best accuracy in buckling analysis.

6. **Applicable Parameters.** Parameters CENTER, LFLAG, LFTEND, RFLAG, RHTEND and SHFSCL from *CONTROL_IMPLICIT_EIGENVALUE are applicable to buckling analysis. For buckling analysis CENTER, LFTEND, RHTEND and SHFSCL are in units of the eigenvalue spectrum.

*CONTROL_IMPLICIT_CONSISTENT_MASS

Purpose: Use the consistent mass matrix in implicit dynamics and eigenvalue solutions.

Card 1	1	2	3	4	5	6	7	8
Variable	IFLAG							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
IFLAG	Consistent mass matrix flag: EQ.0: Use the standard lumped mass formulation (default) EQ.1: Use the consistent mass matrix.

Remarks:

The consistent mass matrix formulation is currently available for:

- three-node and four-node shell elements,
- solid elements types 1, 2, 10, 15, 16, and 18 (see *SECTION_SOLID),
- all thick shell element types, and
- beam types 1, 2, 3, 4, and 5 (see *SECTION_BEAM).
- all IGA shell and solid element types (see *SECTION_IGA_SHELL/SOLID)

All other element types continue to use a lumped mass matrix.

*CONTROL_IMPLICIT_DYNAMICS_{OPTION}

Available options include:

<BLANK>

DYN

SPR

Purpose: Activate implicit dynamic analysis and define time integration constants (see also *CONTROL_IMPLICIT_GENERAL). The DYN option allows setting controls specifically for the dynamic relaxation phase. The SPR option allows setting control specifically for the springback phase.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_DYNAMICS. For example, if ALPHA = 0.5 for *CONTROL_IMPLICIT_DYNAMICS and the value of ALPHA is left blank for *CONTROL_IMPLICIT_DYNAMICS_DYN, then the value of ALPHA for *CONTROL_IMPLICIT_-DYNAMICS_DYN will also be 0.5.

Card 1	1	2	3	4	5	6	7	8
Variable	IMASS	GAMMA	BETA	TDYBIR	TDYDTH	TDYBUR	IRATE	ALPHA
Type	I	F	F	F	F	F	I	F
Default	0	.50	.25	0.0	10^{28}	10^{28}	0	0.0

Rotational Dynamics Card. If ALPHA ≤ -1, specify |*ALPHA*| cards with this format.

Card 2	1	2	3	4	5	6	7	8
Variable	PSID	ANGLE						
Type	I	F						
Default	none	90.0						

VARIABLE	DESCRIPTION
IMASS	Implicit analysis type: LT.0: -IMASS is a curve ID with ordinate values used to control the amount of implicit dynamic effects applied to the analysis. TDYBIR, TDYDTH and TDYBUR are ignored with this option. EQ.0: Static analysis EQ.1: Dynamic analysis using Newmark time integration. EQ.2: Dynamic analysis by modal superposition following the solution of the eigenvalue problem EQ.3: Dynamic analysis by modal superposition using the eigenvalue solution in the d3eigv files that are in the runtime directory.
GAMMA	Newmark time integration constant (see Remark 2).
BETA	Newmark time integration constant (see Remark 2).
TDYBIR	Birth time for application of dynamic terms. See Figure 12-69.
TDYDTH	Death time for application of dynamic terms.
TDYBUR	Burial time for application of dynamic terms.
IRATE	Rate effects switch (see Remark 4). If RATESET on *CONTROL_IMAT is nonzero, this flag applies only to the parts included in that part set. EQ.-1: Rate effects are on in constitutive models even in implicit statics EQ.0: Rate effects are on in constitutive models, except implicit

VARIABLE	DESCRIPTION
	statics
	EQ.1: Rate effects are off in constitutive models
	EQ.2: Rate effects are off in constitutive models for both explicit and implicit.
ALPHA	Composite time integration constant (see Remark 2):
	GT.0: Bathe composite scheme is activated.
	LT.0.AND.GT.-1: HHT scheme is activated,
	LE.-1: Specify part sets for finite rotational dynamics.
PSID	Part set ID for a body undergoing rotational (spinning) motion
ANGLE	Target angle increment during a single time step in degrees

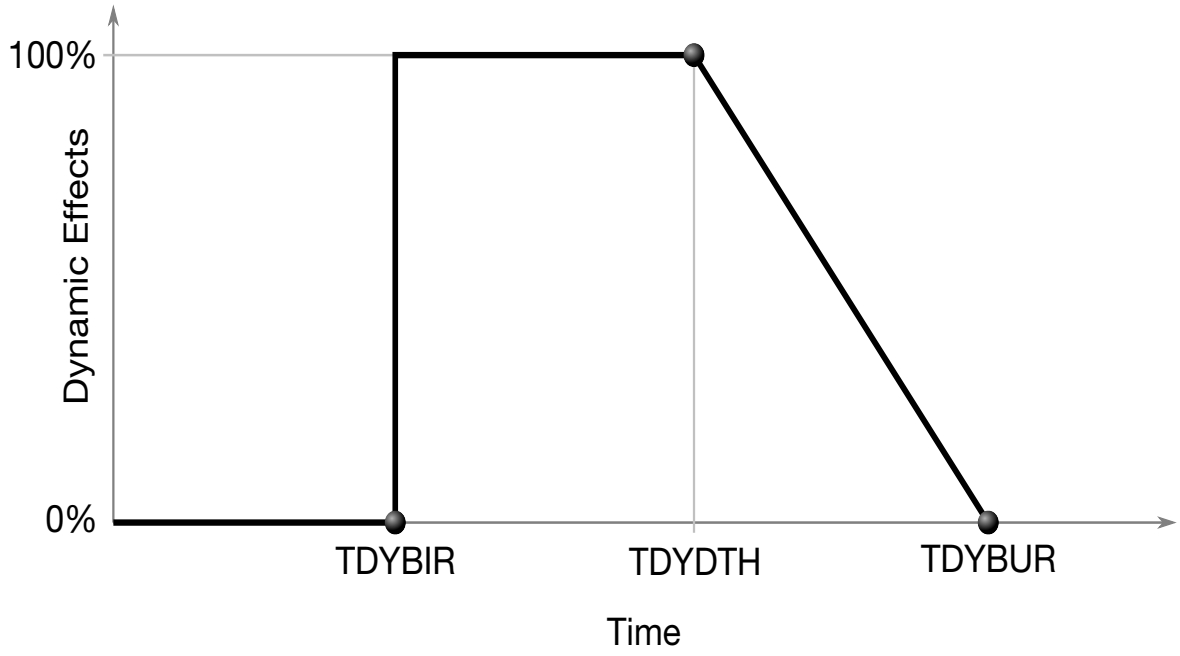


Figure 12-69. Birth, death, and burial time for implicit dynamics. The terms involving **M** and **D** are scaled by a factor ranging between 1 and 0 to include or exclude dynamical effects, respectively.

Remarks:

1. **Equilibrium equations and dynamic terms.** For the dynamic problem, the linearized equilibrium equations may be written in the form

$$\mathbf{M}\ddot{\mathbf{u}}^{n+1} + \mathbf{D}\dot{\mathbf{u}}^{n+1} + \mathbf{K}_t(\mathbf{x}^n)\Delta\mathbf{u} = \mathbf{P}(\mathbf{x}^n)^{n+1} - \mathbf{F}(\mathbf{x}^n) \ ,$$

where

$$\begin{aligned}
\mathbf{M} &= \text{lumped mass matrix} \\
\mathbf{D} &= \text{damping matrix} \\
\mathbf{u}^{n+1} = \mathbf{x}^{n+1} - \mathbf{x}^0 &= \text{nodal displacement vector} \\
\dot{\mathbf{u}}^{n+1} &= \text{nodal point velocities at time } n+1 \\
\ddot{\mathbf{u}}^{n+1} &= \text{nodal point acceleration at time } n+1
\end{aligned}$$

Between the birth and death times 100% of the dynamic terms, that is, the terms involving **M** and **D**, are applied. Between the death and burial time, the dynamic terms are decreased linearly with respect to time until 0% of the dynamic terms are applied after the burial time. This feature is useful for problems that are initially singular because the parts are not in contact initially such as in metal stamping. For these problems dynamics is required for stable convergence. When contact is established, the problem becomes well conditioned and the dynamic terms are no longer required for stable convergence. For such problems setting the death time to be after contact is established and the burial time for 2 or 3 time steps after the death time is recommended.

For problems with more extensive loading and unloading patterns the amount of dynamic effects added to the model can be controlled by using a load curve; see IMASS < 0. This curve should have ordinate values between 0.0 and 1.0. The user should use caution in ramping the load curve and the associated dynamic effects from 1.0 to 0.0. Such a ramping down should take place over 2 or 3 implicit time steps.

2. **Time integration.** The time integration is by default the unconditionally stable, one-step, Newmark-β time integration scheme

$$\begin{aligned}
\ddot{\mathbf{u}}^{n+1} &= \frac{\Delta\mathbf{u}}{\beta\Delta t^2} - \frac{\dot{\mathbf{u}}^n}{\beta\Delta t} - \frac{1}{\beta}\left(\frac{1}{2} - \beta\right)\ddot{\mathbf{u}}^n \\
\dot{\mathbf{u}}^{n+1} &= \dot{\mathbf{u}}^n + \Delta t(1-\gamma)\ddot{\mathbf{u}}^n + \gamma\Delta t\ddot{\mathbf{u}}^{n+1} \\
\mathbf{x}^{n+1} &= \mathbf{x}^n + \Delta\mathbf{u}
\end{aligned}$$

Here, Δt is the time step size, and β and γ are the free parameters of integration. For $\gamma = 1/2$ and $\beta = 1/4$, the method reduces to the trapezoidal rule and is energy conserving. If

$$\begin{aligned}
\gamma &> \frac{1}{2} \\
\beta &> \frac{1}{4}\left(\frac{1}{2} + \gamma\right)^2,
\end{aligned}$$

numerical damping is induced into the solution leading to a loss of energy and momentum.

The Newmark method, and the trapezoidal rule in particular, is known to lack the robustness required for simulating long term dynamic implicit problems.

Even though numerical damping may improve the situation from this aspect, it is difficult to know how to set γ and β without deviating from desired physical properties of the system. In the literature, a vast number of *composite* time integration algorithms have been proposed to handle this, and a family of such methods is implemented and governed by the value of α (ALPHA, parameter 8 on Card 1).

For $\alpha > 0$, every other implicit time step is a three point backward Euler step given as

$$\ddot{\mathbf{u}}^{n+1} = \frac{(1+\alpha)}{\Delta t}\left(\dot{\mathbf{u}}^{n+1} - \dot{\mathbf{u}}^{n}\right) - \frac{\alpha}{\Delta t_{-}}\left(\dot{\mathbf{u}}^{n} - \dot{\mathbf{u}}^{n-1}\right)$$
$$\dot{\mathbf{u}}^{n+1} = \frac{(1+\alpha)}{\Delta t}\Delta\mathbf{u} - \frac{\alpha}{\Delta t_{-}}\Delta\mathbf{u}_{-}$$

where $\Delta t_{-} = t^{n} - t^{n-1}$ and $\Delta\mathbf{u}_{-} = \mathbf{u}^{n} - \mathbf{u}^{n-1}$ are constants. Because of this three step procedure, the method is particularly suitable for nodes/bodies undergoing curved motion as it better accounts for curvature than the default Newmark step. For $\alpha = 1/2$, and default values of γ and β, the method defaults to the Bathe time integration scheme, Bathe [2007], and is reported to preserve energy and momentum to a reasonable degree. The improvement in stability over the Newmark method is primarily attributed to numerical dissipation, but fortunately this dissipation appears to mainly be due to damping of high frequency content and the underlying physics is therefore not affected as such; see Bathe and Noh [2012].

For a negative value of ALPHA (strictly between −1 and 0), the HHT, Hilber-Hughes-Taylor [1977], scheme is activated. This scheme is similar to that of the Newmark method, but the equilibrium is sought at time step $n + 1 + \alpha$ instead of at $n + 1$. As a complement to the Newmark scheme above, we introduce

$$\dot{\mathbf{u}}^{\alpha} = -\alpha\dot{\mathbf{u}}^{n} + (1+\alpha)\dot{\mathbf{u}}^{n+1}$$
$$\mathbf{x}^{\alpha} = -\alpha\mathbf{x}^{n} + (1+\alpha)\mathbf{x}^{n+1}$$

and solve a modified system of equilibrium equations

$$\mathbf{M}\ddot{\mathbf{u}}^{n+1} + \mathbf{D}\dot{\mathbf{u}}^{\alpha} + \mathbf{F}(\mathbf{x}^{\alpha}) = \mathbf{P}(\mathbf{x}^{\alpha}).$$

This method is stable for $-1/3 \le \alpha \le 0$ and $\gamma = (1-2\alpha)/2$ and $\beta = (1-\alpha)^2/4$, which becomes the default values of γ and β if not explicitly set. Parameter α controls the amount of dissipation in the problem; for $\alpha = 0$ an undamped Newmark scheme is obtained, whereas $\alpha = -1/3$ introduces significant damping. From the literature, a value of $\alpha = -0.05$ appears to be a good choice.

Finite rotational dynamics can be activated for a part set by specifying a negative integer for ALPHA, the absolute value of ALPHA then indicates for how many rotating bodies this feature is activated. This option assumes that each body, meaning the parts within each part set PSID, is "almost" a rigid body in the sense that

- the assembly as a whole undergoes relatively large rigid body motion and small deformation in each step;
- the part set includes all parts that makes up the assembly;
- and the assembly is not merged to some other elements in the model by sharing nodes, that is, interaction with other parts should be through joints, contacts, and similar constraints.

The target angle, ANGLE, is meant to be used as a reasonable (in terms of accuracy and convergence) rotation angle for each time step and may vary depending on application. For relatively simple inputs a value of 90 may be ok while for advanced applications it may have to be reduced. The time step will be adjusted to not exceed this target angle. The intention with this option is to use Newmark time integration with large time steps while maintaining high accuracy and robustness and hence serves as an undamped alternative to the damped Bathe and HHT schemes.

3. **Modal superposition.** When modal superposition is invoked, NEIGV on *CONTROL_IMPLICIT_EIGENVALUE indicates the number of modes to be used. With modal superposition, stresses are computed only for linear shell formulation 18.

4. **Rate effects in constitutive models.** The option IRATE is currently supported by the following material models: 3, 15, 24, 26, 57, 58, 63, 73, 75, 77, 81, 83, 89, 98, 103, 123, 124, 133, 142, 153, 157, 179, 181, 183, 187, 187L, 199, 215, 224, 225, 240, 251, 252, 264, 269, 273, 280, and GISSMO (see *MAT_ADD_DAMAGE). IRATE = 2 is also supported for material model 1 when used with truss elements. Output to d3hsp reflects how IRATE applies to the simulation at hand, including the influence of RATESET on *CONTROL_MAT.

*CONTROL_IMPLICIT_EIGENVALUE

Purpose: Activates implicit eigenvalue analysis and defines associated input parameters (see also *CONTROL_IMPLICIT_GENERAL). The available methods are Block Shift and Invert Lanczos, ARPACK for the nonsymmetric eigenvalue problem (see Remark 3), MCMS (SMP only) for problems requiring the computation of thousands of eigenmodes, Fast Lanczos for computing thousands of approximate eigenmodes in MPP, LOBPCG for problems requiring a small number of eigenmodes, and Sectoral Symmetry for problems with sectoral (rotational) symmetry.

Card Summary:

Card 1. This card is required.

NEIG	CENTER	LFLAG	LFTEND	RFLAG	RHTEND	EIGMTH	SHFSCL

Card 2. Card 2 and beyond are optional. If Card 3a or 3b are included, then Card 2 must be included; Card 2 can be a blank line in this case, so default values are used. Similarly, Card 4 requires Cards 2 and 3 (if applicable) to be included.

ISOLID	IBEAM	ISHELL	ITSHELL	MSTRES	EVDUMP	MSTRSCL	

Card 3a. Card 3a is read only when EIGMTH = 2, 3, 5, or 6. It is optional.

		IPARM3					RPARM4

Card 3b. Card 3b is read only when EIGMTH = 101. It is optional.

IPARM1	IPARM2	IPARM3	IPARM4	RPARM1			RPARM4

Card 3c. Card 3c is read only when EIGMTH = 102. It is optional.

IPARM1	IPARM2	IPARM3		RPARM1	RPARM2		RPARM4

Card 3d. Card 3d is read only when EIGMTH = 103. It is optional.

IPARM1	IPARM2	IPARM3		RPARM1	RPARM2		RPARM4

Card 3e. Card 3e is read only when EIGMTH = 111. It is optional.

IPARM1	IPARM2	IPARM3	IPARM4	IPARM5	IPARM6		

Card 4. This card is optional.

ROTSCL	EIGMSCL						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NEIG	CENTER	LFLAG	LFTEND	RFLAG	RHTEND	EIGMTH	SHFSCL
Type	I	F	I	F	I	F	I	F
Default	none	0.0	0	-infinity	0	+infinity	2	0.0

VARIABLE	DESCRIPTION
NEIG	Number of eigenvalues to extract. This must be specified. The other parameters below are optional. See Remark 1. LT.0: Curve ID = (-NEIG) used for intermittent eigenvalue analysis
CENTER	Center frequency. This option finds the nearest NEIG eigenvalues located about this value. See Remarks 2, 3, and 10.
LFLAG	Left endpoint finite flag (see Remarks 2 and 3): EQ.0: Left endpoint is -infinity. EQ.1: Left endpoint is LFTEND.
LFTEND	Left endpoint of the interval. Only used when LFLAG = 1. See Remarks 2, 3, and 10.
RFLAG	Right endpoint finite flag (see Remarks 2 and 3): EQ.0: Right endpoint is +infinity. EQ.1: Right endpoint is RHTEND.
RHTEND	Right endpoint of the interval. Only used when RFLAG = 1. See Remarks 2, 3, and 10.
EIGMTH	Eigenvalue extraction method (see Remark 3): EQ.2: Block Shift and Invert Lanczos (default). See Remark 2. EQ.3: Lanczos with [M] = [I] (for debugging only) EQ.5: Same as 3 but include Dynamic Terms EQ.6: Same as 2 but include Dynamic Terms

VARIABLE	DESCRIPTION
	EQ.101: MCMS. See Remark 4.
	EQ.102: LOBPCG. See Remark 7.
	EQ.103: Fast Lanczos. See Remark 5.
	EQ.111: Sectoral Symmetry. See Remark 8.
SHFSCL	Shift scale. Generally, not used. See Remarks 3, 9, and 10.

Card 2	1	2	3	4	5	6	7	8
Variable	ISOLID	IBEAM	ISHELL	ITSHELL	MSTRES	EVDUMP	MSTRSCL	
Type	I	I	I	I	I	I	F	
Default	0	0	0	0	0	0	0.001	

VARIABLE	DESCRIPTION
ISOLID	If nonzero, reset all solid element formulations to ISOLID for the implicit computations. It can be used for all implicit analyses, not just eigenvalue computations.
IBEAM	If nonzero, reset all beam element formulations to IBEAM for the implicit computations. It can be used for all implicit computations, not just eigenvalue computations.
ISHELL	If nonzero, reset all shell element formulations to ISHELL for the implicit computations. It can be used for all implicit computations, not just eigenvalue computations.
ITSHELL	If nonzero, reset all thick shell element formulations to ITSHELL for the implicit computations. It can be used for all implicit computations, not just eigenvalue computations.
MSTRES	Flag for computing the stresses for the eigenmodes (see Remark 11):
	EQ.0: Do not compute the stresses.
	EQ.1: Compute the stresses.

VARIABLE	DESCRIPTION
EVDUMP	Flag for writing eigenvalues and eigenvectors to file Eigen_Vectors in SMP or files Eigen_Vectors.xxxx in MPP where xxxx is the process ID (see Remark 11): EQ.0: Do not write eigenvalues and eigenvectors. GT.0: Write eigenvalues and eigenvectors using an ASCII format. LT.0: Write eigenvalues and eigenvectors using a binary format.
MSTRSCL	Scaling for computing the velocity based on the mode shape for the stress computation. See Remark 11.

Block Shift and Invert Lanczos Card. Omitted unless EIGMTH = 2, 3, 5, or 6. Additional card for eigenvalue extraction method Block Shift and Invert Lanczos and its variations.

Card 3a	1	2	3	4	5	6	7	8
Variable			IPARM3					RPARM4
Type			I					F
Default			{∅}					0.0

VARIABLE	DESCRIPTION
IPARM3	Node set to reduce output of entries from each eigenvector. See Remark 6.
RPARM4	Control output of eigenvectors to the d3eigv database (see Remark 6): LT.0.0: No output EQ.0.0: Write all eigenmodes to d3eigv. GT.0.0: Write only the first RPARM4 eigenmodes to d3eigv.

MCMS Card. Omitted unless EIGTH = 101. Additional card for eigenvalue extraction method MCMS.

Card 3b	1	2	3	4	5	6	7	8
Variable	IPARM1	IPARM2	IPARM3	IPARM4	RPARM1			RPARM4
Type	I	I	I	I	F			F
Default	100	↓	{∅}	1500	4.0			0.0

VARIABLE	DESCRIPTION
IPARM1	Minimum block size for the Cholesky factorization
IPARM2	Maximum block size for the Cholesky factorization. The default is the model size.
IPARM3	Node set ID specifying particular nodes in the model where increased accuracy is desired. See Remark 4.
IPARM4	MCMS minimum group/substructure size. See Remark 4.
RPARM1	Eigenvalue expansion factor τ. See Remark 4.
RPARM4	Control output of eigenvectors to the d3eigv database: LT.0.0: No output EQ.0.0: Write all eigenmodes to d3eigv. GT.0.0: Write only the first RPARM4 eigenmodes to d3eigv.

LOBPCG Card. Omitted unless EIGMTH = 102. Additional card for eigenvalue extraction method LOBPCG.

Card 3c	1	2	3	4	5	6	7	8
Variable	IPARM1	IPARM2	IPARM3		RPARM1	RPARM2		RPARM4
Type	I	I	I		F	F		F
Default	100	100	{∅}		10^{-12}	10^{-5}		0.0

VARIABLE	DESCRIPTION
IPARM1	Maximum number of iterations
IPARM2	Block size
IPARM3	Node set to reduce output of entries from each eigenvector. See Remark 6.
RPARM1	Convergence tolerance
RPARM2	Flag for using a Block Low-Rank (BLR) factorization for the preconditioner: GT.0.0: Use BLR preconditioner with compression threshold RPARM2. LT.0.0: Use exact factorization instead of a BLR preconditioner
RPARM4	Control output of eigenvectors to the d3eigv database (see Remark 6): LT.0.0: No output EQ.0.0: Write all eigenmodes to d3eigv. GT.0.0: Write only the first RPARM4 eigenmodes to d3eigv.

Fast Lanczos Card. Omitted unless EIGTH = 103. Additional card for eigenvalue extraction method Fast Lanczos.

Card 3d	1	2	3	4	5	6	7	8
Variable	IPARM1	IPARM2	IPARM3		RPARM1	RPARM2		RPARM4
Type	I	I	I		F	F		F
Default	40	40	0		200.0	0.80		0.0

VARIABLE	DESCRIPTION
IPARM1	Maximum number of Block Lanczos iterations
IPARM2	Block size for the Block Lanczos recurrence
IPARM3	Node set to reduce output of entries from each eigenvector. See Remark 6.

VARIABLE	DESCRIPTION
RPARM1	First shift. Supply the approximate value of the 100th eigenvalue.
RPARM2	Shift factor. This field reduces the aggressiveness of the shift logic.
RPARM4	Control output of eigenvectors to the d3eigv database (see Remark 6): LT.0.0: No output EQ.0.0: Write all eigenmodes to d3eigv. GT.0.0: Write only the first RPARM4 eigenmodes to d3eigv.

Sectoral Symmetry Card. Omitted unless EIGTH = 111. Additional card for eigenvalue extraction method MCMS.

Card 3e	1	2	3	4	5	6	7	8
Variable	IPARM1	IPARM2	IPARM3	IPARM4	IPARM5	IPARM6		
Type	I	I	I	I	I	I		
Default	none	0	none	none	none	0		

VARIABLE	DESCRIPTION
IPARM1	Node set ID for nodes on the left surface of the sector
IPARM2	Node set ID for nodes on the axis of rotation. EQ.0: No nodes on the axis of rotation (default)
IPARM3	Node set ID for nodes on the right surface of the sector
IPARM4	Number of sectors
IPARM5	Harmonic index
IPARM6	Vector ID for the axis of rotation. EQ.0: Axis of rotation is the global Z-axis (default)

Optional Control Card.

Card 4	1	2	3	4	5	6	7	8
Variable	ROTSCL	EIGMSCL						
Type	F	I						
Default	0.001	0						

VARIABLE	DESCRIPTION
ROTSCL	Scale factor for the inertia of rotational degrees of freedom. See Remark 13. EQ.0.0: Default value of 0.001 EQ.1.0: Inertia not scaled.
EIGMSCL	For intermittent eigenvalue extractions, control whether to use the original (unscaled) or the scaled mass (when mass scaling is active through parameter DT2MS on *CONTROL_TIMESTEP). Prior to version R15 (unless noted otherwise in the release notes), the scaled mass was incorrectly used. We added this option for backwards compatibility. EQ.0: Use original mass (default). EQ.1: Use scaled mass.

Remarks:

1. **Performing eigenvalue analysis.** To perform an eigenvalue analysis, activate the implicit method by selecting IMFLAG = 1 on *CONTROL_IMPLICIT_GENERAL, and indicate a nonzero value for NEIG above. By default, the lowest NEIG eigenvalues will be found. Specifying a nonzero center frequency causes LS-DYNA to find the NEIG eigenvalues nearest to CENTER.

   When NEIG > 0, LS-DYNA computes eigenvalues at time = 0 and then terminates.

   When NEIG < 0, LS-DYNA performs an intermittent eigenvalue analysis, which can be in both transient and dynamic relaxation (DR) phases. This is a transient/DR simulation during which loads are applied, with eigenvalues computed periodically during the simulation. Changes in geometry, stress, material, and

contact conditions will affect the eigenvalues. The transient simulation can be either implicit or explicit according to IMFLAG = 1 or IMFLAG = 6, respectively, on *CONTROL_IMPLICIT_GENERAL. The curve ID = -NEIG indicates when to extract eigenvalues and how many to extract. Define one curve point at each desired extraction time, with a function value equal to the number of eigenvalues needed at that time. If requiring eigenvalues during the dynamic relaxation phase, set SIDR = 2 in *DEFINE_CURVE, and then set the extraction time to a negative value, corresponding to the pseudo time during the dynamic phase. LS-DYNA produces a d3plot database for the transient solution results. It produces consecutively numbered d3eigv and eigout databases for each intermittent extraction. Each database's analysis title indicates the extraction time.

2. **Block Shift and Invert Lanczos method.** The Block Shift and Invert Lanczos code is from BCSLIB-EXT, Boeing's Extreme Mathematical Library.

   When using Block Shift and Invert Lanczos, you can specify a semifinite or finite interval region in which to compute eigenvalues. Setting LFLAG = 1 changes the left endpoint from -infinity to the value specified by LFTEND. Setting RFLAG = 1 changes the right endpoint from +infinity to the values given by RHTEND. If the interval includes CENTER (default value of 0.0), LS-DYNA computes the NEIG eigenvalues nearest to CENTER. If the interval does not include CENTER, LS-DYNA computes NEIG eigenvalues with the smallest magnitude.

   If all of the eigenvalues are needed in an interval where both endpoints are finite, input a large number for NEIG. The software automatically computes the number of eigenvalues in the interval and decreases NEIG to that value. The most general problem specification is to compute NEIG eigenvalues nearest CENTER in the interval [LFTEND,RHTEND]. Computing the lowest NEIG eigenvalues is equivalent to computing the NEIG eigenvalues nearest 0.0.

3. **Nonsymmetric eigenvalue problems.** When the field LCPACK in *CONTROL_IMPICIT_SOLVER equals 3, LS-DYNA assumes the eigenvalue problem is nonsymmetric. This feature allows you to compute the eigensolutions for problems with both nonsymmetric terms in the stiffness matrix and with damping terms. *Note that the fields CENTER, LFLAG, LFTEND, RFLAG, RHTEND, EIGMTH, and SHFSCL are ignored for the nonsymmetric eigenvalue problem.*

   By setting LCPACK to 3, ARPACK will automatically be chosen to solve the eigenvalue problem. All damping terms in the model will also be added to the first order terms of the eigenvalue problem. Implicit Rotational Dynamics will also cause the use of ARPACK due to the added first order terms.

4. **MCMS method.** MCMS is an SMP only implementation of the AMLS algorithm available as of LS-DYNA R11. See the discussion of Fast Lanczos in Remark 5 for an MPP capability. The target application is automotive NVH models, where

thousands of approximate eigenmodes are computed for Frequency Response Analysis. MCMS calculates all the eigenmodes with eigenvalues in the interval $[-\infty, \text{RHTEND}]$. NEIG must be positive to turn on the eigencomputation. If the number of eigenvalues in the interval is less than NEIG, then the next few eigenmodes will be included to reach NEIG. NEIG should not be too large compared to the expected number of eigenvalues in the interval, as the accuracy of the eigenmodes past RHTEND will be less than that of the eigenmodes before RHTEND.

Increasing the group/substructure size (IPARM4) and the eigenvalue expansion factor (RPARM1) improves the accuracy with the cost of increasing the time of the MCS eigen computation. To improve accuracy at a few nodes, such as the nodes of the mounting bracket for an automotive chassis, specify these nodes in a node set using IPARM3.

5. **Fast Lanczos method.** Fast Lanczos is designed to compute thousands of approximate eigenmodes for Noise, Vibration, and Harshness (NVH) applications. It is an implementation of the Block Shift and Invert Lanczos algorithm used by EIGMTH = 2, with many adjustments to make it faster and less accurate. The goal of this method is to compute the lowest thousands of eigenmodes with an accuracy of 1% for the eigenvalues. It is assumed that the factorization computed by the MF solver (LSOLVR = 2 on *CONTROL_IMPLICIT_SOLVER) ends up with the factorization in memory. At each shift, Block Lanczos runs IPARM1 iterations using a block size of IPARM2. These parameters may be adjusted to allow the computation to fit entirely in memory.

   Fast Lanczos requires an approximate value of the 100^{th} eigenvalue (RPARM1). The default of 200.0 works well for most problems, but the choice of units and the actual model can impact this value. The default value was chosen under the assumption that the time unit is seconds. For example, for models with milliseconds as the time unit, we recommend using a value of 0.2. The most robust way to find this value is to compute 100 modes with Standard Lanczos (EIGMTH = 2) and use the 100^{th} eigenvalue from that computation.

   RPARM2 controls the aggressiveness of the shift logic.

   The interval controls of RFLAG and RHTEND are honored for the case where you desire all of the eigenmodes less than RHTEND.

6. **Reducing the cost of outputting eigenvectors for EIGMTH = 2, 3, 5, 6, 102, and 103.** The cost of outputting the entirety of eigenvectors to the d3mode database can be substantial. Some applications using the eigenmodes only need a subset of the entries for each eigenvector. For EIGMTH = 2, 3, 5, 6, 102, and 103, set IPARM3 to a node set to define this subset. Then, the output for that subset for each eigenvector is written to an LSDA database called EigenVectors. Using IPARM3 can vastly reduce the I/O requirements.

Some post-processing applications require only a subset of the eigenvectors. RPARM4 for the previously mentioned eigenvalue extraction methods controls the number of eigenvectors written to the d3eigv database. The choice is no eigenvectors, some eigenvectors, or all of the eigenvectors. A node set specified with IPARM3 and RPARM4 > 0.0 can be used at the same time. If IPARM3 is nonzero and RPARM4 is zero or blank, no eigenmodes are written to d3eigv. The parameter MSTRES specifies whether to perform stress computations. Computing stresses is expensive for large finite element models and when outputting thousands of eigenvectors. Care should be used in setting MSTRES in combination with RPARM4. Visualizing the first few eigenmodes for model verification can be accomplished by setting RPARM4 to a small value, such as 10.

7. **LOBPCG method.** LOBPCG [1] is a preconditioned eigensolver that can be faster and use less memory than Lanczos when computing a few eigenmodes. The preconditioner is based on a Block Low-Rank (BLR) factorization. The user can adjust the BLR compression threshold as well as various parameters. It is available in SMP as of LS-DYNA R11 and in MPP as of R14.

8. **Sectoral symmetry method.** Sectoral symmetry uses the rotational symmetry of a model to reduce the model size to that of a sector but at the cost of performing many eigenvalue computations based on the harmonic index. This method requires specifying the nodes on the left surface of the sector, the nodes on the right surface of the sector, and any nodes in the sector on the axis of rotation. The default for the axis of rotation is the global *Z*-axis. The axis is otherwise given by a vector ID from *DEFINE_VECTOR or *DEFINE_VECTOR_NODES. The model is checked to verify that the right nodes align with the left nodes when properly rotated given the axis and the number of sectors.

   The eigenvalue problem is partitioned based on the left, center, right, and interior node sets. The boundary condition given by the harmonic index is solved.

   Note that the solver only computes a subset of the eigenmodes, those associated with a given harmonic index. Obtaining the complete spectrum requires multiple runs with harmonic indices varying from 0 to the number of sectors – 1.

   The sectoral symmetry method is available for both SMP and MPP, but in MPP it only works for jobs running on one processor.

9. **Initial shift.** For some problems, it is helpful to override the internal heuristic for picking a starting point for Lanczos shift strategy, that is, the initial shift. In these rare cases, you may specify the initial shift via the parameter SHFSCL. SHFSCL should be in the range of the first few nonzero frequencies.

10. **Units.** Parameters CENTER, LFTEND, RHTEND, and SHFSCL are in units of Hertz for eigenvalue problems. These four parameters, along with LFLAG and

RFLAG, apply to buckling problems. For buckling problems, CENTER, LFTEND, RHTEND, and SHFSCL are in units of the eigenvalue spectrum.

11. **Output files.** Eigenvectors are written to an auxiliary binary plot database named d3eigv, which is automatically created. These can be viewed using a postprocessor in the same way as a standard d3plot database. The time value associated with each eigenvector plot is the corresponding frequency in units of cycles per unit time. A summary table of eigenvalue results is printed to the eigout file. In addition to the eigenvalue results, modal participation factors and modal effective mass tables are written to the eigout" file. The user can export individual eigenvectors using LS-PrePost.

    Set MSTRES = 1 to request computing the stresses and write them to d3eigv. A velocity is computed by dividing the displacements from the eigenmode by MSTRSCL. The element routine then computes the stresses based on this velocity, but LS-DYNA inversely scales those stresses by MSTRSCL before writing them to d3eigv. Thus, MSTRSCL does not affect the results of linear element formulations. The strains associated with the stresses output using the MSTRES option can be obtained by setting the STRFLG on *DATABASE_EXTENT_BINARY.

    Setting EVDUMP to a nonzero value causes writing the eigenvalues and eigenvectors to file Eigen_Vectors in SMP or files Eigen_Vectors.xxxx in MPP. In MPP, xxxx is the process ID. For example, for a run with 128 processes, LS-DYNA outputs files Eigen_Vectors.0001, Eigen_Vectors.0002, …, Eigen_Vectors.0127. The sign of EVDUMP determines the file type for both SMP and MPP. For EVDUMP > 0, the files are output in ASCII format, while for EVDUMP < 0, they are output in a simple binary format. The binary format reduces file space. The written eigenvectors are orthonormal with respect to the mass matrix.

12. **Solver controls.** The print control parameter, LPRINT, and ordering method parameter, ORDER, from the *CONTROL_IMPLICIT_SOLVER keyword card also apply to the various eigensolvers.

13. **Inertia of rotational DOF.** By default, the inertia of the rotational degrees of freedom is scaled by a thousand. This scaling causes an increase in the rotational eigenfrequencies and exclusion of the associated eigenmodes (which often are of little interest) from the lowest NEIG eigenmodes. This approach is acceptable when no coupling exists between the translational and rotational degrees of freedom. Coupling occurs when the neutral and reference planes do not coincide, such as in nonsymmetric laminates or when NLOC ≠ 0. When coupling exists, use correct rotational inertia. Setting ROTSCL = 1.0 tells LS-DYNA to use the correct rotational inertia (the same as in explicit analyses).

References:

[1] A. V. Knyazev, Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. SIAM journal on scientific computing, 23(2), 517-541 (2001).

.

***CONTROL_IMPLICIT_FORMING**_*{OPTION}*

Available options include:

<BLANK>

DYN

SPR

Purpose: This keyword is used to perform implicit static analysis, especially for metal forming processes, such as gravity loading, binder closing, flanging, and stamping subassembly simulation. A systematic study was conducted to identify the key factors affecting implicit convergence, and the preferred values are automatically set using this keyword. In addition to forming applications, this keyword can also be used in other applications, such as dummy loading and roof crush, etc. The DYN option allows you to set controls specifically for the dynamic relaxation phase. The SPR option allows you to set controls specifically for the springback phase.

A related keyword is *CONTACT_AUTO_MOVE, where an empty distance between the tool and blank can be automatically eliminated in a combined gravity and closing simulation for implicit static simulations.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_FORMING. For example, if NSMAX = 50 for *CONTROL_IMPLICIT_FORMING and the value of NSMAX is left blank for *CONTROL_IMPLICIT_FORMING_DYN, then the value of NSMAX for *CONTROL_IMPLICIT_FORMING_DYN will also be 50.

Card 1	1	2	3	4	5	6	7	8
Variable	IOPTION	NSMIN	NSMAX	BIRTH	DEATH	PENCHK	DT0	
Type	I	I	I	F	F	F	F	
Default	1	↓	2	0.0	10^{20}	0.0	↓	

VARIABLE	DESCRIPTION
IOPTION	Solution type: EQ.1: Gravity loading simulation; see Remark 2 and examples: Gravity Loading, Flanging Simulation Using IOPTION = 1, and Switching between Implicit Dynamic and Implicit Static for Gravity Loading. EQ.2: Binder closing and flanging simulation; see Remark 2 and examples: Binder Closing, Binder Closing with Real Beads, and Flanging.
NSMIN	Minimum number of implicit steps for IOPTION = 2.
NSMAX	Maximum number of implicit steps for IOPTION = 2.
BIRTH	Birth time to activate this feature.
DEATH	Death time.
PENCHK	Relative allowed penetration with respect to the part thickness in contact for IOPTION = 2.
DT0	Initial time step size that overrides the DT0 field defined in *CONTROL_IMPLICIT_GENERAL

Remarks:

1. **Implicit settings.** This keyword provides a simplified interface for implicit static analysis. If no other implicit cards are used, the stiffness matrix is reformed every iteration. Convergence tolerances (DCTOL, ECTOL, etc.) are automatically set and recommended not to be changed. In almost all cases, only two additional implicit control cards (*CONTROL_IMPLICIT_GENERAL and *CONTROL_IMPLICIT_AUTO) may be needed to control the time step size with fields DT0, DTMIN, and DTMAX.

   If multiple steps are required for IOPTION = 1, *CONTROL_IMPLICIT_GENERAL must be placed *after* *CONTROL_IMPLICIT_FORMING with DT0 specified as a certain fraction of the ENDTIM (see *CONTROL_TERMINATION). Otherwise, even with DT0 specified as a fraction of the ENDTIM, only one step (with a step size of ENDTIM) will be performed.

2. **Contact.** As always, the field IGAP should be set to "2" in *CONTACT_FORMING… cards to achieve a more realistic contact simulation during forming. The contact type *CONTACT_FORMING_SURFACE_TO_SURFACE is recommended to be used with implicit analysis.

Smaller penalty stiffness scale factor SLSFAC on *CONTROL_CONTACT produces a certain amount of contact penetration but yields faster simulation time, and therefore is recommended for gravity and closing (in case of no physical beads) simulation. A subsequent forming process is likely to follow, and contact conditions will be reestablished there, where a tighter, default SLSFAC value of 0.1 should be used.

3. **Element type.** We recommend using the fully integrated shell element type 16 for all implicit calculations. For solids, type -2 is recommended.

4. **Double precision.** Executable with double precision is to be used for all implicit calculations.

5. **MPP.** MPP is more efficient for models with over 100,000 deformable elements.

Gravity Loading Example:

An example of implicit gravity is provided below, where a blank is loaded with gravity into a toggle die. Five steps are used, controlled by the field DT0. The results are shown in Figure 12-70. If this binder closing is simulated with explicit dynamics, the inertia effects on the blank need to be reduced since contact with the upper binder only happens along the periphery, and a large middle portion of the blank is not driven or supported by anything. With the implicit static method, there is no inertia effect at all on the blank during the closing, and no tool speed, time step size, etc. to be concerned about.

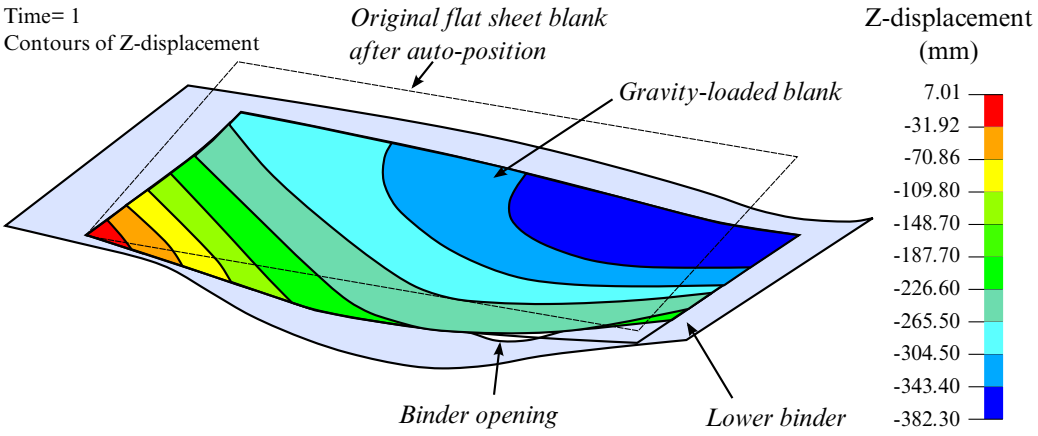


Figure 12-70. Gravity loading on a box side outer toggle die (courtesy of Autodie, LLC).

The implicit gravity application for both air and toggle draw processes is available through LS-PrePost in Metal Forming Application → eZ Setup.

```
*KEYWORD
*PARAMETER
⋮
*CONTROL_TERMINATION
```

```
1.0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_IMPLICIT_FORMING
$  IOPTION
         1
*CONTROL_IMPLICIT_GENERAL
$    IMFLAG       DT0
         1       0.2
*CONTROL_CONTACT
$    SLSFAC    RWPNAL    ISLCHK    SHLTHK    PENOPT    THKCHG     ORIEN
      0.03       0.0         2         1         4         0         4
$    USRSTR    USRFAC     NSBCS    INTERM     XPENE     SSTHK      ECDT   TIEDPRJ
         0         0        10         0       1.0         0
*PART
 Blank
   &blkpid   &blksec   &blkmid
*SECTION_SHELL
$       SID    ELFORM      SHRF       NIP     PROPT   QR/IRID     ICOMP     SETYP
&blksec            16     0.833         7       1.0
$        T1        T2        T3        T4      NLOC
&bthick,&bthick,&bthick,&bthick
*CONTACT_FORMING_SURFACE_TO_SURFACE
$     SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
   &blksid  &lpunsid         2         2                               1         1
$         FS        FD        DC        VC       VDC    PENCHK        BT        DT
       0.12       0.0       0.0       0.0      20.0         0       0.0     1E+20
$      SFSA      SFSB      SAST      SBST     SFSAT     SFSBT      FSF       VSF
        1.0       1.0       0.0     &mstp
$      SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
$    PENMAX    THKOPT    SHLTHK     SNLOG      ISYM     I2D3D    SLDTHK    SLDSTF
                                       1
$      IGAP    IGNORE    DPRFAC    DTSTIF                        FLANGL
         2
⋮
*LOAD_BODY_Z
90994
*DEFINE_CURVE_TITLE
Body Force on blank
90994
0.0,9810.0
10.0,9810.0
*LOAD_BODY_PARTS
&blksid
*END
```

Binder Closing Example:

An example of binder closing and its progression is shown in Figures 12-71, 12-72, 12-73, and 12-74, using the NUMISHEET'05 deck lid inner, where a blank is being closed in a toggle die (modified). An adaptive level of three was used in the closing process. Gravity is and should always be applied at the same time, regardless of whether a prior gravity loading simulation is performed or not, as listed at the end of the input deck. The presence of gravity helps the blank establish an initial contact with the tool, thus improving the convergence rate. The upper binder is moved down by a closing distance (defined by a parameter &bindmv) using a displacement boundary condition (VAD = 2), with a simple linearly increased triangle-shaped load curve. The variable DT0 is set to 0.01, based on the expected total deformation. The solver will automatically adjust based on the initial contact condition. The maximum step size is controlled by the variable DT-

MAX; it must be sufficiently small (<0.02) to avoid missing contact but not so small as to cause a long running time. In some cases, this variable can be set to a larger value, but the current value works for most cases.

```
*KEYWORD
*PARAMETER
⋮
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_FORMING
$  IOPTION     NSMIN     NSMAX
         2         2       100
*CONTROL_IMPLICIT_GENERAL
$   IMFLAG       DT0
         1      0.01
*CONTROL_IMPLICIT_AUTO
$    IAUTO    ITEOPT    ITEWIN     DTMIN     DTMAX
         0         0         0      0.01      0.03
*CONTROL_ADAPTIVE
⋮
*CONTROL_CONTACT
$   SLSFAC    RWPNAL    ISLCHK    SHLTHK    PENOPT    THKCHG     ORIEN
      0.03       0.0         2         1         4         0         4
$   USRSTR    USRFAC     NSBCS    INTERM     XPENE     SSTHK      ECDT   TIEDPRJ
         0         0        10         0       1.0         0
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
⋮
*PART
 Blank
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
   &blkpid   &blksec   &blkmid                                  &adpyes
*SECTION_SHELL
$      SID    ELFORM      SHRF       NIP     PROPT   QR/IRID     ICOMP     SETYP
&blksec           16     0.833         7       1.0
$       T1        T2        T3        T4      NLOC
&bthick,&bthick,&bthick,&bthick
⋮
*CONTACT_FORMING_SURFACE_TO_SURFACE
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
   &blksid  &lpunsid         2         2                              1         1
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
      0.12       0.0       0.0       0.0      20.0         0       0.0     1E+20
$     SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
       1.0       1.0       0.0     &mstp
$     SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
$   PENMAX    THKOPT    SHLTHK     SNLOG      ISYM     I2D3D    SLDTHK    SLDSTF
                                       1
$     IGAP    IGNORE    DPRFAC    DTSTIF                          FLANGL
         2
*CONTACT_...
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
&bindpid           3         2         3      -1.0         0
*DEFINE_CURVE
3
0.0,0.0
1.0,&bindmv
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$ Activate gravity on blank:
*LOAD_BODY_Z
90994
*DEFINE_CURVE_TITLE
```

```
Body Force on blank
90994
0.0,9810.0
10.0,9810.0
*LOAD_BODY_PARTS
&blksid
*END
```

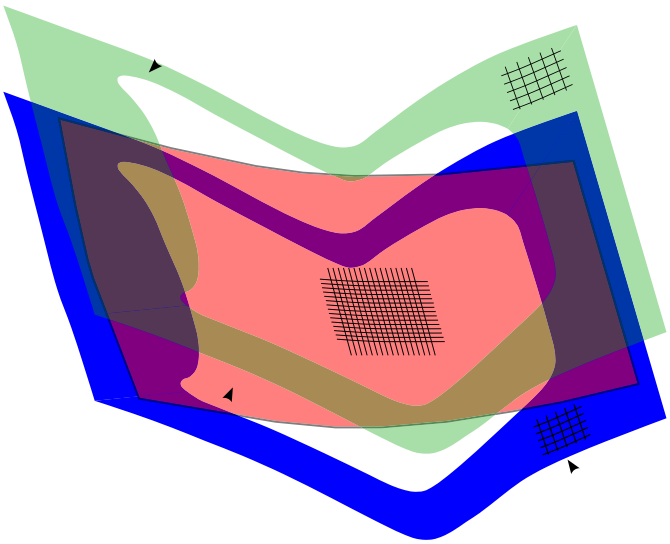


Figure 12-71. Initial auto-positioning (NUMISHEET2005 decklid inner).

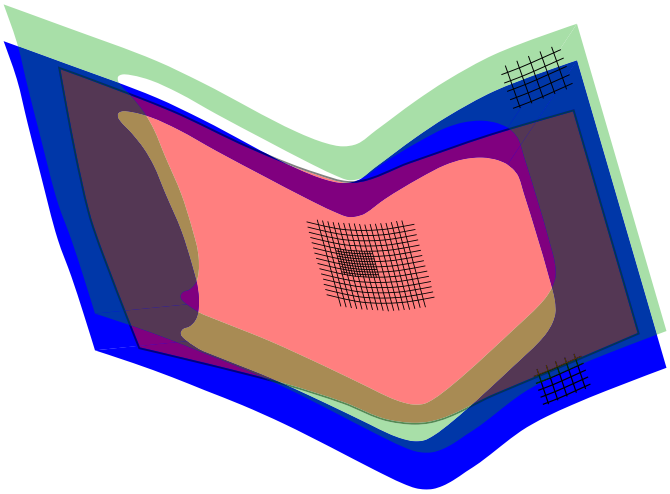


Figure 12-72. At 50% upper travel.

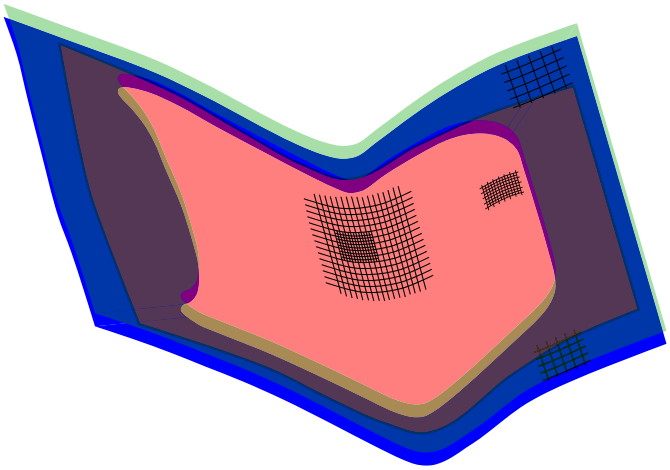


Figure 12-73. At 80% upper travel.

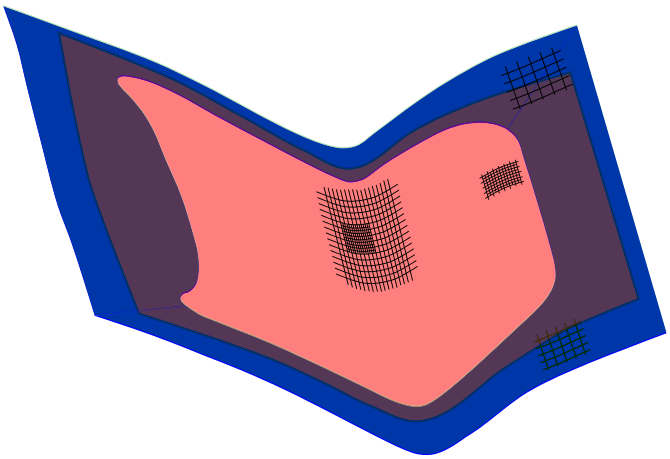


Figure 12-74. Upper travels to home.

Binder Closing with Real Beads Example:

Binder closing with real beads can also be done with implicit static and an adaptive mesh. An example is shown in Figure 12-75, where the hood outer is implicitly closed. Note that a small buckle can be seen near the draw bead region along the fender line. These kinds of small forming effects can be more accurately detected with the implicit static method.

The implicit static closing can now be set up in LS-PrePost Metal Forming Application → eZ Setup.

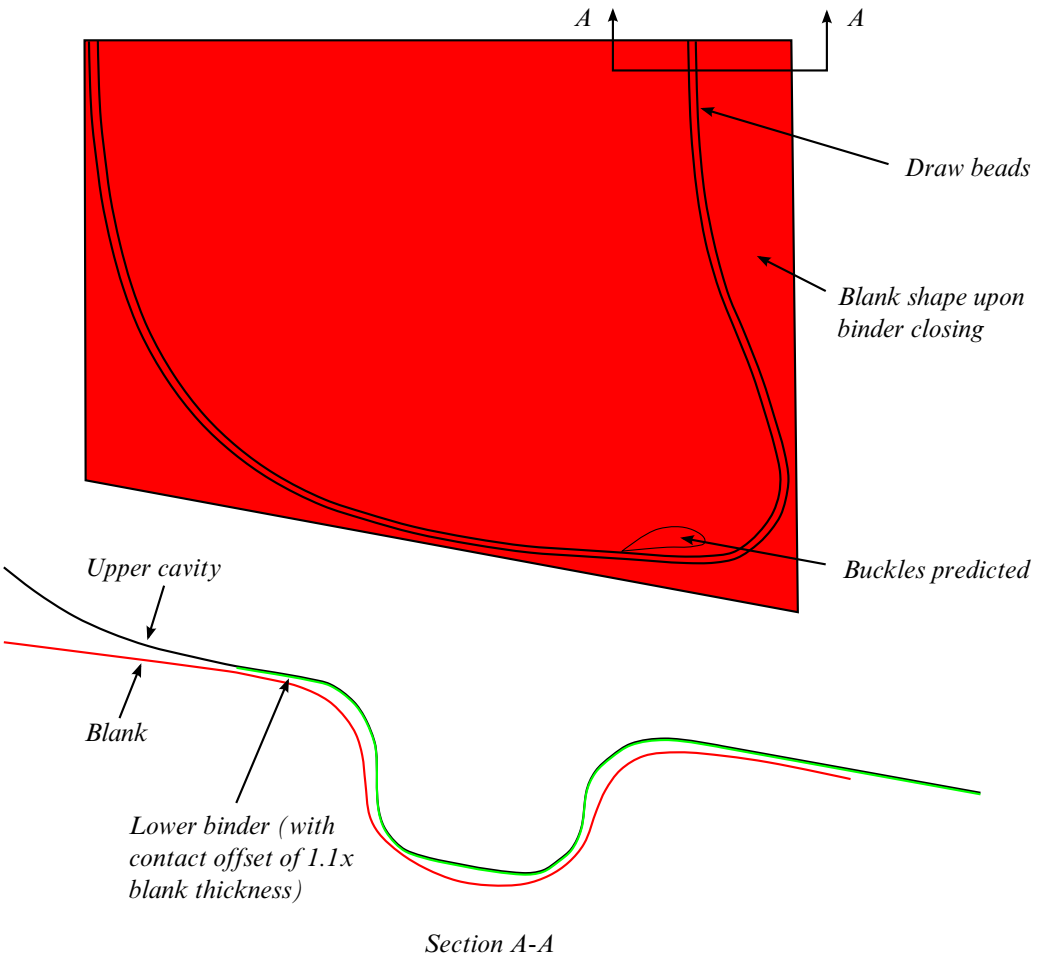


Figure 12-75. Binder closing with beads on a hood outer.

Flanging Example:

An example flanging simulation using this feature is shown in Figures 12-76, 12-77 and 12-78, with NUMISHEET'02 fender outer, where flanging is conducted along the hood line. A partial input is provided below, where DTMAX is controlled by a load curve for contact and speed; a load curve input for DTMAX is usually not necessary. Gravity, pad closing, and flanging were set to 10%, 10%, and 80% of the total step size, respectively. The pad travels a distance of &padtrav, starting at 0.1, when it is automatically moved to close the gap with the blank due to gravity loading (*CONTACT_AUTO_MOVE), then finishes at 0.2 and is held in that position until the end. Flanging steel travels a distance of '&flgtrav' starting at 0.2 and completing at 1.0. A detailed section view of the simulation follows in Figure 12-79.

```
*KEYWORD
*PARAMETER ...
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_FORMING
$  IOPTION     NSMIN     NSMAX
```

```
         2         2       200
*CONTROL_IMPLICIT_GENERAL
         1     0.100
*CONTROL_IMPLICIT_AUTO
$    IAUTO    ITEOPT    ITEWIN     DTMIN     DTMAX
         0         0         0     0.005     -9980
*DEFINE_CURVE
9980
0.0,0.1
0.1,0.1
0.2,0.1
0.7,0.005
1.0,0.005
*CONTROL_ADAPTIVE...
*CONTROL_CONTACT...
*PART...
*SECTION_SHELL...
*CONTACT_...
*CONTACT_FORMING_SURFACE_TO_SURFACE_ID_MPP
2
0,200,,3,2,1.005
$    SURFA     SURFB  SURFATYP  SURFBTYP   SABOXID   SBBOXID      SAPR      SBPR
   &blksid   &padsid         2         2
$       FS        FD        DC        VC       VDC    PENCHK        BT        DT
      0.12       0.0       0.0       0.0      20.0         0       0.0     1E+20
$     SFSA      SFSB      SAST      SBST     SFSAT     SFSBT       FSF       VSF
       1.0       1.0       0.0     &mstp
$     SOFT    SOFSCL    LCIDAB    MAXPAR    PENTOL     DEPTH     BSORT    FRCFRQ
         0
$   PENMAX    THKOPT    SHLTHK     SNLOG      ISYM     I2D3D    SLDTHK    SLDSTF
                                       1
$     IGAP    IGNORE    DPRFAC    DTSTIF                        FLANGL
         2
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
   &padpid         3         2         3      -1.0         0
   &flgpid         3         2         4      -1.0         0
*DEFINE_CURVE
3
0.0,0.0
0.1,0.0
0.2,&padtrav
1.0,&padtrav
*DEFINE_CURVE
4
0.0,0.0
0.2,0.0
1.0,&flgtrav
$ Activate gravity on blank:
*LOAD_BODY_PARTS
&blksid
*LOAD_BODY_Z
90994
*DEFINE_CURVE_TITLE
Body Force on blank
90994
0.0,9810.0
10.0,9810.0
*CONTACT_AUTO_MOVE
$       ID    ContID       VID      LCID     ATIME
        -1         2        89         3       0.1
*END
```

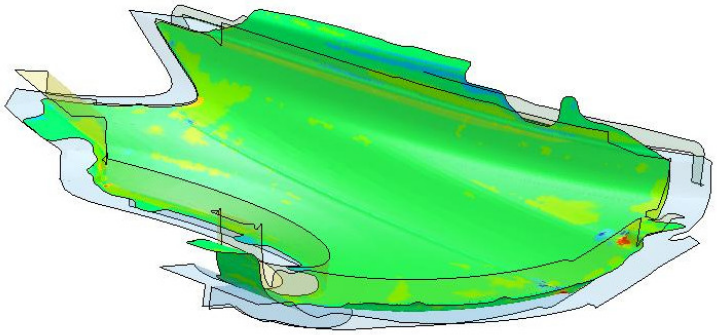


Figure 12-76. Mean stress at pad closing.

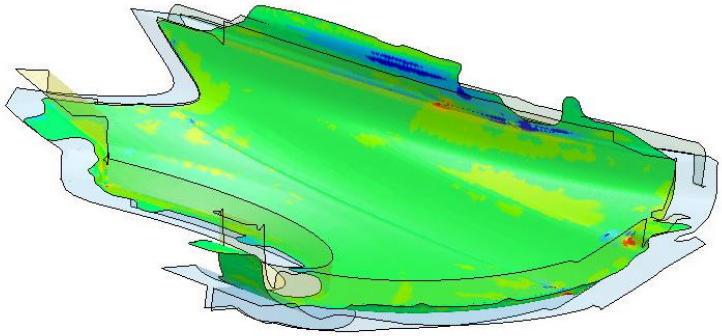


Figure 12-77. Mean stress at 40% Travel.

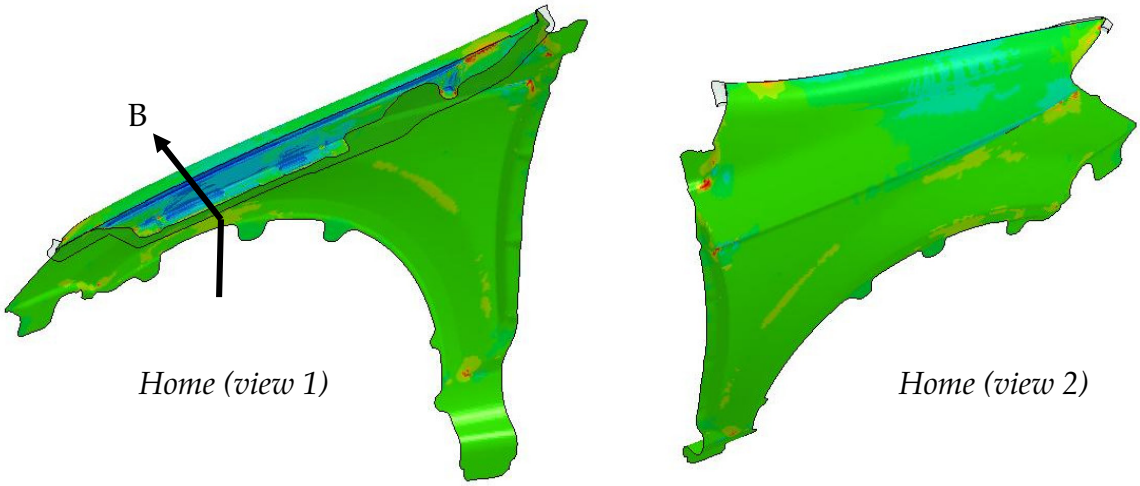


Figure 12-78. Mean stress at flanging home (compression/surface lows in red).

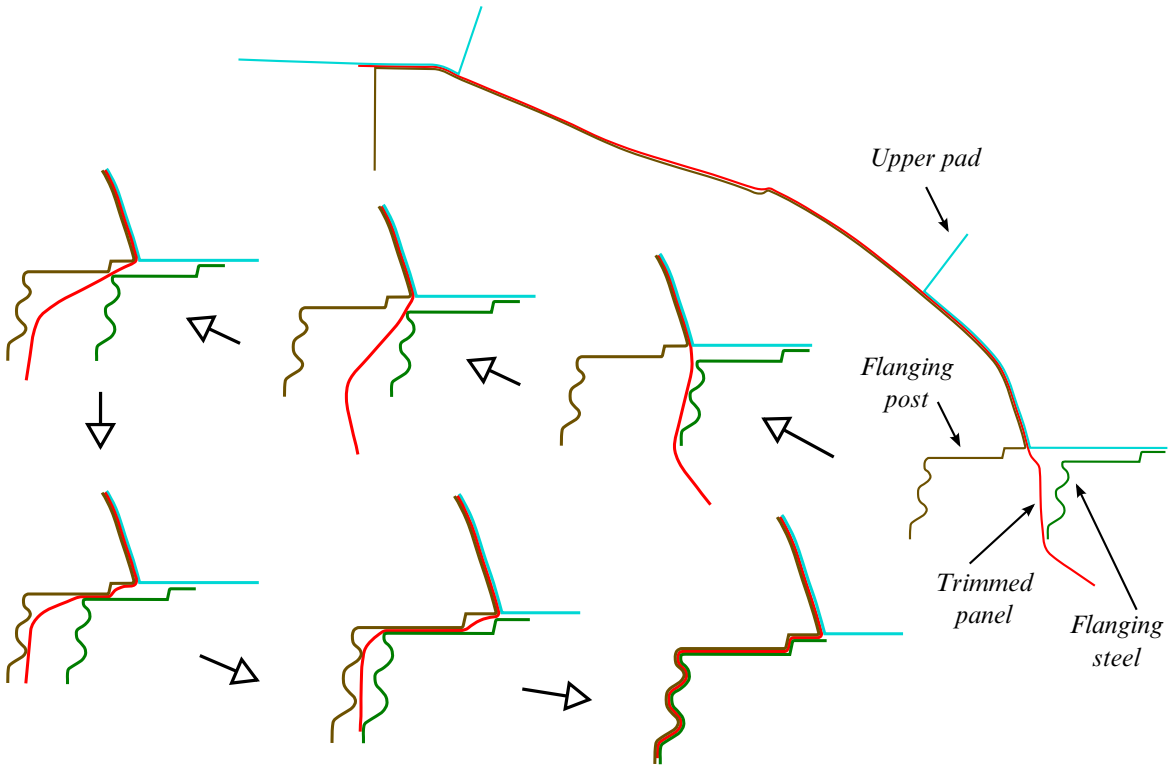


Figure 12-79. Flanging progression along section B (flanging post stationary).

Flanging Simulation Using IOPTION = 1:

IOPTION = 1 can also be used for closing and flanging simulation, or other applications in which there are large plastic strains or deformations. This is used when an equal step size throughout the simulation is desired and is done by specifying the equal step size in the variable DT0 in *CONTROL_IMPLICIT_GENERAL, as shown in the following partial keyword deck, where DT0 of 0.014 is chosen. An application of this is shown in Figures 12-80 and 12-81.

```
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_FORMING
$   IOPTION
         1
*CONTROL_IMPLICIT_GENERAL
$    IMFLAG        DT0
         1      0.014
```

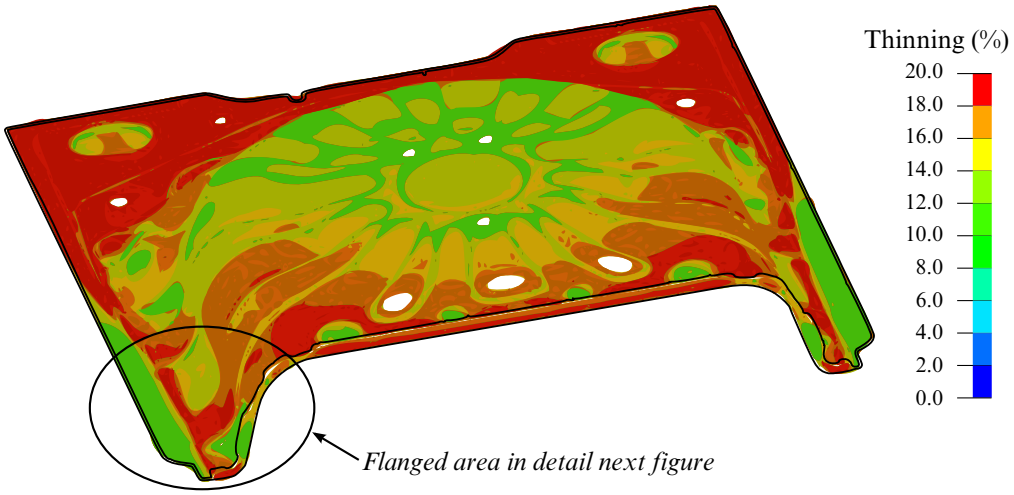


Figure 12-80. Flanging simulation of a rear floor pan using IOPTION 1 (Courtesy of Chrysler, LLC).

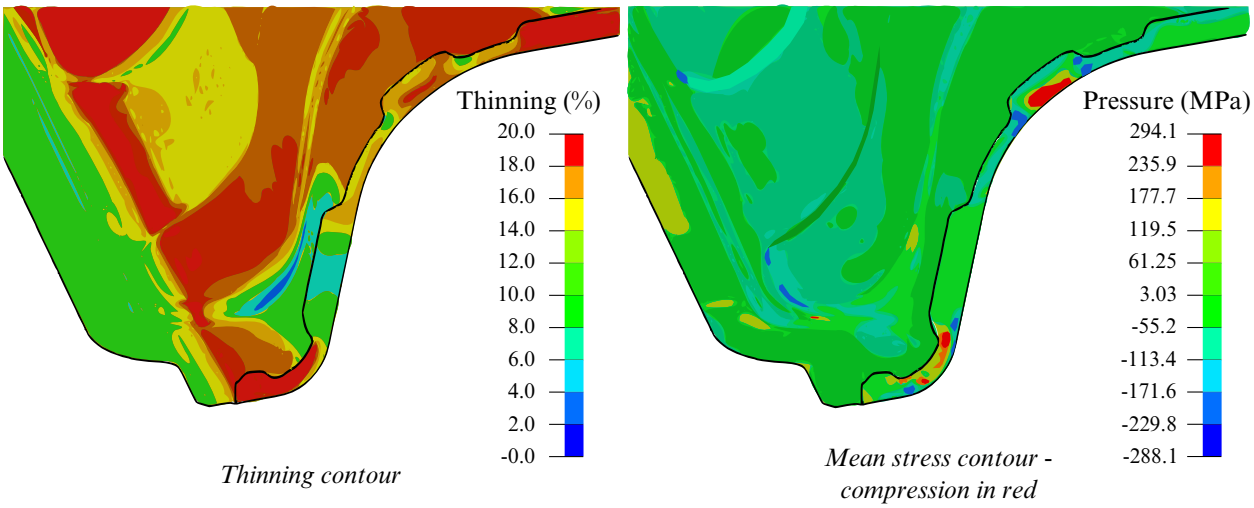


Figure 12-81. Localized view of the last figure.

Switching between Implicit Dynamic and Implicit Static for Gravity Loading:

For sheet blank gravity loading, it is now possible to start the simulation using the implicit dynamic method and then switch to the implicit static method at a user-defined time until completion. This feature is activated by setting the variable TDYDTH in *CONTROL_IMPLICIT_DYNAMICS and was recently (Rev. 81400) linked together with *CONTROL_IMPLICIT_FORMING. In a partial keyword example below, the death time for the implicit dynamic is set at 0.55 seconds. The test model shown in Figure 12-82 (left) results in a gravity-loaded blank shape in Figure 12-82 (right). Without the switch from dynamic to static, the blank at the end of the simulation will be as shown in Figure 12-83. The result with switching is more reasonable. The energy history in Figure 12-84 reveals that the kinetic energy dissipates completely at 0.60 seconds.

```
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_FORMING
$   IOPTION      NSMIN      NSMAX      BIRTH      DEATH     PENCHK
         1
*CONTROL_IMPLICIT_DYNAMICS
$     IMASS      GAMMA       BETA     TDYBIR     TDYDTH     TDYBUR      IRATE
         1      0.600      0.380                  0.55
```

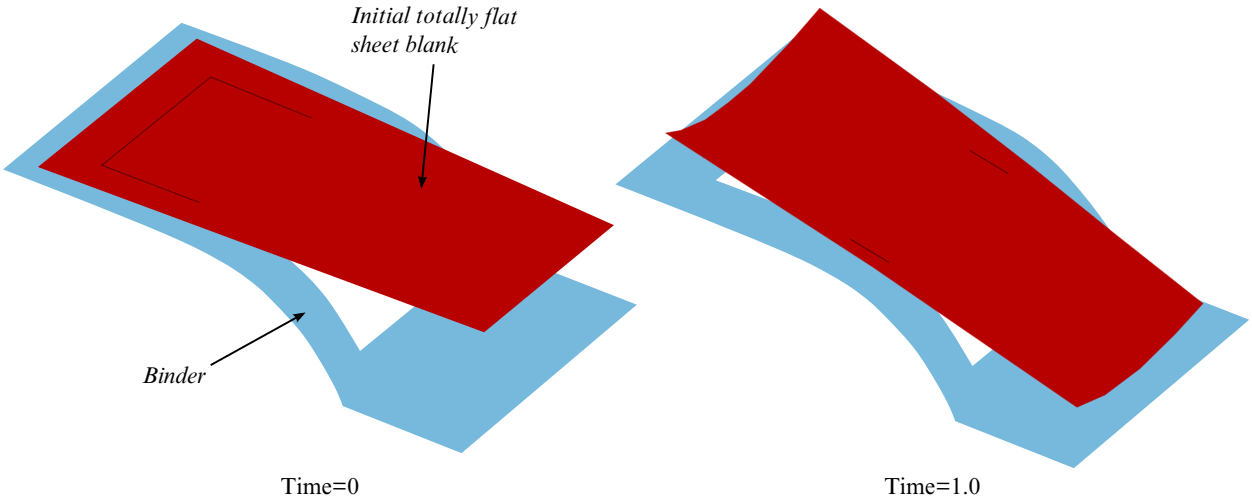


Figure 12-82. Test model (left) and gravity loaded blank (right) with switching from implicit dynamic to implicit static.

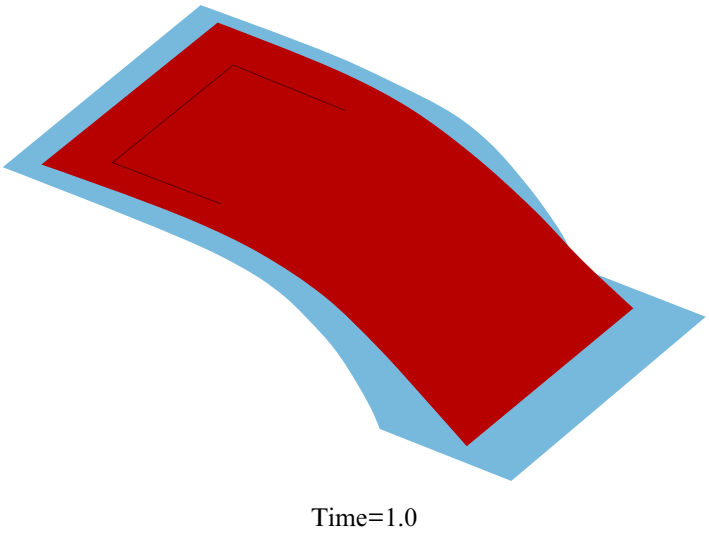


Figure 12-83. Gravity loaded blank without the “switching”.

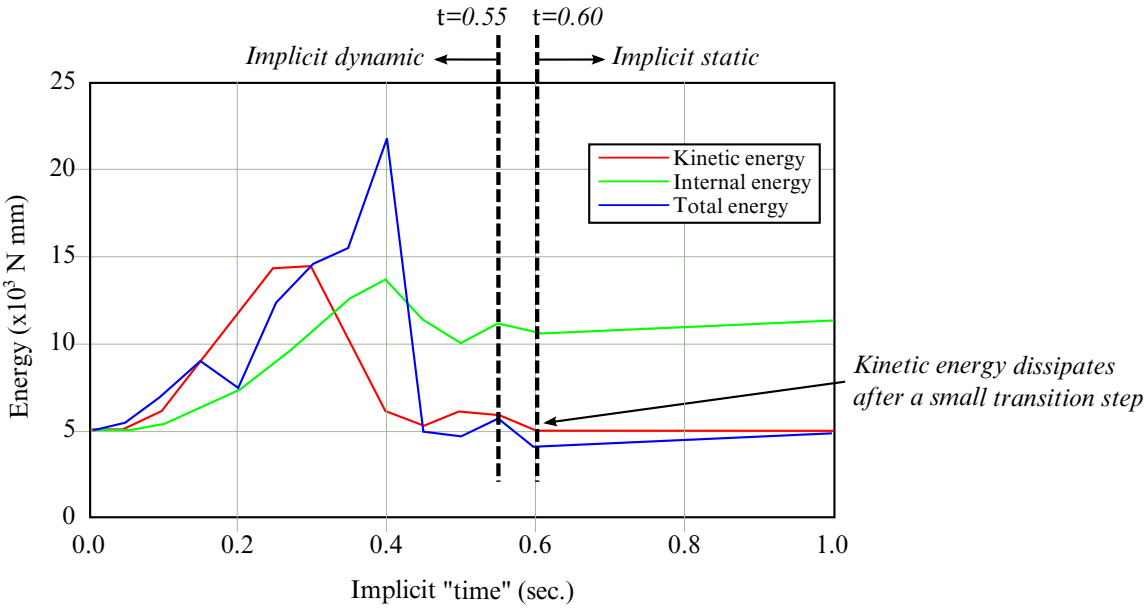


Figure 12-84. Switching between implicit dynamic and implicit static.

*CONTROL_IMPLICIT_GENERAL_{OPTION}

Available options include:

<BLANK>

DYN

SPR

Purpose: Activate implicit analysis and define associated control parameters. This keyword is required for all implicit analyses. The DYN option enables setting controls specifically for the dynamic relaxation phase. The SPR option enables setting controls specifically for the springback phase.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_GENERAL. For example, if ZERO_V = 1 for *CONTROL_IMPLICIT_GENERAL and the value of ZERO_V is left blank for *CONTROL_IMPLICIT_GENERAL_DYN, then the value of ZERO_V for *CONTROL_IMPLICIT_GENERAL_DYN will also be 1.

Card 1	1	2	3	4	5	6	7	8
Variable	IMFLAG	DT0	IMFORM	NSBS	IGS	CNSTN	FORM	ZERO_V
Type	I	F	I	I	I	I	I	I
Default	0	none	2	1	2	0	0	0

VARIABLE	DESCRIPTION
IMFLAG	Implicit/explicit analysis type flag (see Remark 1): EQ.0: Explicit analysis EQ.1: Implicit analysis EQ.2: Explicit followed by implicit, (seamless springback). *INTERFACE_SPRINGBACK_SEAMLESS is required to

VARIABLE	DESCRIPTION
	activate seamless springback. EQ.4: Implicit with automatic implicit-explicit switching EQ.5: Implicit with automatic switching and mandatory implicit finish EQ.6: Explicit with intermittent eigenvalue extraction LT.0: Curve ID = -IMFLAG specifies IMFLAG as a function of time.
DT0	Initial time step size for implicit analysis. See Remarks 2 and 5. LT.0.0: Eliminate negative principal stresses in geometric (initial stress) stiffness. Initial time step is |DT0|.
IMFORM	Element formulation flag for seamless springback; see *INTERFACE_SPRINGBACK_SEAMLESS. See Remark 3. EQ.1: Switch to fully integrated shell formulation for springback EQ.2: Retain original element formulation (default)
NSBS	Number of implicit steps in seamless springback; see *INTERFACE_SPRINGBACK_SEAMLESS. See Remark 4.
IGS	Geometric (initial stress) stiffness flag (see Remark 5): EQ.1: Include EQ.2: Ignore LT.0: Include on part set |IGS|
CNSTN	Indicator for consistent tangent stiffness (solid materials 3 & 115 only): EQ.0: Do not use (default) EQ.1: Use
FORM	Fully integrated element formulation (IMFLAG = 2 and IMFORM = 1 only). See Remark 3. EQ.0: Type 16 EQ.1: Type 6
ZERO_V	Zero out the velocity before switching from explicit to implicit. EQ.0: The velocities are not zeroed out.

VARIABLE	DESCRIPTION
	EQ.1: The velocities are set to zero.

Remarks:

1. **Analysis types.** The default value 0 for IMFLAG indicates a standard explicit analysis will be performed. Using value 1 causes an entirely implicit analysis to be performed. Value 2 is automatically activated when the keyword *INTERFACE_SPRINGBACK_SEAMLESS is present, causing the analysis type to switch from explicit to implicit when the termination time is reached. Other nonzero values for IMFLAG can also be used with *INTERFACE_SPRINGBACK_SEAMLESS. After this switch, the termination time is extended by NSBS × DT0 or reset to twice its original value if DT0 = 0.0. The implicit simulation then proceeds until the new termination time is reached. Contact interfaces are automatically disabled during the implicit phase of seamless springback analysis. Furthermore, implicit stabilization (*CONTROL_IMPLICIT_STABILIZATION) and automatic step size adjustment (*CONTROL_IMPLICIT_AUTO) are on by default for seamless springback.

   When the automatic implicit-explicit switching option is activated (IMFLAG = 4 or 5), the solution method will begin as implicit. If convergence of the equilibrium iterations fails, the solution will automatically switch to explicit for a time interval of DTEXP (see *CONTROL_IMPLICIT_AUTO). After this time interval, the solution method will switch back to implicit and attempt to proceed. The implicit simulation may be either static or dynamic. When this feature is used in a static implicit job, simulation time is no longer arbitrary and must be chosen along with DTEXP in a realistic way to cause efficient execution of any explicit phases. Mass scaling may also be activated (see *CONTROL_TIMESTEP) and will apply only during the explicit phases of the calculation. In cases where much switching occurs, users must exercise caution to ensure that negligible dynamic effects are introduced by the explicit phases.

   When IMFLAG = 5, the final step of the simulation must be implicit. The termination time will be extended automatically as necessary, until a successfully converged implicit step can be obtained. This is useful for example in difficult metal forming springback simulations.

   When IMFLAG = 6, an explicit simulation will be performed. Eigenvalues will be extracted intermittently according to a curve indicated by curve -NEIG on *CONTROL_IMPLICIT_EIGENVALUE. Beware that dynamic stress oscillations which may occur in the explicit simulation will influence the geometric (initial stress) stiffness terms used in the eigen solution, potentially producing misleading results and/or spurious modes. As an alternative, eigenvalues can

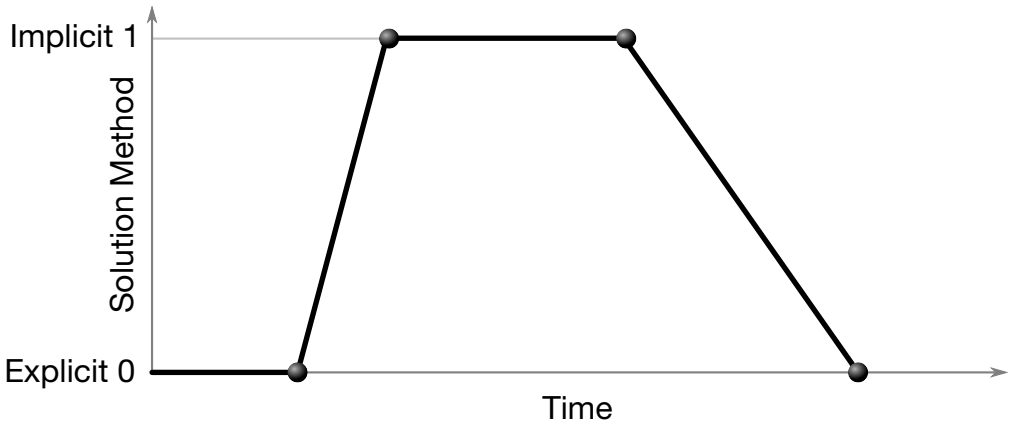


Figure 12-85. Solution method, implicit or explicit, controlled by a load curve.

also be extracted intermittently during an implicit analysis, using IMFLAG = 1 and curve -NEIG.

IMFLAG < 0 indicates a curve ID that gives the solution method as a function of time. Define a curve value of zero during explicit phases, and a value of one during implicit phases. Use steeply sloping sections between phases. An arbitrary number of formulation switches may be activated with this method. See Figure 12-85.

2. **Implicit phase initial time step size.** DT0 selects the initial time step size for the implicit phase of a simulation. The step size may change during a multiple step simulation if the automatic time step size control feature is active (see *CONTROL_IMPLICIT_AUTO).

3. **Element formulation switching.** An adaptive mesh must be activated when using element formulation switching. For best springback accuracy, use of shell type 16 is recommended during the entire stamping and springback analysis, in spite of the increased cost of using this element during the explicit stamping phase.

4. **Seamless springback.** The NSBS option allows a seamless springback analysis, invoked with *INTERFACE_SPRINGBACK_SEAMLESS, to use multiple unloading steps. Implicit seamless springback begins at time, t = ENDTIM and finishes at t = ENDTIM + NSBS × DT0 were ENDTIM is specified in *CONTROL_TERMINATION and DT0 is specified in *CONTROL_IMPLICIT_GENERAL.

5. **Geometric stiffness.** The geometric stiffness (IGS) adds the effect of initial stress to the global stiffness matrix. This effect is seen in a piano string whose natural frequency changes with tension. Geometric stiffness does not always improve nonlinear convergence, especially when compressive stresses are present, so its inclusion is optional. Furthermore, the geometric stiffness may lead

to convergence problems with incompressible, or nearly incompressible, materials. To eliminate compressive stresses in the geometric stiffness, set DT0 to a negative value.

*CONTROL_IMPLICIT_INERTIA_RELIEF

Purpose: Allows analysis of models that have rigid body modes by suppressing the rigid body motion. It can be used for implicit problems as well as implicit/explicit switching.

Card 1	1	2	3	4	5	6	7	8
Variable	IRFLAG	THRESH	IRCNT	MXMODES				
Type	I	F	I	I				
Default	0	0.1	0	100				

Additional Mode List Cards. This card should be included only when the user wants to specify the modes to use. Include as many cards as needed to provide all values. This input ends at the next keyword ("*") card. The mode numbers do not have to be consecutive.

Card 2	1	2	3	4	5	6	7	8
Variable	MODE1	MODE2	MODE3	MODE4	MODE5	MODE6	MODE7	MODE8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
IRFLAG	Inertia relief flag EQ.0: Do not perform inertia relief EQ.1: Do perform inertia relief and use for both implicit and explicit EQ.2: Do perform inertia relief but only use for implicit time steps
THRESH	Threshold for what is a rigid body mode. The default is set to 0.1. Thus, the default threshold is 0.1 Hz when using seconds for the unit of time. If the unit of time is not seconds, you may need to adjust this parameter appropriately. For instance, a unit of time of ms means that the default is 0.1 kHz which leads to too many modes.

VARIABLE	DESCRIPTION
IRCNT	The user can specify to use the lowest IRCNT modes instead of using THRESH to determine the number of modes.
MXMODES	Maximum number of allowed modes that can be stored in memory
MODE*i*	Ignore THRESH and IRCNT and use a specific list of modes, skipping those that should not be used.

*CONTROL_IMPLICIT_JOINTS

Purpose: Specify penalty or constraint treatment of joints for implicit analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	ISPHER	IREVOL	ICYLIN					
Type	I	I	I					
Default	1	1	1					

VARIABLE	DESCRIPTION
ISPHER	Treatment of spherical joints EQ.1: use constraint method for all spherical joints (default) EQ.2: use penalty method for all spherical joints
IREVOL	Treatment of revolute joints EQ.1: use constraint method for all revolute joints (default) EQ.2: use penalty method for all revolute joints
ICYLIN	Treatment of cylindrical joints EQ.1: use constraint method for all cylindrical joints (default) EQ.2: use penalty method for all cylindrical joints

Remarks:

For most implicit applications one should use the constraint (default) method for the treatment of joints. When explicit-implicit switching is used the joint treatment should be consistent. This keyword allows the user to choose the appropriate treatment for their application.

*CONTROL_IMPLICIT_MODAL_DYNAMIC

Purpose: Activate implicit modal dynamic analysis. Eigenmodes are used to linearize the model by projecting the model onto the space defined by the eigenmodes. The eigenmodes can be computed or read from a file. All or some of the modes can be used in the linearization. Modal damping can be applied.

Card 1	1	2	3	4	5	6	7	8
Variable	MDFLAG	ZETA	MD_STRS	DTOUT	INTEG	NSID		
Type	I	F	I	F	I	I		
Default	0	0.0	0	0	1	{all}		

Eigenmodes Card. This card is optional if MDFLAG = 1, but it is required, even if blank, if MDFLAG > 1.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

Constraints Card. Include this card if MDFLAG = 2 or 4.

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME2							
Type	A80							

Residual Vector Card. Include this card if MDFLAG = 3 or 4.

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME3							
Type	A80							

VARIABLE	DESCRIPTION
MDFLAG	Modal Dynamic flag: EQ.0: No modal dynamic analysis EQ.1: Perform modal dynamic analysis. EQ.2: Perform modal dynamic analysis with prescribed motion constraints on the constraint modes input with Card 3. See Remark 7. EQ.3: Perform modal dynamic analysis with the addition of residual vectors to the eigenvectors. EQ.4: Combine analysis types 2 and 3.
ZETA	Modal Dynamic damping constant. See Remark 4.
MD_STRS	Modal dynamic stress flag: EQ.0: No modal stress calculated NE.0: Compute modal stresses for output to d3plot using the modal stresses from the d3eigv database. See Remark 3.
DTOUT	Modal dynamics output interval. LE.0: No modal variable output GT.0: Output modal displacement, velocity, and acceleration to file moddynout every DTOUT of simulation time.
INTEG	Integration method EQ.1: perform modal dynamic analysis with explicit time integration (default_. EQ.2: perform modal dynamic analysis with implicit time integration. See Remark 6.
NSID	The node set ID of the nodes subjected to loads in the modal model. If the set is not specified, then the forces are summed over all the nodes, and that is usually much more expensive than summing over only those subjected to a load.
FILENAME	If specified, the eigenmodes are read from the specified file. Otherwise, the eigenmodes are computed as specified on *CONTROL_IMPLICIT_EIGENVALUE. See Remark 2.

VARIABLE	DESCRIPTION
FILENAME2	If specified, constraint modes are read from the specified file. Prescribed motion constraints can then be applied to the constraint modes. See Remark 7.
FILENAME3	If specified, the residual vectors are read from the specified file. Residual vectors can be used in conjunction with eigenmodes for the analysis. See Remark 8.

Remarks:

1. **Modal dynamics.** Modal Dynamic uses the space spanned by the eigenmodes of the generalized eigenvalue problem

$$\mathbf{K}\boldsymbol{\phi}_i = \lambda_i \mathbf{M}\boldsymbol{\phi}_i.$$

The matrix of eigenmodes, $\boldsymbol{\Phi}$, diagonalizes $\mathbf{K}$ and $\mathbf{M}$

$$\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{K}\boldsymbol{\Phi} = \boldsymbol{\Lambda}$$

and

$$\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{M}\boldsymbol{\Phi} = \mathbf{I}.$$

Multiplication by $\boldsymbol{\Phi}$ changes coordinates from amplitude space to displacement space as

$$\mathbf{u} = \boldsymbol{\Phi}\mathbf{a}\ ,$$

where $\mathbf{a}$ is a vector of modal amplitudes. The equations of motion,

$$\mathbf{M}\ddot{\mathbf{u}}^{n+1} + \mathbf{K}\Delta\mathbf{u} = \mathbf{F}(\mathbf{x}^n)\ ,$$

when multiplied on the left by $\boldsymbol{\phi}^{\mathrm{T}}$ and substituting $\mathbf{u} = \boldsymbol{\Phi}\mathbf{a}$ become the linearized equations of motion in its spectral form as

$$\mathbf{I}\ddot{\mathbf{a}}^{n+1} + \boldsymbol{\Lambda}(\Delta\mathbf{a}) = \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{F}(\mathbf{x}^n).$$

The modal damping feature adds a velocity-dependent damping term,

$$\mathbf{I}\ddot{\mathbf{a}}^{n+1} + 2\mathbf{Z}\dot{\mathbf{a}}^n + \boldsymbol{\Lambda}(\Delta\mathbf{a}) = \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{F}(\mathbf{x}^n)\ ,$$

where $Z_{ii} = \zeta_i\omega_i$, $\omega_i = \sqrt{\lambda_i}$, and each ζ_i is a user-specified damping coefficient.

The matrices in the reduced equations are diagonal and constant. Thus, Modal Dynamics can quickly compute the acceleration of the amplitudes and, hence, the motion of the model. The space spanned by the eigenmodes, however, restricts the motion.

2. **Eigenmodes.** Eigenmodes are either computed based on *CONTROL_IMPLICIT_EIGENVALUE or read from file FILENAME. By default, all modes are used

in the projection. Selected modes can be specified via *CONTROL_IMPLICIT_-MODAL_DYNAMIC_MODE to reduce the size of the projection.

3. **Stress output.** Stresses are computed only for linear shell formulation 18, linear solid formulation 18, and linear beam formulation 13.

4. **Modal damping.** Modal damping on all modes can be specified using ZETA. More options for specifying modal damping can be found on *CONTROL_IM-PLICIT_MODAL_DYNAMIC_DAMPING.

5. **Similar keywords.** Using MDFLAG = 1, ZETA = 0.0, and FILENAME = " " is the same as using IMASS = 2 with *CONTROL_IMPLICIT_DYNAMICS. Using MDFLAG = 1, ZETA = 0.0 and FILENAME = d3eigv is the same as IMASS = 3. The new keywords *CONTROL_IMPLICIT_MODAL_DYNAMIC_MODE and *CONTROL_IMPLICIT_MODAL_DYNAMIC_DAMPING provide additional options for mode selection and modal damping.

6. **Time integration method.** When INTEG is set to 2, the implicit Newmark time integration method is used instead of the explicit central difference method, and the time step size is taken from the *CONTROL_IMPLICIT_GENERAL input. The integration parameters are taken from the *CONTROL_IMPLICIT_DY-NAMICS input. If they are not specified, the default values of GAMMA and BETA are used. This option permits a time step size limited only by the accuracy requirements of the user.

7. **Prescribed Motion Constraints.** To apply prescribed motion constraints in global coordinates at specified nodes, first use *CONTROL_IMPLICIT_MODES to compute constraint modes for these nodes for this model. The resulting d3mode file is then input as FILENAME2. The computation of the constraint modes must be performed separately from the computation of the eigenmodes specified in file FILENAME. The prescribed motion constraint (*BOUNDARY_-PRESCRIBED_MOTION) should only be present in the execution with *CON-TROL_IMPLICIT_MODAL_DYNAMIC and should not be present during the computation of the eigenmodes or the constraint modes. Any prescribed motion constraints on any node *not* in the constraint mode set will be ignored.

8. **Residual vectors.** Residual vectors can be used in conjunction with eigenmodes in the analysis to better represent specific loading conditions.

*CONTROL_IMPLICIT_MODAL_DYNAMIC_DAMPING_{OPTION}

Available options include:

<BLANK>

SPECIFIC

FREQUENCY_RANGE

Purpose: Define damping terms to be used in implicit modal dynamics.

Card Summary:

Card 1a. Include this card if and only if the keyword option is unused, that is, <BLANK>.

ZETA1							

Card 1b. Include this card if and only if the SPECIFIC keyword option is used. This input ends at the next keyword ("*") card.

MID1	ZETA1	MID2	ZETA2	MID3	ZETA3	MID4	ZETA4

Card 1c. Include this card if and only if the FREQUENCY_RANGE keyword option is used. This input ends at the next keyword ("*") card.

FREQ1	ZETA1	FREQ2	ZETA2	FREQ3	ZETA3	FREQ4	ZETA4

Data Card Definitions:

Damping Card. Card for option set to <BLANK>.

Card 1a	1	2	3	4	5	6	7	8
Variable	ZETA1							
Type	F							

Specific Damping Cards. Card for the SPECIFIC option. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	MID1	ZETA1	MID2	ZETA2	MID3	ZETA3	MID4	ZETA4
Type	I	F	I	F	I	F	I	F

Frequency Range Damping Cards. Card for FREQUENCY_RANGE option. This input ends at the next keyword ("*") card.

Card 1c	1	2	3	4	5	6	7	8
Variable	FREQ1	ZETA1	FREQ2	ZETA2	FREQ3	ZETA3	FREQ4	ZETA4
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
ZETA*n*	Modal Dynamic damping coefficient *n*.
MID*n*	Mode ID *n*.
FREQ*n*	Frequency value *n*.

Remarks:

1. **No keyword option.** If no option is specified, the value of ZETA1 becomes the damping coefficient for all modes involved in implicit modal dynamic analysis. This value overrides the value on *CONTROL_IMPLICIT_MODAL_DYNAMIC.

2. **SPECIFIC keyword option.** If option SPECIFIC is specified, the integers MID*n* indicate which modes involved in *CONTROL_IMPLICIT_MODAL_DYNAMIC will have modal damping applied to them. The associated value ZETA*n* will be the modal damping coefficient for that mode.

3. **FREQUENCY_RANGE keyword option.** If option FREQUENCY_RANGE is specified, all modes involve will have modal damping applied. The damping coefficient will be computed by linear interpolation of the pairs (FREQ*i*,ZETA*i*). If the modal frequency is less than FREQ1, then the modal damping coefficient will be ZETA1. If the modal frequency is greater than FREQ*n*, then the modal

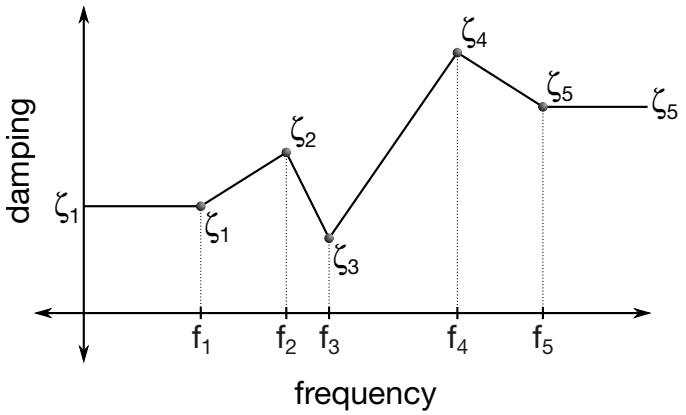


Figure 12-86. Schematic illustration of frequency range damping.

damping coefficient will be ZETA*n*. The values of FREQ*i* must be specified in ascending order.

CONTROL_IMPLICIT_MODAL_DYNAMIC_MODE_*OPTION

Available options include:

LIST

GENERATE

Purpose: Define vibration modes to be used in implicit modal dynamic.

Mode ID Cards. Card 1 for the LIST keyword option. Set one value per mode. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	MID1	MID2	MID3	MID4	MID5	MID6	MID7	MID8
Type	I	I	I	I	I	I	I	I

Mode Range Cards. Card 1 for the GENERATE keyword option. Set one pair of M*n*BEG and M*n*END values per block of modes. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	M1BEG	M1END	M2BEG	M2END	M3BEG	M3END	M4BEG	M4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
MID*n*	Mode ID *n*.
M*n*BEG	First mode ID in block *n*.
M*n*END	Last mode ID in block *n*. All mode IDs between and including M*n*BEG and M*n*END are added to the list.

Remarks:

1. **Related Keywords.** This keyword may be used with *CONTROL_IMPLICIT_-MODAL_DYNAMIC if some of the vibration modes have less contribution to

the total structural response and can be removed from the implicit modal dynamic analysis.

*CONTROL_IMPLICIT_MODES_{OPTION}

Available options include:

<BLANK>

BINARY

Purpose: Request calculation of constraint, attachment, and/or eigenmodes for later use in modal analysis using *PART_MODES (see also *CONTROL_IMPLICIT_GENERAL) or *ELEMENT_DIRECT_MATRIX_INPUT.

Card Summary:

Card 1. This card is required.

NSIDC	NSIDA	NEIG	IBASE	SE_MASS	SE_DAMP	SE_STIFF	SE_INERT

Card 1.1. Include this card if at least one of SE_MASS, SE_DAMP, SE_STIFF, or SE_INERT is not blank.

SE_FILENAME

Card 2. This card is optional.

IRESVEC	ISTRESS	ID3MODE	UNITSC		BUFSIZ		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NSIDC	NSIDA	NEIG	IBASE	SE_MASS	SE_DAMP	SE_STIFF	SE_INERT
Type	I	I	I	I	A	A	A	A
Default	{∅}	{∅}	0	0	blank	blank	blank	blank

Card 1.1	1	2	3	4	5	6	7	8
Variable	SE_FILENAME							
Type	A							

Card 2	1	2	3	4	5	6	7	8
Variable	IRESVEC	ISTRESS	ID3MODE	UNITSC		BUFSIZ		
Type	I	I	I	I		I		
Default	0	0	0	0		1		

VARIABLE	DESCRIPTION
NSIDC	Node set ID for constraint modes. See Remark 2. EQ.0: No constraint modes will be generated
NSIDA	Node set ID for attachment modes. See Remark 2. EQ.0: No attachment modes will be generated
NEIG	Number of eigenmodes (normal modes). See Remark 2. EQ.0: No eigenmodes will be generated.
IBASE	Offset for numbering of the generalized internal degrees of freedom for the superelement
SE_MASS	Name of the superelement mass matrix. If left blank, it is not generated.
SE_DAMP	Name of the superelement damping matrix. If left blank, it is not generated.
SE_STIFF	Name of the superelement stiffness matrix. If left blank, it is not generated.
SE_INERT	Name of the superelement inertia matrix, required for gravity loading applications of the superelement. If left blank, it is not generated.

VARIABLE	DESCRIPTION
SE_FILENAME	If any of SE_MASS, SE_DAMP, SE_STIFF, or SE_INERT are not blank, then the second line is required and contains the filename for the superelement. See Remark 3.
IRESVEC	Converting the attachment modes to residual vectors flag (see Remark 4): EQ.0: No conversion EQ.1: Conversion by orthogonalizing them to the eigenvectors.
ISTRESS	Flag to compute stresses: EQ.0: No stresses are computed. EQ.1: Stresses are computed for the mode shapes and added to the d3mode database.
ID3MODE	Write d3mode file flag: EQ.0: Write the d3mode file. EQ.1: Do not write the d3mode file. This is useful when *CONTROL_IMPLICIT_MODES is being used to create superelement matrices and you do not care about the contents of the d3mode file.
UNITSC	Set to 1 to use consistent units in subsequent analyses that use the matrices. In other words, the solution variables for the eigenmodes get units that are consistent with the displacement and rotations for norm calculations. When left as zero, the matrices are such that the mode degrees of freedom are always unitless. This can cause inconsistent norm calculations in the implicit solver.
BUFSIZ	Buffer size in i/o treatment of large superelement generations. The default is 1. It may be worthwhile setting BUFSIZ to a larger number if large superelements are to be generated for large model files. The intention is to speed up execution of the block linear algebraic operations needed for creating the reduced matrices and decrease the number of i/o calls.

Remarks:

1. **Input and output files.** To use this feature, an implicit analysis must be requested using IMFLAG = 1 on *CONTROL_IMPLICIT_GENERAL, and a nonzero termination time must be specified on *CONTROL_TERMINATION. A double precision version of LS-DYNA should be used for best accuracy. Care must be taken to apply a sufficient number of constraints to the model to eliminate static rigid body motion. Computed modes are written to binary output file d3mode, with the order of output being constraint modes, followed by attachment modes, and then eigenmodes. The modes can be viewed by reading d3mode into LS-PrePost. Eigenmodes are also written to binary output file d3eigv.

2. **Modes.** Constraint and attachment modes are generated by applying unit displacements and unit forces, respectively, to each specified degree of freedom. By default, modes are computed for all degrees of freedom for each node in sets NSIDC and NSIDA. The first and second node set attribute parameters can be optionally used to restrict the translational and rotational degrees of freedom for which modes are requested, respectively, according to the following syntax:

   a) Node set attribute parameters DA1 and A1: translational degree of freedom codes

   b) Node set attribute parameters DA2 and A2: rotational degree of freedom codes

code	modes computed
0	(See note below.)
1	X degree of freedom only
2	Y degree of freedom only
3	Z degree of freedom only
4	X, Y degrees of freedom only
5	Y, Z degrees of freedom only
6	X, Z degrees of freedom only
7	X, Y, Z degrees of freedom

   Setting both node set attributes to zero is equivalent to setting both node set attributes to 7 (X, Y, and Z for translational and rotational degrees of freedom).

   If one node set attribute is nonzero (codes 1 to 7) and the other node set attribute is zero, then the zero attribute means NO degrees of freedom are considered. For example, if DA1 = 2 and DA2 = 0, then only the Y-translational degree of freedom modes are calculated.

Eigenmodes are generated for the model with single point constraints applied on the constraint modes. The number of eigenmodes is specified here. If the user wants to compute eigenmodes other than the lowest ones, the controls on *CONTROL_IMPLICIT_EIGENVALUE can be used.

The combination of constraint modes and eigenmodes form the Hurty-Craig-Bampton linearization for a model. Using only constraint modes is the same as static condensation.

Some broad guidelines for appropriate selection of constraint modes, attachment modes, and eigenmodes include:

c) Use constraint modes for the nodal degrees-of-freedom that are to be "constrained" with SPCs or prescribed motion.

d) Use attachment modes for nodal degrees-of-freedom that are under the influence of point loads.

e) Use eigenmodes in the construction of the superelement to capture the reaction of the part being modeled by the superelement and the associated feedback to the rest of the model.

Note: This node set should only contain deformable nodes, using a node that is part of a rigid body will result in errors.

3. **Superelements.** When the superelement is created, an internal numbering must be applied to the attachment and eigenmodes. This numbering starts at IBASE+1.

   The user can create the superelement representation of the reduced model by specifying the SE_MASS, SE_DAMP, SE_STIFF, SE_INERT and SE_FILENAME fields. The inertia matrix is necessary if body forces, such as gravity loads, are applied to the superelement. The file, by default, is written in the Nastran DMIG file format and can be used as input to *ELEMENT_DIRECT_MATRIX_INPUT. The BINARY keyword option can be used to create a binary representation for the superelement which can be used with *ELEMENT_DIRECT_MATRIX_INPUT_BINARY to reduce the file size.

4. **Residual vectors.** For some applications residual vectors may be desired. These are attachment modes (response of the model to a unit force) orthogonalized to the eigenmodes. To do this, set IRESVEC to 1.

*CONTROL_IMPLICIT_ORDERING

Purpose: Provide user control for ordering algorithms used by implicit linear algebra.

Card 1	1	2	3	4	5	6	7	8
Variable	ORDER	NMETIS	REUSE					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
ORDER	Ordering option (see Remarks 1 and 2): EQ.0: Method set automatically by LS-DYNA EQ.1: MMD (Multiple Minimum Degree) EQ.2: METIS EQ.3: ParMETIS EQ.4: LS-GPart
NMETIS	Number of times to use METIS on each compute node for MPP. See Remark 2.
REUSE	Control the ordering reuse feature (MPP only). See Remark 3.

Remarks:

1. **Ordering option.** The system of linear equations is reordered to optimize the sparsity of the factorization when using direct methods. Reordering is a hard problem, and there is no one best algorithm for all problems. LS-DYNA offers four different methods:

   a) *MMD (Multiple Minimum Degree)*. This reordering method is inexpensive but should only be used for small problems.

   b) *METIS.* METIS is from the University of Minnesota. It is the default method for most problems. It is very effective for large problems. METIS, however, is a serial algorithm. For large problems in MPP, however, a bottleneck in both time and memory will occur. Thus, for a large number of

MPP processes, you might want to use one of the two parallel algorithms mentioned next.

c) *ParMETIS.* ParMETIS is also from the University of Minnesota. It is the MPP version of METIS. For a large number of processes, it should run much faster than METIS, but there might be some variation in the factorization cost (storage and number of operations). It is available as of R14.

d) *LS-GPart.* LS-GPart is an MPP algorithm developed by Ansys and available as of R11. For a large number of processes, it should run faster than METIS. Quality (in terms of factorization cost) compared to METIS or ParMETIS will depend on the problem.

Reordering cost is included in the symbolic factorization phase of the linear solver. For large models, if this cost exceeds 20% of the numeric factorization cost, it may be more efficient to select the MMD method.

Note that the value of ORDER also affects the eigensolution software. That is ORDER from this keyword card is applicable to eigensolution.

2. **METIS in MPP.** In MPP LS-DYNA will run METIS on each compute node based on the estimate of required storage for METIS and the amount of free storage. The estimate is usually an overestimate. If insufficient storage is available, then MMD is used instead of METIS, usually leading to a more expensive solution. You can force the use of METIS with NMETIS > 0. However, a memory error and fatal termination of LS-DYNA may result in this case. If you do choose to try this option, NMETIS = 1 is recommended.

3. **Ordering reuse.** In MPP the cost of reordering for the direct factorization can be a substantial proportion of the overall cost of a linear algebra solution. LS-DYNA attempts to reuse the ordering from the previous factorization as the default. If the matrix structure changes due to contact or another dynamic feature, a reordering is performed. Otherwise, the previous ordering is reused. REUSE < 0 causes no attempt to reuse the ordering to be made. REUSE > 0 invokes attempting to predict matrix changes due to contact so that an ordering might be able to be reused for future factorization. Except for special problems, it is best to use REUSE = 0, the default. This is an MPP-only feature as the cost of ordering for SMP execution is not a significant portion of the overall cost.

*CONTROL_IMPLICIT_RESIDUAL_VECTOR

Purpose: Activate and control the computations of residual vectors. Residual vectors are the linear response of a model to a specified load which is then orthogonalized to a set of eigenmodes and any previously computed residual vectors. The eigenmodes can be computed during the execution or read from an input file. The computation of residual vectors is the same as multi-step linear (see NSOLVR = -1 on *CONTROL_IMPLICIT_SOLUTION) but has the additional step of orthogonalization.

Card Summary:

Card 1a. Include this card if NEIG > 0.

IRESVEC	NEIG	IFORMAT					

Card 1a.1. Include this card if IFORMAT ≠ 0.

RV_FILENAME

Card 1b. Include this card if NEIG = 0.

IRESVEC	NEIG	IFORMAT					

Card 1b.1. Include this card if IRESVEC ≠ 0.

RV_FILENAME

Data Card Definitions:

Compute Eigenmodes Card. This card is included if and only if NEIG > 0.

Card 1a	1	2	3	4	5	6	7	8
Variable	IRESVEC	NEIG	IFORMAT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
IRESVEC	Residual vector control flag: EQ.0: Do not compute residual vectors.

VARIABLE	DESCRIPTION
	GT.0: Compute residual vectors.
NEIG	Number of eigenmodes to compute to orthogonalize the computed load (NEIG > 0)
IFORMAT	Format for processing eigenmodes (NEIG > 0): EQ.0: Do not dump the computed eigenmodes. LT.0: Dump the computed eigenmodes in binary format. GT.0: Dump the computed eigenmodes in ASCII format.

Eigenmodes File Card. This card is included if and only if IFORMAT ≠ 0.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	RV_FILENAME							
Type	A80							

VARIABLE	DESCRIPTION
RV_FILENAME	Name of the file for dumping the eigenvector

Read Eigenmodes Card. This card is included if and only if NEIG = 0.

Card 1b	1	2	3	4	5	6	7	8
Variable	IRESVEC	NEIG	IFORMAT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
IRESVEC	Residual vector control flag: EQ.0: Do not compute residual vectors. GT.0: Compute residual vectors.

VARIABLE	DESCRIPTION
NEIG	Read the eigenmodes from the file RV_FILENAME (if IRESVEC ≠ 0), which is the file used for dumping eigenvectors; see EVDUMP on *CONTROL_IMPLICIT_EIGENVALUE.
IFORMAT	Format for processing eigenmodes read from RV_FILENAME when NEIG = 0 (if IRESVEC > 0): LT.0: File is in binary format. GT.0: File is in ASCII format. If IRSEVEC > 0 and NEIG = 0, IFORMAT = 0 is not allowed.

Eigenmodes File Card. This card is included if and only if IRESVEC ≠ 0.

Card 1b.1	1	2	3	4	5	6	7	8
Variable	RV_FILENAME							
Type	A80							

VARIABLE	DESCRIPTION
RV_FILENAME	Name of the file read to obtain the eigenvectors. See EVDUMP on *CONTROL_IMPLICIT_EIGENVALUE.

Remarks:

For each load step, LS-DYNA resets the geometry to the original geometry and computes the response for the given load. This feature is similar to multi-step linear (NSOLVR = -1 for *CONTROL_IMPLICIT_SOLUTION). Then, the algorithm orthogonalizes the response with respect to the eigenmodes using the mass matrix norm and with respect to any previously computed residual vectors. Finally, LS-DYNA normalizes the residual vector with respect to the mass matrix.

LS-DYNA reads the eigenmodes from an eigenvector dump file or computes them depending on the values of NEIG. RV_FILENAME gives the file name.

LS-DYNA creates an LSDA database named residual_vectors to hold the results of these computations. For each load step, the database includes the following:

- the specified load
- the residual vector, **v**

- the residual frequency ($\mathbf{v}^T\mathbf{K}\mathbf{v}/\mathbf{v}^T\mathbf{M}\mathbf{v}$)

Please contact Technical Support for instructions for reading this database.

In addition, LS-DYNA writes a binary database d3resvec, which LS-PrePost can read to animate residual vectors.

*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS

Purpose: Model rotational dynamics using the implicit time integrator. Applications for this feature include the transient and vibration analysis of rotating parts, such as turbine blades, aircraft propellers, and rotating disks in hard disk drives. This feature requires the use of a double precision executable. For transient simulations (NOMEG = 0), both SMP and MPP are supported. The MPP implementation of the eigenvalue computations (NOMEG > 0) is not yet available.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	OMEGA	VID	NOMEG	IREF	OMEGADR	ISTFNS
Type	I	I	F	I	I	I	F	I
Default	none	0	none	none	0	0	0.0	3

Additional Rotational Speed Cards. This card should be included only when NOMEG > 0. Include as many cards as needed to provide all NOMEG values. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	OMEG1	OMEG2	OMEG3	OMEG4	OMEG5	OMEG6	OMEG7	OMEG8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
SID	Set ID of the rotational components
STYPE	Set type: EQ.0: Part EQ.1: Part set
OMEGA	Rotating speed in units of radians/(time unit). GT.0: Rotating speed LT.0: Curve ID = |OMEGA| gives rotating speed as a function of time.

VARIABLE	DESCRIPTION
VID	Vector ID to define the rotating axis. It can be defined by *DEFINE_VECTOR and *DEFINE_VECTOR_NODES. The tail of the vector should be set as the rotating center.
NOMEG	Number of rotating speeds. This feature is intended to automatically perform parameter studies with respect to the rotation speed. The keyword *CONTROL_IMPLICIT_EIGENVALUE must be included if NOMEG > 0.
IREF	Reference frame (see Remark 7): EQ.0: Rotating coordinate system. Rotating parts will not rotate in the visualization. Solid element and thick shell element will use IREF = 0. EQ.1: Fixed coordinate system. Using rotational dynamics will add Coriolis and spin softening terms to the matrices but will not rotate the structures. EQ.2: Rotating coordinate system, only rotate rotating parts for visualization purpose.
OMEGADR	Rotating speed defined in dynamic relaxation. GT.0: Rotating speed defined in dynamic relaxation. LT.0: Curve ID = |OMEGADR| gives rotating speed as a function of time.
ISTFNS	Flag for adding spin softening and gyroscopic stiffness terms: EQ.1: Add no stiffness for rotational dynamics effects. EQ.2: Add only the spin softening stiffness terms which keep the stiffness matrix symmetric. EQ.3: Add both the spin softening and gyroscopic stiffness terms which make the stiffness matrix nonsymmetric.
OMEG*n*	The *n*th rotating speed

Remarks:

1. **Equilibrium equation.** The equilibrium equations depend on the reference frame. For the rotating coordinate system, the linearized equilibrium equation is given by

$$\mathbf{M}\ddot{\mathbf{u}} + (\mathbf{D} + 2\Omega\mathbf{C})\dot{\mathbf{u}} + \left(\mathbf{K} - \Omega^2\mathbf{K}_G\right)\mathbf{u} = \mathbf{F}\,.$$

Whereas, in a fixed coordinate system, the linearized equilibrium equation is

$$\mathbf{M}\ddot{\mathbf{u}} + (\mathbf{D} + \Omega\mathbf{C})\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{F}$$

with

$\mathbf{M}$ = lumped mass matrix
$\mathbf{D}$ = damping matrix
$\mathbf{K}$ = stiffness matrix
$\mathbf{C}$ = gyroscopic matrix
$\mathbf{K}_G$ = centrifugal stiffness matrix
$\mathbf{u}$ = nodal displacement vector
$\dot{\mathbf{u}}$ = nodal point velocities at time
$\ddot{\mathbf{u}}$ = nodal point acceleration at time
Ω = rotating speed

The chief difference between the equations for the rotating and fixed frames is the inclusion of the centrifugal stiffness matrices $\mathbf{K}_g$. Additionally, the coefficient on the gyroscopic matrix, $\mathbf{C}$, as well as its content are modified in the rotating-frame case. Specifically, the rotating system includes an additional Coriolis contribution to $\mathbf{C}$.

2. **Spin softening and gyroscopic effects.** Rotational dynamics generate additional terms of spin softening and gyroscopic effects. For transient simulations, adding both terms makes the stiffness matrix nonsymmetric, doubling the storage and computational cost of the numerical factorization. For large problems, the user might consider using ISTFNS = 2, which adds only the spin softening terms. This feature maintains the symmetry of the stiffness matrix, hence avoiding the doubling of the cost of the numerical factorization. No matter the value of ISTFNS, all the effects of rotational dynamics are included in the force computation.

3. **Eigenfrequencies.** In many applications of rotational dynamics, the critical speed – the theoretical angular velocity that excites the natural frequency of a rotating object – is of particular concern. Therefore, the study of the mode frequency response as the rotating speed changes is very important. The Campbell diagram, which represents a system's eigenfrequencies as a function of rotating speed, is introduced for this purpose. To do this, you need to define a set of rotating speeds on Card 2, and the solver will perform a modal analysis for each of them. NOMEG should be defined as the number of rotating speeds specified in Card 2. A keyword file example in this application can be set as follows:

```
*KEYWORD
*CONTROL_TERMINATION...
*CONTROL_IMPLICIT_EIGENVALUE
         5

*CONTROL_IMPLICIT_GENERAL
         1      0.05
*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS
$#     SID     STYPE     OMEGA       VID    NOMEGA      IREF
         1         0       0.0         1         4         1
$#   OMEG1     OMEG2     OMEG3     OMEG4
```

```
      50.0     100.0     150.0     200.0
*DEFINE_VECTOR
$#     VID        XT        YT        ZT        XH        YH        ZH       CID
         1       0.0       0.0       0.0       1.0       0.0       0.0
*DATABASE_...
*PART...
*SECTION...
*MAT...
*ELEMENT...
*NODE...
*END
```

4. **Transient analysis.** Along with modal analysis, transient analysis can also be done using this keyword. A keyword file example can be set as follows:

```
*KEYWORD
*CONTROL_TERMINATION...
*CONTROL_IMPLICIT_GENERAL
         1      0.05
*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS
$#     SID     STYPE     OMEGA       VID    NOMEGA      IREF
         1         0       0.0         1         0         0
*DEFINE_VECTOR
$#     VID        XT        YT        ZT        XH        YH        ZH       CID
         1       0.0       0.0       0.0       1.0       0.0       0.0
*DATABASE_...
*PART...
*SECTION...
*MAT...
*ELEMENT...
*NODE...
*END
```

5. **Limitations.** The rotational dynamics effects are currently not available for solid and thick shell elements in a fixed coordinate system (IREF = 1), so the Campbell diagram in a fixed coordinate system only shows horizontal lines for these two element types. You may use IREF = 0 to see the rotational dynamics effects. The rotational effects are available for other element types, such as shells and beams, in both fixed (IREF = 1) and rotating (IREF = 0) coordinate systems.

   Brake squeal analysis requires a nonsymmetric linear solver and thus needs two keywords: *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS and *CONTROL_IMPLICIT_SOLVER with LCPACK = 3. We recommend setting IREF to 1 in this case, which is supported for all element types.

6. **Plotting Campbell diagrams in LS-PrePost.** To plot a Campbell diagram (frequency as a function of rotating speed) using LS-PrePost, select Post → ASCII → eigout → Load → Frequency → (select modes of interest) → Plot.

7. **Choosing the reference frame.** The following table indicates how to choose the reference frame:

Fixed Coordinate System	Rotating Coordinate System
The structure must be axisymmetric about the spin axis.	The structure need not be axisymmetric about the spin axis.

Fixed Coordinate System	Rotating Coordinate System
A rotating structure can be part of a stationary structure.	The rotating structure must be the only part of an analysis model.

8. **Comparison to similar keywords.** The following table compares the capabilities for keywords *LOAD_BODY, *LOAD_BODY_GENERALIZED, and *CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS:

Feature	*LOAD_BODY	*LOAD_BODY_GENERALIZED	*CONTROL_IMPLICIT_ROTATIONAL_DYNAMICS
Explicit	✓	✓	×
Implicit	✓	✓	✓
Centrifugal force	✓	✓	✓
Coriolis force	✓	✓	✓
Angular acceleration force	×	✓	×
Rotating reference frame	✓	✓	✓
Fixed reference frame	×	×	✓
Mode frequency depends on spin softening	✓	✓	✓
Mode frequency depends on gyroscopic effects	×	×	✓
Complex eigensolver	×	×	✓
Moving rotational axis	×	×	✓
Campbell diagram	×	×	✓

*CONTROL_IMPLICIT_SOLUTION_{OPTION}

Available options include:

<BLANK>

DYN

SPR

Purpose: These optional cards apply to implicit calculations. Use these cards to specify whether a linear or nonlinear solution is desired. Parameters are also available to control the implicit nonlinear and arc length solution methods (see also *CONTROL_IMPLICIT_GENERAL). The DYN option allows setting controls specifically for the dynamic relaxation phase. The SPR option allows setting controls specifically for the springback phase.

> **WARNING:** For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_SOLUTION. For example, if MAXREF = 5 for *CONTROL_IMPLICIT_SOLUTION and the value of MAXREF is left blank for *CONTROL_IMPLICIT_SOLUTION_DYN, then the value of MAXREF for *CONTROL_IMPLICIT_SOLUTION_DYN will also be 5.

Card Summary:

Card 1. This card is required.

NSOLVR	ILIMIT	MAXREF	DCTOL	ECTOL	RCTOL	LSTOL	ABSTOL

Card 2. Card 2 and beyond are optional.

DNORM	DIVERG	ISTIF	NLPRINT	NLNORM	D3ITCTL	CPCHK	

Card 2.1. Card 2.1 is read if DNORM is less than 0.

DMTOL	EMTOL	RMTOL		NTTOL	NRTOL	RTTOL	RRTOL

Card 3. The contents of this card are ignored unless an arc-length method is activated (6 ≤ NSOLVR ≤ 9, or NSOLVR = 12 and ARCMTH = 3).

ARCCTL	ARCDIR	ARCLEN	ARCMTH	ARCDMP	ARCPSI	ARCALF	ARCTIM

Card 4. This card is optional.

LSMTD	LSDIR	IRAD	SRAD	AWGT	SRED	KSSIZE	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NSOLVR	ILIMIT	MAXREF	DCTOL	ECTOL	RCTOL	LSTOL	ABSTOL
Type	I	I	I	F/I	F/I	F/I	F	F
Default	12	11	15	0.001	0.01	10^{10}	0.90	10^{-10}

VARIABLE	DESCRIPTION
NSOLVR	Solution method for implicit analysis: EQ.-12: Linear analysis with nonlinear contacts (see Remark 1) EQ.-1: Multistep linear (see Remark 1) EQ.1: Linear (see Remark 1) EQ.6: Nonlinear with BFGS updates + arc length (see Remark 2) EQ.7: Nonlinear with Broyden updates + arc length (see Remark 2) EQ.8: Nonlinear with DFP updates + arc length (see Remark 2) EQ.9: Nonlinear with Davidon updates + arc length (see Remark 2) EQ.12: Nonlinear with BFGS updates (default). This solver incorporates different line search and integration schemes as compared to obsolete NSOLVR = 2. This solver optionally allows the inclusion of an arc length method. Setting ARCMTH = 3 invokes an arc length method.

VARIABLE	DESCRIPTION
	EQ.13: Nonlinear with JFNK updates. See parameter KSSIZE on Card 4 and Remark 8.
ILIMIT	Iteration limit between automatic stiffness reformations (see Remark 3).
	LT.0: -ILIMIT references a curve that defines reformation limit as a function of time.
MAXREF	Stiffness reformation limit per time step.
	LT.0: If |MAXREF| matrix reformations occur, convergence for that time step is forced; see Remark 4.
DCTOL	Displacement relative convergence tolerance (see Remark 5).
	LT.0: -DCTOL references a curve that defines tolerance as a function of time.
ECTOL	Energy relative convergence tolerance (see Remark 5).
	LT.0: -ECTOL references a curve that defines tolerance as a function of time.
RCTOL	Residual (force) relative convergence tolerance (see Remark 5).
	LT.0: -RCTOL references a curve that defines tolerance as a function of time.
LSTOL	Line search convergence tolerance; see Remark 6.
	LT.0: -LSTOL is the line search tolerance, but this option activates an alternate strategy where line search acts only on the independent degrees of freedom. This is opposed to the default strategy, where prescribed motions on nodes and rigid bodies are also incorporated, sometimes leading to unnecessarily small time steps because of the requirement of fulfilling these boundary conditions.
ABSTOL	Absolute convergence tolerance.
	LT.0: Convergence detected when the residual norm is less than |ABSTOL|. Note: To drive convergence based on |ABSTOL|, set DCTOL and ECTOL to 10^{-20}.

Card 2	1	2	3	4	5	6	7	8
Variable	DNORM	DIVERG	ISTIF	NLPRINT	NLNORM	D3ITCTL	CPCHK	
Type	I	I	I	I	F/I	I	I	
Default	2	1	1	0	2	0	0	

Strict Tolerances Optional Card. Define this card if DNORM is less than 0.

Card 2.1	1	2	3	4	5	6	7	8
Variable	DMTOL	EMTOL	RMTOL		NTTOL	NRTOL	RTTOL	RRTOL
Type	F	F	F		F	F	F	F
Default	0.0	0.0	0.0		0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
DNORM	Displacement norm for convergence test (see Remark 5): EQ.1: Increment as a function of displacement over current step EQ.2: Increment as a function of total displacement (default) LT.0: |DNORM|; also activates reading of optional Card 2.1.
DIVERG	Divergence flag (force imbalance increases during equilibrium iterations): EQ.1: Reform stiffness if divergence detected (default) EQ.2: Ignore divergence (see Remark 3)
ISTIF	Initial stiffness formation flag (see Remark 3): EQ.1: Reform stiffness at start of each step (default) EQ.n: Reform stiffness at start of every n^{th} step LT.0: Curve ID = (-ISTIF) gives the number of steps to wait before reforming the stiffness as a function of time.
NLPRINT	Nonlinear solver print flag (see Remark 5):

VARIABLE	DESCRIPTION
	EQ.1: Print iteration information to screen, message, d3hsp files (default) EQ.2: Print extra norm information (NLNORM = 1) EQ.3: Same as 2, but also print information from line search NOTE: during execution, interactive commands can be used: interactive command — response <ctrl-c> **diag**nostic — toggle NLPRINT between 1 and 2 <ctrl-c> **info**rmation — set NLPRINT = 2 for one iteration
NLNORM	Nonlinear convergence norm type (see Remark 5; input is an integer if ≥ 0 and a float if < 0): LT.0: Same as 4, but rotational degrees of freedom are scaled with characteristic length |NLNORM| to account for units. EQ.1: Consider translational and rotational degrees of freedom EQ.2: Consider translational degrees of freedom only (default) EQ.4: Consider sum of translational and rotational degrees of freedom, that is, no separate treatment.
D3ITCTL	Control d3iter database. If nonzero, the search directions for the nonlinear implicit solution are written to the d3iter database. To reduce the size of the d3iter database, the database is reset every D3ITCTL time steps.
CPCHK	Contact penetration check flag. This flag does not apply to mortar contacts. EQ.0: No contact penetration check is performed (default). EQ.1: Check for contact penetration during the nonlinear solution procedure. If such penetration is found, modify the line search to prevent unnecessary penetration.
DMTOL	Maximum displacement convergence tolerance; convergence is detected when the *relative* maximum nodal or rigid body displacement is less than this value. See Remark 5.
EMTOL	Maximum energy convergence tolerance; convergence is detected when the *relative* maximum nodal or rigid body energy increment is less than this value. See Remark 5.
RMTOL	Maximum residual convergence tolerance; convergence is detected

VARIABLE	DESCRIPTION
	when the *relative* maximum nodal or rigid body residual is less than this value. See Remark 5.
NTTOL	Nodal translational convergence tolerance; convergence is detected when the *absolute* maximum nodal translational residual is less than this value. See Remark 5.
NRTOL	Nodal rotational convergence tolerance; convergence is detected when the *absolute* maximum nodal rotational residual is less than this value. See Remark 5.
RTTOL	Rigid body translational convergence tolerance; convergence is detected when the *absolute* maximum rigid body translational residual is less than this value. See Remark 5.
RRTOL	Rigid body rotational convergence tolerance; convergence is detected when the *absolute* maximum rigid body rotational residual is less than this value. See Remark 5.

Arc Length Optional Card. The contents of this card are ignored unless an arc-length method is activated (6 ≤ NSOLVR ≤ 9, or NSOLVR = 12 and ARCMTH = 3).

Card 3	1	2	3	4	5	6	7	8
Variable	ARCCTL	ARCDIR	ARCLEN	ARCMTH	ARCDMP	ARCPSI	ARCALF	ARCTIM
Type	I	I	F	I	I	F	F	F
Default	0	↓	0.	1	2	0.	0.	0.

VARIABLE	DESCRIPTION
ARCCTL	Arc length controlling node ID (see Remark 7). EQ.0: Generalized arc length method
ARCDIR	Arc length controlling node direction (ignored if ARCCTL = 0 above; see Remark 7): EQ.1: Global x-translation EQ.2: Global y-translation EQ.3: Global z-translation

VARIABLE	DESCRIPTION
ARCLEN	Relative arc length size. See Remark 7. LE.0.0: Use automatic size. GT.0.0: Use ARCLEN × (automatic step size).
ARCMTH	Arc length method (see Remark 7): EQ.1: Crisfield (default) EQ.2: Ramm EQ.3: Modified Crisfield (used with NSOLVR = 12 only) **Note**: When using the arc-length method, termination should be based on some *TERMINATION_*OPTION* criteria.
ARCDMP	Arc length damping option (see Remark 7): EQ.2: Off (default) EQ.1: On. Oscillations in the static solution are suppressed.
ARCPSI	Relative influence of load/time parameter in spherical arclength constraint; default value is 0 which corresponds to a cylindrical arclength constraint. Applies to ARCMTH = 3.
ARCALF	Relative influence of predictor step direction for positioning of the arc center; default is 0 which means that the center is at the origin. Applies to ARCMTH = 3.
ARCTIM	Optional time when arclength method is initiated. Applies to ARCMTH = 3.

Line Search Parameter Optional Card.

Card 4	1	2	3	4	5	6	7	8
Variable	LSMTD	LSDIR	IRAD	SRAD	AWGT	SRED	KSSIZE	
Type	I	I	F	F	F	F	I	
Default	4	2	0.0	0.0	0.0	0.0	10	

VARIABLE	DESCRIPTION
LSMTD	Line search convergence method (see Remark 6): EQ.1: Energy method using only translational variables EQ.2: Residual method EQ.3: Energy method using both translational and rotational variables EQ.4: Energy method using sum of translational and rotational degrees of freedom (default), that is, no separate treatment EQ.5: Same as 4, but account for residual norm growth to be extra conservative in step length EQ.6: Same as 5, but minimizes the residual norm whenever convenient
LSDIR	Line search direction method (see Remark 6): EQ.1: Search on all variables (traditional approach used in versions prior to 971) EQ.2: Search only on the independent (unconstrained) variables EQ.3: Use adaptive line search (see AWGT, SRED) EQ.4: Use curved line search (see IRAD, SRAD)
IRAD	Normalized curvature factor for curved line search, where 0 indicates a straight line search and 1 indicates full curved line search. See Remark 6.
SRAD	Radius of influence for determining curve in curved line search. For each independent node, all nodes within this radius are used for determining the curve. If 0, then all nodes connected to the same element as the independent node are used. See Remark 6.
AWGT	Adaptive line search weight factor between 0 and 1. A high value tends to restrict the motion of oscillating nodes during the implicit process. See Remark 6.
SRED	Initial step reduction between 0 and 1 for adaptive line search; use large number for conservative start in implicit procedure. See Remark 6.
KSSIZE	Size of Krylov space in JFNK iterative method. The default is 10. Decreasing the default value tends to make the problem more

VARIABLE	DESCRIPTION
	robust but potentially more expensive to solve. Increasing it could save simulation time but may also result in a less robust scheme. See Remark 8.

Remarks:

1. **Linear solution methods.** NSOLVR = 1 invokes a linear analysis for a user specified number of time steps. The results are cumulative for those time steps. NSOLVR = -1 resets the geometry back to the initial state for each time step. With this method, a linear analysis for several different loading cases with only one stiffness matrix formation and factorization can be done. If a linear analysis is selected, equilibrium checking and iterations are not performed. NSOLVR = -12 causes a hybrid approach, where all features are linearized except for contacts. This option is suitable for situations with small displacement for which the state of the contact is important.

2. **Unmaintained nonlinear solution methods.** The nonlinear solution methods 6-9 are not being maintained. NSOLVR = 12 with ARCMTH = 3 is the suggested method for arc length solvers.

3. **Stiffness matrix reformation.** In the default BFGS method, the global stiffness matrix is only reformed every ILIMIT iterations. Otherwise, an inexpensive stiffness update is applied. By setting ILIMIT = 1, a stiffness reformation is performed every iteration, which is equivalent to the Full Newton method (with line search). A higher value of ILIMIT (20-25) can reduce the number of stiffness matrix reformations and factorizations, which may lead to a significant reduction in cost. However, increasing the value of ILIMIT may increase the number of iterations. Note that the storage requirements for implicit include storing two vectors per iteration. Therefore, large values of ILIMIT will cause a substantial increase in storage requirements. By using a negative ILIMIT, it is possible to switch between Full Newton and various degrees of BFGS methods, if, for instance, nonlinearity varies significantly during the simulation.

   By default, a new stiffness matrix is formed whenever divergence (growing out-of-balance force) is detected. The DIVERG flag can be used to suppress this stiffness reformation. Also, by default, a new stiffness matrix is formed at the start of every time step. Suppressing this stiffness reformation by changing the value of ISTIF can decrease the cost of simulations which have many tiny steps that are mostly linear, such as transient dynamics.

4. **MAXREF.** The nonlinear equilibrium search will continue until the stiffness matrix has been reformed |MAXREF| times, with ILIMIT iterations between each reformation. If equilibrium has not been found and MAXREF > 0, control will

be passed to the automatic time step controller if it is activated. If the automatic time step controller is not active, error termination will result. When the auto time step controller is active, it is often efficient to choose MAXREF = 5 and try another stepsize quickly, rather than wasting too many iterations on a difficult step.

When MAXREF < 0 and |MAXREF| matrix reformations have occurred, convergence for the current time step is declared, with a warning, and the simulation moves to the next time step. This option should be used with caution as the results for that particular time step may be wrong.

5. **Convergence criteria.** For convergence criteria DCTOL, ECTOL, and RCTOL, the conditions are satisfied when the displacement norm ratio, energy norm ratio, and residual norm ratio are reduced below DCTOL, ECTOL, and RCTOL, respectively. Smaller tolerances lead to a stricter determination of equilibrium, but also result in more iterations and higher costs. By default, residual norm ratio (RCTOL) criterion is effectively disabled (RCTOL = 10^{10}).

When computing the displacement ratio, the norm of the incremental displacement vector is divided by the norm of "total" displacement. This "total" displacement may be either the total over the current step, or the total over the entire simulation. The latter tends to be more lax and can be poor at the end of simulations where large motions develop. For these problems, an effective combination is DNORM = 1 and DCTOL = 0.01 or larger.

For all nonzero values of the strict tolerance fields in optional Card 2.1, the associated criterion must be satisfied *in addition* to the ones defined through DCTOL, ECTOL, and RCTOL. These criteria are based on the maximum norm, which is regarded as stronger than the Euclidian norm used for the other parameters; using these criteria will likely result in higher accuracy at the price of more iterations. For NLPRINT greater than or equal to 2, a table is listed in the messag and d3hsp files for each iteration, providing the values associated with all the criteria activated. The first three (DMTOL, EMTOL, and RMTOL) of these extra fields are unitless and honor the meaning of both DNORM and NLNORM. The last four (NTTOL, NRTOL, RTTOL, and RRTOL) are to be given in units of force, torque, force, and torque, respectively, and the values used should account for the representative loads in the problem as well as the discretization size.

By default, only translational degrees of freedom are used when evaluating convergence norms. NLNORM can be set to include rotational degrees of freedom or to make additional data available for diagnosing convergence problems. This additional data includes the worst offending node and degree of freedom contributing to each norm.

By setting NLNORM to 1, rotational degrees of freedom can be considered independently from the translational degrees of freedom, meaning that two

separate scalar products are used for evaluating norms, $\langle \mathbf{u}, \mathbf{v} \rangle_t = \mathbf{u}^T \mathbf{J}_t \mathbf{v}$ and $\langle \mathbf{u}, \mathbf{v} \rangle_r = \mathbf{u}^T \mathbf{J}_r \mathbf{v}$. Here $\mathbf{J}_t$ and $\mathbf{J}_r$ are diagonal matrices with ones on the diagonal to extract the translational and rotational degrees of freedom, respectively. In this case, the convergence criteria must be satisfied for both translational and rotational degrees of freedom simultaneously.

Alternatively they can be included by defining the single scalar product $\langle \mathbf{u}, \mathbf{v} \rangle = \langle \mathbf{u}, \mathbf{v} \rangle_t + \lambda_u \lambda_v \langle \mathbf{u}, \mathbf{v} \rangle_r$, where λ_u and λ_v are scale factors to account for different units of the rotational degrees of freedom. For NLNORM = 4 these scale factors are equal to a characteristic element size that is calculated internally. But for NLNORM < 0 λ_u is equal to |NLNORM| if $\mathbf{u}$ is a displacement vector and $|\text{NLNORM}|^{-1}$ if it is a force vector; the same goes for the pair λ_v and $\mathbf{v}$. So |NLNORM| is a characteristic length that appropriately weighs translational and rotational degrees of freedom together. For NLNORM = 4 the internally calculated size is echoed to the screen, d3hsp and messag files for observation and based on this NLNORM < 0 can be used if another scale factor is more suitable.

6. **Line search.** A line search is performed on stiffening systems to guard against divergence of Newton-based nonlinear solvers. With the Full Newton method, it is sometimes helpful to define a large value of LSTOL (LSTOL = 9999.0) to effectively disable line search.

The default method for determining convergence of the nonlinear line search is to find the minimum of the energy. LMSTD allows choosing the energy on only the translational variables, energy of both the translational and rotational variables, or for minimizing the residual (forces). The effect of using a residual based line search is not always positive; sometimes it is too restrictive and stops convergence. But it is a more conservative approach than using the energy based method since it explicitly controls the norm of the residual. It should not be seen as a better strategy than the energy method but as an alternative to try in cases when the default method seems to be working poorly. Line search methods 5 and 6 are conservative line search methods to be used for highly nonlinear problems; these should not be used as default but as final resorts to potentially resolve convergence issues. The rule of thumb is that the LSMTD = 5 is slow but robust and LSMTD = 6 is even slower but more robust.

In Version 971 of LS-DYNA, new line search options were added. The traditional approach (LSDIR = 1) computes the line search direction using all variables. The new (default) approach of LSIDR = 2 computes the line search direction only on the unconstrained variables. It has proven to be both robust and more efficient. We have also included two new approaches to try for problems where the default and traditional approach fail, and the user is using Full Newton (ILIMIT = 1). These methods are described below:

a) *Curve Line Search.* The parameters IRAD and SRAD are for the curved line search (LSDIR = 4). The first parameter is a switch (0 or 1) to invoke this line search; an intermediate value is interpreted as weighted combination of a straight and curved line search (the curvature radius is decreased with increasing IRAD). A value of unit is recommended in situations with rather smooth responses, such as springback. Also, IRAD = 1 seems to work best with full Newton iterations. The SRAD parameter should be equal to 0 for most cases; this means that the search curve for a node is determined from the search direction of nodes connected to the same elements as that node. SRAD > 0 is interpreted as a radius of influence, meaning that the search curve for a node is determined from the search direction of nodes within a distance SRAD of this node. This option was introduced as an experiment to see if this had a smoothing and stabilizing effect. A value of 0.0 is currently recommended.

b) *Adaptive Line Search.* The parameters AWGT and SRED are for the adaptive line search. The intention is to improve robustness for problems that have tendencies to oscillate or diverge, indicated by the dnorm and enorm parameter outputs in the iterations (stdout). A value of 0.5 is recommended for AWGT as a starting point. With a nonzero value the motions of individual nodes are tracked. For nodes that are oscillating (going back and forth in space), the maximum step size for the next iteration is reduced in proportion to the parameter AWGT, and for nodes that are not oscillating but going nicely along a straight path, the maximum step size for the next iteration is increased in proportion to 1-AWGT.

In test problems, the introduction of the adaptive line search has stabilized the implicit procedure in the sense that the dnorm and enorm values are more monotonically decreasing until convergence with virtually no oscillations. If a problem is still oscillating or diverging, the user should try to increase the AWGT parameter since this is a more restrictive approach but probably gives a slower convergence rate. An option for nasty problems is also to use SRED > 0 which is the initial step reduction factor (less than 1). This means that the initial step size is reduced by this value but the maximum step size will increase by an amount that is determined by the success in the iterative procedure; eventually it will reach unity. It can never decrease. Also here, it is intended to be used with full Newton method.

7. **Arc length methods.** In the neighborhood of limit points the Newton based iteration schemes often fail. The arc length method of Riks and Wempner (combined here with the BFGS method) adds a constraint equation to limit the load step to a constant "arc length" in load-displacement space. This method is frequently used to solve snap through buckling problems. *When applying the arc-length method, the curves that define the loading should contain only two points, and the first point should be at the origin* (0,0). LS-DYNA will extrapolate, if necessary,

to determine the load. In this way, time and load magnitude are related by a constant. It is possible that time can become negative in case of load reversal. The arc length method cannot be used in a dynamic analysis.

In many cases the arc length method has difficulty tracking the load-displacement curve through critical regions. Using 0 < ARCLEN < 1 will reduce the step size to assist tracking the load-displacement curve with more accuracy. Use of ARCLEN < 1 will cause more steps to be taken. Suggested values are 1.0 (the default), 0.5, 0.25, and 0.10.

The arc length method can be controlled based on the displacement of a single node in the model using ARCCTL. For example, in dome reversal problems the node at the center of the dome can be used. By default, the generalized arc length method is used, where the norm of the global displacement vector controls the solution. This includes all nodes.

Some static problems exhibit oscillatory response near instability points. ARCDMP numerically suppresses these oscillations and may improve the convergence behavior of the post-buckling solution.

8. **JFNK.** The Jacobian-Free Newton-Krylov (JFNK) method is a multidisciplinary variant of the linear GMRES method that may potentially bridge the gap between full Newton and BFGS. In each iteration, the method constructs a Krylov subspace of dimension KSSIZE based on recursively computing force vectors from ϵ-perturbations of the current geometry. The method then obtains a search direction within this subspace by means of a minimization principle. Utilizing the factorization of the stiffness matrix from the last stiffness reformation as a preconditioner accelerates the method. Since the method exploits the quasi-linear characteristics of the force in the vicinity of a geometry iterate, this method potentially increases the robustness when compared to BFGS. The method seems particularly promising for very nonlinear problems.

9. **Deprecated keyword name.** *CONTROL_IMPLICIT_NONLINEAR is a deprecated name for this keyword.

***CONTROL_IMPLICIT_SOLVER**_*{OPTION}*

Available options include:

<BLANK>

DYN

SPR

Purpose: These optional cards apply to implicit calculations. The linear equation solver performs the CPU-intensive stiffness matrix solution (see also *CONTROL_IMPLICIT_GENERAL). The DYN option allows setting controls specifically for the dynamic relaxation phase. The SPR option enables setting controls specifically for the springback phase.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_SOLVER. For example, if NEGEV = 1 for *CONTROL_IMPLICIT_SOLVER and the value of NEGEV is left blank for *CONTROL_IMPLICIT_SOLVER_DYN, then the value of NEGEV for *CONTROL_IMPLICIT_SOLVER_DYN will also be 1.

Card 1	1	2	3	4	5	6	7	8
Variable	LSOLVR	LPRINT	NEGEV	ORDER	DRCM	DRCPRM	AUTOSPC	AUTOTOL
Type	I	I	I	I	I	F	I	F
Default	7	0	2	0	4	↓	1	↓

Card 2 is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	LCPACK	MTXDMP	IPARM1	RPARM1	RPARM2		RPARM4	RPARM5
Type	I	I	I	F	F		F	F
Default	2	0	↓	10^{-12}	10^{-8}		240.0	0.0

Card 3 is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	EMXDMP	RDCMEM	ABSMEM	ISINGLE	IBLROPT		ISPD	MEMLVL
Type	I	F	F	I	I		I	F
Default	0	0.85	optional	0	0		0	0

VARIABLE **DESCRIPTION**

LSOLVR Linear equation solver method (see Remark 1).

EQ.2: Parallel multi-frontal sparse solver (deprecated)

EQ.7: Parallel multi-frontal sparse solver (default)
Iterative if ISINGLE > 0 or IBLROPT > 0.

EQ.22: Iterative, CG with diagonal preconditioner when LCPACK = 2 (default) or GMRES with diagonal preconditioner when LCPACK = 3 (nonsymmetric matrix assembly)

EQ.23: Iterative, CG with local Symmetric Gauss-Seidel (SGS) preconditioner

EQ.24: Iterative, CG with local Symmetric Successive Over-Relaxation (SSOR) preconditioner

EQ.25: Iterative, CG with local 0-fill incomplete factorization (ILDLT0) preconditioner

EQ.26: Iterative, CG with local 0-fill incomplete factorization (ILDLT0) preconditioner that requires extra storage

VARIABLE	DESCRIPTION
	EQ.30: Parallel direct/hybrid solver MUMPS (see Remark 2) EQ.90: User-Supplied Linear Equation Solver (see Remark 3) SMP only: EQ.6: BCSLIB-EXT, direct, sparse, double precision
LPRINT	Linear solver print flag controls screen and message file output (see Remarks 4 and 6). EQ.0: No printing EQ.1: Output summary statistics on memory, CPU requirements EQ.2: More statistics EQ.3: Even more statistics and debug checking During execution, use the interactive command "<ctrl-c> lprint" to toggle this print flag between 0 and 1.
NEGEV	Negative eigenvalue flag. Selects procedure when negative eigenvalues are detected during stiffness matrix inversion (see Remark 5). EQ.1: Stop or retry step if auto step control is active EQ.2: Print warning message, try to continue (default)
ORDER	Ordering option (see Remark 6): EQ.0: Method set automatically by LS-DYNA EQ.1: MMD, Multiple Minimum Degree EQ.2: METIS EQ.3: ParMETIS EQ.4: LS-GPart
DRCM	Drilling rotation constraint method for shells (see Remark 7): EQ.1: Add drilling stiffness (old Version 970 method) EQ.2: Same as 4 below EQ.3: Add no drilling stiffness EQ.4: Add drilling stiffness (improved method) (default)
DRCPRM	Drilling rotation constraint parameter for shells. This parameter scales the drilling stiffness. For the old method (DRCM = 1), the default value of DRCPRM is 1.0 for linear analysis, 100.0 for nonlinear implicit analysis, and either 10^{-12} or 10^{-8} for eigenvalue

VARIABLE	DESCRIPTION
	analysis, depending on the shell element type. For eigenvalue analysis, the input value for DRCPRM is ignored. For the improved method (default, DRCM = 4), the default value of DRCPRM is as described above for the old method, except the default value of DRCPRM is 1.0 for nonlinear implicit analysis.
AUTOSPC	Automatic Constraint Scan flag EQ.1: Scan the assembled stiffness matrix looking for unconstrained, unattached degrees of freedom. Generate additional constraints as necessary to avoid negative eigenvalues. EQ.2: Do not add constraints.
AUTOTOL	AUTOSPC tolerance. The test for singularity is the ratio of the smallest singular value and the largest singular value. If this ratio is less than AUTOTOL, then the triple of columns is declared singular, and a constraint is generated. Default values in single and double precision are 10^{-4} and 10^{-8}, respectively.
LCPACK	Matrix assembly package: EQ.2: Default. EQ.3: Same as 2, but incorporates a nonsymmetric linear solver; see Remark 8.
MTXDMP	Matrix and right-hand-side dumping. LS-DYNA has the option of dumping the globally assembled stiffness matrix and right-hand-side vector files in Harwell-Boeing sparse matrix format. Such output may help with comparing to other linear equation solution packages. EQ.0: No dumping GT.0: Dump all matrices and right-hand-side vectors every MTXDMP time steps. The first dump occurs at the first time step, then at time step MTXDUMP+1, 2*MTXDUMP+1, and so on. Output is written as ASCII text, and the involved filenames are of the following form: K_*xxxx*_*yyy*.mtx.rb This file contains the stiffness matrix at step *xxxx*, iteration *yyy*. M_*xxxx*_*yyy*.mtx.rb This file contains the mass matrix at step *xxxx*,

VARIABLE	DESCRIPTION
	iteration *yyy*. Only for eigenvalue analysis. W_*xxxx*_*yyy*.mtx.rb This file contains the damping matrix at step *xxxx*, iteration *yyy*. Only for simulations with damping. K_*xxxx*_*yyy*_*zzz*.rhs.rb This file contains the right-hand side at step *xxxx*, iteration *yyy*, where *yyy* is the iteration at which a stiffness matrix is formed, and *zzz* is the cumulative iteration number for the step. The values of *yyy* and *zzz* don't always coincide because the stiffness matrix is not necessarily reformed every iteration. Node_Data_*xxxx*_*yyy* This file maps the stiffness matrix to nodes and provides nodal coordinates.
	LT.0: Like positive values of MTXDMP, but dumped data is binary.
	EQ.|9999|: Simulation is terminated after dumping matrices and right-hand side before factorization.
IPARM1	For iterative solvers, maximum number of iterations. The default is 1000 for LSOLVR = 7. The default is 10000 for 22 ≤ LSOLVR ≤ 26.
RPARM1	For iterative solvers, absolute tolerance for convergence. The default is 10^{-12}.
RPARM2	For iterative solvers, relative tolerance for convergence. The default is 10^{-8}.
RPARM4	For LSOLVR = 7 with LCPACK = 2 (symmetric) and IBLROPT > 0, nominal block size. The default is 240.
RPARM5	For LSOLVR = 7 with LCPACK = 2 (symmetric) and IBLROPT > 0, compression tolerance for the low-rank approximation. For LSOLVR = 30, RPARM5 is the compression tolerance used to compute a low-rank factorization with the MUMPS solver (see Remark 2).
EMXDMP	Flag for dumping elemental stiffness and mass matrices: EQ.0: No dumping GT.0: Dump all elemental matrices every EMXDMP time steps. Output is written as ASCII text, and the involved filenames are of the following form:

VARIABLE	DESCRIPTION
	ElmStfMtx *xxxx yyy* This file contains the elemental stiffness matrix at step *xxxx*, iteration *yyy*. ElmMssMtx *xxxx yyy* This file contains the elemental mass matrix at step *xxxx*, iteration *yyy*. LT.0: Like positive values of MTXDMP, but dumped data is binary. EQ.|9999|: Simulation is terminated after dumping matrices and right-hand side before factorization.
RDCMEM	Starting with R11, the memory for linear algebra has been moved from static to dynamic memory allocation. For implicit applications, we found that some operating systems are not "robust" when queried about how much dynamic memory is free. This factor caps the amount of dynamic memory requested for linear algebra applications to RDCMEM times the amount the operating system declares available. 0.85 seems to work well for most systems. If you are using a workstation and starting up other applications while running LS-DYNA, you may need to use a number like 0.50.
ABSMEM	Absolute upper bound for the dynamic memory allocated for factorization. The allocated memory will be bounded above by the min(RDCMEM × NWORDS, ABSMEM) where NWORDS is the number of available words determined by the operating system. If the predicted amount of required memory is less than this value, then less memory than this bound may be allocated. Note that the memory can be specified in this input in the same way as memory is specified on the command line. For instance, an allocation of 3.5 Gwords can be input as 3.5G or 3.5e+9 for ABSMEM.
ISINGLE	Only active if LSOLVR = 7 and LCPACK = 2. This field controls tasks performed in single precision (see Remark 10): EQ.0: No single precision used. EQ.1: The factors of the matrix, which are computed in double precision, are stored in single precision, reducing the storage required to solve the linear system. EQ.2: Same as 1, but additionally some of the factorization

VARIABLE	DESCRIPTION
	operations are performed in single precision, reducing the runtime.
IBLROPT	Flag providing options for the block low-rank (BLR) computations. It is only active if LSOLVR = 7 and LCPACK = 2. As the value of IBLROPT increases, the treatment is increasingly aggressive. However, increasingly aggressive compression can introduce additional error into the factors of the linear system, which can slow down the convergence of the subsequent numerical solve or even prevent convergence. See Remark 10. EQ.0: BLR is not used. EQ.1: Factor, Solve, Update, and Compress algorithm (FSUC). A full-rank factorization is performed, and off-diagonal blocks of the factors are then subject to compression with BLR. This method should reduce the storage required to factor and solve the linear system. However, the additional BLR computations increase the runtime. EQ.2: Factor, Solve, Compress, and Update algorithm (FSCU). FSUC is applied with the fully assembled equations of the frontal matrix. Where BLR compression is applied, low-rank updates are performed to compute the Schur complement of the frontal matrix. This method should reduce both storage and runtime. EQ.3: Update, Factor, Solve, and Compress algorithm (UFSC). Low-rank approximations are used to update blocks within the fully assembled equations in a frontal matrix. Where BLR compression is applied, low-rank computations are also performed when computing the Schur complement of the frontal matrix. This option should reduce storage and further reduce runtime. EQ.4: UFSC is performed. In addition, BLR is applied to the contribution blocks of low-rank frontal matrices before they are placed on the real stack. This step further reduces the working storage needed for the low-rank factorization. The additional BLR computations increase the runtime relative to UFSC.
ISPD	Symmetric positive definite flag. Only active when LSOLVR = 7 and LCPACK = 2. EQ.0: The solver only assumes that the linear system is symmetric positive definite if the diagonals of the factorization are

VARIABLE	DESCRIPTION
	all positive. EQ.1: The solver assumes that the linear system is symmetric positive definite and ignores spurious negative diagonal entries.
MEMLVL	Control the load-balancing behavior. Only active when LSOLVR = 7, LCPACK = 2, and ISINGLE > 0. The solver's default behavior is to load balance to reduce runtime. That behavior can cause a subset of processors to have a disproportionate share of the factor entries and drive such processors out-of-core, even when enough storage should be available. MEMLVL enables instructing the solver to consider storage as well. EQ.0.0: Load balancing is entirely based on minimizing in-core runtime. GT.1.0: Perform load balancing to reduce storage disparity among the processors. GT.1.0: The first load balance is performed to reduce runtime. Then, it is adjusted to reduce any outlier cores, such that they have no more than MEMLVL times an equal share of the factor entries. In this case, MEMLVL is an aspirational target that the solver cannot always achieve.

Remarks:

1. **Direct versus iterative solvers.** The linear solver computes the inverse of the global stiffness matrix, which is costly in both memory and CPU time. Direct solvers apply Gaussian elimination, while iterative solvers successively improve "guesses" at the correct solution. Iterative solvers require far less memory than direct solvers but may suffer from convergence problems. Generally, iterative solvers are poor for automotive applications but can be superior for large brick element models, such as solid models in civil engineering.

2. **MUMPS solver.** Starting with R11.1, we have coupled the external sparse linear solver MUMPS with LS-DYNA. MUMPS can be used as a direct solver (similar to LSOLVR = 7) with RPARM5 = 0 or as a hybrid direct-iterative solver when RPARM5 > 0. When RPARM5 > 0, an approximate factorization is computed using RPARM5 as a compression threshold for Block Low-Rank approximations. The approximate factorization is then used as a preconditioner inside a Conjugate Gradient solver; IPARM1, RPARM1, and RPARM2 are used to drive the Conjugate Gradient solver as described above.

Using MUMPS as the linear solver in LS-DYNA should only be used in conjunction with LS-DYNA Implicit Staff.

MUMPS is the property of CERFACS, CNRS, ENS Lyon, INP Toulouse, Inria, Mumps Technologies, and the University of Bordeaux. See [1] and [2] for references about this solver.

3. **User-supplied solver.** Starting with R11, we have added a capability for a user-supplied solver. Users can follow the template in the file UserLE.F90 to supply their own linear equation solver. Some examples are available in the usermat delivery and from Technical Support.

4. **Solver print flag.** Select printing of the timing and storage information (LPRINT = 1) if you are comparing linear equation solvers' performance or running out of memory for large models. Minimum memory requirements for in-core and out-of-core solutions are printed. This flag can also be toggled using sense switch "<ctrl-c> lprint." *For best performance, increase available memory using "memory= " on the command line until an IN-CORE solution is indicated.*

   When using solver option 6, LPRINT = 2 and 3 will increase the printed output of statistics and performance information.

5. **Negative eigenvalues.** Negative eigenvalues result from underconstrained models (rigid body modes), severely deformed elements, or non-physical material properties. When negative eigenvalues are detected, the NEGEV flag allows control to be passed directly to the automatic time step controller. Otherwise, significant numerical roundoff error will likely occur during factorization, and equilibrium iterations may fail (see *CONTROL_IMPLICIT_AUTO).

6. **Ordering algorithms and cost.** The system of linear equations is reordered to optimize the sparsity of the factorization when using direct methods. Reordering is a challenging problem, and no best algorithm exists for all problems. LS-DYNA offers four different methods:

   a) *MMD (Multiple Minimum Degree)*. This reordering method is inexpensive but should only be used for small problems.

   b) *METIS.* METIS is from the University of Minnesota. It is the default method for most problems. It is very effective for large problems. METIS, however, is a serial algorithm. Thus, for a large number of MPP processes, you might want to use one of the two parallel algorithms mentioned next.

   c) *ParMETIS.* ParMETIS is also from the University of Minnesota. It is the MPP version of METIS. For a large number of processes, it should run much faster than METIS, but there might be some variation in the factorization cost (storage and number of operations). It is available as of R14.

d) *LS-GPart.* LS-GPart is an MPP algorithm developed by Ansys and available as of R11. For a large number of processes, it should run faster than METIS. Quality (in terms of factorization cost) compared to METIS or ParMETIS will depend on the problem.

The symbolic factorization phase of the linear solver (LPRINT ≥ 1) includes the reordering cost. For large models, if this cost exceeds 20% of the numeric factorization cost, it may be more efficient to select the MMD method.

Note that the values of LPRINT and ORDER also affect the eigensolution software. That is, LPRINT and ORDER from this keyword card apply to the eigensolution.

Please refer to *CONTROL_IMPLICIT_ORDERING for additional parameters for ordering methods.

7. **Drilling degree of freedom.** To avoid a singular stiffness matrix for the implicit analysis of flat shell topologies, some constraint on the drilling degree of freedom is needed. The default method of applying this constraint, DRCM = 4, adds the consistent force vector for consistency and improved convergence compared to the old method, DRCM = 1.

   In explicit analysis, an unconstrained drilling degree of freedom is usually not a concern since a stiffness matrix is not used. However, special situations may arise where the user wishes to include additional resisting rotational force in the drilling degree of freedom for improved robustness and/or accuracy. To activate the consistent drilling constraint in explicit analysis, use the input variables DRCPSID and DRCPRM for *CONTROL_SHELL.

8. **Nonsymmetric matrix solver.** Certain features may break the symmetry of the stiffness matrix. Unless LCPACK is set to 3, the default linear solver suppresses or symmetrizes these contributions. However, LCPACK set to 3 activates a more general linear solver, lifting the symmetry requirement. The solver for nonsymmetric matrices is more computationally expensive than for symmetric matrices. LSOLVR = 7 (Parallel multi-frontal sparse solver), 6 (BCSLIB-EXT, direct, sparse, double precision), 22 (CG or GMRES), and 30 (MUMPS) can be used in combination with LCPACK = 3. Other values will default to a direct solver.

   For eigenvalue analysis, LCPACK = 3 is only implemented in SMP. LCPACK = 3 is implemented for both SMP and MPP for static and dynamic implicit analyses.

   Keywords for which the nonsymmetric contribution is included when LCPACK = 3 are listed below. If none of these features are included in the model, setting LCPACK = 3 offers no benefit but will incur additional computational cost.

***CONTACT_..._MORTAR:**

The mortar contact accounts for frictional nonsymmetry in the resulting tangent stiffness matrix; the effects on convergence characteristics have not yet been shown to be significant.

***LOAD_SEGMENT_NONUNIFORM:**

The nonsymmetric contribution may be significant for the follower load option, LCID < 0.

***LOAD_SEGMENT_SET_NONUNIFORM:**

The nonsymmetric contribution may be significant for the follower load option, LCID < 0.

***MAT_FABRIC_MAP:**

This stress map fabric model accounts for nonsymmetry in the material tangent modulus, representing the non-linear Poisson effect due to the complex interaction of yarns.

***SECTION_SHELL, *SECTION_SOLID:**

Solid element formulation 2, -1, -2 and user-defined resultant elements (ELFORM = 101, 102, 103, 104, 105 with NIP = 0) support the assembly and solution of nonsymmetric element matrices.

***SECTION_BEAM:**

Belytschko-Schwer beam (ELFORM = 2) nonsymmetric geometric stiffness contribution is supported.

9. **Constraints.** Explicit treats all the constraints in sequence, which introduces the limitation of applying several constraints on the same DOF. Implicit (with LCPACK), on the other hand, treats all constraints simultaneously, which precludes (circumvents) the limitation on applying multiple constraints on the same nodes or DOFs, as long as they are not in conflict.

10. **Reduced precision.** Using ISINGLE or BLR can save time and storage when using the default, multi-frontal sparse solver (LSOLVR = 7). The trade-off is that they reduce the precision of the factors of the matrix, turning them into high-quality preconditioners for a subsequent iterative linear solver. If the symmetric matrix is positive definite, the conjugate gradient method is used. If the symmetric matrix is indefinite, the iterative refinement method is used. There is no guarantee that the iterative solver will converge to the requested tolerance. If the iterative solver does not converge, LS-DYNA implicit may not either.

11. **Deprecated keyword name.** *CONTROL_IMPLICIT_LINEAR is a deprecated name for this keyword.

References:

[1] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent, A fully asynchronous multifrontal solver using distributed dynamic scheduling, SIAM Journal on Matrix Analysis and Applications, Vol 23, No 1, pp 15-41 (2001).

[2] P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary, Performance and scalability of the block low-rank multifrontal factorization on multicore architectures, ACM Transactions on Mathematical Software, Vol 45, Issue 1, pp 2:1-2:26 (2019).

*CONTROL_IMPLICIT_SSD_DIRECT

Purpose: Request a direct complex solution to steady-state vibration. Currently, this keyword has limited loading and damping capabilities. It is only available for double-precision executables. Refer to Appendix W for a list of keywords this feature supports.

Card 1	1	2	3	4	5	6	7	8
Variable	ISSFLG	FMIN	FMAX	NFREQ	LOSS	FSPACE	FRACTN	
Type	I	F	F	I	F	I	I	
Default	0	none	FMIN	1	0.0	0	1	

VARIABLE	DESCRIPTION
ISSFLG	Complex steady state vibration flag: EQ.0: Off EQ.1: On
FMIN	Minimum frequency in the solution
FMAX	Maximum frequency in the solution
NFREQ	Number of frequencies in the solution
LOSS	Structural loss factor
FSPACE	Solution frequency assignment strategy: EQ.0: The frequency is interpolated linearly between FMIN and FMAX. This is the default strategy. EQ.1: The frequency is interpolated on a log scale between FMIN and FMAX, so they are biased to lower frequencies. EQ.2: The frequency is interpolated on a fractional octave scale starting with FMIN. Integer FRACTN is the octave fraction. The formula for the active frequency in Hertz is $\text{FACTIVE} = \text{FMIN}\left(2.0^{1/\text{FRACTN}}\right)^{(\text{IFREQ}-1)}$. IFREQ is the i^{th} frequency in the solution. FMAX is ignored. LT.0: |FSPACE| is a load curve ID for assigning active frequencies. The abscissa is frequencies in the solution and the

VARIABLE	DESCRIPTION
	ordinate is the active frequency in Hertz. FMIN and FMAX are ignored.
FRACTN	Octave fraction. For example, FRACTN = 3 means 1/3 octave spacing.

Remarks:

1. **Plot files.** The complex solution is written to file d3ssd at every solution step. If *DATABASE_BINARY_D3PLOT output is also requested, then the real part of the solution is written to the d3plot file.

2. **Structural loading.** Currently, the structural loads are identified with *LOAD_... keywords and limited to zero phase angle.

3. **Acoustics.** Acoustic elements (ELFORMs 8 and 14 on *SECTION_SOLID) with *MAT_ACOUSTIC as the material model are supported for *CONTROL_IMPLICIT_SSD_DIRECT analyses. The coupling of the acoustic fluid and the structural elements is achieved with *BOUNDARY_ACOUSTIC_COUPLING_MISMATCH or by merging acoustic and structural nodes with compatible element faces.

*CONTROL_IMPLICIT_STABILIZATION_{OPTION}

Available options include:

<BLANK>

DYN

SPR

Purpose: This optional card applies to implicit calculations. Artificial stabilization is required for multi-step unloading in implicit springback analysis (see also *CONTROL_IMPLICIT_GENERAL). The DYN option allows setting controls specifically for the dynamic relaxation phase. The SPR option allows setting controls specifically for the springback phase.

WARNING: For this keyword including the DYN and SPR keyword option variants, setting a field value to zero means zero and not the default value. For the DYN and SPR keyword options, leaving a field blank causes it to default to the value set for the version without a keyword option, meaning *CONTROL_IMPLICIT_STABILIZATION. For example, if SCALE = 0.001 for *CONTROL_IMPLICIT_STABILIZATION and the value of SCALE is left blank for *CONTROL_IMPLICIT_STABILIZATION_DYN, then the value of SCALE for *CONTROL_IMPLICIT_STABILIZATION_DYN will also be 0.001.

Card 1	1	2	3	4	5	6	7	8
Variable	IAS	SCALE	TSTART	TEND				
Type	I	F	F	F				
Default	2	1.0	↓	↓				

VARIABLE	DESCRIPTION
IAS	Artificial stabilization flag (see Remark 2): EQ.1: Active EQ.2: Inactive (default)

VARIABLE	DESCRIPTION
SCALE	Scale factor for artificial stabilization (see Remark 3). For flexible parts with large springback, like outer body panels, a value of 0.001 may be required. EQ.-n: Curve ID = n gives SCALE as a function of time
TSTART	Start time. (Default: immediately upon entering implicit mode)
TEND	End time. (Default: termination time)

Remarks:

1. **Artificial stabilization.** Artificial stabilization allows springback to occur over several steps. This is often necessary to obtain convergence during equilibrium iterations on problems with large springback deformation. Stabilization is introduced at the start time TSTART, and slowly removed as the end time TEND is approached. Intermediate results are not accurate representations of the fully unloaded state. The end time TEND must be reached exactly for total springback to be predicted accurately.

2. **IAS defaults.** The default for IAS depends on the analysis type in *CONTROL_IMPLICIT_GENERAL. For "seamless" springback analysis, automatic time step control and artificial stabilization are activated by default. Otherwise, IAS is inactive by default.

3. **Scale factor.** SCALE is a penalty scale factor similar to that used in contact interfaces. If modified, it should be changed in order-of-magnitude increments at first. Large values suppress springback deformation until very near the termination time, making convergence during the first few steps easy. Small values may not stabilize the solution enough to allow equilibrium iterations to converge.

*CONTROL_IMPLICIT_STATIC_CONDENSATION_{OPTION}

NOTE: As of R16, this keyword is obsolete. For SC_FLAG = 1, LS-DYNA automatically converts this keyword to *CONTROL_IMPLICIT_MODES. SC_FLAG = 2 leads to an error termination. To obtain the behavior of SC_FLAG = 2, compute the superelement with *CONTROL_IMPLICIT_MODES and then create a second run that uses that superelement.

Available options include:

<BLANK>

BINARY

Purpose: Request static condensation of a part to build a reduced linearized model for later computation with *ELEMENT_DIRECT_MATRIX_INPUT. Optionally, the analysis can continue using the linearization for the current analysis.

Note: Implicit Static Condensation is not supported in MPP. Please use *CONTROL_IMPLICIT_MODES.

Card Summary:

Card 1. This card is required.

SC_FLAG	SC_NSID	SC_PSID	SE_MASS	SE_STIFF	SE_INERT		

Card 1.1. Include this card if at least one of SE_MASS, SE_STIFF, or SE_INERT is not blank.

SE_FILENAME

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SC_FLAG	SC_NSID	SC_PSID	SE_MASS	SE_STIFF	SE_INERT		
Type	I	I	I	A10	A10	A10		
Default	0	0	0	↓	↓	↓		

Card 1.1	1	2	3	4	5	6	7	8
Variable	SE_FILENAME							
Type	A80							

VARIABLE	DESCRIPTION
SC_FLAG	Static Condensation Control Flag: EQ.0: No static condensation will be performed. EQ.1: Create superelement representation based on static condensation. EQ.2: Use static condensation to build a linearized representation for a part and use that linearized representation in the following analysis.
SC_NSID	Node set ID for nodes to be preserved in the static condensation procedure. Required when SC_FLAG = 1.
SC_PSID	Part set ID for parts to be included in the static condensation procedure. When SC_FLAG = 1, SC_PSID can be used to specify a subset of the model with the default being the entire model. When SC_FLAG = 2, SC_PSID is required. SC_PSID = 0 implies that the entire model is condensed.
SE_MASS	Name of the superelement mass matrix. If left blank, it is not generated.
SE_STIFF	Name of the superelement stiffness matrix. If left blank, it is not generated.

VARIABLE	DESCRIPTION
SE_INERT	Name of the superelement inertia matrix, required for gravity loading applications of the superelement. If left blank, it is not generated.
SE_FILENAME	If any of SE_MASS, SE_STIFF, or SE_INERT is blank, then the second line is required and contains the file name for the superelement.

Remarks:

1. **Input and output files.** To use this feature, an implicit analysis must be requested using IMFLAG = 1 on *CONTROL_IMPLICIT_GENERAL, and a nonzero termination time must be specified on *CONTROL_TERMINATION. A double-precision version of LS-DYNA should be used for best accuracy. The superelement model is written to file SE_FILENAME.

2. **Modes.** Static condensation is the reduction of the global stiffness and mass matrices to a specified set of rows and columns associated with the nodes in the node set SC_NSID. The first and second node set attribute parameters can be optionally used to restrict the translational and rotational degrees of freedom for which modes are requested, respectively, according to the following syntax:

   a) Node set attribute parameters DA1 and A1: translational degree of freedom codes

   b) Node set attribute parameters DA2 and A2: rotational degree of freedom codes

Code	Modes Computed
0	(See note below.)
1	*X* degree of freedom only
2	*Y* degree of freedom only
3	*Z* degree of freedom only
4	*X*, *Y* degrees of freedom only
5	*Y*, *Z* degrees of freedom only
6	*X*, *Z* degrees of freedom only
7	*X*, *Y*, *Z* degrees of freedom

Setting both node set attributes to zero is equivalent to setting both node set attributes to 7 (*X*, *Y*, and *Z* for translational and rotational degrees of freedom).

If one node set attribute is nonzero (codes 1 to 7) and the other node set attribute is zero, then the zero attribute means NO degrees of freedom are considered. For example, if DA1 = 2 and DA2 = 0, then only the *Y*-translational degree of freedom modes are calculated.

3. **Superelements.** The user can create the superelement representation of the reduced model by specifying the SE_MASS, SE_STIFF, SE_INERT and SE_FILENAME fields. This implementation does not include SE_DAMP. The file, by default is written in the Nastran DMIG file format and can be used as input to *ELEMENT_DIRECT_MATRIX_INPUT. The keyword option BINARY can be used to create a binary representation for the superelement that can be used with *ELEMENT_DIRECT_MATRIX_INPUT_BINARY to reduce the file size.

4. **Comparison to *CONTROL_IMPLICIT_MODES.** Static condensation is equivalent to using only constraint modes with *CONTROL_IMPLICIT_MODES. Static condensation does have the ability to continue the analysis using the linear representation for a part set.

*CONTROL_IMPLICIT_TERMINATION

Purpose: Specify termination criteria for implicit transient simulations.

Card 1	1	2	3	4	5	6	7	8
Variable	DELTAU	DELTA1	KETOL	IETOL	TETOL	NSTEP	ABSTOL	
Type	F	F	F	F	F	I	F	
Default	0.0	0.0	0.0	0.0	0.0	3	0.0	

VARIABLE	DESCRIPTION
DELTAU	Terminate based on relative total displacement in the Euclidean norm: GT.0.0: Terminate when the displacement in the Euclidean norm for the last time step relative to the total displacement in the Euclidean norm is less than DELTAU
DELTA1	Terminate based on relative total displacement in the max norm: GT.0.0: Terminate when the displacement in the max norm for the last time step relative to the total displacement in the max norm is less than DELTAU
KETOL	Terminate based on kinetic energy GT.0.0: Terminate when kinetic energy drops below KETOL for NSTEP consecutive implicit time steps.
IETOL	Terminate based on internal energy: GT.0.0: Terminate when internal energy drops below IETOL for NSTEP consecutive implicit time steps
TETOL	Terminate based on total energy: GT.0.0: Terminate when total energy drops below TETOL for NSTEP consecutive implicit time steps
NSTEP	Number of steps used in the early termination tests for kinetic, internal, and total energy

VARIABLE	DESCRIPTION
ABSTOL	Terminate based on absolute total displacement in the Euclidean norm: GT.0.0: Terminate when displacement in the Euclidean norm for last time step is less than ABSTOL.

Remarks:

For some implicit applications, it is useful to terminate when the displacement does not change or when the energy is low. This keyword provides the ability to specify such a stopping criterion to terminate the simulation prior to ENDTIM.

*CONTROL_LSDA

This keyword is used to globally control data present in included dynain.lsda files generated by *INTERFACE_SPRINGBACK_LSDYNA with FTYPE = 3. The primary function is to provide the flexibility of not having to know *a priori* how to define *INTERFACE_-SPRINGBACK but be able to modify this *a posteriori* without having to rerun the first simulation. For examples, see remarks.

Card 1	1	2	3	4	5	6	7	8
Variable	NPEXCL							
Type	I							
Default	0							

Excluded Parts Card. If NPEXCL ≠ 0, include as many of this card as required to exclude NPEXCL part IDs.

Card 1.1	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
NPEXCL	Number of parts to exclude from the dynain.lsda files
P1 - P8	Part IDs to exclude from dynain.lsda

Remarks:

1. **Excluding Parts Example.** Assume a forming simulation is performed with 2 deformable parts (say part 1 and part 2) and both are for some reason included in the resulting dynain.lsda. In a later stage, a springback of only part 1 is desired without the need of part 2. Instead of rerunning the forming simulation and outputting only part 1 to dynain.lsda, the springback can be performed by excluding part 2 from the springback simulation. This is done through NPEXCL = 1 and P1 = 2.

*CONTROL_MAPPING_SHIFT_BY_DT

Purpose: For *INITIAL_ALE_MAPPING and *INITIAL_LAG_MAPPING, two subsequent models are run with the second run initialized by data from the last cycle of the first run. This keyword shifts the initial time of the second run by the time step read from the mapping file, meaning the final time step of the first run.

*CONTROL_MAT

Purpose: Define global control parameters for properties related to the material model.

Card 1	1	2	3	4	5	6	7	8
Variable	MAEF		UMCHK	OLDINTP	IMPCONT		PSMAEF	RATESET
Type	I		I	I	I		I	I
Default	0		0	0	0		0	0

VARIABLE	DESCRIPTION
MAEF	Failure options: EQ.0: All *MAT_ADD_EROSION, *MAT_ADD_DAMAGE_-DIEM, and *MAT_ADD_DAMAGE_GISSMO definitions are active. EQ.1: Switch off all *MAT_ADD_EROSION, *MAT_ADD_-DAMAGE_DIEM, and *MAT_ADD_DAMAGE_GISSMO definitions globally. This feature is helpful for larger models where removing those cards is inconvenient. EQ.2: Same as 1 but also deactivate plastic strain to failure in some material models. In other words, FAIL is internally set to zero in materials 24, 103, 105, 106, 114, 123, 124, 155, 195, 225, 251, and 255.
UMCHK	User-material check. In the first calculation cycle, the solver checks if user-defined material models are applied or whether only the default, unmodified subroutines already present in the native dyn21 files are called. It also works for user-defined friction in MPP. EQ.0: Warning is issued if only unmodified subroutines are called. EQ.1: Error termination occurs if only unmodified subroutines are called.
OLDINTP	Because of changes in results when correcting the interpolation of tables in the piecewise plasticity model (*MAT_024), you can set this flag to 1 to invoke the old behavior. Note that the old behavior may result in an incorrect stress response, so we generally do not

VARIABLE	DESCRIPTION
	recommend it.
IMPCONT	Flag to ignore issues coming from the material models, such as the plasticity routine not converging, when using implicit (see Remark 1): EQ.0: When issues arise in the material model routines, the job error terminates unless IAUTO is greater than 0 on *CONTROL_IMPLICIT_AUTO. If IAUTO > 0, the time step size reduces. EQ.1: Continue the job, ignoring the issue coming from the material model routine.
PSMAEF	Part set ID for option MAEF: EQ.0: MAEF is applied to all parts in the model (default). GT.0: MAEF is applied to all parts in this set. LT.0: MAEF is applied to all parts not in set |PSMAEF|.
RATESET	Part set ID for IRATE on *CONTROL_IMPLICIT_DYNAMICS. If nonzero, then IRATE applies only to the parts included in this part set. See *SET_PART.

Remarks:

1. **IMPCONT.** Enabling the IMPCONT causes the implicit solver to ignore certain issues coming from the material routines that normally lead to an error termination or time step reduction. This flag causes implicit to behave more like explicit, which automatically ignores these issues. This flag should be used with caution and only in cases where the material routine becomes "stuck" for a small set of elements in the model.

*CONTROL_MPP

Purpose: Set control parameters for MPP-specific features.

All options with DECOMPOSITION in the name control the distribution of the model to the processors. However, these keywords do not affect the decomposition of seat-belt-related elements because these types of elements are decomposed in an unalterable, predetermined way. All seat belt elements in a single seat belt are allocated to a single processor. For a model with more than one seat belt, the decomposition algorithm assigns each seat belt to a different processor if available.

*CONTROL_MPP_CONTACT_GROUPABLE

*CONTROL_MPP_DECOMPOSITION_ADJUST_MTYPE_COST

*CONTROL_MPP_DECOMPOSITION_ADJUST_PART_COST

*CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS

*CONTROL_MPP_DECOMPOSITION_AUTOMATIC

*CONTROL_MPP_DECOMPOSITION_BAGREF

*CONTROL_MPP_DECOMPOSITION_CHECK_SPEED

*CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE

*CONTROL_MPP_DECOMPOSITION_DEFORMED_GEOMETRY

*CONTROL_MPP_DECOMPOSITION_DISABLE_UNREF_CURVES

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_SPH_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_ELCOST

*CONTROL_MPP_DECOMPOSITION_FILE

*CONTROL_MPP_DECOMPOSITION_FLAG_STRESS_STRAIN_CURVE

*CONTROL_MPP_DECOMPOSITION_METHOD

*CONTROL_MPP_DECOMPOSITION_NODISTRIBUT_DES_ELEMENTS

*CONTROL_MPP_DECOMPOSITION_NUMPROC

*CONTROL_MPP_DECOMPOSITION_OUTDECOMP

*CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE

*CONTROL_MPP_DECOMPOSITION_RCBLOG

*CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION

*CONTROL_MPP_DECOMPOSITION_SCALE_CONTACT_COST

*CONTROL_MPP_DECOMPOSITION_SCALE_FACTOR_SPH

*CONTROL_MPP_DECOMPOSITION_SHOW

*CONTROL_MPP_DECOMPOSITION_TRANSFORM_KEYWORD

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION

*CONTROL_MPP_IO_LSTC_REDUCE

*CONTROL_MPP_IO_NOBEAMOUT

*CONTROL_MPP_IO_NOD3DUMP

*CONTROL_MPP_IO_NODUMP

*CONTROL_MPP_IO_NOFULL

*CONTROL_MPP_IO_NOTIEDIO

*CONTROL_MPP_IO_SWAPBYTES

*CONTROL_MPP_MATERIAL_MODEL_DRIVER

*CONTROL_MPP_PFILE

*CONTROL_MPP_REBALANCE

*CONTROL_MPP_CONTACT_GROUPABLE

Purpose: Allow for global specification that the GROUPABLE algorithm should be enabled/disabled for contacts when running MPP.

Card 1	1	2	3	4	5	6	7	8
Variable	GRP							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
GRP	The sum of these available options (in any combination that makes sense): EQ.1: Turn on GROUPABLE for all non-tied contacts. This flag does not apply to segment-to-segment contact (SOFT = 2). EQ.2: Turn on GROUPABLE for all tied contacts EQ.4: Turn off GROUPABLE for all non-tied contacts. This flag does not apply to segment-to-segment contact (SOFT = 2). EQ.8: Turn off GROUPABLE for all tied contacts. EQ.16: Turn on GROUPABLE for segment-to-segment contact (SOFT = 2). GRP overrides the setting of GRPABLE on the MPP Cards of *CONTACT_..._MPP for GRPABLE ≥ 0.

Remarks:

1. **Availability.** The groupable algorithm is an alternate MPP communication algorithm for various single surface (including AUTOMATIC_GENERAL), NODES_TO_SURFACE, SURFACE_TO_SURFACE, ERODING, and option SOFT = 2 contacts. This groupable algorithm does not support all contact options, including MORTAR, and is still under development. It can be significantly faster and scale better than the normal algorithm when there are more than two or three applicable contact types defined in the model. Its intent is to speed up the contact processing but not to change the behavior of the contact.

2. **Priority of this keyword.** This keyword overrides GRPABLE on the MPP cards of *CONTACT_…_MPP for GRPABLE ≥ 0. It is intended as a way to quickly experiment with this feature. The equivalent pfile option is "contact { groupable GRP }" where GRP is an integer as described above.

3. **Shell reference surface for tied contacts in MPP.** Unless the MPP contact is groupable (see GRPABLE on the MPP cards of *CONTACT_…_MPP and GRP here), MPP tied contacts ignore the influence of NLOC on *SECTION_SHELL, OFFSET on *ELEMENT_SHELL_OFFSET, CNTCO on *CONTROL_SHELL, and SHLOFF on Optional Card G on the shell reference surface. Non-groupable tied contacts always use the plane determined by the nodes for the shell reference surface. Thus, these fields for non-groupable tied contacts have no effect on determining what is tied or not. Groupable tied contacts honor the settings of these fields.

*CONTROL_MPP_DECOMPOSITION_ADJUST_MTYPE_COST

Purpose: Modify the decomposition cost of elements according to their material type.

Any number of cards may appear with one material per card.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPEID	A	B					
Type	I	F	F					
Default	none	1.0	0.0					

VARIABLE	DESCRIPTION
TYPEID	Material type ID as a number. For example, 1 here indicates *MAT_ELASTIC.
A	Cost multiplier
B	Added cost

Remarks:

1. **Adjusting computational cost.** During decomposition, all elements having the specified material type have their computational cost adjusted according to this formula:

$$\text{new cost} = A \times (\text{original cost}) + B$$

2. **Same material type in more than one card.** If a material appears in more than one instance of this card, only the first instance is used, and the others are ignored.

*CONTROL_MPP_DECOMPOSITION_ADJUST_PART_COST

Purpose: Modify the decomposition cost of elements according to their part ID.

Any number of cards may appear with one part per card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	A	B					
Type	I	R	R					
Default	none	1.0	0.0					

VARIABLE	DESCRIPTION
PID	Part ID
A	Cost multiplier
B	Added cost

Remarks:

1. **Adjusting computational cost.** During decomposition, all elements belonging to the specified part ID have their computational cost adjusted according to this formula:

$$\text{new cost} = A \times (\text{original cost}) + B$$

2. **Same part in more than one card.** If a part appears in more than one instance of this card, only the first instance is used, and the others are ignored. Any values given here override conflicting data entered with *CONTROL_MPP_DECOMPOSITION_ADJUST_MTYPE_COST because this keyword is more specific.

*CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS_{OPTION}

Purpose: Distribute certain part(s) to all processors or isolate certain part(s) to a single processor. This keyword supports multiple entries. Each entry is processed as a separate region for decomposition.

Available options include:

<BLANK>

LOCAL

When this keyword is part of an included file and the LOCAL option is given, the decomposition will be done in the coordinate system of the included file, which may be different from the global system if the file is included using *INCLUDE_TRANSFORM.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NPROC	FRSTP	NPTOT			
Type	I	I	I	I	I			
Default	none	none	none	none	0			

VARIABLE	DESCRIPTION
ID	Part ID/part set ID
TYPE	Type of distribution: EQ.0: The part is distributed to all processors. EQ.1: The part set is distributed to all processors. EQ.10: The part is lumped onto one processor. EQ.11: The part set is lumped onto one processor. EQ.20: The part is lumped onto one processor with MPP load balanced. EQ.21: The part set is onto one processor with MPP load balanced.
NPROC	Used only for TYPE equal to 0 or 1. Number of processors that will be used for decomposition. This part/part set will be distributed to NPROC of the processors.

VARIABLE	DESCRIPTION
FRSTP	Used only for TYPE equal to 0 or 1. Starting MPP rank ID (rank starts from 0).
NPTOT	Total number of processors for (NPROC, FRSTP): EQ.0: Use number of processors currently being used in the run GT.0: If max(number of processors currently being used, number of processors upon which the job is decomposed) differs from NPTOT, (NPROC, FRSTP) will be rescaled by NPTOT. This feature allows the same input to be used with changing computing resources.

Remarks:

1. **pfile.** Because this keyword supports multiple entries, each line will be treated as a region for the decomposition. Therefore, the equivalent method using a pfile will be different depending on TYPE and whether the entity being decomposed is a part or parts set as shown in the table below.

Type	pfile Equivalent
0	region { parts PID nproc NPROC FRSTP }
1	region { partsets PSID nproc NPROC FRSTP }
10	region { parts PID lumped }
11	region { partsets PSID lumped }
20	region { parts PID together }
21	region { partsets PSID together }

*CONTROL_MPP_DECOMPOSITION_AUTOMATIC

Purpose: Instructs the program to apply a simple heuristic to try to determine the proper decomposition for the simulation.

There are no input parameters. The existence of this keyword triggers the automated decomposition. This option should not be used if there is more than one occurrence of any of the following options in the model:

*INITIAL_VELOCITY

*CHANGE_VELOCITY

*BOUNDARY_PRESCRIBED_MOTION

And the following control card must not be used:

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION

For the general case, it is recommended that you specify the proper decomposition using the command *CONTROL_MPP_DECOMPOSITION_TRANSFORMATION instead.

*CONTROL_MPP_DECOMPOSITION_BAGREF

Purpose: Perform decomposition according to the airbag's reference geometry, rather than the folded geometry.

Other than BAGID values this card takes no input parameters. The initial geometry may lead to a poor decomposition once the bag is deployed. This option improves load balancing for the fully deployed geometry.

Optional card(s) for selected reference geometry ID

Card 1	1	2	3	4	5	6	7	8
Variable	BAGID1	BAGID2	BAGID3	BAGID4	BAGID5	BAGID6	BAGID7	BAGID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BAGID*i*	ID defined in *AIRBAG_REFERENCE_GEOMETRY_ID or *AIRBAG_SHELL_REFERENCE_GEOMETRY_ID

Bags specified in the optional cards will be decomposed based on the reference geometry. If there is no card given, *all* bags will be decomposed by their reference geometry.

Remarks:

Command in partition file (pfile): BAGREF. The option for selecting particular airbags is only available when using keyword input.

*CONTROL_MPP_DECOMPOSITION_CHECK_SPEED

Purpose: Modifies the decomposition depending on the relative speed of the processors involved.

There are no input parameters. Use of this keyword activates a short floating point timing routine to be executed on each processor. The information gathered is used during the decomposition, with faster processors being given a relatively larger portion of the problem. This option is not recommended on homogeneous systems.

*CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE_{OPTION}

Purpose: Ensure that the indicated contact interfaces are distributed across all processors, which can lead to better load balance for large contact interfaces.

Available options include:

<BLANK>

LOCAL

If this keyword is part of an included file and the LOCAL option is given, the decomposition will be done in the coordinate system of the included file, which may be different from the global system if the file is included through *INCLUDE_TRANSFORM.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
ID1	First contact interface ID to distribute. If no contact IDs are specified, the number given here corresponds to the order of the interfaces as they appear in the input, with the first being 1.
ID2, ID3, ID4, ID5	Remaining interfaces IDs to distribute

Remarks:

Up to 5 contact interface IDs can be specified. The decomposition is modified as follows: First, all the elements involved in the first contact interface are decomposed across all the processors. Then all the elements involved in the second contact interface (excluding any already assigned to processors) are distributed, and so on. After all the contact interfaces given are processed, the rest of the input is decomposed in the normal manner. This will result in each processor having possibly several disjoint portions of the input assigned to it, which will increase communications somewhat. However, this can be offset by improved load balance for the contact. It is generally recommended that at most one or two

interfaces be specified, and then only if they are of substantial size relative to the whole problem.

*CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE

Purpose: Ensures that the indicated contact interfaces are isolated on a single processor, which can lead to decreased communication.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
ID1	First contact interface ID to isolate. If no contact ID's are specified, the number given here corresponds to the order of the interfaces as they appear in the input, with the first being 1.
ID2, ID3, ID4, ID5	Remaining interfaces ID's to isolate.

Remarks:

Up to 5 contact interfaces can be specified. The decomposition is modified as follows: First, all the elements involved in the first contact interface ID are assigned to the first processor. Then all the elements involved in the second contact interface ID (excluding any already assigned to processors) are assigned to the next processor, and so on. After all the contact interfaces given are processed, the rest of the input is decomposed in the normal manner. This will result in each of the interfaces being processed on a single processor. For small contact interfaces this can result in better parallelism and decreased communication.

*CONTROL_MPP_DECOMPOSITION_DEFORMED_GEOMETRY

Purpose: Because of distortion due to deformation or changing contacts due to rotation, a new decomposition of the deformed geometry may need to be performed during a full deck restart for better load balancing. This option causes the new load balance. This feature may be useful, for instance, with bird strike since the contacts change as the fan blades rotate.

SPH Card. This card is optional.

Card 1	1	2	3	4	5	6	7	8
Variable	SPH							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
SPH	Flag for inclusion of inactive SPH elements: EQ.0: Exclude inactive SPH elements in the new decomposition (default). EQ.1: Include inactive SPH elements in the new decomposition.

Remarks:

Command in partition file (pfile): decomposition { defgeo } or decomposition { defgeo 1 }.

*CONTROL_MPP_DECOMPOSITION_DISABLE_UNREF_CURVES

Purpose: Disable unreferenced time dependent load curves for the following keywords:

*BOUNDARY_PRESCRIBED_MOTION_NODE

*LOAD_NODE

*LOAD_SHELL_ELEMENT

*LOAD_THERMAL_VARIABLE_NODE

The details of this operation are reported in each processor's scratch scr#### file. This will skip the curve evaluation on each cycle, and improve the parallel efficiency.

Remarks:

Command in partition file (pfile): DUNREFLC.

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_ALE_ELEMENTS

Purpose: Ensure ALE elements are evenly distributed to all processors.

There are no input parameters and the card below is optional. ALE elements usually have a larger computational cost than other element types, and it is better to distribute them to all CPUs for better load balance. This keyword causes DYNA/MPP to extract ALE parts from the input and then evenly distribute them to all the processors.

FSI/ALE Mesh Card. This card is optional.

Card 1	1	2	3	4	5	6	7	8
Variable	OVERLAP							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
OVERLAP	For FSI models where structures are inside ALE meshes (see *CONSTRAINED_LAGRANGE_IN_SOLID), flag to decompose the structure and ALE domains together instead of first the structure and then the ALE (see Remark 2). EQ.0: Off EQ.1: On

Remarks:

1. **Partition file.** Command in partition file (pfile): ALEDIST.

2. **ALE/FSI overlap.** Most of the processors will have to deal with MPP subdomains from the structure and ALE meshes: a portion of the ALE computational domain and a portion of the structure meshes. The default decomposition (first divide the structures, then ALE) does not always overlap these subdomains. The more they overlap, the less the MPP communications due to the coupling cost. Cutting the ALE and structure meshes together allows their MPP subdomains to be as inclusive as possible.

*CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_SPH_ELEMENTS

Purpose: Ensures SPH elements are evenly distributed to all processors

There are no input parameters. SPH elements usually have higher computational cost than other type of elements and it is better to distribute them to all CPU for better load balance. The existence of this keyword causes DYNA/MPP to extract SPH parts from input and then evenly distributed to all processors.

Remarks:

Command in partition file (pfile): SPHDIST.

*CONTROL_MPP_DECOMPOSITION_ELCOST

Purpose: Instructs the program to use a hardware specific element cost weighting for the decomposition

Card 1	1	2	3	4	5	6	7	8
Variable	ITYPE							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
ITYPE	Hardware specific cost profile. EQ.1: Fujitsu PrimePower EQ.2: Intel IA 64, AMD Opteron EQ.3: Intel Xeon 64 EQ.4: General profile

Remarks:

Command in partition file (pfile): elcost itype.

*CONTROL_MPP_DECOMPOSITION_FILE_{OPTION}

Purpose: Create a pre-decomposition file that can be used in subsequent runs without having to do the decomposition. This keyword also reads the pre-decomposition file from a previous run. This file cannot be used to visualize the decomposition. For that, see *CONTROL_MPP_DECOMPOSITION_OUTDECOMP (for versions prior to R18, see *CONTROL_MPP_DECOMPOSITION_SHOW).

OPTION specifies the action type. The following options are accepted:

<BLANK>

READ

WRITE

Without a keyword option, this keyword causes the solver to look for a pre-decomposition file in the working directory. If the file is not there, the file is created during the decomposition phase. If the file exists, then the existing pre-decomposition file is used for the decomposition. When the keyword option READ is used, the pre-decomposition file must be in the working directory. The job will terminate if the file is not there. When WRITE is used, the file is created, but the job will terminate if the pre-decomposition file already exists in the working directory.

Card 1	1	2	3	4	5	6	7	8
Variable	NAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
NAME	Name of a file containing (or to contain) a decomposition record

Remarks:

1. **Number of processors for creating and using a pre-decomposition file.** The original run that created the pre-decomposition file must be for a number of processors that is a multiple of the number of processors currently being used. Thus, a problem can be decomposed once for, say, 48 processors. Subsequent runs are then possible on any number that divides 48: 1, 2, 3, 4, 6, etc. Since the

decomposition phase generally requires more memory than execution, this allows large models to be decomposed on one system and run on another (provided the systems have compatible binary formats). See Remark 1 of *CONTROL_MPP_DECOMPOSITION_NUMPROC for a discussion of creating this file with a single processor for a decomposition on N domains.

2. **pfile command.** The equivalent pfile command to this keyword is: decomp { file_*option filename* }. In this command, *filename* is the name of the file that contains or will contain the decomposition information, and *option* can be unused or set to READ or WRITE. These values for *option* work exactly like their equivalent keyword options.

3. ***r*-adaptivity restriction.** This keyword is *not* supported for simulations using 2D/3D *r*-adaptivity (see *CONTROL_ADAPTIVE). When using *r*-adaptivity, the adaptive parts are remeshed at each adaptive step, followed by an automatic restart procedure that involves MPP decomposition. If this keyword is included in the input deck for *r*-adaptive simulations, the job will crash during the automatic restart.

*CONTROL_MPP_DECOMPOSITION_FLAG_STRESS_STRAIN_CURVE

Purpose: Flag to skip evaluating non-time dependent stress strain curves on every cycle to save CPU time. The stress strain curves will only be evaluated as needed in the material routines. Currently, this keyword only affects curves used with *MAT_024.

Remarks:

Command in partition file (pfile): SORTSSLC.

*CONTROL_MPP_DECOMPOSITION_METHOD

Purpose: Specify the decomposition method to use.

Card 1	1	2	3	4	5	6	7	8
Variable	NAME							
Type	A80							
Default	RCB							

VARIABLE	DESCRIPTION
NAME	Name of the decomposition method to use. There are currently two options: EQ."RCB": recursive coordinate bisection EQ."GREEDY": a simple heuristic method In almost all cases the RCB method is superior and should be used.

*CONTROL_MPP_DECOMPOSITION_NODISTRIBUTE_DES_ELEMENTS

Purpose: By default, DES elements are treated as a single region and distributed to all processors. For traditional MPP execution in which LS-DYNA performs a single decomposition at the beginning of the run, the default treatment of distributing to all processors yields better load balancing. For MPP that includes repartition of the model during execution with *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION, putting DES elements and couple FE elements on the same processor leads to better performance. This keyword is for the second case. It disables the default distribution and causes current geometry to be used for decomposition.

There are no input parameters.

Remarks:

Command in partition file (pfile): DESNODIST.

*CONTROL_MPP_DECOMPOSITION_NUMPROC

Purpose: Specify the number of domains onto which the problem is decomposed.

Card 1	1	2	3	4	5	6	7	8
Variable	N							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
N	Number of domains for decomposition

Remarks:

1. **Pre-decomposition file.** This keyword can be used in conjunction with *CONTROL_MPP_DECOMPOSITION_FILE to create a pre-decomposition file, allowing for later runs on a different number of processors. By default, the decomposition is performed for the number of processors currently being used. However, a different value can be specified here. If N > 1 and only one processor is currently being used, the problem is decomposed for N domains, and then the program terminates. If the current number of processors is greater than 1 and N is *not* a multiple of the the current number of processors, then N is ignored, and the execution proceeds with the current number of processors. Otherwise, the decomposition is performed for N domains, and the execution continues using the current number of processors.

2. **Visualizing the decomposition.** This keyword can be used with *CONTROL_MPP_DECOMPOSITION_OUTDECOMP (as of R18) or *CONTROL_MPP_DECOMPOSITION_SHOW (before R18) to create a file for visualizing the final decomposition using a single processor. This keyword gives the number of domains upon which the problem is decomposed. See Remark 3 of *CONTROL_MPP_DECOMPOSITION_OUTDECOMP and Remark 1 of *CONTROL_MPP_DECOMPOSITION_SHOW for details.

3. **pfile command.** The following pfile command is equivalent to this keyword: decomp { numproc *n* }. In this command, *n* is the number of domains for which the problem is decomposed.

*CONTROL_MPP_DECOMPOSITION_OUTDECOMP

Purpose: Instruct the program to output the element's ownership data to file for a post-processor to show state data from different processors. In versions R18 and later, this keyword additionally can output the final decompostion and terminate the job after the decomposition phase like *CONTROL_MPP_DECOMPOSITION_SHOW. See Remark 3 for details.

Card 1	1	2	3	4	5	6	7	8
Variable	ITYPE							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
ITYPE	Set the format for the output file. EQ.1: Database in LS-PrePost format: decomp_parts.lsprepost EQ.2: Database in animator format: decomp_parts.ses EQ.3: Database in LS-PrePost format with a d3plot state number: decomp_parts.lsprepost_s######. This file allows LS-PrePost to show the matching d3plot with the decomposition for *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION.

Remarks:

1. **PFILE.** Command in partition file (pfile): decomp { outdecomp *itype* }.

2. **Viewing results in LS-PrePost for elements on a particular core.** When ITYPE is set to 1, the elements assigned to any particular core can be viewed and animated by LS-PrePost by (1) reading the d3plot data, and then (2) selecting *Models → Views → MPP → Load → decomp_parts.lsprepost*.

3. **Outputting the decomposition for visualization.** In versions R18 and later, an input deck containing this keyword with ITYPE set to 1 or 3 run on a single processor gives the same behavior as *CONTROL_MPP_DECOMPOSITION_SHOW. The solver outputs the final decomposition in an LS-PrePost format file and terminates the job after the decomposition phase. To specify the number of

domains upon which the problem is decomposed, use *CONTROL_MPP_DECOMPOSITION_NUMPROC. Note that the decomposition for a subsequent run cannot be initialized with this file. See *CONTROL_MPP_DECOMPOSITION_FILE for obtaining a file with the decomposition that can be used for subsequent jobs.

This visualization file can also be obtained by running the job on a single processor with the following pfile commands instead of their corresponding keywords:

```
decomp { outdecomp itype }
decomp ( numproc n }
```

In the above, *itype* must be set to 1 or 3, and *n* is the number of domains upon which the problem is decomposed.

*CONTROL_MPP_DECOMPOSITION_PARTS_DISTRIBUTE_{OPTION}

Purpose: Distribute the parts given by this keyword to all processors before the decomposition for the rest of the model is performed.

If this appears in an included file and the **LOCAL** option is used, the decomposition will be done in the coordinate system of the included file, which may be different from the global system if the file is included via *INCLUDE_TRANSFORM.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ID1, ID2, ID3, …	For each ID: GT.0: ID is a part number. LT.0: |ID| is a part set number. All parts defined in this card will be treated as a single region to be decomposed.

Remarks:

1. **Decomposition.** Up to 1000 parts/part sets can be specified. The decomposition is modified as follows: the elements involved in the given parts are put into a separate domain from rest of the model and then distributed to all processors to balance their computational cost. Then the remainder of the model will be distributed in the usual way.

2. **PFILE.** This is equivalent to the pfile command (for example, if ID1 - ID3 are part IDs and ID4 - ID6 are part set IDs):

   ```
   decomp { region { parts ID1 ID2 ID3 or partsets ID4 ID5 ID6 } }
   ```

   The part set IDs, however, are positive when used in the pfile.

CONTROL_MPP_DECOMPOSITION_PARTSET_DISTRIBUTE_*{OPTION}

Purpose: Distribute the part sets given in this option to all processors before the decomposition for the remainder of the model is performed.

If this appears in an included file and the **LOCAL** option is given, the decomposition will be done in the coordinate system of the included file, which may be different from the global system if the file is included via *INCLUDE_TRANSFORM.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ID1, ID2, ID3, …	Part Set ID to be distributed. All parts in ID1 will be shared across all processors. Then all parts in ID2 will be distributed, and so on.

Remarks:

Any number of part sets can be specified. Each part set is distributed across all processors, in the order given. The order may be significant if, in particular, a part ID is in more than one set. Distribution of these parts is done before any decomposition specifications given in the pfile.

*CONTROL_MPP_DECOMPOSITION_RCBLOG

Purpose: Causes the program to record decomposition information in the indicated file, for use in subsequent analyses.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of output file where decomposition history will be recorded. This file can be used as the pfile for later analyses.

Remarks:

Command in parallel option file (pfile): rcblog filename.

*CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION_{OPTION}

Available options are:

<BLANK>

ONCE

ONCE causes this keyword to be disabled after the model has been redecomposed based on information collected from the initial step.

Purpose: Redecompose a model at a given time interval. Other decomposition directives given in a pfile or defined using *CONTROL_MPP_DECOMPOSITION_*OPTION* remain in effect for each redecomposition. Redecomposition is intended to improve computational efficiency in simulations where large relative displacements between nodes develop and change as the simulation progresses.

Redecomposition relies on LS-DYNA's full deck restart capability and therefore will work correctly only to the extent that a full deck restart works correctly. We advise examining results carefully to ensure that history variable values, such as stress values, present before redecomposition are present after redecomposition.

Card 1	1	2	3	4	5	6	7	8
Variable	FREQ	DEFGEO	WEIGHT	REMSPH	STIME	SAMPT		
Type	F	I	F	I	F	F		
Default	none	1	1.0	0	0.0	None		

VARIABLE	DESCRIPTION
FREQ	Determines the number of redecompositions during the solution: LT.0: |FREQ| rounded to the nearest integer is the number of redecompositions during the solution. GT.0: FREQ is the time interval between redecompositions.
DEFGEO	Geometry for decomposition: EQ.1: Use current geometry for decomposition. When applied to a model containing SPH, deactivated SPH elements are not considered in the partition. This will give

VARIABLE	DESCRIPTION
	better load balancing if SPH elements are deleted during the simulation. EQ.2: Use current geometry for decomposition (same as 1 if applied to a non-SPH model). When applied to a model containing SPH, all SPH elements are considered in the partition. This will give better load balancing if SPH elements are reactivated during the simulation.
WEIGHT	Element cost scale factor for an element in contact or an element that has undergone plastic strain. If the element is under contact and has plastic strain, the weight will be doubled. Since the element cost is measured from calculated quantities, the results will remain consistent between runs with the same input and decomposition unlike using SAMPT below.
REMSPH	Flag to remove deactivated SPH particles: EQ.0: Keep deactivated particles. EQ.1: Remove deactivated particles. EQ.2: Remove deactivated particles. Also, particles generated by the *DEFINE_SPH_MESH_BOX keyword are fragmented and added to the input file as needed during each redecomposition phase.
STIME	Start time for redecomposition
SAMPT	Time interval for collecting element cost profile to use in the next REDECOMP step. Since the cost profile is based on the measurement of elapsed time, the timing profile may change between runs. It may produce different numerical results from multiple runs using identical input. GT.0: Sampling from beginning of each redecomposition for length SAMPT (t to t + SAMPT). If SAMPT ≥ FREQ, then the sampling will occur for the entire time interval, FREQ. LT.0: Sampling from before ending of each redecomposition through to the next redecomposition (t + FREQ − SAMPT to t + FREQ)

*CONTROL_MPP_DECOMPOSITION_SCALE_CONTACT_COST

Purpose: Instructs the program to apply a scale factor to the list of contacts to change the partition weight for the decomposition.

Card 1	1	2	3	4	5	6	7	8
Variable	SF	ID1	ID2	ID3	ID4	ID5	ID6	ID7
Type	F	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
SF	Scale factor for the contact segments listed in the interface ID.
ID1, ID2, …	interfaces ID's to be considered for scaling. Include second card if necessary.

Remarks:

Up to 15 contact interfaces ID can be specified. The decomposition is modified by applying this scale factor to the default computational cost of elements for the given contact interface ID.

Command in partition file (pfile): CTCOST ID1, ID2, …, SF.

*CONTROL_MPP_DECOMPOSITION_SCALE_FACTOR_SPH

Purpose: Instructs the program to apply a scale factor to SPH elements to change the partition weight for the decomposition.

Card 1	1	2	3	4	5	6	7	8
Variable	SF							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
SF	Scale factor

Remarks:

Command in partition file (pfile): SPHSF SF.

*CONTROL_MPP_DECOMPOSITION_SHOW

Purpose: Write the final decomposition to the d3plot database. There are no input parameters. This output is used for visualizing the decomposition. It cannot be used to initialize a subsequent run. See *CONTROL_MPP_DECOMPOSITION_FILE for creating a pre-decomposition file for subsequent runs.

This keyword causes the MPP version to terminate immediately after the decomposition phase without performing an analysis. The resulting d3plot database is designed to allow visualization of the decomposition by making each part correspond to the group of solids, shells, beams, thick shells, or SPH particles assigned to a particular processor. For example, in a model that includes various element types including solids, part 1 corresponds to the solid elements assigned to processor 1, part 2 corresponds to the solid elements assigned to processor 2, and so on.

> **WARNING:** As of R18, this keyword is deprecated. To obtain the final decomposition without performing the full analysis, run the job on a single processor using an input deck that contains *CONTROL_MPP_DECOMPOSITION_OUTDECOMP with ITYPE = 1 or 3 and *CONTROL_MPP_DECOMPOSITION_NUMPROC with N set to the number of domains upon which the problem is decomposed. Like for *CONTROL_MPP_DECOMPOSITION_SHOW, the run terminates after the decomposition phase and outputs a file with the final decomposition. However, the file produced for viewing the final decomposition is in LS-PrePost format.

Remarks:

1. **Using one processor to obtain the decomposition over multiple domains.** Obtaining the final decomposition for *n* domains can be performed with a single processor. To do this, include *CONTROL_MPP_DECOMPOSITION_NUMPROC to specify the number of domains upon which the problem is to be decomposed in the input deck along with this keyword, and run the job on a single processor.

2. **pfile command.** The equivalent pfile command to this keyword is: decomp { show }.

*CONTROL_MPP_DECOMPOSITION_TRANSFORM_KEYWORD

Purpose: Cause all *CONTROL_MPP_DECOMPOSITION_TRANSFORMATION keywords to apply not just to the primary decomposition region but also to all the specific decomposition regions created by other *CONTROL_MPP_DECOMPOSITION_... keywords.

This keyword has no input parameters.

Remarks:

1. **PFILE.** The related command in the partition file (pfile) is: decomposition { transform_keyword}.

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION

Purpose: Specifies transformations to apply to modify the decomposition.

There are 10 different kinds of decomposition transformations available. For a detailed description of each, see Appendix O the LS-DYNA MPP user guide.

The data cards for this keyword consist of transformation operations. Each operation, depending on its type, involves either one or two additional cards. The input deck may include an arbitrary number of transformations with the next keyword, "*" card, terminating this input.

Transformation Card 1. For each transformation this card is required.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPE	V1	V2	V3	V4	V5	V6	
Type	A10	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	

Transformation Card 2. Additional card for TYPE set to one of VEC3, C2R, S2R, MAT.

Card 2	1	2	3	4	5	6	7	8
Variable	V7	V8	V9					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
TYPE	Which transformation to apply. The allowed values are RX, RY, RZ, SX, SY, SZ, VEC3, C2R, S2R, and MAT.
V1 - V9	For type set to either RX, RY, RZ, SX, SY, or SZ: The parameter V1 gives either the angle of rotation (RX, RY, RZ) or the magnitude for the scaling (SX, SY, SZ). The remaining parameters are ignored.

VARIABLE	DESCRIPTION
	For type set to either VEC3, C2R, S2R, or MAT: All parameters are used. See the appendix for the "pfile."

Remarks:

When using scale factors (SX, SY, SZ), specify a non-zero scale factor; if left blank, it will be set to use 0.0.

*CONTROL_MPP_IO_LSTC_REDUCE

Purpose: Use LST's own reduce routine to consistently sum floating point data among processors. There are no input parameters.

Remarks:

1. **PFILE.** The related command in the partition file (pfile) is: lstc_reduce.

2. **Motivation behind this Keyword.** LS-DYNA/MPP uses the Recursive Coordinates Bisection (RCB) algorithm to divide the model element-wise into sub-domains; each sub-domain (group of elements) will be executed in its own CPU core. For shared nodes with data on more than one processor, both internal and external forces must be added across the processors to determine the correct loading; this is called the "REDUCE" operation in MPI.

   Adding up the values across processors uses the standard MPI interface provided by different vendors, such as Platform MPI, Intel MPI, Open MPI, etc. Each vendor has its own algorithm for better communication. Therefore, the order of summation for shared nodes varies depending on the vendor's algorithm as well as hardware differences, namely, the number of cores per socket. This numerical noise triggers undesired bifurcation for explicit analysis with lots of integration cycles.

   LSTC_REDUCE ensures a deterministic summation order to provide numerical consistency. This keyword gives identical results while using the same decomposition and core counts but different MPI algorithms.

*CONTROL_MPP_IO_NOBEAMOUT

Purpose: Suppress beam, shell, and solid element failure messages in the d3hsp and message files. There are no parameters for this keyword.

Remarks:

Command in parallel option file (pfile): nobeamout.

*CONTROL_MPP_IO_NOD3DUMP

Purpose: Suppresses the output of the d3dump and runrsf files.

There are no input parameters for this keyword.

*CONTROL_MPP_IO_NODUMP

Purpose: Suppresses the output of all dump files and full deck restart files.

There are no input parameters. The existence of this keyword causes the d3dump and runrsf file output routines to be skipped. It also suppresses output of the full deck restart files d3full and runfull.

*CONTROL_MPP_IO_NOFULL

Purpose: Suppresses the output of the full deck restart files.

There are no input parameters. The existence of this keyword suppresses the output of the full deck restart files d3full and runfull.

*CONTROL_MPP_IO_NOTIEDIO

Purpose: Store the tied_nodes temp files in memory instead of outputting to disk to avoid problems on distributed file systems. This keyword has no input parameters.

Remarks:

1. **PFILE.** The related command in the partition file (pfile) is: notiedio.

*CONTROL_MPP_IO_SWAPBYTES

Purpose: Swap bytes on some of the output files.

There are no input parameters. The existence of this keyword causes the d3plot file and the "interface component analysis" file to be output with bytes swapped. This is to allow further processing of data on a different machine that has big endian vs. little endian incompatibilities compared to the system on which the analysis is running.

*CONTROL_MPP_MATERIAL_MODEL_DRIVER

Purpose: Enable this feature in MPP mode. To allow MPP reader to pass the input phase even without any nodes and elements but using only one processor.

*CONTROL_MPP_PFILE

Purpose: Provide keyword support for the MPP "p=" pfile options

All lines of input up to the next keyword card will be copied to a temporary file which is effectively pre-pended to the "p=" file given on the command line (even if no such file is given). This allows all options available via the "p=" file to be specified in the keyword input. The only restriction is that pfile directives in the "directory" section are not available, as those must be processed before the keyword input file is read. See the "LS-DYNA MPP User Guide" in the appendix for details of the available pfile commands and their syntax.

*CONTROL_MPP_REBALANCE

Purpose: Specify parameters associated with dynamic load balancing

Card 1	1	2	3	4	5	6	7	8
Variable	NCYCLE	ICOOR	ICOST	THRES				
Type	I	I	I	I				
Default	optional	0	0	1				

VARIABLE	DESCRIPTION
NCYCLE	Number of cycles between rebalance steps
ICOOR	Coordinates used in rebalance: EQ.0: Current coordinates NE.0: Coordinates at $t = 0$
ICOST	Element costs used in rebalance: EQ.0: Time costs EQ.1: Original
THRES	Percent threshold for rebalancing when performing in-core adaptivity (see Remark 1). For in-core adaptivity, only include this field.

Remarks:

1. **In-Core Adaptivity Threshold.** After each adaptive step, the percent increase in the number of shell elements since the last rebalance is computed on each processor. If the maximum of these values minus the minimum exceeds this threshold, then a rebalance is performed.

2. **Supported Features.** This keyword is supported for the features listed below. Models that contain features not included in this list will have unpredictable results.

   - Nodes
   - Shells, solids, beams, thick shells, and discrete elements

- Seatbelt features (sliprings, retractors, etc.)
- Contact except SOFT = 2, CONSTRAINT, and beam
- Control volumes
- Velocity boundary conditions
- Nodal SPCs
- Files: d3plot, elout, nodout, bndout, glstat, abstat, deforc, jntforce, rcforc. sbtout, and sleout
- Adaptive constraints
- In-core adaptivity
- Rigid walls
- Joints
- Nodal rigid bodies
- Pressure loads
- Loads applied to nodes
- Lumped masses
- Cross sections
- Constrained sets (welds)

*CONTROL_NONLOCAL

Purpose: Allocate additional memory for *MAT_NONLOCAL option.

Card 1	1	2	3	4	5	6	7	8
Variable	MEM							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
MEM	This parameter is no longer used and any value input will be ignored. Memory is allocated dynamically as needed.

*CONTROL_OUTPUT

Purpose: Set miscellaneous output parameters.

Card Summary:

Card 1. This card is required.

NPOPT	NEECHO	NREFUP	IACCOP	OPIFS	IPNINT	IKEDIT	IFLUSH

Card 2. This card and all remaining cards are optional.

IPRTF	IERODE	TET10S8	MSGMAX	IPCURV	GMDT	IP1DBLT	EOCS

Card 3. This card is optional.

TOLEV	NEWLEG	FRFREQ	MINFO	SOLSIG	MSGFLG	CDETOL	IGEOM

Card 4. This card is optional.

PHSCHNG	DEMDEN	ICRFILE	SPC2BND	PENOUT	SHLSIG	HISNOUT	ENGOUT

Card 5. This card is optional.

INSF	ISOLSF	IBSF	ISSF	MLKBAG	KINENG	ISFCNT	

Card 6. This card is optional

IELOGKEY	IELOGINI	IELOGSOL					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NPOPT	NEECHO	NREFUP	IACCOP	OPIFS	IPNINT	IKEDIT	IFLUSH
Type	I	I	I	I	F	I	I	I
Default	0	0	0	0	0.	0	100	5000

VARIABLE	DESCRIPTION
NPOPT	Print suppression during input phase flag for the d3hsp file:

VARIABLE	DESCRIPTION
	EQ.0: No suppression, EQ.1: Nodal coordinates, element connectivities, rigid wall definitions, nodal SPCs, initial velocities, initial strains, adaptive constraints, and SPR2/SPR3 constraints are not printed.
NEECHO	Print suppression during input phase flag for echo file: EQ.0: All data printed, EQ.1: Nodal printing is suppressed, EQ.2: Element printing is suppressed, EQ.3: Both nodal and element printing is suppressed.
NREFUP	Flag to update reference node coordinates for beam formulations 1, 2, and 11. This option requires that each reference node is unique to the beam: EQ.0: Do not update reference node. EQ.1: Update reference node. This update is required for proper visualization of the beam cross-section orientation in LS-PrePost beyond the initial ($t = 0.0$) plot state. NREFUP does not affect the internal updating of the beam cross-section orientation in LS-DYNA.
IACCOP	Flag to average or filter nodal accelerations output to file nodout and the time history database d3thdt: EQ.0: No average (default), EQ.1: Averaged between output intervals, EQ.2: Accelerations for each time step are stored internally and then filtered over each output interval using a filter from General Motors [Sala, Neal, and Wang, 2004] based on a low-pass Butterworth frequency filter. See also [Neal, Lin, and Wang, 2004]. DT2MS in *CONTROL_TIMESTEP must be set to a negative value when IACCOP = 2 so that the maximum possible number of time steps for an output interval is known and adequate memory can be allocated.
OPIFS	Output time interval for interface file written per *INTERFACE_COMPONENT_*OPTION*.

VARIABLE	DESCRIPTION
IPNINT	Flag controlling output of initial time step sizes for elements to d3hsp: EQ.0: 100 elements with the smallest time step sizes are printed. EQ.1: Time step sizes for all elements are printed. GT.1: IPNINT elements with the smallest time step sizes are printed.
IKEDIT	Problem status report interval steps to the d3hsp file. This flag is ignored if the glstat file is written; see *DATABASE_GLSTAT.
IFLUSH	Number of time steps interval for flushing I/O buffers. The default value is 5000. If the I/O buffers are not emptied and an abnormal termination occurs, the output files can be incomplete. The I/O buffers for restart files are emptied automatically whenever a restart file is written so these files are not affected by this option.

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	IPRTF	IERODE	TET10S8	MSGMAX	IPCURV	GMDT	IP1DBLT	EOCS
Type	I	I	I	I	I	F	I	I
Default	0	0	2	50	0	0.	0	0

VARIABLE	DESCRIPTION
IPRTF	Default print flag for rbdout and matsum files. This flag defines the default value for the print flag which can be defined in the part definition section; see *PART. This option is meant to reduce the file sizes by eliminating data which is not of interest. EQ.0: Write part data into both matsum and rbdout EQ.1: Write data into rbdout file only EQ.2: Write data into matsum file only EQ.3: Do not write data into rbdout and matsum
IERODE	Output the eroded internal and kinetic energies into the matsum file. Also, (1) under the heading of part ID 0 in matsum, output the

VARIABLE	DESCRIPTION
	kinetic energy from nonstructural mass, lumped mass elements, and lumped inertia elements; and (2) under the heading of part ID -1in matsum, output the kinetic energy associated with distributed mass from *ELEMENT_MASS_PART. See Remark 3. EQ.0: Do not output extra data. EQ.1: Output the eroded internal and kinetic energies and other additional kinetic energies.
TET10S8	Output ten connectivity nodes for the 10-node solid tetrahedral (solid formulations 16/17) and the eight connectivity nodes for the 8-node shell (shell formulation 23) into the d3plot, d3part, d3eigv, and d3mode databases. The current default is set to 2 since this change in the databases may make the data unreadable for many popular post-processors and older versions of LS-PrePost. EQ.1: Write the full node connectivity into the databases EQ.2: Write only the corner nodes of the elements into the databases
MSGMAX	Maximum number of each error/warning message: GT.0: Number of messages to screen output; all messages written to d3hsp/messag LE.0: Number of messages to screen output and d3hsp/messag EQ.0: Default, 50
IPCURV	Flag to output digitized curve data to messag and d3hsp files: EQ.0: Off EQ.1: On
GMDT	Output interval for recorded motions from *INTERFACE_SSI_AUX
IP1DBLT	Output information of 1D (bar-type) seatbelt created for 2D (shell-type) seatbelt to sbtout. EQ.0: The analysis results of internally created 1D seatbelts are extracted and processed to yield the 2D belt information. The 2D belt information is stored in sbtout. EQ.1: The analysis results of internally created 1D retractors and slip rings are stored in sbtout. Belt load can be yielded by

VARIABLE	DESCRIPTION
	*DATABASE_CROSS_SECTION. This might lead to different results from that of IP1DBLT = 0 in MPP if the model is not robust. EQ.2: Same as IP1DBLT = 1, but the model is decomposed in the same way as IP1DBLT = 0 in MPP, which guarantees result consistency.
EOCS	elout Coordinate System: controls the coordinate system to be used when writing out shell data to the elout file. EOCS has no effect on eloutdet. EOCS has no effect on elout if OPTION2 in *DATABASE_ELOUT is greater than zero. EQ.0: Default (local element coordinate system, or if an orthotropic material model and CMPFLG = 1, then material coordinate system) EQ.1: Local element coordinate system EQ.2: Global coordinate system

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	TOLEV	NEWLEG	FRFREQ	MINFO	SOLSIG	MSGFLG	CDETOL	IGEOM
Type	I	I	I	I	I	I	F	I
Default	2	0	1	0	0	0	10.0	1

VARIABLE	DESCRIPTION
TOLEV	Timing Output Levels: controls the number of levels output in the timing summary at termination. The default is 2.
NEWLEG	New Legends: controls the format of the LEGEND section of various ASCII output files. EQ.0: Use the normal format EQ.1: Use the optional format with extra fields.
FRFREQ	Output frequency for failed element report in cycles. The default is to report the summary every cycle in which an element fails.

VARIABLE	DESCRIPTION
	If > 1, the summary is reported every FRFREQ cycles whether an element fails that cycle or not, provided some element has failed since the last summary report. Individual element failure is still reported as it occurs.
MINFO	Output penetration and memory information for Mortar contact after each implicit step. Reporting this information is also available for explicit analysis when MINFO is a negative number. LT.0: Report for each contact interface at the time frequency |MINFO|. This is available for explicit and implicit analysis. EQ.0: No information reported. EQ.1: Report for each contact interface at each implicit time step. No information is output for explicit analysis.
SOLSIG	Flag to extrapolate stresses and other history variables for multi-integration point solids from integration points to nodes. These extrapolated nodal values replace the integration point values normally stored in d3plot. When a nonzero SOLSIG is invoked, NINTSLD in *DATABASE_EXTENT_BINARY should be set to 8 as any other value of NINTSLD will result in only one value being reported for each element. Supported solid formulations are -1, -2, 2, 3, 4, 16, 17, -18, 18, 23, and 62. EQ.0: No extrapolation. EQ.1: Extrapolate the stress for linear materials only. EQ.2: Extrapolate the stress if plastic strain is zero. EQ.3: Extrapolate the stress always. EQ.4: Extrapolate all history variables. **NOTE:** Do not use "Setting – Extrapolate" in LS-Pre-Post when this field, SOLSIG, is nonzero.
MSGFLG	Flag for writing detailed error/warning message to d3msg. MSGFLG has no effect on output of standard length error/warning messages; such messages are written to messag or mes****. NOTE: Most errors/warnings offer only standard length messages. Only a few also offer optional, detailed messages. EQ.0: Do not write detailed messages to d3msg.

VARIABLE	DESCRIPTION
	EQ.1: Write detailed messages to d3msg at the conclusion of the run. Each detailed message is written only once even in cases where the associated error or warning occurs multiple times. A detailed message written to d3msg should only be used to help interpret the standard-length message better. The information in d3msg could contain fictitious IDs and names.
CDETOL	Tolerance for output of *DEFINE_CURVE discretization warnings. After each curve is discretized, the resulting curve is evaluated at each of the original definition points, and the values compared. A warning will be issued for any curve where this comparison results in an error of more than CDETOL/100 × M, where the curve-specific value M is computed as the median of the absolute values of the non-zero curve values.
IGEOM	Flag to control whether nodal coordinates or displacements for the nodes in the mesh are output to d3plot, d3part, and d3drlf: EQ.1: Nodal coordinates (default) EQ.2: Displacements. IGEOM = 2 is useful when the precision of the binary files is insufficient to accurately calculate the displacements from the coordinates in post-processing. This problem arises when the displacements are very small relative to the coordinate values. Note that IGEOM = 2 is supported for post-processing in LS-PrePost 4.10 and later.

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	PHSCHNG	DEMDEN	ICRFILE	SPC2BND	PENOUT	SHLSIG	HISNOUT	ENGOUT
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
PHSCHNG	Message to messag file when materials 216, 217, and 218 change phase.

VARIABLE	DESCRIPTION
	EQ.0: No message (default). EQ.1: The time and element ID are written.
DEMDEN	Output DEM density data to d3plot database. EQ.0: No output (default). EQ.1: Output data. EQ.2: Output data with modification of the boundary density calculation to avoid the low-density distribution near the DES domain boundary.
ICRFILE	Flag to output node sets and element sets used in computing secforc data; see *DATABASE_CROSS_SECTION_*OPTION* and *DATABASE_SECFORC. These sets are written in keyword format (*SET_...) and thus can be displayed using LS-PrePost. The assigned set IDs are the same as the ID of the cross-section. EQ.0: Do not write sets (default). EQ.1: Write a separate file for each cross-section called cross_section_# where # is the cross-section ID. EQ.2: Write sets for all cross-sections to a file called cross_sections.
SPC2BND	Flag to convert constraints on rigid bodies (see CMO/CON1/CON2 on *MAT_RIGID and *CONSTRAINED_NODAL_RIGID_BODY_SPC) to equivalent *BOUNDARY_PRESCRIBED_MOTION_RIGID motion. The purpose of this field is to obtain reaction forces in bndout, without changing the results of the simulation. EQ.0: Not active EQ.1: Active
PENOUT	Flag to output contact penetration to sleout (binout format only) and d3plot for Mortar contact. In sleout the maximum absolute and/or relative penetration per interface is output, in magnitude only. In d3plot a nodal vector field is output for absolute and/or relative penetration, respectively, each giving the maximum penetration (magnitude and direction) for all nodes in any sliding interface. See also NPEN on *DATABASE_EXTENT_INTFOR. EQ.0: Do not output.

VARIABLE	DESCRIPTION
	EQ.1: Output absolute penetration.
	EQ.2: Output absolute and relative penetration.
SHLSIG	Flag to extrapolate stresses for shells with 8 integration points to nodes. These extrapolated nodal values replace the integration point values normally stored in d3plot. When a nonzero SHLSIG is invoked, MAXINT in *DATABASE_EXTENT_BINARY should be set to -2 to indicate 4 in-plane integration points and 2 in the thickness direction. Supported shell formulations are: 16, 20, and 21.
	EQ.0: No extrapolation.
	EQ.1: Extrapolate the stress for linear materials only.
	EQ.2: Extrapolate the stress if plastic strain is zero.
	EQ.3: Extrapolate the stress always.
	EQ.4: Extrapolate all history variables.
	NOTE: Do not use "Setting – Extrapolate" in LS-PrePost when this field, SHLSIG, is nonzero. SHLSIG is only supported for Guass integration.
HISNOUT	Flag to invoke output of extra history variable names. Usually, the extra history variables of material models are given as just numbers. The corresponding meaning of these variables can be determined, for example, using this website: https://lsdyna.ansys.com/history-variables-for-certain-material-models/. As an alternative, this option allows the output of the history variable names, listed for each part separately, to d3hsp. In addition, XML files that can be read by a post-processor can be output. Currently, two XML files are available: • hisnames.xml (read by LS-PrePost and GS Animator), which lists the history variable names in each part (as with d3hsp), and • d3labels.xml (read by LS-PrePost), which is similar to hisnames.xml but contains additional information regarding history variable names for each element and integration

VARIABLE	DESCRIPTION
	point when necessary (such as in the case of composites). The number of supported material models is continuously increasing. EQ.0: No output (default) EQ.1: Information written to d3hsp EQ.2: Information written to d3hsp and XML file hisnames.xml EQ.3: Information written to d3hsp and XML file d3labels.xml (and hisnames.xml).
ENGOUT	Flag to output contact sliding energy densities to d3plot for Mortar contact. If set to 1, a nodal scalar field is output giving the minimum sliding energy density for each node in any sliding interface. See also NENG on *DATABASE_EXTENT_INTFOR.

This card is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	INSF	ISOLSF	IBSF	ISSF	MLKBAG	KINENG	ISFCNT	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	1	

VARIABLE	DESCRIPTION
INSF	Flag to invoke output of *SET_NODE data: EQ.0: No output (default) EQ.1: Information written to file. See Remark 1.
ISOLSF	Flag to invoke output of *SET_SOLID data: EQ.0: No output (default) EQ.1: Information written to file. See Remark 1.
IBSF	Flag to invoke output of *SET_BEAM data: EQ.0: No output (default) EQ.1: Information written to file. See Remark 1.

VARIABLE	DESCRIPTION
ISSF	Flag to invoke output of *SET_SHELL data: EQ.0: No output (default) EQ.1: Information written to file. See Remark 1.
MLKBAG	Flag to invoke output of accumulated airbag mass leakage in ABSTAT: EQ.0: Airbag mass leakage rate is output (default). EQ.1: Accumulated airbag mass leakage is output.
KINENG	Flag to output kinetic energy density as a nodal field: EQ.0: Do not output (default). EQ.1: Output, will appear as "kinetic energy density" as a nodal field in the d3plot and d3eigv databases.
ISFCNT	Continuity level in applying Interface Linking Data: LE.1: Continuity in displacements. Velocities and accelerations excluded. EQ.2: Continuity in displacements and velocities. Accelerations excluded. EQ.3: Continuity in displacements, velocities, and accelerations. This option may result in a smoother response overall.

This card is optional.

Card 6	1	2	3	4	5	6	7	8
Variable	IELOGKEY	IELOGINI	IELOGSOL					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
IELOGKEY	Flag to invoke diagnostic output to a consolidated error file for MPP called error.log during the keyword input phase (see Remark 2):

VARIABLE	DESCRIPTION
	EQ.0: No output during this phase (default) EQ.1: Output errors only EQ.2: Output errors and warnings EQ.3: Output errors, warnings, and informational output
IELOGINI	Flag to invoke diagnostic output to a consolidated error file for MPP called error.log during the initialization phase (see Remark 2): EQ.0: No output during this phase (default) EQ.1: Output errors only EQ.2: Output errors and warnings EQ.3: Output errors, warnings, and informational output
IELOGSOL	Flag to invoke diagnostic output to a consolidated error file for MPP called error.log during the solution phase (see Remark 2): EQ.0: No output during this phase (default) EQ.1: Output errors only EQ.2: Output errors and warnings EQ.3: Output errors, warnings, and informational output

Remarks:

1. **Set output format.** Set data is written to an output file based on the type of set (see table below). Note that the output file has the job ID as a prefix.

Set type	Output file name	Set prefix(ID) in the file
node	nodeset_file	SetN
solid	solidset_file	SetH
beam	beanset_file	SetB
shell	shellset_file	SetS

2. **Consolidated error file.** Nonzero IELOGKEY, IELOGINI, or IELOGSOL cause all processors to output diagnostic data in MPP to a single file named error.log for the corresponding phase. Depending on the setting, different information can be output to this file. By default, error.log is not created.

3. **Energy in glstat and matsum.** The internal and kinetic energies in glstat always include the eroded internal and kinetic energies. In contrast, the internal and kinetic energies in matsum depend on IERODE. Thus, setting IERODE to 0 causes the energies in glstat to differ from that in matsum. This difference is intentional because glstat reflects the global system energy, while in matsum, the IERODE flag dictates outputting the internal and kinetic energies of interest.

*CONTROL_PARALLEL

Purpose: Control parallel processing usage by defining the number of processors and invoking the optional consistency of the global vector assembly. This command applies only to shared memory parallel (SMP) LS-DYNA. It does not apply to distributed memory parallel (MPP) LS-DYNA.

Card 1	1	2	3	4	5	6	7	8
Variable	NCPU	NUMRHS	CONST	PARA				
Type	I	I	I	I				
Default	1	0	2	0				
Remarks		1	2	3				

VARIABLE	DESCRIPTION
NCPU	Number of cpus used. (This parameter is disabled in 971 R5 and later versions. Set number of cpus using "ncpu =" on the execution line — see Execution Syntax section of Getting Started — or on the *KEYWORD line of the input.)
NUMRHS	Number of right-hand sides allocated in memory: EQ.0: Same as NCPU, always recommended, EQ.1: Allocate only one.
CONST	Consistency flag. (Including "ncpu = *n*" on the execution line or on the *KEYWORD line of input overrides CONST. The algebraic sign of *n* determines the consistency setting.) EQ.1 or n < 0: On (recommended) EQ.2 or n > 0: Off, for a faster solution (default).
PARA	Flag for parallel force assembly if CONST = 1. (Including "para =" on the execution line overrides PARA.) EQ.0: Off EQ.1: On (see Remark 3)

VARIABLE	DESCRIPTION
	EQ.2: On (see Remark 3)

Remarks:

1. **NUMHRS.** It is recommended to always set NUMRHS = NCPU because then the force assembly is done in parallel which greatly improves the parallel performance. Setting NUMRHS to one reduces storage by one right-hand side vector for each additional processor after the first. If the consistency flag is active, that is, CONST = 1, NUMRHS defaults to unity.

2. **Consistency.** For any given problem with the consistency option off, i.e., CONST = 2, slight differences in results are seen when running the same job multiple times with the same number of processors and also when varying the number of processors. Comparisons of nodal accelerations often show wide discrepancies; however, it is worth noting that the results of accelerometers often show insignificant variations due to the smoothing effect of the accelerometers which are generally attached to nodal rigid bodies.

   The accuracy issues are not new and are inherent in numerical simulations of automotive crash and impact problems where structural bifurcations under compressive loads are common. This problem can be easily demonstrated by using a perfectly square thin-walled tubular beam of uniform cross-section under a compressive load. Typically, every run on one processor that includes a minor input change (i.e., element or hourglass formulation) will produce dramatically different results in terms of the final shape, and, likewise, if the same problem is again run on a different brand of computer. If the same problem is run on multiple processors the results can vary dramatically from run to run WITH NO INPUT CHANGE. The problem here is due to the randomness of numerical round-off which acts as a trigger in a "perfect" beam.

   Since summations will (CONST = 2) occur in a different order from run to run, the round-off is also random. The consistency flag, CONST = 1, provides for identical results (or nearly so) whether one, two, or more processors are used while running in the shared memory parallel (SMP) mode. This is done by requiring that all contributions to global vectors be summed in a precise order independently of the number of processors used. When checking for consistent results, nodal displacements or element stresses should be compared. The NODOUT and ELOUT files should be digit-to-digit identical. However, the GLSTAT, SECFORC, and many of the other ASCII files will not be identical since the quantities in these files are summed in parallel for efficiency reasons and the ordering of summation operations is not enforced. The biggest drawback of this option is the CPU cost penalty which is at least 15 percent if PARA = 0 and is much less if PARA = 1 and 2 or more processors are used. Unless the PARA flag

is on (for non-vector processors), parallel scaling is adversely affected. The consistency flag does not apply to MPP parallel.

3. **PARA.** The PARA flag set to 1 or 2 will cause the force assembly for the consistency option to be performed in parallel for the SMP version, so better scaling will be obtained. However, PARA = 1 will increase memory usage while PARA = 2 will not. This flag does not apply to the MPP version. If PARA = 0, CONST = 0, and NUMRHS = NCPU, the force assembly by default is done in parallel, but without consistency. The value of the flag may also be given by including "para = <value>" on the execution line, and the value given in this manner will override the value of PARA in *CONTROL_PARALLEL.

*CONTROL_PORE_AIR

Purpose: Set parameters for pore air pressure calculations.

Card 1	1	2	3	4	5	6	7	8
Variable	AIR_RHO	AIR_P	ETERM	ANAMSG	OPTAPA	PA_DR		
Type	F	F	F	I	I	I		
Default	none	none	endtim	0	0	0		

VARIABLE	DESCRIPTION
AIR_RHO	Density of atmospheric air, = 1.184 kg/m^3 at 25°C
AIR_P	Pressure of atmospheric air, = 101.325 kPa at 25°C
ETERM	Event termination time. The default is ENDTIM of *CONTROL_TERMINATION. If ETERM is defined and smaller than ENDTIM, the simulation terminates, by default, when the simulation time reaches ETERM. However, OTAPA below provides options for continuing the simulation beyond ETERM.
ANAMSG	Flag to turn off the printing of pore air analysis status message, including the analysis time, the node with the highest pressure change. EQ.0: Status messages are printed, the default value. EQ.1: Status messages are not printed
OPTAPA	Option when 0.0 < ETERM < ENDTIM: EQ.0: The simulation terminates when time reaches ETERM. EQ.1: The simulation continues beyond time ETERM. However, pore air pressure is not updated after ETERM, and the last updated pore air pressure applies for the rest of the simulation. EQ.2: The simulation continues beyond time ETERM, but pore air pressure does not apply after time ETERM.
PA_DR	Flag to turn on pore air flow analysis during dynamic relaxation. Starting with R13, the pore air analysis is disabled by default

VARIABLE	DESCRIPTION
	during dynamic relaxation. This flag enables reproducing results comparable to R12 or earlier versions in which pore air analysis is active during the dynamic relaxation phase.
	EQ.0: Disable pore air analysis is disabled during dynamic relaxation.
	EQ.1: Enable pore air analysis during dynamic relaxation.

*CONTROL_PORE_FLUID

Purpose: Set parameters for pore water pressure calculations.

This control card is intended for soil analysis but is also applicable to any other materials containing pore fluid. The pore-pressure capabilities invoked by this card are available for explicit analysis only, not for implicit, and are restricted to solid element formulations 1, 2, 10, and 15 and thick shell formulations 1, 2, 3, 5, 6 and 7.

LS-DYNA uses Terzaghi's Effective Stress to model materials with pore pressure. The pore fluid and soil skeleton are assumed to occupy the same volume and to carry loads in parallel. Thus, the total stress in an element is the sum of the "effective stress" in the soil skeleton, plus the pressure in the pore fluid. LS-DYNA calculates the "effective stress" with standard material models. The pore fluid treatment, then, is independent of material model. The pore pressure is calculated at nodes and interpolated onto the elements. The pore fluid's hydrostatic stress is equal to the negative of the element pore pressure.

Card 1	1	2	3	4	5	6	7	8
Variable	ATYPE	(blank)	WTABLE	PF_RHO	GRAV	PF_BULK	OUTPUT	TMF
Type	I	F	F	F	F	F	I	F
Default	0	0.0	0.0	none	none	none	0	1.0

Card 2	1	2	3	4	5	6	7	8
Variable	TARG	FMIN	FMAX	FTIED	CONV	CONMAX	ETERM	THERM
Type	F	F	F	F	F	F	F	F
Default	TMF	0.0	0.0	0.0	10^{-4}	10^{20}	0.0	0.0

Card 3 is optional

Card 3	1	2	3	4	5	6	7	8
Variable	ETFLAG							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ATYPE	Analysis type for pore water pressure calculations (see Analysis Types): EQ.0: No pore water pressure calculation EQ.1: Undrained analysis EQ.2: Drained analysis EQ.3: Time dependent consolidation (coupled) EQ.4: Consolidate to steady state (uncoupled) EQ.5: Drained in dynamic relaxation, undrained in transient EQ.6: Same as 4 but do not check convergence, continue to end time
WTABLE	Default z-coordinate of water table (where pore pressure is zero)
PF_RHO	Default density for pore water
GRAV	Gravitational acceleration used to calculate hydrostatic pore water pressure
PF_BULK	Default bulk modulus of pore fluid (stress units)
OUTPUT	Flag controlling stresses output to d3plot, d3thdt, and elout: EQ.0: Total stresses are output. EQ.1: Effective stresses are output; see Output section below.
TMF	Initial Time Magnification Factor for seepage (ATYPE = 3 and 4 only):

VARIABLE	DESCRIPTION
	GT.0: Factor (can be used with automatic control, see TARG, FMIN, FMAX)
	LT.0: Load curve ID (see *DEFINE_CURVE) giving Time Magnification Factor as a function of analysis time
TARG	Target for maximum change of excess pore pressure head at any node, per timestep. Head is defined in Remark 2. If the actual change falls below the target, the time factor for the seepage calculation will be increased (see Time Factoring). If zero, the constant value of TMF is used. If non-zero, TMF is taken as the initial factor.
FMIN	Minimum time factor for seepage calculation
FMAX	Maximum time factor for seepage calculation
FTIED	Analysis type for pore water pressure calculations (see Remark 1): EQ.0.0: Tied contacts act as impermeable membranes, EQ.1.0: Fluid may flow freely through tied contacts.
CONV	Convergence tolerance for ATYPE = 4. Maximum head change per time step at any node. See Remark 2 below for the definition of "head".
CONMAX	Maximum factor on permeability for ATYPE = 4
ETERM	Event time termination (ATYPE = 3)
THERM	Thermal expansion: Volumetric strain per degree increase for undrained soil.
ETFLAG	Flag for interpretation of time (see Time Factoring): EQ.0: Time means analysis time. EQ.1: Time means event time.

Analysis Types (see ATYPE):

1. **Undrained.** For analyses of the "undrained" type the pore fluid is trapped within the material. Volume changes result in pore pressure changes. This approximation is used to simulate the effect of rapidly applied loads on relatively impermeable soil.

2. **Drained.** For analyses of the "drained" type the pore fluid is free to move within the material such that the user-defined pressure as a function of z-coordinate relationship is always maintained. This approximation is used to model high-permeability soils.

3. **Time-Dependent Consolidation.** For the analysis type "time dependent consolidation" pressure gradients cause pore fluid to flow through the material according to Darcy's law:

$$\mathbf{v} = \kappa \nabla (p + z)$$

where

$\mathbf{v}$ = fluid seepage velocity vector
κ = permeability
p = pressure head
z = z-coordinate.

The seepage velocity is defined as fluid volume flow rate per unit cross-sectional area. Net inflow or outflow at a node leads to a theoretical volume gain or loss. The analysis is coupled, that is, any difference between actual and theoretical volume leads to pore pressure change, which in turn affects the fluid flow. The result is a prediction of response as a function of time.

4. **Steady-State Consolidation.** For the analysis type "steady-state consolidation" an iterative method is used to calculate the steady-state pore pressure. The analysis is uncoupled, that is, only the final state is meaningful, not the response as a function of time.

Time Factoring:

Consolidation occurs over time intervals of days, weeks, or months. To simulate this process using explicit time integration, a time factor is used. The permeability of the soil is increased by the time factor so that consolidation occurs more quickly. The output times in d3plot, d3thdt, and the ascii files are modified to reflect the time factor. The factored time ("Event Time") is intended to represent the time taken in the real-life consolidation process and will usually be much larger than the analysis time (the analysis time is the sum of the LS-DYNA timesteps). The time factor may be chosen explicitly (using TMF), but we recommend using automatic factoring instead. The automatic scheme adjusts the time factor according to how quickly the pore pressure is changing. Usually at the start of consolidation the pore pressure changes quickly, and the time factor is low. The time factor increases gradually as the rate of pore pressure change reduces. Automatic time factoring is input by setting TARG (the target pore pressure head change per timestep) and maximum and minimum allowable time factors; for example, TARG = 0.001 to 0.01 m head, FMIN = 1.0, and FMAX = 10^6. Optimum settings for these are model-dependent.

Loading, other input data from load curves, and output time-intervals on *DATABASE cards use the analysis time by default (for example, the *x*-axis of a load curve used for pressure loading is analysis time). When performing consolidation with automatic time-factoring, the relationship between analysis time and event time is unpredictable. Termination based on event time may be input using ETERM.

It may also be desired to apply loads as functions of event time rather than analysis time, since the event time is representative of the real-life process. By setting ETFLAG = 2, the time axis of all load curves used for any type of input as a function of time, and output intervals on *DATABASE cards, will be interpreted as event time. This method also allows consolidation to be used as part of a staged construction sequence – when ETFLAG = 2, the stages begin and end at the "real time" stage limits and input curves of pore pressure analysis type as a function of time may be used to enforce, for example, consolidation in some stages and undrained behavior in others.

Output:

Five extra variables for solid elements are automatically written to the d3plot and d3thdt files when the model contains *CONTROL_PORE_FLUID, in addition to the NEIPH variables requested on *DATABASE_EXTENT_BINARY. The first of the five is the pore pressure in stress units; the second is the excess pore pressure, meaning actual minus hydrostatic pressure. These follow the NEIPH extra variables requested by the user, so for example if NEIPH = 3 then there will be a total of 8 extra variables in the d3plot and d3thdt files of which the fourth is pore pressure. Even if NEIPH = 0, the d3plot and d3thdt files will still contain the five extra variables related to pore pressure. The same five extra variables are also written to the elout file, but only if OPTION1 > 0 on *DATABASE_ELOUT.

Further optional output to d3plot, d3thdt, and nodout files is available for nodal pore pressure variables; see *DATABASE_PWP_OUTPUT.

For time-dependent and steady-state consolidation, information on the progress of the analysis is written to d3hsp file.

Remarks:

1. **Tied Contacts.** By default, the mesh discontinuity at a tied contact will act as a barrier to fluid flow. If the flag FTIED is set to 1.0, then pore fluid will be transmitted across tied nodes in tied contacts (*CONTACT_TIED_SURFACE_TO_SURFACE and *CONTACT_TIED_NODES_TO_SURFACE, including OFFSET and non-OFFSET types). This algorithm has an effect only when the analysis type of at least one of the contacting parts is 3, 4 or 6.

2. **Pressure and Head.** The term "head" means pressure, defined in terms of the height of fluid needed to generate that pressure hydrostatically. Head is given in length units. Head relates to pressure measured in stress units as follows:

$$h = \frac{p}{\rho g}$$

where h is head (in length units), p is pressure (in stress units), ρ is pore fluid density (PF_RHO), and g is gravitational acceleration (GRAV).

3. **See also *BOUNDARY_PORE_FLUID.** Only the parts for which *BOUNDARY_PORE_FLUID is defined will be treated as containing pore fluid. Parts not included in a *BOUNDARY_PORE_FLUID definition will be treated as dry, meaning they do not have pore fluid; the settings on *CONTROL_PORE_FLUID will not affect them. Pore fluid properties, water table and analysis type are input on a per part basis on *BOUNDARY_-PORE_FLUID. Nonzero values given on *BOUNDARY_PORE_FLUID override those on *CONTROL_PORE_FLUID for that particular part.

4. **Material Properties and Density.** For parts containing pore fluid, the properties given on the *MAT card determine the effective stresses. For example, the bulk modulus on the *MAT card should not include the bulk stiffness of the pore water. The only material property on the *MAT card that relates to the soil/water mixture (as opposed to the soil skeleton) is the density, RO, for which the density of saturated soil should be input. This is somewhat higher than the density of dry soil. The mass and weight of the soil are determined solely from the RO on the *MAT card, and not from the pore fluid density PF_RHO given on *CONTROL_PORE_FLUID or *BOUNDARY_PORE_FLUID.

5. **Part Associativity.** Pore pressure is a nodal variable, but analysis type and other pore pressure related inputs are properties of parts. When a node is shared by elements of different parts, and those parts have different pore pressure inputs, the following rules are followed to determine which part's properties should be applied to the node.

   a) Dry parts (meaning parts without a *BOUNDARY_PORE_FLUID card) will never be used (lowest priority).

   b) If a part is initially dormant (due to staged construction inputs), it has next-lowest priority.

   c) Parts with analysis type set to drained have highest priority.

   d) Next, higher permeability gives higher priority.

   e) If two or more parts have equal-highest priority at a node, the part with lowest ID will win.

6. **Related Keywords.** The following are cards related to this keyword:

*BOUNDARY_PORE_FLUID

BOUNDARY_PWP_*OPTION

*DATABASE_PWP_OUTPUT

*DATABASE_PWP_FLOW

*MAT_ADD_PERMEABILITY

*CONTROL_PZELECTRIC

Purpose: Set solver options for a piezoelectric material; see *MAT_ADD_PZELECTRIC.

Card 1	1	2	3	4	5	6	7	8
Variable	SOLVER	MSGITR	MAXITR	ABSTOL	RELTOL	NDTRFK	EPZMSG	
Type	I	I	I	F	I	I	I	
Default	11	0	500	10^{-20}	10^{-10}	1	0	

VARIABLE	DESCRIPTION
SOLVER	Piezoelectric solver type: EQ.11: Direct solver EQ.12: Diagonal scaling conjugate gradient iterative, recommended for MPP for better scalability
MSGITR	Output iteration message level for SOLVER = 12: EQ.0: No output (default) EQ.1: Summary information
MAXITR	Maximum number of iterations for SOLVER = 12. EQ.0: Use default value 100.
ABSTOL	Absolute convergence tolerance, for SOLVER = 12. EQ.0.0: Use default value 10^{-20}.
RELTOL	Relative convergence tolerance, for SOLVER = 12. EQ.0.0: Use default value 10^{-10}.
NDTRFK	Reform the dielectric stiffness matrix every NDTRFK time steps. LT.0: Curve |NDTRFK| defines the stiffness reformation time step as a function of time.
EPZMSG	Flag to determine if electric flux and electric field at the element center of piezoelectric material is output to d3plot: EQ.0: No electric flux or electric field output to d3plot

VARIABLE	DESCRIPTION
	EQ.1: x, y, and z strain slots in d3plot store the electric flux along the x, y, and z directions, respectively. xy, yz, and zx strain slots in d3plot store the electric field along the x, y, and z directions, respectively.

*CONTROL_REFERENCE_CONFIGURATION_{OPTION}

Available options include:

<BLANK>

ITER

Purpose: Find an approximate reference geometry given a measured/known deformed geometry. A solution to this inverse problem is found iteratively by means of an optimization method. This keyword needs to be in the main keyword file, not in any of the include files. case must be used on the command line (see Remark 2).

Card 1	1	2	3	4	5	6	7	8
Variable	MAXITER	TARGETFILE						
Type	I	A70						
Default	none	none						

Card 2	1	2	3	4	5	6	7	8
Variable	METHOD	STEP	TOL					
Type	I	F	F					
Default	0	1.0	0.0					

Optional card for the ITER keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	ITERFILE							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
MAXITER	Maximum number of iterations. See Remark 1.
TARGET-FILE	File containing all nodes of the target geometry. See Remark 2.
METHOD	Iterative method: EQ.1: Sellier's method EQ.2: Rausch's method EQ.3: Rausch's method with an additional line search.
STEP	Step size used in the iterations to update the current approximate reference geometry for Sellier's method. It must be > 0. See Remark 1.
TOL	Tolerance used to determining convergence of the iterative method. This is given in the unit of length. See Remark 1.
ITERFILE	Base name of two files for the ITER keyword option. These files are used to start, or restart, the iterative method. ITERFILE.guess gives an initial guess of the approximate stress free reference geometry. It has the same format as the TARGET file. The second file, ITERFILE.algo, gives algorithmic parameters and uses an internal format. ITERFILE is optional. If ITERFILE is not supplied, TARGET is used as an initial guess. See Remark 2 and 3 for details.

Remarks:

1. **Iterative Method.** The implemented iterative method is a backward displacement method of either Sellier [1] (METHOD = 1) or Rausch et.al. [2] (METHOD = 2 or 3). It attempts to solve the inverse problem by a sequence of so-called forward solves. In each forward solve the underlying mechanical problem is solved. The reference geometry is then adjusted by subtracting a suitable amount of the obtained displacement.

   Let $\boldsymbol{x}^*$ be the target geometry specified by TARGETFILE. Then we want to find the unknown reference geometry $\boldsymbol{X}^*$. We can state the inverse problem as a minimization of the difference $\mathfrak{I}(\boldsymbol{X}) = \|\boldsymbol{x} - \boldsymbol{x}^*\|$ under the constraint that $\boldsymbol{x}$ must be in equilibrium with respect to the applied load. In iteration n of the optimization method, we have a guess of the reference geometry, $\boldsymbol{X}_n$. For $n = 0$ this geometry is given by ITERFILE.guess if specified; otherwise it is given by TARGETFILE.

The stated problem provides the solution x_n as the resulting deformed geometry when applying the prescribed load. It can be shown that an approximate descent direction for $\mathfrak{I}$ yields the update $\boldsymbol{X}_{n+1} = \boldsymbol{X}_n - \alpha_n(\boldsymbol{x}_n - \boldsymbol{x}^*)$. α_n is the step size STEP for Sellier's method, while for Rausch's method it is given as

$$\alpha_n = \alpha_{n-1} \times \frac{(\boldsymbol{x}_{n-1} - \boldsymbol{x}^*)^T(\boldsymbol{x}_n - \boldsymbol{x}_{n-1})}{(\boldsymbol{x}_n - \boldsymbol{x}_{n-1})^T(\boldsymbol{x}_n - \boldsymbol{x}_{n-1})} \ .$$

For METHOD = 3, a line search is also attempted if the error $\boldsymbol{x}_n - \boldsymbol{x}^*$ does not decrease at iteration n. It amounts to bisecting the step at iteration n into $\alpha_n/2^k$ and re-solving with $k = 1,2,3,\ldots$ until either the error decreases or $k = 5$.

For all METHOD types, the iterations continue until $\mathfrak{I}(\boldsymbol{X}_n)$ is smaller than the tolerance TOL, or until the maximum number of iterations MAXITER is exceeded. In the implementation, $\boldsymbol{x}$ and $\boldsymbol{X}$ are vectors of nodal positions.

2. **Iteration using Cases.** The overall iterative structure is based on the *CASE construct. Starting from a main keyword file (*.key, *.k, *.dyn) which acts as a template, the iterates are automatically generated by parsing this input file. Each iterate is a "case" with a corresponding *.inp file containing the necessary keywords. The input for iterate number n is named jobid.iter{n}.inp, where jobid is an optional job ID. This is essentially a copy of the main keyword file, but with minor modifications reflecting input that is unique to iterate n.

   You must have *CONTROL_REFERENCE_CONFIGURATION in the main keyword file. For disk space reasons we recommend reducing the size of the main keyword file to a minimum. The modifications to the main keyword file chiefly pertain to change of node coordinates for the current iterate and associated iteration parameters. At the end of the iteration jobid.iter{$n+1$}.guess and jobid.iter{$n+1$}.algo are written with the updated (reference) geometry and algorithmic parameters, respectively. These files are used as starting guess in the next iterate, $n+1$.

3. **Node Files.** The TARGETFILE and ITERFILE.guess files both contain the nodes subjected to optimization and have the same simple structure

```
*KEYWORD
*NODE
$#   nid               x               y               z
       1         -11.776         -23.849          14.234
…
*END
```

   The ITERFILE.algo files have an internal format whose description is currently omitted.

Example 1:

Below is a sketch of a main file showing the use of the *CONTROL_REFERENCE_CONFIGURATION keyword. To run it, the syntax on the command line should be: lsdyna case i=main.k

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
*KEYWORD
*CONTROL_REFERENCE_CONFIGURATION
$# maxiter                                                      target node file
         4target.k
$#  method      step       tol
         1       1.0     0.001
$ all nodes (ie.  target nodes as well as any other nodes)
*INCLUDE
target.k
*ELEMENT_SOLID
$#   eid     pid      n1      n2      n3      n4      n5      n6      n7      n8
       1       1       1     127     136      10       2     128     137      11
*CONTROL_IMPLICIT_GENERAL
$#  imflag       dt0    imform      nsbs       igs     cnstn      form    zero_v
         1       0.1         2         0         1         0         0         0
*CONTROL_TERMINATION
$#  endtim    endcyc     dtmin    endeng    endmas     nosol
       1.0         0       0.0       0.0       0.0         0
*BOUNDARY_SPC_NODE
$#     nid       cid      dofx      dofy      dofz     dofrx     dofry     dofrz
        30         0         1         1         1         1         1         1
*SECTION_SOLID
$#   secid    elform       aet
         1        -2         0
*MAT_ELASTIC
$#     mid        ro         e        pr        da        db  not used
         11.00000E-0     100.0       0.3       0.0       0.0       0.0
*LOAD_SEGMENT_ID
$#      id                                                               heading
         0oneseg
$#    lcid        sf        at        n1        n2        n3        n4        n5
         1       1.0       0.0       185       186       192       191         0
*PART
$# title
boxsolid
$#     pid     secid       mid     eosid      hgid      grav    adpopt      tmid
         1         1         1
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

References:

[1] Sellier, M, “An iterative method for the inverse elasto-static problem,” Journal of Fluids and Structures, Volume 27, issue 8, (2011)

[2] Rausch, M.K., Genet M., and Humphrey J.D, “An augmented iterative method for identifying a stress-free reference configuration in image-based biomechanical modeling” Journal of Biomechanics 58 (2017) 227-231.

*CONTROL_REFINE_ALE

Purpose: Refine ALE hexahedral solid elements locally. Each parent element is replaced by 8 child elements with a volume equal to $1/8^{th}$ the parent volume. If only the 1^{st} card is defined, the refinement occurs during the initialization. The 2^{nd} card defines a criterion CRITRF to automatically refine the elements during the run. If the 3^{rd} card is defined, the refinement can be removed if a criterion CRITRM is reached, that is, the child elements can be replaced by their parents.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NLVL	MMSID	IBOX	IELOUT		
Type	I	I	I	I	I	I		
Default	none	0	1	0	0	0		

Remaining cards are optional.†

Automatic refinement card. Optional card for activating automatic refinement whereby each element satisfying certain criteria is replaced by a cluster of 8 child elements (2 × 2 × 2 elements)

Card 2	1	2	3	4	5	6	7	8
Variable	NTOTRF	NCYCRF	CRITRF	VALRF	BEGRF	ENDRF	LAYRF	DELAYRF
Type	I	F	I	F	F	F	I	F
Default	0	0.0	0	0.0	0.0	0.0	0	0.0

Automatic Refinement Remove Card. Optional card for activating automatic refinement removal whereby, when, for a cluster of 8 child elements, certain criteria are satisfied, the cluster (2 × 2 × 2 child elements) is replaced by its parent.

Card 3	1	2	3	4	5	6	7	8
Variable	MAXRM	NCYCRM	CRITRM	VALRM	BEGRM	ENDRM	MMSRM	DELAYRM
Type	I	F	I	F	F	F	I	F
Default	0	0.0	0	0.0	0.0	0.0	0	0.0

VARIABLE	DESCRIPTION
ID	Set ID. LT.0: Parent elements can be hidden in LS-PrePost as they are replaced by their children.
TYPE	Set type: EQ.0: ALE part set EQ.1: ALE part EQ.2: Lagrangian part set coupled to ALE (see Remarks 1 and 2) EQ.3: Lagrangian part coupled to ALE (see Remarks 1 and 2) EQ.4: Lagrangian shell set coupled to ALE (see Remarks 1 and 2) EQ.5: ALE solid set
NLVL	Number of refinement levels (see Remark 3).
MMSID	Multi-Material Set ID (see Remark 4): LT.0: Only ALE elements with all the multi-material groups listed in *SET_MULTI-MATERIAL_GROUP_LIST can be refined (or removed otherwise) GT.0: ALE elements with at least one of the multi-material groups can be refined (or removed)
IBOX	Box ID (See *DEFINE_BOX) defining a region in which the ALE elements are refined. The options LOCAL and ADAPTIVE for *DEFINE_BOX are supported.
IELOUT	Flag to handle child data in elout (see Remark 9).

VARIABLE	DESCRIPTION
NTOTRF	Total number of ALE elements to refine (see Remark 5): GT.0: Number of elements to refine EQ.0: Number of solid elements in IBOX (see Remark 2) EQ.-1: Add clusters of 8 solids for the refinement during the run as needed. LT.-1: |NTOTRF| is the ID of *CONTROL_REFINE_MPP_DISTRIBUTION that computes the number of extra elements required by the processors.
NCYCRF	Number of cycles between each refinement. LT.0: |NCYCRF| is the time interval.
CRITRF	Refinement criterion: EQ.-3: Volume fraction < VALRF EQ.-2: Relative volume (V/V_0) > VALRF EQ.-1: Pressure < VALRF EQ.0: Static refinement (as if only Card 1 is defined) EQ.1: Pressure > VALRF EQ.2: Relative volume (V/V_0) < VALRF EQ.3: Volume fraction > VALRF EQ.5: User defined criterion. The fortran routine alerfn_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRF	Criterion value to reach for the refinement
BEGRF	Time to begin the refinement
ENDRF	Time to end the refinement
LAYRF	Number of element layers to refine around an element satisfying the refinement criterion (see Remark 6).
DELAYRF	Period of time after removing the refinement of an element, during which this element will not be refined again.

VARIABLE	DESCRIPTION
MAXRM	Maximum number of child clusters (2 × 2 × 2 child elements) to remove (see Remark 8): LT.0: For the whole run GT.0: Every NCYCRM cycles
NCYCRM	Number of cycles between each check for refinement removal. LT.0: |NCYCRM| is the time interval.
CRITRM	Criterion for refinement removal (a negative CRITRM reverses the conditions below): EQ.-3: Volume fraction > VALRM EQ.-2: Relative volume (V/V_0) < VALRM EQ.-1: Pressure > VALRM EQ.0: No refinement removal (as if only Cards 1 and 2 are defined) EQ.1: Pressure < VALRM EQ.2: Relative volume (V/V_0) > VALRM EQ.3: Volume fraction < VALRM EQ.5: User defined criterion. The fortran routine alermv_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRM	Criterion value to reach in each child element of a cluster for its removal (child elements replaced by parent element).
BEGRM	Time to begin the check for refinement removal. LT.0: |BEGRM| represents a critical percent of NTOTRF below which the check for refinement removal should begin (0.0 < |BEGRM| < 1.0). See Remark 7.
ENDRM	Time to end the check for refinement removal
MMSRM	Multi-Material Set ID for the refinement removal. See Remark 4. LT.0: |MMSRM| represents the radius of a sphere centered on a newly refined element, in which the refinement can not be removed.

VARIABLE	DESCRIPTION
DELAYRM	Period of time after refining an element, during which this refinement will not be removed

Remarks:

1. **First Card Definition Only and TYPE.** If only the 1st card is defined, only TYPE = 0, 1, and 5 can be defined.

2. **Lagrangian Coupling.** *CONSTRAINED_LAGRANGE_IN_SOLID needs to be defined for TYPE = 2, 3, and 4. If an ALE element has at least one coupling point (see NQUAD in *CONSTRAINED_LAGRANGE_IN_SOLID), this element will be selected to be refined (or removed). The number of elements to refine is computed during the initialization. NTOTRF can be zero. Otherwise it can be used to add more elements.

3. **Refinement Levels.** If NLVL = 1, there is only one level of refinement: the ALE elements in *ELEMENT_SOLID are the only ones to be replaced by clusters of 8 child elements. If NLVL > 1, there are several levels of refinement; not only the initial ALE elements in *ELEMENT_SOLID are refined but also their child elements.

4. **Multi-Material Groups.** If only Card 1 is defined, a multi-material set ID is not used. It can be left as zero. For Cards 2 and 3, MMSID is the ID of *SET_MULTI-MATERIAL_GROUP_LIST in which the multi-material group IDs (as defined in *ALE_MULTI-MATERIAL_GROUP) are listed to select the ALE elements to be refined (or removed). If MMSID < 0, only mixed ALE elements containing all the multi-material groups can be refined. Otherwise clusters of 8 elements without a mix of the listed multi-material groups can be removed.

   If MMSRM = 0, all the child clusters meeting the removal criteria can be deleted. If MMSRM is defined, only ALE child clusters fully filled by the multi-material groups listed by the set MMSRM can be removed (if the refinement removal criterion is reached).

5. **Number of ALE Elements to be Refined.** NTOTRF defines the total number of ALE elements to be refined. For example, NTOTRF = 100 with NLVL = 1 means that only 100 ALE elements can be replaced by 800 finer ALE elements (or 100 clusters of 8 child elements). For NLVL = 2, these 800 elements can be replaced by 6400 finer elements.

6. **Neighbor Element Refinement.** If an element is refined, the neighbor elements can be refined as well. LAYRF defines the number of neighbor layers to refine. For example:

a) with LAYRF = 1 an element that meets the refinement criterion at the center of a block of 3 × 3 × 3 elements will cause the refinement of these 27 elements.

b) with LAYRF = 2 an element that meets the refinement criterion at the center of a block of 5 × 5 × 5 elements will cause the refinement of these 125 elements.

c) with LAYRF = 3 an element that meets the refinement criterion at the center of a block of 7 × 7 × 7 elements will cause the refinement of these 343 elements.

7. **Refinement Removal Activation.** If BEGRM < 0, the check for refinement removal is activated when the number of 8 element clusters for the refinement is below a limit defined by |BEGRM| × NTOTRF. If |BEGRM| = 0.1, then the check for refinement removal starts when 90% of the stock of clusters is used for the refinement.

8. **Number of Child Clusters to Remove.** MAXRM < 0 defines a total number of child clusters to remove for the whole run. If positive, MAXRM defines an upper limit for the number of child clusters to remove every NCYCRM cycles.

9. **Output Child Element Data.** If only Card 1 is defined, the child data is always output in elout, that is, IELOUT is always activated. Since the refinement occurs during the initialization, every refined element is replaced by its 8 children in the set defined for *DATABASE_ELOUT.

   If the optional cards are included, the child data is output in elout if the IELOUT flag is set to 1. Since the refinement occurs during the run, the parent IDs in the set defined for *DATABASE_ELOUT are duplicated 8^{NLVL} times. The points of integration in the elout file are incremented to differentiate the child contributions to the database.

*CONTROL_REFINE_ALE2D

Purpose: Refine ALE quadrilateral shell elements locally. Each parent element is replaced by 4 child elements with a volume equal to $1/4^{th}$ the parent volume. If only the 1^{st} card is defined, the refinement occurs during the initialization. The 2^{nd} card defines a criterion CRITRF to automatically refine the elements during the run. If the 3^{rd} card is defined, the refinement can be removed if a criterion CRITRM is reached, that is, the child elements can be replaced by their parents.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NLVL	MMSID	IBOX	IELOUT	INITRN	
Type	I	I	I	I	I	I	I	
Default	none	0	1	0	0	0	0	

Remaining cards are optional.†

Automatic refinement card. Optional card for activating automatic refinement whereby each element satisfying certain criteria is replaced by a cluster of 4 child elements (2 × 2 child elements)

Card 2	1	2	3	4	5	6	7	8
Variable	NTOTRF	NCYCRF	CRITRF	VALRF	BEGRF	ENDRF	LAYRF	
Type	I	F	I	F	F	F	I	
Default	0	0.0	0	0.0	0.0	0.0	0	

Automatic Refinement Remove Card. Optional card for activating automatic refinement removal whereby, when, for a cluster of 4 child elements, certain criteria are satisfied the cluster (2 × 2 child elements) is replaced by its parent.

Card 3	1	2	3	4	5	6	7	8
Variable	MAXRM	NCYCRM	CRITRM	VALRM	BEGRM	ENDRM	MMSRM	
Type	I	F	I	F	F	F	I	
Default	0	0.0	0	0.0	0.0	0.0	0	

VARIABLE	DESCRIPTION
ID	Set ID. LT.0: Parent elements can be hidden in LS-PrePost as they are replaced by their children.
TYPE	Set type: EQ.0: ALE part set EQ.1: ALE part EQ.2: Lagrangian part set coupled to ALE (see Remarks 1 and 2) EQ.3: Lagrangian part coupled to ALE (see Remarks 1 and 2) EQ.4: Lagrangian beam set coupled to ALE (see Remarks 1 and 2) EQ.5: ALE shell set
NLVL	Number of refinement levels (see Remark 3)
MMSID	Multi-Material Set ID (see Remark 4): LT.0: Only ALE elements with all the multi-material groups listed in *SET_MULTI-MATERIAL_GROUP_LIST can be refined (or removed otherwise) GT.0: ALE elements with at least one of the multi-material groups can be refined (or removed)
IBOX	Box ID (see *DEFINE_BOX) defining a region in which the ALE elements are refined. The options LOCAL and ADAPTIVE for *DEFINE_BOX are supported.

VARIABLE	DESCRIPTION
IELOUT	Flag to handle child data in elout (see Remark 9)
INITRN	Flag to add mesh transitions for CRITRF = 0: EQ.0: Do not add mesh transitions. EQ.1: Add mesh transitions.
NTOTRF	Total number of ALE elements to refine (see Remark 5): GT.0: Number of elements to refine EQ.0: Number of shell elements in IBOX (see Remark 2) EQ.-1: Add clusters of 4 shells for the refinement during the run as needed
NCYCRF	Number of cycles between each refinement. LT.0: -NCYCRF is the time interval.
CRITRF	Refinement criterion: EQ.-3: Volume fraction < VALRF EQ.-2: Relative volume (V/V_0) > VALRF EQ.-1: Pressure < VALRF EQ.0: Static refinement (as if only the 1st card is defined) EQ.1: Pressure > VALRF EQ.2: Relative volume (V/V_0) < VALRF EQ.3: Volume fraction > VALRF EQ.5: User defined criterion. The fortran routine al2rfn_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRF	Criterion value to reach for the refinement
BEGRF	Time to begin the refinement
ENDRF	Time to end the refinement
LAYRF	Number of element layers to refine around an element that has satisfied the refinement criterion (see Remark 6)

VARIABLE	DESCRIPTION
MAXRM	Maximum number of child clusters (cluster is 2 × 2 child elements) to remove (see Remark 8): LT.0: For the whole run GT.0: Every NCYCRM cycles
NCYCRM	Number of cycles between each check for refinement removal. LT.0: |NCYCRM| is the time interval.
CRITRM	Criterion for refinement removal (a negative CRITRM reverses the conditions below): EQ.3: Volume fraction > VALRM EQ.2: Relative volume (V/V_0) < VALRM EQ.1: Pressure > VALRM EQ.0: No refinement removal (as if only Cards 1 and 2 are defined) EQ.1: Pressure < VALRM EQ.2: Relative volume (V/V_0) > VALRM EQ.3: Volume fraction < VALRM EQ.5: User defined criterion. The fortran routine al2rmv_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRM	Criterion value to reach in each child element of a cluster for its removal (child elements of a cluster replaced by parent element)
BEGRM	Time to begin the check for refinement removal. LT.0: |BEGRM| represents a critical percent of NTOTRF below which the check for refinement removal should begin (0.0 < |BEGRM| < 1.0). See Remark 6.
ENDRM	Time to end the check for refinement removal
MMSRM	Multi-Material Set ID for the refinement removal. See Remark 4.

Remarks:

1. **First card definition only and TYPE.** If only the 1st card is defined, only TYPE = 0, 1, and 5 can be defined.

2. **Lagrangian coupling.** *CONSTRAINED_LAGRANGE_IN_SOLID needs to be defined for TYPE = 2, 3, and 4. If an ALE element has at least one coupling point (see NQUAD in *CONSTRAINED_LAGRANGE_IN_SOLID), this element will be selected to be refined (or removed).

3. **Refinement levels.** If NLVL = 1, there is only one level of refinement; the ALE elements in *ELEMENT_SHELL are the only ones to be replaced by clusters of 4 child elements. If NLVL > 1, there are several levels of refinement; not only the initial ALE elements in *ELEMENT_SHELL are refined but also their child elements. If NLVL = 2, for example, an initial ALE element can be replaced by a cluster of 16 child elements.

4. **Multi-material groups.** If only Card 1 is defined, a multi-material set ID is not used. It can be left as zero. For Cards 2 and 3, MMSID is the ID of *SET_MULTI-MATERIAL_GROUP_LIST in which the multi-material group IDs (as defined in *ALE_MULTI-MATERIAL_GROUP) are listed to select the ALE elements to be refined (or removed). If MMSID < 0, only mixed ALE elements containing all the multi-material groups can be refined. Otherwise, clusters of 4 elements without a mix of the listed multi-material groups can be removed.

   If MMSRM = 0, all the child clusters meeting the removal criterion can be deleted. If MMSRM is defined, only ALE child elements fully filled by the multi-material groups listed by the set MMSRM can be removed (if the refinement removal criterion is reached).

5. **Number of ALE elements to be refined.** NTOTRF defines the total number of ALE elements to be refined. For example, NTOTRF = 100 means that only 100 ALE elements will be replaced by 400 ALE finer elements (or 100 clusters of 4 child elements). For NLVL = 2, these 400 elements can be replaced by 1600 finer elements.

6. **Neighbor element fefinement.** If an element is refined, neighbor elements can be refined as well. LAYRF defines the number of neighbor layers to refine. For example:

   a) with LAYRF = 1 an element that meets the refinement criterion at the center of a block of 3 × 3 elements will cause the refinement of these 9 elements.

   b) with LAYRF = 2 an element that meets the refinement criterion at the center of a block of 5 × 5 elements will cause the refinement of these 25 elements.

   c) with LAYRF = 3 an element that meets the refinement criterion at the center of a block of 7 × 7 elements will cause the refinement of these 49 elements

7. **Refinement removal activation.** If BEGRM < 0, the check for refinement removal is activated when the number of four element clusters for the refinement

is below a limit defined by |BEGRM| × NTOTRF. If |BEGRM| = 0.1, then the check for refinement removal starts when 90% of the stock of clusters is used for the refinement.

8. **Number of child clusters to remove.** MAXRM < 0 is the exact opposite of NTOTRF > 0; it defines a total number of child clusters to remove for the whole run. If positive, MAXRM defines an upper limit for the number of child clusters to remove every NCYCRM cycles

9. **Output child element data.** If only Card 1 is defined, the child data is output in elout, meaning IELOUT is always activated. Because the refinement occurs during the initialization phase, every refined element is replaced by its 4 children in the set defined for *DATABASE_ELOUT.

   If the optional cards are included, the child data is output in elout if the IELOUT is set to 1. Since the refinement occurs during the run, the parent IDs in the set defined for *DATABASE_ELOUT are duplicated 4^{NLVL} times. The points of integration in the elout file are incremented to differentiate the child contributions to the database.

*CONTROL_REFINE_MPP_DISTRIBUTION

Purpose: Distribute the elements for the refinement over the MPP processes. This keyword addresses the following situation:

> If TYPE = 2, 3, or 4 in *CONTROL_REFINE_ALE, the refinement occurs around a structure. The number of elements for this refinement is computed for each process according the initial position of the structure in each MPP subdomain (after the MPP decomposition of the ALE mesh during phase 3 of the initialization, each process has a subdomain that is a portion of the ALE mesh). If the structure is not in a subdomain, the related process receives no extra element for the refinement. If the structure moves into this subdomain during the computation, the refinement around the structure can not occur. To avoid this problem, the structure can be considered within a box (the structure maxima and minima give the box dimensions and positions). This box moves and expands during the computation to keep the structure inside. An estimation of the maximal displacement and expansion will allow the code to evaluate which subdomains the structure will likely cross and how many extra elements a process may need to carry out the refinement.

The computation of the number of extra elements per process occurs in 2 steps:

- If a file called refine_mpp_distribution does not exist in the working directory, it will be created to list the number of elements by process. Each line in this file matches a process rank (starting from 0). After phase 3 of the MPP decomposition, the run terminates as if *CONTROL_MPP_DECOMPOSITION_SHOW was activated.
- The model can be run again and the file refine_mpp_distribution will be read to allocate the memory for the extra elements and distribute them across the processes.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	DX	DY	DZ	EX	EY	EZ	
Type	I	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	1.0	1.0	1.0	

VARIABLE	DESCRIPTION
ID	ID = -NTOTRF in *CONTROL_REFINE_ALE

VARIABLE	DESCRIPTION
DX	Dimensionless x-displacement of the box. (see Remark 1).
DY	Dimensionless y-displacement of the box. (see Remark 1).
DZ	Dimensionless z-displacement of the box. (see Remark 1).
EX	Dimensionless x-expansion of the box. (see Remark 2).
EY	Dimensionless y-expansion of the box. (see Remark 2).
EZ	Dimensionless z-expansion of the box. (see Remark 2).

Remarks:

1. **Box Displacements.** DX, DY and DZ are the maximal displacements of the box center. These displacements are ratio of the box dimensions. If, for example, the largest length of the structure in the x-direction is 10 m and the maximal displacement in this direction is 2 m, DX should be equal to 0.2.

2. **Maximal Box Dilations.** EX, EY and EZ represent the maximal dilatations of the box in each direction. These expansions are the ratio of the box dimensions. The box expands around its center. If, for example, the maximal thickness of a structure along the z-direction is 1 cm and the structure deforms 30 times the thickness in z-direction, EZ should be equal to 30, and DZ = 15 accounts for the box center motion. The xy-plane is a plane of symmetry for this deformation; DZ can be zero.

*CONTROL_REFINE_SHELL

Purpose: Refine quadrilateral shell elements locally. Each parent element is replaced by 4 child elements, each with a volume equal to 1/4 the parent volume.

Card Summary:

Card 1. This card is required. If only this card is defined, the refinement occurs during the initialization.

ID	TYPE	NLVL	IBOX	IELOUT			

Card 2. This card is optional. If defined, this card sets a criterion, CRITRF, to automatically refine the elements during the run.

NTOTRF	NCYCRF	CRITRF	VALRF	BEGRF	ENDRF	LAYRF	

Card 3. This card is optional. If defined, this card sets a criterion, CRITRM, for which the refinement can be removed; that is, the child elements can be replaced by their parents.

MAXRM	NCYCRM	CRITRM	VALRM	BEGRM	ENDRM		

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NLVL	IBOX	IELOUT			
Type	I	I	I	I	I			
Default	none	0	1	0	0			

VARIABLE	DESCRIPTION
ID	Set ID. LT.0: parent elements can be hidden in LS-PrePost as they are replaced by their children.
TYPE	Set type: EQ.0: part set

VARIABLE	DESCRIPTION
	EQ.1: part EQ.2: shell set
NLVL	Number of refinement levels (see Remark 1)
IBOX	Box ID (See *DEFINE_BOX) defining a region in which the elements are refined. The options LOCAL and ADAPTIVE for *DEFINE_BOX are supported.
IELOUT	Flag to handle child data in the elout file (see Remark 5)

Automatic refinement card. Optional card for activating automatic refinement whereby each element satisfying certain criteria is replaced by a cluster of 4 child elements

Card 2	1	2	3	4	5	6	7	8
Variable	NTOTRF	NCYCRF	CRITRF	VALRF	BEGRF	ENDRF	LAYRF	
Type	I	F	I	F	F	F	I	
Default	0	0.0	0	0.0	0.0	0.0	0	

VARIABLE	DESCRIPTION
NTOTRF	Total number of elements to refine (see Remark 2): GT.0: number of elements to refine EQ.0: number of shell elements in IBOX EQ.-1: add clusters of 4 shells for the refinement during the run.
NCYCRF	Number of cycles between each refinement. LT.0: |NCYCRF| is the time interval.
CRITRF	Refinement criterion: EQ.-4: criterion similar to ADPTYP = 4 in *CONTROL_ADAPTIVE with VALRF replacing ADPTOL (VALRF = ADPTOL), except refinement occurs when the shell error in the energy norm is less than VALRF/100 times the mean energy norm within the part

VARIABLE	DESCRIPTION
	EQ.-3: von Mises stress < VALRF EQ.-1: pressure < VALRF EQ.0: static refinement (as if only the 1st card is defined) EQ.1: pressure > VALRF EQ.2: undefined EQ.3: von Mises stress > VALRF EQ.4: criterion similar to ADPTYP = 4 in *CONTROL_ADAPTIVE with VALRF replacing ADPTOL (VALRF = ADPTOL) EQ.5: user defined criterion. The fortran routine shlrfn_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRF	Criterion value to reach for the refinement
BEGRF	Time to begin the refinement
ENDRF	Time to end the refinement
LAYRF	Number of element layers to refine around an element reaching the refinement criterion (see Remark 3)

Automatic Refinement Remove Card. Optional card for activating automatic refinement removal whereby, when, for a cluster of 4 child elements, certain criteria are satisfied the clusters is replaced by its parent.

Card 3	1	2	3	4	5	6	7	8
Variable	MAXRM	NCYCRM	CRITRM	VALRM	BEGRM	ENDRM		
Type	I	F	I	F	F	F		
Default	0	0.0	0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
MAXRM	Maximum number of child clusters to remove: LT.0: |MAXRM| is the total number of child clusters to be removed for the whole run, GT.0: max number removed every NCYCRM cycles
NCYCRM	Number of cycles between each check for refinement removal: LT.0: |NCYCRM| is the time interval.
CRITRM	Criterion for refinement removal: EQ.-4: criterion similar to ADPTYP = 4 in *CONTROL_ADAPTIVE with VALRM replacing ADPTOL (VALRM = ADPTOL) EQ.-3: von Mises stress > VALRM EQ.-1: pressure > VALRM EQ.0: no refinement removal (as if only the 1st and 2nd cards are defined) EQ.1: pressure < VALRM EQ.2: undefined EQ.3: von Mises stress < VALRM EQ.4: criterion similar to ADPTYP = 4 in *CONTROL_ADAPTIVE with VALRM replacing ADPTOL (VALRM = ADPTOL), except refinement occurs when the shell error in the energy norm is less than VALRM/100 times the mean energy norm within the part EQ.5: user defined criterion. The fortran routine shlrmv_criteria5 in the file dynrfn_user.f should be used to develop the criterion. The file is part of the general package usermat.
VALRM	Criterion value to reach in each child elements of a cluster for its removal (child elements replaced by parent element)
BEGRM	Time to begin the check for refinement removal. LT.0: |BEGRM| represents a critical percent of NTOTRF below which the check for refinement removal should begin (0.0 < |BEGRM| < 1.0). See Remark 4.

VARIABLE	DESCRIPTION
ENDRM	Time to end the check for refinement removal

Remarks:

1. **Refinement Level.** If NLVL = 1, there is only one level of refinement; the elements in *ELEMENT_SHELL are the only ones to be replaced by clusters of 4 child elements. If NLVL > 1, there are several levels of refinement; not only the initial elements in *ELEMENT_SHELL are refined but also their child elements. If NLVL = 2 for example, the initial elements can be replaced by clusters of 16 child elements.

2. **Number of Parent Elements.** NTOTRF > 0 defines the total number of elements to be refined. For example, NTOTRF = 100 with NLVL = 1 means that only 100 elements can be replaced by 400 finer elements (or 100 clusters of 4 child elements). For NLVL = 2, these 400 elements can be replaced by 1600 finer elements.

3. **Neighboring Element Refinement.** If an element is refined, neighboring elements may be refined as well. LAYRF defines the number of neighbor layers to refine. For example:

   a) with LAYRF = 1 an element that meets the refinement criterion at the center of a block of 3 × 3 elements will cause the refinement of these 9 elements.

   b) with LAYRF = 2 an element that meets the refinement criterion at the center of a block of 5 × 5 elements will cause the refinement of these 25 elements.

   c) with LAYRF = 3 an element that meets the refinement criterion at the center of a block of 7 × 7 elements will cause the refinement of these 49 elements

4. **Refinement Removal Activation.** If BEGRM < 0, the check for refinement removal is activated when the number of 4-element clusters for the refinement is below a limit defined by |BEGRM| × NTOTRF. If |BEGRM| = 0.1, the check for refinement removal starts when 90% of the stock of clusters is used for the refinement.

5. **IELOUT.** If only the 1st card is defined, the code for IELOUT is always activated. Since the refinement occurs during the initialization, every refined element is replaced by its 4 children in the set defined for *DATABASE_ELOUT.

   If more than the 1st card is defined, the code for IELOUT is activated if the flag is equal to 1. Since the refinement occurs during the run, the parent IDs in the set defined for *DATABASE_ELOUT are duplicated 4^{NLVL} times. The points of

integration in the elout file are incremented to differentiate the child contributions to the database.

*CONTROL_REFINE_SOLID

Purpose: Refine hexahedral solid elements locally. Each parent element meeting the refinement criterion is replaced by 8 child elements with a volume equal to $1/8^{th}$ the parent volume. If only the 1^{st} card is defined, the refinement occurs during the initialization. The 2^{nd} card defines a criterion CRITRF to automatically refine the elements during the run. If the 3^{rd} card is defined, the refinement can be removed if a criterion CRITRM is reached: the child elements can be replaced by their parents.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NLVL	IBOX	IELOUT			
Type	I	I	I	I	I			
Default	none	0	1	0	0			

Remaining cards are optional.†

Automatic refinement card. Optional card for activating automatic refinement whereby each element satisfying certain criteria is replaced by a cluster of 8 child elements

Card 2	1	2	3	4	5	6	7	8
Variable	NTOTRF	NCYCRF	CRITRF	VALRF	BEGRF	ENDRF	LAYRF	
Type	I	F	I	F	F	F	I	
Default	0	0.0	0	0.0	0.0	0.0	0	

Automatic Refinement Remove Card. Optional card for activating automatic refinement removal whereby, when, for a cluster of 8 child elements, certain criteria are satisfied the clusters is replaced by its parent.

Card 3	1	2	3	4	5	6	7	8
Variable	MAXRM	NCYCRM	CRITRM	VALRM	BEGRM	ENDRM		
Type	I	F	I	F	F	F		
Default	0	0.0	0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ID	Set ID. LT.0: parent elements can be hidden in LS-PrePost as they are replaced by their children.
TYPE	Set type: EQ.0: part set EQ.1: part EQ.2: solid set
NLVL	Number of refinement levels. See Remark 1.
IBOX	Box ID (see *DEFINE_BOX) defining a region in which the elements are refined. The keyword options LOCAL and ADAPTIVE for *DEFINE_BOX a re supported.
IELOUT	Flag to handle child data in elout. See Remarks 6 and 7.
NTOTRF	Total number of elements to refine. See Remark 2. GT.0: number of elements to refine EQ.0: NTOTRF = number of solid elements in IBOX EQ.-1: add clusters of 8 solids for the refinement during the run.
NCYCRF	Number of cycles between each refinement. LT.0: |NCYCRF| is the time interval.

VARIABLE	DESCRIPTION
CRITRF	Refinement criterion: EQ.-5: user defined criterion as defined in subroutine sldrfn_criteria5 in the file dynrfn_user.f, which is included in the general usermat package. "Effective plastic strain" < VALRF is the criterion if sldrfn_criteria5 is not modified by the user. EQ.-3: von Mises stress < VALRF EQ.-1: pressure < VALRF EQ.0: static refinement as if only the 1st card is defined EQ.1: pressure > VALRF EQ.2: no refinement occurs. EQ.3: von Mises stress > VALRF EQ.5: user defined criterion as defined in subroutine sldrfn_criteria5 in the file dynrfn_user.f, which is included in the general usermat package. "Effective plastic strain" > VALRF is the criterion if sldrfn_criteria5 is not modified by the user.
VALRF	Value of the refinement criterion that triggers refinement
BEGRF	Time to begin the refinement
ENDRF	Time to end the refinement
LAYRF	Number of element layers to refine around an element reaching the refinement criterion. See Remark 3.
MAXRM	Maximum number of child clusters to remove. See Remark 5. LT.0: for the whole run GT.0: every NCYCRM cycles
NCYCRM	Number of cycles between each check for refinement removal: LT.0: |NCYCRM| is the time interval.
CRITRM	Criterion for removal of refinement: EQ.-3: von Mises stress > VALRM EQ.-1: pressure > VALRM

VARIABLE	DESCRIPTION
	EQ.0: no removal of refinement as if only the 1st and 2nd cards are defined. EQ.1: pressure < VALRM EQ.2: same effect as CRITRM = 0 EQ.3: von Mises stress < VALRM EQ.5: user defined criterion as defined in subroutine sldrmv_criteria5 in the file dynrfn_user.f.
VALRM	Criterion value to reach in each child element of a cluster for its removal (replace child elements with parent element)
BEGRM	Time to begin check for refinement removal: LT.0: |BEGRM| represents a critical percent of NTOTRF below which the check for refinement removal should begin (0.0 < |BEGRM| < 1.0). See Remark 4.
ENDRM	Time to end the check for refinement removal

Remarks:

1. **Number of Refinement Levels.** If NLVL = 1, there is only one level of refinement: the elements in *ELEMENT_SOLID are the only ones to be replaced by clusters of 8 child elements. If NLVL > 1, there are several levels of refinement: not only the initial elements in *ELEMENT_SOLID are refined but also their child elements. If NLVL = 2 for example, the initial elements can be replaced by clusters of 64 child elements.

2. **Maximum Number of Elements to Refine.** NTOTRF defines the total number of elements to be refined. For example, NTOTRF = 100 with NLVL = 1 means that only 100 elements can be replaced by 800 finer elements (or 100 clusters of 8 child elements). For NLVL = 2, these 800 elements can be replaced by 6400 finer elements.

3. **Number of Layers to Refine.** If an element is refined, it is possible to refine the neighbor elements as well. LAYRF defines the number of neighbor layers to refine. For example:

   a) with LAYRF = 1 an element that meets the refinement criterion at the center of a block of 3 × 3 × 3 elements will cause the refinement of these 27 elements.

b) with LAYRF = 2 an element that meets the refinement criterion at the center of a block of 5 × 5 × 5 elements will cause the refinement of these 125 elements.

c) with LAYRF = 3 an element that meets the refinement criterion at the center of a block of 7 × 7 × 7 elements will cause the refinement of these 343 elements.

4. **Onset of Refinement Removal.** If BEGRM < 0, the check for refinement removal is activated when the number of 8-element clusters for the refinement is below a limit defined by |BEGRM| × NTOTRF. If |BEGRM| = 0.1, it means that the check for refinement removal starts when 90% of the stock of clusters are used for the refinement.

5. **Maximum Refinement Removal.** MAXRM < 0 defines a total number of child clusters to remove for the whole run. If positive, MAXRM defines an upper limit for the number of child clusters to remove every NCYCRM cycles.

6. **The elout Database and Initial Refinement.** If only the 1st card is defined, the code for IELOUT is always activated. Since the refinement occurs during the initialization, every refined element is replaced by its 8 children in the set defined for *DATABASE_ELOUT.

7. **The elout Database and Refinement at Run Time.** If there is more than 1 line, the code for IELOUT is activated if the flag is equal to 1. Since the refinement occurs during the run, the parent IDs in the set defined for *DATABASE_ELOUT are duplicated 8^{NLVL} times. The points of integration in the elout file are incremented to differentiate the child contributions to the database.

8. **Contact.** When a refined part involves a contact that is not *CONTACT_ERODING_SINGLE_SURFACE with SOFT = 2, that contact is automatically replaced with *CONTACT_ERODING_SINGLE_SURFACE with SOFT = 2.

***CONTROL_REMESHING**_*{OPTION}*

Available options include:

<BLANK>

EFG

Purpose: Provide control over the adaptive remeshing of solids. See also *CONTROL_-ADAPTIVE and the variable ADPOPT in *PART.

There are two different types of 3D solid adaptivity affected by *CONTROL_REMESH-ING.

1. Tetrahedral adaptivity is invoked by ADPOPT = 2 in *PART. The initial mesh must be comprised of only tetrahedrons. We recommend FEM solid formulations 10 and 13 and EFG solid formulation 42 (see details in *SECTION_SOLID). When adaptivity is triggered by any of several available criteria (volume loss, minimum time step, etc.), the surface mesh consisting of triangles is first improved, subject to the element size criteria set forth in *CONTROL_REMESH-ING. Then, the automatic remesher creates the tetrahedral elements of the new mesh starting from the improved surface mesh. When the EFG option is included, additional adaptivity criteria may be invoked using Card 2 and, optionally, Card 3.

2. Axisymmetric adaptivity, sometimes called orbital adaptivity, in which remeshing is done with hexahedral and pentahedral elements, is invoked by AD-POPT = 3 in *PART. Only the variables RMIN, RMAX, CID, and SEGANG are used in this type of adaptivity. Though EFG solid formulation 41 is permissible, the EFG option of *CONTROL_REMESHING does not apply for this type of adaptivity.

Card 1	1	2	3	4	5	6	7	8
Variable	RMIN	RMAX	VF_LOSS	MFRAC	DT_MIN	ICURV	CID	SEGANG
Type	F	F	F	F	F	I	I	F
Default	none	none	1.0	0.0	0.	4	0	0.0

Additional card for EFG option or FEM tetrahedral adaptivity. With FEM tetrahedral adaptivity, only MM option is available.

Card 2	1	2	3	4	5	6	7	8
Variable	IVT	IAT	IAAT	ICONOPT	MM			
Type	I	I	I	I	I			
Default	1	0	0	0	0			

Second additional card for EFG option. This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	IAT1	IAT2	IAT3					
Type	F	F	F					
Default	10^{20}	10^{20}	10^{20}					

VARIABLE	DESCRIPTION
RMIN	Minimum edge length for the surface mesh surrounding the parts which should be remeshed. See Remark 1. LT.0.0: |RMIN| is the ID of a load curve controlling the time-dependent minimum edge length for 3D *r*-adaptivity.
RMAX	Maximum edge length for the surface mesh surrounding the parts which should be remeshed. See Remark 1.
VF_LOSS	Volume fraction loss required in a type 10/13 tetrahedral elements to trigger a remesh. For the type 13 solid elements, the pressures are computed at the nodal points; therefore, it is possible for overall volume to be conserved but for individual tetrahedrons to experience a significant volume loss or gain. The volume loss can lead to numerical problems. Recommended values for VF_LOSS in the range of 0.10 to 0.30 may be reasonable.
MFRAC	Mass ratio gain during mass scaling required for triggering a remesh. For a one percent increase in mass, set MFAC = 0.010. This variable applies to both to general three-dimensional

VARIABLE	DESCRIPTION
	tetrahedral remeshing and to three dimensional axisymmetric remeshing.
DT_MIN	Time step size required for triggering a remesh. This option applies only to general three-dimensional tetrahedral remeshing and is checked before mass scaling is applied and the time step size reset.
ICURV	Define number of elements along the radius in the adaptivity. See Remark 3.
CID	Coordinate system ID for three dimensional axisymmetric remeshing. The z-axis in the defined coordinate system is the orbital axis and must be parallel to the global z-axis in the current axisymmetric remesher. EQ.0: Use global coordinate, and the global z-axis is the orbital axis (default)
SEGANG	*For axisymmetric 3D remeshing.* Angular element size in degrees. *For general (tet) 3D remeshing.* Critical angle specified in radians needed to preserve feature lines.
IVT	Internal variable transfer in adaptive EFG (see Remark 4): EQ.1: Moving least square approximation with Kronecker-delta property (recommended in general case) EQ.-1: Moving least square approximation without Kronecker-delta property EQ.2: Partition of unity approximation with Kronecker-delta property EQ.-2: Partition of unity approximation without Kronecker-delta property EQ.-3: Finite element approximation
IAT	Flag for interactive adaptivity (see Remark 2): EQ.0: No interactive adaptivity EQ.1: Interactive adaptivity combined with predefined adaptivity. EQ.2: Purely interactive adaptivity. The time interval between two successive adaptive steps is bounded by ADPFREQ.

VARIABLE	DESCRIPTION
	EQ.3: Purely interactive adaptivity
IAAT	Interactive adaptivity adjustable tolerance: EQ.0: The tolerance to trigger interactive adaptivity is not adjusted. EQ.1: The tolerance is adjusted in run-time to avoid over-activation.
ICONOPT	Remeshing option for tied contact surfaces between two adaptive parts: EQ.0: No special remeshing EQ.-1: The two contact surfaces are matched at the first adaptive step and not remeshed in later adaptive steps.
MM	Adaptive remeshing with monotonic resizing for either EFG or FEM: EQ.1: The adaptive remeshing cannot coarsen a mesh. The current implementation only supports IAT = 1, 2, 3 for EFG. It is available for FEM tetrahedral remeshing too.
IAT1	Shear strain tolerance for interactive adaptivity. If the shear strain in any formulation 42 EFG tetrahedral element exceeds IAT1, remeshing is triggered. (0.1 ~ 0.5 is recommended).
IAT2	$L_{\max}/L_{\min}$ tolerance where $L_{\max}$ and $L_{\min}$ are the maximum and minimum edge lengths of any given formulation 42 EFG tetrahedral element. If this ratio in any element exceeds IAT2, remeshing is triggered. (RMAX/RMIN is recommended.)
IAT3	Volume change tolerance. If the normalized change in volume of any formulation 42 tetrahedral element, defined as $\|v_1 - v_0\|/\|v_0\|$ where v_1 is the current element volume and v_0 is the element volume immediately after the most recent remeshing, exceeds IAT3, remeshing is triggered. (0.5 is recommended.)

Remarks:

1. **Edge length.** The value of RMIN and RMAX should be of the same order. The value of RMAX can be set to 2-5 times greater than RMIN. RMIN and RMAX may be superseded in specific regions of the part being adapted; see the variables PID, BRMIN, and BRMAX in *DEFINE_BOX_ADAPTIVE.

2. **Interactive adaptivity.** Interactive adaptivity is only supported for the adaptive 4-noded mesh-free (EFG) solid formulation (ELFORM = 42 in *SECTION_SOLID_EFG). When interactive adaptivity is invoked (IAT > 0), even if none of the tolerances IAT1, IAT2, and IAT3 for the three respective indicators (shear strain, edge length ratio, normalized volume change) are exceeded, remeshing will still be triggered if any of the three indicators over a single explicit time step changes by more than 50%, that is, if

$$\frac{|[\text{value}]_n - [\text{value}]_{n-1}|}{|[\text{value}]_{n-1}|} > 0.5$$

where $[\text{value}]_n$ denotes value of indicator in n^{th} (current) time step and $[\text{value}]_{n-1}$ denotes value of indicator in previous time step. This condition is checked only if $[\text{value}]_{n-1}$ is nonzero.

3. **ICURV.** ICURV represents a number of elements and applies only when ADPENE > 0 in *CONTROL_ADAPTIVE. The "desired element size" at each point on the SURFA contact surface is computed based on the tooling radius of curvature (see the description of ADPENE in *CONTROL_ADAPTIVE), so that ICURV elements would be used to resolve a hypothetical 90 degree arc at the tooling radius of curvature. The value of ICURV is (internally) limited to be ≥ 2 and ≤ 12. The final adapted element size is adjusted as necessary to fall within the size range set forth by RMIN and RMAX.

4. **IVT.** IVT defines different remapping schemes for the adaptive 4-noded mesh-free (EFG) solid formulation (ELFORM = 42 in *SECTION_SOLID_EFG). The finite element approximation (IVT = -3) is the most efficient one in terms of computational cost. The moving least square approximation (IVT = 1 or -1) can significantly reduce the remapping error when there are many adaptive steps and the solution fields have local features, such as a high gradient. But this option has a high CPU and memory cost. The partition of unity approximation (IVT = 2 or -2) is slightly more accurate than the finite element approximation (IVT = -3).

*CONTROL_REQUIRE_REVISION

Purpose: To prevent the model from being run in old versions of LS-DYNA. This might be desirable due to known improvements in the program, required capability, etc.

Card 1	1	2	3	4	5	6	7	8
Variable	RELEASE	SVNREV	GITREV					
Type	C	I	I					
Default	Rem 2	none	none					

VARIABLE	DESCRIPTION
RELEASE	The release of code required. This should be a string such as "R6.1.0" or "R7.0"
SVNREV	The minimum SVN revision required (ignored by executables made after our move to git). This corresponds to the "SVN Version" field in the d3hsp file for versions of LS-DYNA prior to our move to git for version control. R12.0 was the last release version made while we were using SVN for version control.
GITREV	The minimum git revision required corresponding to the release of code (ignored by executables made prior to our move to git). This corresponds to part of the git hash given in the "Revision" field in the d3hsp file for versions of LS-DYNA after our move to git for version control. In d3hsp, the last string is "R[Release #]-[number]-[alphanumeric]". If RELEASE is given, LS-DYNA compares GITREV to that [number]. If RELEASE is not given, see Remark 2. R11.2 was the first release version made when we moved to git.

Remarks:

1. **Number of Lines.** Any number of lines can appear, indicating for example that a particular feature was introduced in different release branches at different times.

2. **RELEASE.** If the RELEASE field is left empty, then any executable whose development split from the main SVN or git trunk after the given SVNREV or GITREV, respectively, will be allowed.

Example:

```
*CONTROL_REQUIRE_REVISION
R6.1    79315
R7.0    78310
        78304
```

The above example would prevent execution by any R6.1 executable before r79315, any R7.0 before r78310, and all other executables whose development split from the main trunk before r78304. Note that no versions of R6.0, R6.0.0, or R6.1.0 are allowed: R6.1 does NOT imply R6.1.0, no matter what the revision of R6.1.0 – R6.1.0 would have to be explicitly listed. Similarly, R7.0.0 would not be allowed because it is not listed, and it split from the trunk in r76398. Any future R8.X executable would be allowed, since it will have split from the trunk after r78304.

*CONTROL_RIGID

Purpose: Special control options related to rigid bodies and linearized flexible bodies; see *PART_MODES.

Card 1	1	2	3	4	5	6	7	8
Variable	LMF	JNTF	ORTHMD	PARTM	SPARSE	METALF	PLOTEL	RBSMS
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Remaining cards are optional.†

Card 2	1	2	3	4	5	6	7	8
Variable	NORBIC	GJADSTF	GJADVSC	TJADSTF	TJADVSC	RCVLR2D		
Type	I	F	F	F	F	I		
Default	0	0.0	0.0	0.0	0.0	0		

VARIABLE	DESCRIPTION
LMF	Joint formulation flag for explicit analysis. This flag can be used to switch to an implicit formulation for joints (*CONSTRAINED_JOINT_*OPTION*), which uses Lagrange multipliers to impose prescribed kinematic boundary conditions and joint constraints. There is a slight cost overhead due to the assembly of sparse matrix equations, which are solved using standard procedures for nonlinear problems in rigid multi-body dynamics. EQ.0: Penalty formulation for joints (default) EQ.1: Lagrange-multiplier-based formulation for joints
JNTF	Generalized joint stiffness formulation; see Remark 1 below: EQ.0: Incremental update EQ.1: Total formulation (exact)

VARIABLE	DESCRIPTION
	EQ.2: Total formulation intended for implicit analysis
ORTHMD	Orthogonalize modes with respect to each other: EQ.0: True EQ.1: False. The modes are already orthogonalized.
PARTM	Use global mass matrix to determine part mass distribution. This mass matrix may contain mass from other parts that share nodes. See Remark 2 below. EQ.0: True EQ.1: False
SPARSE	Use sparse matrix multiply subroutines for the modal stiffness and damping matrices. See Remark 3. EQ.0: False. Do full matrix multiplies (frequently faster). EQ.1: True
METALF	Use fast update of rigid body nodes. Only applicable to sheet metal forming analysis using 2D and 3D shell elements. If this option is active, the rotational motion of all rigid bodies should be suppressed. EQ.0: Full treatment EQ.1: Fast update for metal forming applications
PLOTEL	Automatic generation of *ELEMENT_PLOTEL for *CONSTRAINED_NODAL_RIGID_BODY: EQ.0: No generation EQ.1: One part is generated for all nodal rigid bodies with the PID set to 10000000. EQ.2: One part is generated for each nodal rigid body in the problem with a part ID of 10000000 + PID, where PID is the nodal rigid body ID.
RBSMS	Flag to apply consistent treatment of rigid bodies in selective and conventional mass scaling, Remark 4. EQ.0: Off EQ.1: On

VARIABLE	DESCRIPTION
NORBIC	Circumvent rigid body inertia check, see Remark 5. EQ.0: Off EQ.1: On
GJADSTF	Rotational stiffness is added to all joints in the model to model a small resistance, such as joint friction, or to avoid zero-energy modes in implicit. This is equivalent to defining a *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED to the free rotational degrees of freedom of all joints, using a slope in LCIDPH, LCIDT, and LCIDPS equal to GJADSTF. This field is advantageous for models with many joints because it only has to be defined once for all joints, whereas the equivalent keyword must be set for each joint.
GJADVSC	Rotational damping is added to all joints in the model to model a small resistance, such as joint friction, or to avoid zero-energy modes in implicit. This is equivalent to defining a *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED to the free rotational degrees of freedom of all joints, using a slope in DLCIDPH, DLCIDT, and DLCIDPS equal to GJADVSC. Like GJADSTF, this field is advantageous for models with many joints because it only has to be defined once for all joints, whereas the equivalent keyword must be set for each joint.
TJADSTF	Translational stiffness is added to all joints in the model to model a small resistance, such as joint friction, or to avoid zero-energy modes in implicit. This is equivalent to defining a *CONSTRAINED_JOINT_STIFFNESS_TRANSLATIONAL to the free translational degrees of freedom of all joints, using a slope in LCIDX, LCIDY, and LCIDZ equal to TJADSTF. Like GJADSTF, this field is advantageous for models with many joints because it only has to be defined once for all joints, whereas the equivalent keyword must be set for each joint.
TJADVSC	Translational damping is added to all joints in the model to model a small resistance, such as joint friction, or to avoid zero-energy modes in implicit. This is equivalent to defining a *CONSTRAINED_JOINT_STIFFNESS_TRANSLATIONAL to the free translational degrees of freedom of all joints, using a slope in DLCIDX, DLCIDY, and DLCIDZ equal to TJADVSC. Like GJADSTF, this field is advantageous for models with many joints because it only has to be defined once for all joints, whereas the equivalent

VARIABLE	DESCRIPTION
	keyword must be set for each joint.
RCVLR2D	Recover the lead rigid body of constrained rigid bodies, which was changed due to *DEFORMABLE_TO_RIGID_AUTOMATIC (see Remark 6): EQ.0: Off EQ.1: On

Remarks:

1. **JNTF.** The default behavior is for the relative angles between the two coordinate systems to be done incrementally. This is an approximation, in contrast to the total formulation where the angular offsets are computed exactly. The disadvantage of the latter approach is that a singularity exists when an offset angle equals 180 degrees. In most applications, the stop angles exclude this possibility and JNTF=1 should not cause a problem. JNTF=2 is implemented with smooth response and especially intended for implicit analysis.

2. **PARTM.** If the determination of the normal modes included the mass from both connected bodies and discrete masses, or if there are no connected bodies, then the default is preferred. When the mass of a given part ID is computed, the resulting mass vector includes the mass of all rigid bodies that are merged to the given part ID, but does not included discrete masses. See the keyword: *CONSTRAINED_RIGID_BODIES. A lumped mass matrix is always assumed.

3. **SPARSE.** Sparse matrix multipliers save a substantial number of operations if the matrix is truly sparse. However, the overhead will slow the multipliers for densely populated matrices.

4. **RBSMS.** In selective mass scaling, rigid bodies connected to deformable elements can significantly add inertia due to missing terms in the SMS mass matrix. This problem has been observed in automotive applications where spot welds are modeled using constrained nodal rigid bodies. Applying consistent rigid body treatment can significantly improve accuracy and robustness at the expense of increased CPU intensity. One aspect of this treatment is that all different types of rigid bodies are treated equally. For example, like for *CONSTRAINED_NODAL_RIGID_BODIES, this treatment merges parts with *MAT_RIGID that share nodes into one rigid body.

   RBSMS = 1 also applies the same rigid body treatment to conventional mass scaling because inconsistencies, for various reasons, may result in unstable solution schemes, even in this case.

5. **NORBIC.** During initialization, the determinant of the rigid body inertia tensor is checked. If it falls below a tolerance value of 10^{-30}, LS-DYNA issues an error message and the calculation stops. In some rare cases (for example with an unfavorable system of units), such tiny values would still be valid. In this case, NORBIC should be set to 1 to circumvent the implied inertia check.

6. **RCVLR2D.** When a deformable part PID is switched to rigid and becomes constrained to lead rigid part LRB, the rigid part PIDL sharing common nodes with PID will be constrained to LRB as well. At the same time, the rigid part PIDC having PIDL as the lead rigid body through *CONSTRAINED_RIGID_BODIES will now have LRB as its new lead rigid body (see *DEFORMABLE_TO_RIGID_AUTOMATIC and *CONSTRAINED_RIGID_BODIES). When PID is switched back to deformable, PIDC will be constrained to its old lead PIDL if RCVLR2D = 1. Otherwise, PIDC will stay constrained LRB.

*CONTROL_SEGMENTS_IN_ALE_COUPLING

Purpose: Deactivate segments in the ALE penalty coupling (see CTYPE = 4, 5 and 6 in *CONSTRAINED_LAGRANGE_IN_SOLID) if the segments are face to face with other segments. By default, all the segments (element faces with 4 or 3 nodes in 3D or element sides with 2 nodes in 2D) search for the ALE material interface involved in the coupling. If the interface is close to a segment, penalty coupling forces are applied on both to keep them together. Some segments might not need to be involved in the coupling if they face another segment. For example, let 2 side-by-side cubic structures have element faces in contact. As long as these segments face each other, they do not need to apply coupling forces on the material interface. In this particular example, if inside segments near the corners were to drag the material interfaces to them, leakages could occur; see Figure 12-87.

See also *CONSTRAINED_LAGRANGE_IN_SOLID (CTYPE = 4, 5 and 6), *SET_SEGMENT, and *ALE_MULTI-MATERIAL_GROUP.

Segment Set Card. This card is optional.

Card 1	1	2	3	4	5	6	7	8
Variable	RANKEY	SEGSET	NCYCHK	SYM				
Type	I	I	I	I				
Default	0	0	10	0				

Coupling Card. This card is optional card.

Card 2	1	2	3	4	5	6	7	8
Variable	NINTHK	CONTHK						
Type	I	F						
Default	0	0.0						

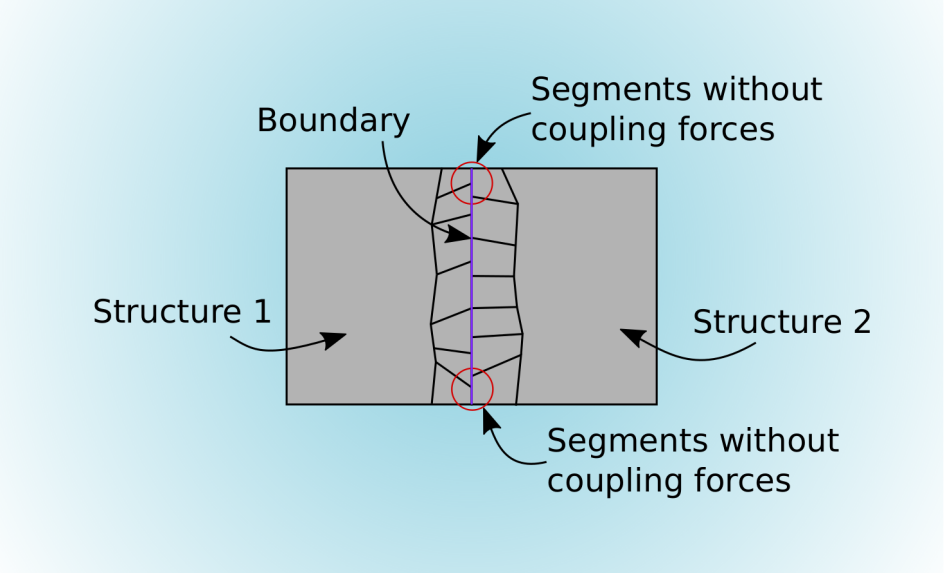


Figure 12-87. An example of when penalty coupling should be deactivated for certain segments.

VARIABLE	DESCRIPTION
RANKEY	Instantiation of *CONSTRAINED_LAGRANGE_IN_SOLID in the input deck controlled by this keyword (see Remarks 1 and 2). For example, if RANKEY = 2, then the second *CONSTRAINED_LAGRANG_IN_SOLID definition is controlled by this keyword.
SEGSET	Set ID of *SET_SEGMENT (see Remark 2)
NCYCHK	Number of cycles between checks to activate/deactivate coupling segments (see Remark 3)
SYM	Flag to deactivate coupling segments with normal boundary constraints. EQ.0: Off EQ.1: On
NINTHK	Minimum number of coupling points in contact to deactivate the segment (see Remark 4).
CONTHK	Contact thickness (see Remark 5)

Remarks:

1. **Keyword without Data Cards.** If *CONTROL_SEGMENTS_IN_ALE_COUPLING is included without any data cards, namely, Cards 1 and 2, all the *CON-

STRAINED_LAGRANGE_IN_SOLID keywords are controlled by *CONTROL_SEGMENTS_IN_ALE_COUPLING.

2. **Segment Sets.** The RANKEYth *CONSTRAINED_LAGRANGE_IN_SOLID keyword definition in the keyword deck determines the segment set controlled by *CONTROL_SEGMENTS_IN_ALE_COUPLING. The parameter LSTRSID with LSTRSTYP = 2 in *CONSTRAINED_LAGRANGE_IN_SOLID refers to a segment set. If LSTRSTYP < 2, LSTRSID refers to a part or part set. However, the segments of these parts are still generated during the initialization. If RANKEY and SEGSET are both nonzero, the segments common to RANKEY and SEGSET will be controlled. If RANKEY and SEGSET do not have segments in common, their segment lists will both be controlled.

3. **Check Frequency.** The parameter NBKT in *CONTROL_ALE controls how often the bucket sorting to search segments in contact is updated.

4. **Coupling Points.** NQUAD in *CONSTRAINED_LAGRANGE_IN_SOLID defines the number of coupling points by segment (NQUAD × NQUAD in 3D and NQUAD in 2D). If NINTHK = 0, all the coupling points of a segment should be in contact to deactivate it.

5. **Contact Thickness.** For this keyword, a segment, SG1, is in contact with another segment, SG2, if at least one coupling point of SG1 is inside a cylinder formed by SG2's area as its base and SG2's normal as its generatrix (see *SET_SEGMENT for the normal definition). The cylinder height is CONTHK. If CONTHK = 0 and the segment is attached to a shell (or beam in 2D), CONTHK by default is half the element thickness. If CONTHK = 0 and the segment is attached to a solid (or shell in 2D), CONTHK by default is a thousandth of the largest segment diagonal.

*CONTROL_SHELL

Purpose: Provide controls for computing shell response.

Card Summary:

Card 1. This card is required.

WRPANG	ESORT	IRNXX	ISTUPD	THEORY	BWC	MITER	PROJ

Card 2. This card and all the following cards are optional.

ROTASCL	INTGRD	LAMSHT	CSTYP6	THSHEL			

Card 3. This card is optional.

PSTUPD	SIDT4TU	CNTCO	ITSFLG	IRQUAD	W-MODE	STRETCH	ICRQ

Card 4. This card is optional.

NFAIL1	NFAIL4	PSNFAIL	KEEPCS	DELFR	DRCPSID	DRCPRM	INTPERR

Card 5. This card is optional.

DRCMTH	LISPSID	NLOCDT	ISWSHL				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	WRPANG	ESORT	IRNXX	ISTUPD	THEORY	BWC	MITER	PROJ
Type	F	I	I	I	I	I	I	I
Default	20.	0	-1	0	2	2	1	0

VARIABLE	DESCRIPTION
WRPANG	Shell element warpage angle in degrees. If a warpage greater than this angle is found, a warning message is printed.
ESORT	Sorting of triangular shell elements to automatically switch degenerate quadrilateral shell formulations to more suitable triangular

VARIABLE	DESCRIPTION
	shell formulations. EQ.0: Do not sort (default). EQ.1: Sort (switch to C0 triangular shell formulation 4, or if a quadratic shell, switch to shell formulation 24, or if a shell formulation with thickness stretch, switch to shell formulation 27). EQ.2: Sort (switch to DKT triangular shell formulation 17, or if the shell is quadratic, switch to shell formulation 24). The DKT formulation will be unstable for an uncommonly thick, triangular shell.
IRNXX	Shell normal update option. This option applies to the Hughes-Liu formulations (ELFORMs 1, 6, 7, and 11), the BWC formulation (ELFORM 10), and if warping stiffness is turned on, the Belytschko-Tsay formulations (ELFORMs 2 and 25 with BWC = 1 and ELFORMs -16, 16, and 26 with hourglass type 8). EQ.-2: Unique nodal fibers that are incrementally updated based on the nodal rotation at the location of the fiber (see Remark 1) EQ.-1: Recomputed fiber directions each cycle (see Remark 1) EQ.0: Default set to -1 EQ.1: Compute on restarts EQ.n: Compute every *n* cycles (Hughes-Liu shells only)
ISTUPD	Shell thickness change option for deformable shells. For crash analysis, neglecting the elastic component of the strains, ISTUPD = 4, may improve energy conservation and stability. The option specified with ISTUPD applies to all shell parts unless PSTUPD (and optionally, SIDT4TU) on Card 3 is specified. See the description of those variables below. EQ.0: No thickness change (default) EQ.1: Membrane straining causes thickness change in 3 and 4 node shell elements. This option is important in sheet metal forming or whenever membrane stretching is important. EQ.2: Membrane straining causes thickness change in 8 node thick shell elements, types 1 and 2. We do not recommend this option for implicit or explicit solutions which use the

VARIABLE	DESCRIPTION
	fully integrated type 2 elements. Types 3 and 5 thick shells are continuum-based, and thickness changes are always considered. EQ.3: Options 1 and 2 apply EQ.4: Option 1 applies, but the elastic strains are neglected for the thickness update. This option only applies to shells (not thick shells) and the most common elastic-plastic materials for which the elastic response is isotropic. See SIDT4TU for selective application of this option. EQ.5: Same as 1, but thickness changes are stored with double precision in single-precision binaries for shell elements with ELFORM = 2, 4, or 16. All other shell element types are stored with single precision in single-precision binaries. These storage changes should make results more comparable (single precision versus double precision) in long-lasting explicit analyses. Additionally, the internal energy computation for shell type 16 in single precision has improved accuracy with this option. EQ.6: Same as 4, but thickness changes are stored with double precision in single-precision binaries for shell elements with ELFORM = 2, 4, or 16. All other shell element types are stored with single precision in single-precision binaries. These storage changes should make results more comparable (single precision versus double precision) in long-lasting explicit analyses. Additionally, the internal energy computation for shell type 16 in single precision has improved accuracy with this option.
THEORY	Default shell formulation. For a complete list of shell formulations, refer to *SECTION_SHELL. For remarks on overriding this default and how THEORY may affect contact behavior, see Remark 3. EQ.1: Hughes-Liu EQ.2: Belytschko-Tsay (default) EQ.3: BCIZ triangular shell (not recommended) EQ.4: C0 triangular shell EQ.5: Belytschko-Tsay membrane EQ.6: S/R Hughes Liu EQ.7: S/R co-rotational Hughes Liu

VARIABLE	DESCRIPTION
	EQ.8: Belytschko-Leviathan shell
	EQ.9: Fully integrated Belytschko-Tsay membrane
	EQ.10: Belytschko-Wong-Chiang
	EQ.11: Fast (co-rotational) Hughes-Liu
	EQ.12: Plane stress (*xy*-plane)
	EQ.13: Plane strain (*xy*-plane)
	EQ.14: Axisymmetric solid (*y*-axis of symmetry) – area weighted. See Remark 8
	EQ.15: Axisymmetric solid (*y*-axis of symmetry) – volume weighted. See Remark 8
	EQ.16: Fully integrated shell element (very fast)
	EQ.17: Discrete Kirchhoff triangular shell (DKT)
	EQ.18: Discrete Kirchhoff linear shell either quadrilateral or Triangular with 6DOF per node
	EQ.20: C0 linear shell element with 6 DOF per node
	EQ.21: C0 linear shell element with 5 DOF per node with the Pian-Sumihara membrane hybrid quadrilateral membrane
	EQ.25: Belytschko-Tsay shell with thickness stretch
	EQ.26: Fully integrated shell with thickness stretch
	EQ.27: C0 triangular shell with thickness stretch
BWC	Warping stiffness for Belytschko-Tsay shells:
	EQ.1: Belytschko-Wong-Chiang warping stiffness added
	EQ.2: Belytschko-Tsay (default)
MITER	Plane stress plasticity option (applies to materials 3, 18, 19, and 24):
	EQ.1: Iterative plasticity with 3 secant iterations (default)
	EQ.2: Full iterative plasticity
	EQ.3: Radial return noniterative plasticity. May lead to false results and must be used with great care.
PROJ	Projection method for the warping stiffness in the Belytschko-Tsay shell (BWC above) and the Belytschko-Wong-Chiang elements (see Remark 2 below). This parameter only applies to explicit

VARIABLE	DESCRIPTION
	calculations since the implicit solver always uses the full projection method. EQ.0: Drill projection EQ.1: Full projection

Card 2	1	2	3	4	5	6	7	8
Variable	ROTASCL	INTGRD	LAMSHT	CSTYP6	THSHEL			
Type	F	I	I	I	I			
Default	1.	0	0	1	0			

VARIABLE	DESCRIPTION
ROTASCL	Define a scale factor for the rotary shell mass. This option is not for general use. The rotary inertia for shells is automatically scaled to permit a larger time step size. A scale factor other than the default, that is, unity, is not recommended.
INTGRD	Default through thickness numerical integration rule for shells and thick shells. If more than 10 integration points are requested, a trapezoidal rule is used unless a user-defined rule is specified. EQ.0: Gauss integration. If 1 - 10 integration points are specified, the default rule is Gauss integration. EQ.1: Lobatto integration. If 3 - 10 integration points are specified, the default rule is Lobatto. For two-point integration, the Lobatto rule is very inaccurate, so Gauss integration is used instead. Lobatto integration has an advantage in that the inner and outer integration points are on the shell surfaces.
LAMSHT	Laminated shell theory flag. Except for those using the Green-Lagrange strain tensor, laminated shell theory is available for all thin shell and thick shell materials (this feature is developed and implemented by Professor Ala Tabiei). It is activated when LAMSHT = 3, 4, or 5 and by using *PART_COMPOSITE or *INTEGRATION_SHELL to define the integration rule. See Remark 9.

VARIABLE	DESCRIPTION
	EQ.0: Do not update shear corrections. EQ.1: Activate laminated shell theory (deprecated). As of R16, the keyword reader changes LAMSHT = 1 to LAMSHT = 3. EQ.3: Activate laminated thin shells. EQ.4: Activate laminated shell theory for thick shells. EQ.5: Activate laminated shell theory for thin and thick shells.
CSTYP6	Coordinate system for the type 6 shell element. The default system computes a unique local system at each in-plane point. The uniform local system computes just one system used throughout the shell element. This involves fewer calculations and is therefore more efficient. The change of systems has a slight effect on results; therefore, the older, less efficient method is the default. EQ.1: Variable local coordinate system (default) EQ.2: Uniform local system
THSHEL	Thermal shell option (applies only to thermal and coupled structural thermal analyses). See parameter THERM on *DATABASE_EXTENT_BINARY keyword. EQ.0: No temperature gradient is considered through the shell thickness (default). EQ.1: A temperature gradient is calculated through the shell thickness.

Card 3	1	2	3	4	5	6	7	8
Variable	PSTUPD	SIDT4TU	CNTCO	ITSFLG	IRQUAD	W-MODE	STRETCH	ICRQ
Type	I	I	I	I	I	F	F	I
Default	0	0	0	0	3	inactive	inactive	0

VARIABLE	DESCRIPTION
PSTUPD	|PSTUPD| is the optional shell part set ID specifying which part IDs have or do not have their thickness updated according to ISTUPD,

VARIABLE	DESCRIPTION
	subject to further specification as given by SIDT4TU below. The shell thickness update option as specified by ISTUPD by default applies to all shell elements in the mesh. LT.0: Parts in shell part set |PSTUPD| are excluded from the shell thickness update. EQ.0: All deformable shells use the shell thickness update as specified by ISTUPD. GT.0: Only parts in shell part set PSTUPD undergo shell thickness update.
SIDT4TU	Shell part set ID for parts that use the type 4 thickness update where elastic strains are ignored. The shell parts in part set SIDT4TU must also be included in the part set defined by PSTUPD. SIDT4TU has no effect unless ISTUPD is set to 1 or 3. Shell parts in the shell part set PSTUPD that are not also in the shell part set SIDT4TU use the type 1 thickness update.
CNTCO	Flag affecting the location of contact surfaces for shells when NLOC is nonzero in *SECTION_SHELL or in *PART_COMPOSITE, or when OFFSET is specified using *ELEMENT_SHELL_OFFSET. CNTCO is not supported for the tracked side of NODES_TO_SURFACE type contacts, nor does it have any effect on Mortar contacts. For Mortar contacts NLOC or OFFSET completely determines the location of the contact surfaces, as if CNTCO = 1. EQ.0: NLOC and OFFSET have no effect on the location of the shell contact surfaces. EQ.1: Contact reference plane (see Remark 4) coincides with the shell reference surface. EQ.2: Contact reference plane (see Remark 4) is affected by contact thickness. This is typically not physical. EQ.3: Similar to 1 but with improved behavior at corners or folds in the mesh in certain cases. See Remarks 4 and 5. EQ.4: Similar to 2 but with improved behavior at corners or folds in the mesh in certain cases. See Remarks 4 and 5.
ITSFLG	Flag to activate/deactivate initial transverse shear stresses (local shell stress components σ_{yz} and σ_{zx}) from *INITIAL_STRESS_SHELL: EQ.0: Keep transverse shear stresses

VARIABLE	DESCRIPTION
	EQ.1: Set transvers shear stresses to zero
IRQUAD	In-plane integration rule for the 8-node quadratic shell element (shell formulation 23): EQ.2: 2 × 2 Gauss quadrature EQ.3: 3 × 3 Gauss quadrature (default)
W-MODE	W-Mode amplitude for element deletion, specified in degrees. See Figure 12-88 and Remark 7 for the definition of the angle.
STRETCH	Stretch ratio of element diagonals for quadrilateral elements or of side edges for triangular elements that results in element deletion. This option is activated only if either NFAIL1 or NFAIL4 are non-zero and STRETCH > 0.0.
ICRQ	Continuous treatment across element edges for some specified result quantities. See Remark 10. EQ.0: Not active EQ.1: Thickness and plastic strain EQ.2: Thickness

Card 4	1	2	3	4	5	6	7	8
Variable	NFAIL1	NFAIL4	PSNFAIL	KEEPCS	DELFR	DRCPSID	DRCPRM	INTPERR
Type	I	I	I	I	I	I	F	I
Default	inactive	inactive	0	0	0	0	1.0	0

VARIABLE	DESCRIPTION
NFAIL1	Flag to check for highly distorted under-integrated shell elements, print a message, and delete the element or terminate. Generally, this flag is not needed for one point elements that do not use warping stiffness. A distorted element is one where a negative Jacobian exist within the domain of the shell, not just at integration points. The checks are made away from the CPU requirements for one point elements. If nonzero, NFAIL1 can be changed in a restart.

VARIABLE	DESCRIPTION
	EQ.1: Print message and delete element. EQ.2: Print message, write d3dump file, and terminate. GT.2: Print message and delete element. When NFAIL1 elements are deleted, then write d3dump file and terminate. These NFAIL1 failed elements also include all shell elements that failed for other reasons than distortion. Before the d3dump file is written, NFAIL1 is doubled, so the run can immediately be continued if desired.
NFAIL4	Flag to check for highly distorted fully-integrated shell elements, print a message, and delete the element or terminate. Generally, this flag is recommended. A distorted element is one where a negative Jacobian exist within the domain of the shell, not just at integration points. The checks are made away from the integration points to enable the bad elements to be deleted before an instability leading to an error termination occurs. If nonzero, NFAIL4 can be changed in a restart. EQ.1: Print message and delete element. EQ.2: Print message, write d3dump file, and terminate. GT.2: Print message and delete element. When NFAIL4 elements are deleted, then write d3dump file and terminate. These NFAIL4 failed elements also include all shell elements that failed for other reasons than distortion. Before the d3dump file is written, NFAIL4 is doubled, so the run can immediately be continued if desired.
PSNFAIL	Optional shell part set ID specifying which part IDs are checked by the NFAIL1, NFAIL4, and W-MODE options. If zero, all shell part IDs are included.
KEEPCS	Flag to keep the contact segments of failed shell elements in the calculation. The contact segments of the failed shells remain active until a node shared by the segments has no active shells attached. Only then are the segments deleted. EQ.0: Inactive EQ.1: Active
DELFR	Flag to delete shell elements whose neighboring shell elements have failed; consequently, the shell is detached from the structure and moving freely in space. This condition is checked if NFAIL1

VARIABLE	DESCRIPTION
	or NFAIL4 are nonzero. EQ.0: Inactive EQ.1: Isolated elements are deleted. EQ.2: Elements that are isolated and triangular elements that are connected by only one node are deleted. EQ.3: Elements that are either isolated or connected by only one node are deleted.
DRCPSID	Part set ID for drilling rotation constraint method (see Remark 6 below).
DRCPRM	Drilling rotation constraint parameter (default = 1.0). GT.0: Constant scaling value. LT.0: |DRCPRM| is a *DEFINE_FUNCTION ID (FID), see Remark 6 below.
INTPERR	Flag for behavior in case of unwanted interpolation/extrapolation of initial stresses from *INITIAL_STRESS_SHELL. EQ.0: Only warning is written, calculation continues (default). EQ.1: Error exit, calculation stops.

Card 5	1	2	3	4	5	6	7	8
Variable	DRCMTH	LISPSID	NLOCDT	ISWSHL				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
DRCMTH	Drilling rotation constraint method. Options to choose how drilling kinematics are determined. EQ.0: Generalized drilling strain rate at shell element nodes involving drill rotation at the specific node plus the translational velocities of two adjacent nodes. See more details in Erhart and Borrvall [2013].

VARIABLE	DESCRIPTION
	EQ.1: Direct use of the spin tensor (for example, see Stress Update Overview in the LS-DYNA Theory Manual) with respect to the shell element's normal direction, numerically integrated at element level. A similar approach is described in Kanok-Nukulchai [1979].
LISPSID	Part set ID related to *INITIAL_STRESS_SHELL. For all parts in this set, the initial stress components SIGXX, SIGYY, …, SIGZX are defined in the local (element) coordinate system.
NLOCDT	Flag for time step handling for shell elements with offset. If the shell reference surface is offset by NLOC (*SECTION_SHELL) or OFFSET (*ELEMENT_SHELL), the time step size of those shell elements is reduced to fix instabilities. The reduction of the time step size is based on numerical tests which show a dependence on the offset distance and the ratio of shell thickness to edge length (T/L). EQ.0: Reduce time step size up to 10% to avoid instabilities. Care must be taken since a smaller time step will lead to larger masses due to mass scaling. EQ.1: No reduction of time step to restore prior behavior if necessary. Instabilities were most likely observed for aspect ratios of $T/L > 0.5$.
ISWSHL	Flag for switching between formulations 16 and 30: EQ.0: Do not convert the shell formulations. EQ.1: Convert all formulation 16 shells to 30. EQ.2: Convert all formulation 30 shells to 16.

Remarks:

1. **IRNXX for curved geometries.** We recommend IRNXX = -1 over IRNXX = -2 for curved geometries. IRNXX = -2 initializes the fibers to be the nodal average normals. Thus, in curved geometries, it can lead to non-objective results. In contrast, IRNXX = -1 sets the fibers to the elementwise constant normals, leading to more objective results.

2. **Drill versus full projections for warping stiffness.** The drill projection is used in the addition of warping stiffness to the Belytschko-Tsay and the Belytschko-Wong-Chiang shell elements. This projection generally works well and is very efficient, but to quote Belytschko and Leviathan:

> "The shortcoming of the drill projection is that even elements that are invariant to rigid body rotation will strain under rigid body rotation if the drill projection is applied. On one hand, the excessive flexibility rendered by the 1-point quadrature shell element is corrected by the drill projection, but on the other hand, the element becomes too stiff due to loss of the rigid body rotation invariance under the same drill projection."

They later went on to add in the conclusions:

> "The projection of only the drill rotations is very efficient and hardly increases the computation time, so it is recommended for most cases. However, it should be noted that the drill projection can result in a loss of invariance to rigid body motion when the elements are highly warped. For moderately warped configurations the drill projection appears quite accurate."

In crashworthiness and impact analysis, elements that have little or no warpage in the reference configuration can become highly warped in the deformed configuration and may affect rigid body rotations if the drill projection is used, that is, DO NOT USE THE DRILL PROJECTION. Of course, it is difficult to define what is meant by "moderately warped." The full projection circumvents these problems but at a significant cost. The cost increase of the drill projection versus no projection as reported by Belytschko and Leviathan is 12 percent and by timings in LS-DYNA, 7 percent, but for the full projection they report a 110 percent increase, and in LS-DYNA an increase closer to 50 percent is observed.

In Version 940 of LS-DYNA, the drill projection was used exclusively, but in one problem the lack of invariance was observed; consequently, the drill projection was replaced in the Belytschko-Leviathan shell with the full projection and the full projection is now optional for the warping stiffness in the Belytschko-Tsay and Belytschko-Wong-Chiang elements. Starting with version 950 the Belytschko-Leviathan shell, which now uses the full projection, is somewhat slower than in previous versions. In general, in light of these problems, the drill projection cannot be recommended. For implicit calculations, the full projection method is used in the development of the stiffness matrix.

3. **THEORY, ELFORM, and contact with tapered shells.** All shell parts need not share the same element formulation. A nonzero value of ELFORM, given either in *SECTION_SHELL or *PART_COMPOSITE, overrides the element formulation specified by THEORY in *CONTROL_SHELL.

   When using MPP, THEORY = 1 in *CONTROL_SHELL has a special meaning when dealing with non-uniform-thickness shells: it serves to set the nodal contact thickness equal to the average of the nodal thicknesses from the shells sharing that node. Thus, when non-uniform-thickness shells comprise a contact surface, we recommend THEORY = 1 and setting the actual shell theory using

ELFORM in *SECTION_SHELL. (This remark concerning THEORY does not apply to SOFT = 2 contact wherein each contact segment is considered to be of uniform thickness.)

4. **Contact surfaces.** For automatic contact types, the shell contact surfaces are always, regardless of CNTCO, offset from a contact reference plane by half a contact thickness. Contact thickness is taken as the shell thickness by default but can be overridden, for example, with input on Card 3 of *CONTACT.

   The parameter CNTCO affects how the location of the contact reference plane is determined. When CNTCO = 0, the contact reference plane coincides with the plane of the shell nodes. Whereas when CNTCO = 1 or 3, the contact reference plane coincides with the shell reference surface as determined by NLOC or by OFFSET. For CNTCO = 2 or 4, the contact reference plane is offset from the plane of the nodes by

$$-\frac{\text{NLOC}}{2} \times \text{contact thickness}$$

   or by

$$\text{OFFSET} \times \left(\frac{\text{contact thickness}}{\text{shell thickness}}\right)$$

   whichever applies.

5. **Improvements with CNTCO = 3 or 4.** CNTCO = 3 and 4 are similar to 1 and 2, respectively, but they include improvements in determining the contact reference plane at edges and corners in the mesh (including internal) where adjacent segments are not coplanar and contact thicknesses or areas of adjacent segments differ.

6. **Drilling rotation constraint method.** The drilling rotation constraint method, which is used by default in implicit calculations (see parameter DRCM = 4 on *CONTROL_IMPLICIT_SOLVER), can be used in explicit calculations as well by defining an appropriate part set DRCPSID. This might be helpful in situations where single constraints (such as spot welds) are connected to flat shell element topologies. The additional drill force can be scaled by DRCPRM (default value is 1.0), where a moderate value should be chosen to avoid excessive stiffening of the structure. A speed penalty of 15% at maximum may be observed with this option.

   As an alternative to a constant drill force scaling parameter (DRCPRM > 0), a general function can be defined (DRCPRM < 0) with *DEFINE_FUNCTION ID |DRCPRM|. The arguments of that function include elastic shear modulus *gmod*, the shear correction factor *shrf* (from *SECTION_SHELL), the mass density *rho*, the shell element area *sarea*, and the element thickness *thk* in the following order:

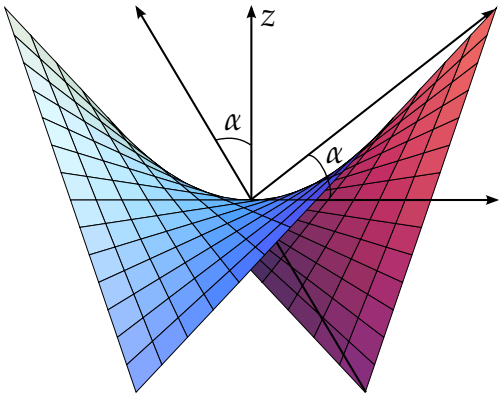


Figure 12-88. Illustration of an element in a *W-Mode*. One pair of opposite corners go up, and the other pair goes down. The angle, α, is formed by the plane of the flat element and by the vector connecting the center to the corner. See Remark 7.

```
*DEFINE_FUNCTION
<FID>
func(gmod,shrf,rho,sarea,thk)=...
```

The function should have the units of a force.

7. **W-mode failure criterion.** The w-mode failure criterion depends on the magnitude of the w-mode, w, compared to the approximate side length ℓ. The magnitude, w, is defined as

$$w = \frac{1}{4}[(\mathbf{x}_1 - \mathbf{x}_2) + (\mathbf{x}_3 - \mathbf{x}_4)] \cdot \mathbf{n} \ ,$$

where $\mathbf{x}_i$ is the position vector for node i, and $\mathbf{n}$ is the element normal vector evaluated at the centroid. The element normal is the unit vector obtained from the cross product of the diagonal vectors $\mathbf{a}$ and $\mathbf{b}$ as,

$$\begin{aligned} \mathbf{a} &= \mathbf{x}_3 - \mathbf{x}_1 \\ \mathbf{b} &= \mathbf{x}_4 - \mathbf{x}_2 \\ \mathbf{n} &= \frac{\mathbf{a} \times \mathbf{b}}{\|\mathbf{a} \times \mathbf{b}\|} \ . \end{aligned}$$

The failure criterion depends on the ratio of w to ℓ, where ℓ is defined as

$$\ell = \frac{1}{2}\underbrace{\left[\sqrt{2}\underbrace{\sqrt{\frac{1}{2}\|\mathbf{a} \times \mathbf{b}\|}}_{\sim\sqrt{\text{area}}}\right]}_{\sim\text{diagonal length}}$$

such that the element is deleted when

$$\frac{|w|}{\ell} \geq \tan(\text{WMODE}).$$

The angle α in the figure may be identified as,

$$\alpha = \arctan\left(\frac{|w|}{\ell}\right) .$$

8. **2D axisymmetric solid elements.** The 2D axisymmetric solid elements come in two types: area-weighted (type 14) and volume-weighted (type 15).

   a) High explosive applications work best with the area-weighted approach and structural applications work best with the volume-weighted approach. The volume-weighted approach can lead to problems along the axis of symmetry under very large deformations. Often the symmetry condition is not obeyed, and the elements will kink along the axis.

   b) The volume-weighted approach must be used if 2D shell elements are used in the mesh. Type 14 and 15 elements cannot be mixed in the same calculation.

9. **Lamination theory.** Lamination theory should be activated when the assumption that shear strain through the shell is uniform and constant becomes violated. Unless this correction is applied, the stiffness of the shell can be grossly incorrect if there are drastic differences in the elastic constants from ply to ply, especially for sandwich-type shells. Generally, without this correction, the results are too stiff. For the discrete Kirchhoff shell elements, which do not consider transverse shear, this option is ignored.

10. **Continuous result quantities**. A nodal averaging technique is used to achieve continuity for some quantities across element edges. Applying this approach to the thickness field and plastic strains (ICRQ = 1) or only thickness (ICRQ=2) can reduce alternating weak localizations sometimes observed in metal forming applications when shell elements get stretch-bended over small radii. This option currently works with shell element types 2, 4, and 16. A maximum number of 9 through-thickness integration points is allowed for this method. A maximum speed penalty of 15% may be observed with this option.

11. **Obsolete Card 2 format**. Prior to 2010, the first three fields of Card 4 were read from Card 2 as fields six through eight. This obsolete card format is still supported by the solver for backward compatibility. However, LS-PrePost may only recognize the current format. Note that if Card 4 is included, the values set on Card 4 override those set on Card 2.

Obsolete Version of Card 2.

Card 2	1	2	3	4	5	6	7	8
Variable	ROTASCL	INTGRD	LAMSHT	CSTYP6	THSHEL	NFAIL1	NFAIL4	PSNFAIL
Type	F	I	I	I	I	I	I	I
Default	1.	0	0	1	0	inactive	inactive	0

*CONTROL_SOLID

Purpose: Provides controls for solid element response.

Card 1	1	2	3	4	5	6	7	8
Variable	ESORT	FMATRX	NIPTETS	SWLOCL	PSFAIL	T10JTOL	ICOH	TET13K
Type	I	I	I	I	I	F	I	I
Default	0	0	4	1	0	0.0	0	0

This card is optional.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none	none	none

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	TET13V	RINRT	COHEQC					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
ESORT	Automatic sorting of tetrahedral and pentahedral elements to avoid using degenerate formulations for these shapes. See *SECTION_SOLID. EQ.0: No sorting (default) EQ.1: Sort tetrahedron to type 10; pentahedron to type 15;

VARIABLE	DESCRIPTION
	cohesive pentahedron types 19 and 20 to types 21 and 22, respectively.
	EQ.2: Sort tetrahedron to type 10; 1-point integrated pentahedron to type 115; fully integrated pentahedron to type 15; cohesive pentahedron types 19 and 20 to types 21 and 22, respectively.
	EQ.3: Same as 1 but also print switched elements in messag file
	EQ.4: Same as 2 but also print switched elements in messag file
	EQ.11: Same a 1 except sort tetrahedron to type 13
	EQ.12: Same a 2 except sort tetrahedron to type 13
	EQ.13: Same a 3 except sort tetrahedron to type 13
	EQ.14: Same a 4 except sort tetrahedron to type 13
FMATRX	Default method used in the calculation of the deformation gradient matrix.
	EQ.1: Update incrementally in time. This is the default for explicit.
	EQ.2: Directly compute **F**: This is the default for implicit and implicit/explicit switching.
NIPTETS	Number of integration points used in the quadratic tetrahedron elements. Either 4 or 5 can be specified. This option applies to the types 4, 16, and 17 tetrahedron elements.
SWLOCL	Output option for stresses in solid elements used as spot welds with material *MAT_SPOTWELD. Affects elout, d3plot, d3part, etc.
	EQ.1: Stresses in the global coordinate system (default)
	EQ.2: Stresses in the element coordinate system
PSFAIL	Solid element erosion from negative volume is limited *only* to solid elements in the part set indicated by PSFAIL. This is similar to setting ERODE = 1 in *CONTROL_TIMESTEP, except that it is not global. In other words, when PSFAIL is nonzero, the time-step-based criterion for erosion (TSMIN) applies to all solid elements (except formulations 11 and 12), and the negative volume criterion

VARIABLE	DESCRIPTION
	for erosion applies only to solids in part set PSFAIL.
T10JTOL	Tolerance for Jacobian in 4-point 10-noded quadratic tetrahedra (type 16). If the quotient between the minimum and maximum Jacobian values falls below this tolerance, a warning message is issued in the messag file. This is useful for tracking (poorly shaped) elements that deteriorate convergence during implicit analysis; a value of 1.0 indicates a perfectly shaped element.
ICOH	Flag for cohesive elements to control deletion, the time step estimate, and the element type used in implicit and explicit. Breaking LS-DYNA convention, ICOH is interpreted digit-wise, namely as, $$\text{ICOH} = [MLK] = K + 10 \times L + 100 \times M\ .$$ The first digit (in the one's place), K, is interpreted as follows: K.EQ.0: No cohesive element deletion due to neighbor failure. K.EQ.1: Solid elements having ELFORM = 19 – 22 (or ELFORM = 1, 2, 15 being used with *MAT_169) are eroded when neighboring shell or solid elements fail. This works for nodewise connected parts and tied contacts. The second digit (in the ten's place), L, controls how the mass used to estimate the element time step is computed. L defaults to zero if ICOH is less than 10 (having a single digit). See Remark 1. L.EQ.0: The mass is the sum of the element nodes. Each node is assumed to be shared between two elements. L.EQ.1: The mass is computed by integrating the cohesive density. L.EQ.2: Same as the default (0), but each node is assumed to be shared by four elements. Compared to the default, the time step is reduced by a factor of $1/\sqrt{2}$. L.EQ.3: The element connectivity is used to compute how much (nodal) mass is associated with each element, meaning no assumption regarding connectivity is made. This setting is recommended. The third digit (in the hundred's place), M, is interpreted as follows: M.EQ.0: Explicit element variants used in explicit calculations and higher-accuracy implicit element variants used in implicit calculations.

VARIABLE	DESCRIPTION
	M.EQ.1: Higher-accuracy implicit element variants used in both implicit and explicit calculations.
TET13K	Set to 1 to invoke a consistent tangent stiffness matrix for the pressure-averaged tetrahedron (type 13). This feature is available only for the implicit integrator. This element type averages the volumetric strain over adjacent elements to alleviate volumetric locking; therefore, the corresponding material tangent stiffness should be treated accordingly. In contrast to a hexahedral mesh, where a node usually connects to fewer than eight elements, tetrahedral meshes offer no such regularity. Consequently, matrix assembly is computationally expensive for nonlinear implicit analysis, so this option is recommended only for linear or eigenvalue analysis to exploit the stiffness characteristics of the type 13 tetrahedron.
PM1 – PM10	Components of a permutation vector for nodes that define the 10-node tetrahedron. The nodal numbering of 10-node tetrahedron elements is somewhat arbitrary. The permutation vector allows other numbering schemes to be used. Unless specified, this permutation vector is not used. PM1 – PM10 are unique numbers between 1 to 10 inclusive that reorder the input node IDs for a 10-node tetrahedron into the order used by LS-DYNA.
TET13V	Choice of implementation for solid element types 10 and 13. If EXACC on *CONTROL_ACCURACY is greater than zero, TET13V is set to one. EQ.0: Efficient version for type 10 and more accurate version for type 13 (default). With the single precision version, a little noise could be observed in the solution for elements moving long distances with rigid body motion. EQ.1: More accurate version for both types 10 and 13 (smoother results) with an additional cost of about 15%. EQ.2: Efficient version for both types 10 and 13.
RINRT	Option to compute rotational inertia for the nodes of solid elements. This ensures consistent results if the applied constraints assume rotational degrees of freedom, as with tied contacts using the option SHELL_EDGE_TO_SURFACE. EQ.0: By default, an average of the existing rotational inertia from the shell and beam elements in the model is distributed to the nodes of the solid elements. This method is sufficient in most situations but might lead to

VARIABLE	DESCRIPTION
	inconsistencies between different model assemblies in the case of rotational motion.
	EQ.1: Compute rotational inertia for each solid element based on its dimensions and mass density to ensure consistency.
COHEQC	Flag for cohesive element quality checks. Note that this check is only performed for cohesive elements with materials that have ROFLG = 0. COHEQC is interpreted digit-wise, namely as, $$\text{COHEQC} = [LK] = K + 10 \times L$$ The first digit, K, controls the behavior of LS-DYNA when cohesive elements with poor quality are detected at $t = 0$:
	K.EQ.0: Error termination with a list of elements (default).
	K.EQ.1: Issue a warning and continue.
	K.EQ.2: Same as 1, but delete the concerned elements.
	The second digit, L, controls the behavior of LS-DYNA when inconsistent cohesive element connectivity is detected:
	L.EQ.0: Error termination with a list of elements (default).
	L.EQ.1: Issue a warning and continue.

Remarks:

1. **Computation of cohesive element time step.** As mentioned in the Theory Manual, we can estimate the stable time step of a cohesive element by representing the element as a mass-spring element. The critical time step is then:

$$\Delta t_e = \sqrt{\frac{2m}{k}},$$

where

$$m = \frac{2m_b m_t}{m_b + m_t}$$

is the harmonic mean of the top and bottom spring masses m_t and m_b. The equivalent spring stiffness (force per length) is

$$k = \sum_{i=1}^{N_\text{IP}} \max[E_n, E_t]\, w_i J_i = \max[E_n, E_t]\, A\ ,$$

where E_n and E_t are the normal/tangential cohesive stiffness (stress per length), w_i are weights, and A is the interface area. Note that the last step assumes a constant stiffness and Jacobian determinant over the element.

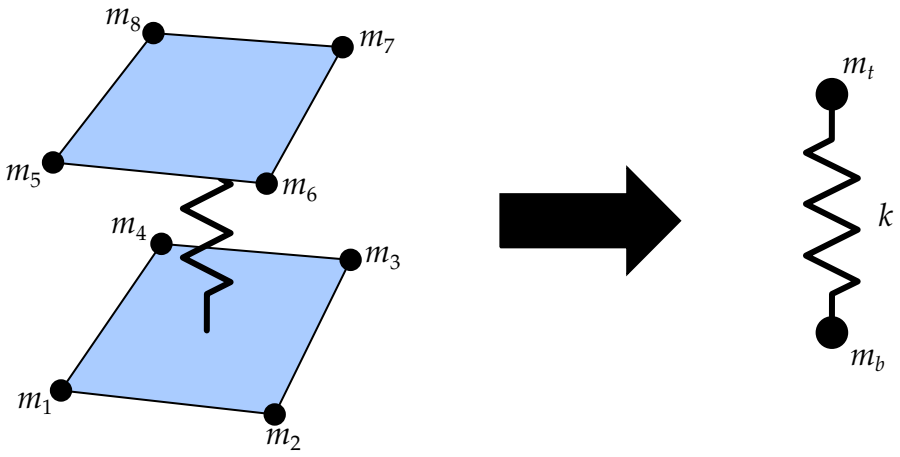


Figure 12-89. Representing a cohesive element as a mass-spring element.

The setting of L in the ICOH parameters determines the method for computing the top and bottom masses. For $L = 0$, LS-DYNA treats the cohesive element like a series of mass-spring elements stacked on each other. Two elements share each mass, resulting in

$$m_t = \frac{1}{2} M_t$$
$$m_b = \frac{1}{2} M_b$$

where M_t and M_b are the sum of the top and bottom masses, respectively. That is,

$$M_t = m_5 + m_6 + m_7 + m_8$$
$$M_b = m_1 + m_2 + m_3 + m_4$$

Note that for a pentahedral element, $m_4 = m_8 = 0$. The resulting time step is

$$\Delta t_e = \sqrt{\frac{2 M_b M_t}{k(M_b + M_t)}}.$$

For $L = 1$, the mass contribution is from the cohesive element only. Thus,

$$m = \frac{1}{2} H \rho A$$

or

$$m = \frac{1}{2} \rho A \ ,$$

depending on whether the provided density is mass per volume or mass per area. In the above, H is the thickness. The resulting time step is

$$\Delta t_e = \sqrt{\frac{H\rho}{\max[E_n, E_t]}}$$

or

$$\Delta t_e = \sqrt{\frac{\rho}{\max[E_n, E_t]}} \ .$$

For $L = 2$, we assume a regular mesh such that four neighboring elements share each corner node. Thus, we have

$$m_t = \frac{1}{4} M_t$$
$$m_b = \frac{1}{4} M_{\mathrm{b}}$$

These masses result in a time-step estimation of

$$\Delta t_e = \sqrt{\frac{M_b M_t}{k(M_b + M_t)}},$$

which is a factor of $\sqrt{2}$ smaller than the default $L = 0$.

For $L = 3$, we make no assumption regarding element connectivity. Instead, we compute how much of the shared mass is associated with each cohesive element. The benefit of this is that the cohesive time step estimate is the same throughout the cohesive interface. Thus, the time step estimate for elements at the edges is the same as elements inside the cohesive interface, which is not the case for options 0 or 2 above.

All options, except $L = 1$, allow for zero cohesive density. Furthermore, option $L = 3$, results in the theoretical time step for two surfaces connected by cohesive interfaces, so we recommend using this option.

*CONTROL_SOLUTION

Purpose: To specify the analysis solution procedure if thermal only or combined thermal analysis is performed. Other solution parameters including the vector length and NaN (not a number) checking can be set.

Card 1	1	2	3	4	5	6	7	8
Variable	SOLN	NLQ	ISNAN	LCINT	LCACC	NCDCF	NOCOPY	CRVP
Type	I	I	I	I	I	I	I	I
Default	0	0	0	100	0	1	0	0

VARIABLE	DESCRIPTION
SOLN	Analysis solution procedure: EQ.0: Structural analysis only EQ.1: Thermal analysis only EQ.2: Combined structural, multiphysics, and thermal analysis
NLQ	Define the vector length used in the solution. This value must not exceed the vector length of the system which varies based on the machine manufacturer. The default vector length is printed at termination in the messag file.
ISNAN	Flag to check for a NaN in the force and moment arrays after the assembly of these arrays is completed. This option can be useful for debugging purposes. A cost overhead of approximately 2% is incurred when this option is active. EQ.0: No checking EQ.1: Checking is active.
LCINT	Number of equally spaced points used in curve (*DEFINE_CURVE) rediscretization. A minimum number of 100 is always used, that is, only larger input values are possible. All curves are automatically rediscretized, but the rediscretized version of a given curve is not always used. The original curves are used when the ordinate is a function of time (such as motion, loads, etc.). In most other cases, the rediscretized curve is used. Note that memory requirements increase as LCINT increases. Thus, extremely large

VARIABLE	DESCRIPTION
	values of LCINT should be avoided if possible.
LCACC	Flag to truncate curves to 6 significant figures for single precision and 13 significant figures for double precision. The truncation is done after applying the offset and scale factors specified in *DEFINE_CURVE. Truncation is intended to prevent curve values from deviating from the input value, such as 0.7 being stored as 0.69999999. This small deviation was seen to have an adverse effect in a particular analysis using *MAT_083. In general, curve truncation is not necessary and is unlikely to have any effect on results. EQ.0: No truncation NE.0: Truncate
NCDCF	Global option to evaluate *DEFINE_CURVE_FUNCTION every NCDCFth cycle.
NOCOPY	*This field is currently disabled with NOCOPY always set to zero.* Avoid copying material history variables to temporary buffers for constitutive evaluations (see Remark 1): EQ.0: Not active EQ.1: Active
CRVP	Bypass time-based evaluation of non-time-dependent curves, e.g., stress-strain curves for materials. This can improve CPU performance when many curves are used (e.g. from a big material database) in smaller models. EQ.0: Not active EQ.1: Active EQ.2: Same as 1, but with half the memory requirement for the rediscretized curves.

Remarks:

1. **NOCOPY.** Setting NOCOPY = 1 reduces overhead during element calculations and could potentially decrease the *Element processing* in *Timing Information* between 5% and 25% depending on the choice of element and material. The most substantial gain is obtained when the material is combined with a modular damage model, such as GISSMO (see *MAT_ADD_DAMAGE_GISSMO) or DIEM (see *MAT_ADD_DAMAGE_DIEM), for which another level of copying is avoided.

Currently this option is *restricted to MPP* (not hybrid). It also only works for materials 24 and 123 in combination with shell elements 2, 4 or 16, solid elements -1, -2, 1, 2, 10 or 15, and all thick shells and for materials 36 and 133 in combination with shell elements 2, 4, or 16. Other combinations of materials and elements ignore this option and will be supported in an order of priority/importance.

*CONTROL_SPH

Purpose: Provide controls related to SPH (Smooth Particle Hydrodynamics).

Card 1	1	2	3	4	5	6	7	8
Variable	NCBS	BOXID	DT	IDIM	NMNEIGH	FORM	START	MAXV
Type	I	I	F	I	I	I	F	F
Default	1	0	10^{20}	none	150	0	0.0	10^{15}

Remaining cards are optional.†

Card 2	1	2	3	4	5	6	7	8
Variable	CONT	DERIV	INI	ISHOW	IEROD	ICONT	IAVIS	ISYMP
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	100

Card 3	1	2	3	4	5	6	7	8
Variable	ITHK	ISTAB	QL		SPHSORT	ISHIFT		
Type	I	I	F		I	I		
Default	0	0	0.01		0	0		

VARIABLE	DESCRIPTION
NCBS	Number of time steps between particle sorting.
BOXID	SPH approximations are computed inside a specified box (see *DEFINE_BOX). When a particle has gone outside the BOX, it is deactivated. This will save computational time by eliminating particles that no longer interact with the structure.

VARIABLE	DESCRIPTION
DT	Death time. Determines when the SPH calculations are stopped.
IDIM	Space dimension for SPH particles: EQ.3: 3D problems EQ.2: 2D plane strain problems EQ.-2: 2D axisymmetric problems (see Remark 2)
NMNEIGH	Defines the initial number of neighbors per particle (see Remark 1 below).
FORM	Particle approximation theory (Remark 2): EQ.0: Default formulation EQ.1: Renormalization approximation EQ.2: Symmetric formulation EQ.3: Symmetric renormalized approximation EQ.4: Tensor formulation EQ.5: Fluid particle approximation EQ.6: Fluid particle with renormalization approximation EQ.7: Total Lagrangian formulation EQ.8: Total Lagrangian formulation with renormalization EQ.9: Adaptive SPH formulation (ASPH) with anisotropic smoothing tensor (Remark 2g) EQ.10: Renormalization approximation for adaptive SPH formulation (ASPH) with anisotropic smoothing tensor EQ.12: Moving least-squares based formulation (MPP only, see Remark 2e) EQ.13: Incompressible formulation (MPP only, see Remark 2h) EQ.15: Enhanced fluid formulation EQ.16: Enhanced fluid formulation with renormalization
START	Start time for particle approximation. Particle approximations will be computed when time of the analysis has reached the value defined in START.

VARIABLE	DESCRIPTION
MAXV	Maximum magnitude of the velocity for SPH particles. Particles with a velocity magnitude greater than MAXV are deactivated. A value of 0.0 will turn off the velocity checking. If a negative value is used, particles with a velocity magnitude greater than |MAXV| will be reassigned a velocity field with a magnitude equal to |MAXV|.
CONT	Flag for inter-part particle interaction by "particle approximation": EQ.0: Particle approximation is used for computing inter-part particle interaction for all SPH parts (default). EQ.1: Particle approximation is not used for inter-part particle interaction, except as specified by the INTERACTION option of *SECTION_SPH. As an alternative to particle approximation, inter-part particle interaction can be accomplished using *DEFINE_SPH_TO_SPH_COUPLING.
DERIV	Time integration type for the smoothing length: EQ.0: $\frac{d}{dt}[h(t)] = \frac{1}{d}h(t)\nabla \cdot \mathbf{v}$, (default) EQ.1: $\frac{d}{dt}[h(t)] = \frac{1}{d}h(t)(\nabla \cdot \mathbf{v})^{1/3}$
INI	Computation of the smoothing length during the initialization: EQ.0: Bucket-sort-based algorithm (default, very fast) EQ.1: Global computation on all the particles of the model EQ.2: Based on the mass of the SPH particle
ISHOW	Display option for deactivated SPH particles: EQ.0: No distinction in active SPH particles and deactivated SPH particles when viewing in LS-PrePost EQ.1: Deactivated SPH particles are displayed only as points and active SPH particles are displayed as spheres when Setting → SPH → Style is set to "smooth" in LS-PrePost.
IEROD	Deactivation control for SPH particles: EQ.0: Particles remain active. See Remark 3. EQ.1: SPH particles are partially deactivated and stress states are set to 0 when erosion criteria are satisfied. See Remark 3.

VARIABLE	DESCRIPTION
	EQ.2: SPH particles are totally deactivated and stress states are set to 0 when erosion criteria are satisfied. See Remark 3.
	EQ.3: SPH particles are partially deactivated and stress states are set to 0 when erosion criteria are satisfied. If an EOS is defined, the volumetric response is unaffected. See Remark 3.
ICONT	Controls contact behavior for deactivated SPH particles:
	EQ.0: Any contact defined for SPH remains active for deactivated particles.
	EQ.1: Contact is inactive for deactivated particles.
IAVIS	Defines artificial viscosity formulation for SPH elements (Remark 4):
	EQ.0: Monaghan-type artificial viscosity formulation is used.
	EQ.1: Standard type artificial viscosity formulation from solid element is used (this option is not supported in SPH 2D and 2D axisymmetric elements).
ISYMP	Defines the percentage of original SPH particles used for memory allocation when generating the ghost nodes for SPH symmetric planes (default is 100%). Setting this field is recommended for large SPH particle models (value range 10~20) to control the memory allocation for SPH ghost particles when using *BOUNDARY_SPH_SYMMETRY_PLANE.
ITHK	Contact thickness option:
	EQ.0: The contact thickness is set to zero (default).
	EQ.1: The contact thickness is automatically calculated based on the volume of each SPH particle.
	This contact thickness calculation is ignored if the SURFA surface has a nonzero contact thickness (SAST on Card 3 of *CONTACT_...).
ISTAB	Stabilization type, only used when IFORM = 12:
	EQ.0: Incremental stabilization (default). Adequate for most materials.
	EQ.1: Total stabilization. Only recommended for hyperelastic materials.

VARIABLE	DESCRIPTION
QL	Quasi-linear coefficient, only used when IFORM = 12. See Remark 5.
SPHSORT	For the implicit solver, sort and move SPH nodes from *NODE list to the end of the list. EQ.0: No sorting (default) EQ.1: Perform sorting
ISHIFT	Flag for applying the shifting algorithm to SPH particles (available from R13.0). With the shifting algorithm, particles are advanced and then shifted slightly across streamlines. This process reduces particle clustering in the maximum compression and stretching directions. EQ.0: Do not apply shifting algorithm applied (default). EQ.1: Apply shifting algorithm applied.

Remark:

1. **NMNEIGH.** NMNEIGH is used to determine the initial memory allocation for the SPH arrays. Its value can be positive or negative. If NMNEIGH is positive, memory allocation is dynamic such that the number of neighboring particles is initially equal to NMNEIGH but that number is subsequently allowed to exceed NMNEIGH as the solution progresses. Note that setting NMNEIGH to a value greater than 300 may cause out-of-memory issues for large models. If NMNEIGH is negative, memory allocation is static and |NMNEIGH| is the maximum allowed number of neighboring particles for each particle throughout the entire solution. Using this static memory option can avoid memory allocation problems.

2. **FORM.** The right value for FORM depends upon the application. Below are some guidelines for selecting FORM:

   a) For most solid structure applications, FORM = 1 is recommended for more accurate results around the boundary area.

   b) For fluid or fluid-like material applications, FORM = 15 or 16 is recommended. FORM = 16 usually has better accuracy but requires more CPU time. Also note that formulations 15 and 16 include a smoothing of the pressure field and are, therefore, not recommended for materials with failure or problems with important strain localization.

c) All SPH formulations with Eulerian kernel, that is, FORM with values 0 to 6, 15 and 16, can be used for large or extremely large deformation applications but will have tensile instability issues. FORM = 2 or 3 is not recommended for any case.

d) All SPH formulations with Lagrangian kernel (FORM = 7 or 8) can be used to avoid tensile instability issue but they cannot endure very large deformations.

e) For improved accuracy and tensile stability, a formulation based on moving least-squares (FORM = 12) is available. This formulation can be used for extremely large deformation applications but entails a significant computational cost. FORM = 12 is available for MPP simulations only. It is strongly recommended to keep a constant smoothing length for this formulation by setting HMIN = 1.0 and HMAX = 1.0 in *SECTION_SPH.

f) Only formulations 0, 1, 15 and 16 are implemented for 2D axisymmetric problems (IDIM = -2).

g) Formulations 9 and 10 are adaptive smoothed particle hydrodynamics formulations with an anisotropic kernel (Eulerian kernel) whose axes evolve automatically to follow the mean particles spacing as it varies in time, space and direction based on the strain rate tensors. These forms must be used with the *SECTION_SPH_ELLIPSE keyword. They have better accuracy and stability than the standard SPH. These forms can be used for extremely large deformation problems and are available for 3D case only.

h) Formulation 13 is an implicit incompressible fluid formulation based on operator splitting, used mainly for water wading simulations. The fluid should be assigned to a *MAT_SPH_INCOMPRESSIBLE_FLUID material, and any structure interacting with the fluid needs to be sampled with SPH particles (see the *DEFINE_SPH_MESH_SURFACE keyword) assigned to a *MAT_SPH_INCOMPRESSIBLE_STRUCTURE material. The density of the structure particles should be the same as the rest density of the interacting fluid. The time step is calculated based on a CFL condition using the particle velocity and usually needs to be limited using the LCTM option in *CONTROL_TIMESTEP. The smoothing length of fluid and structure particles should be set to the initial inter-particle distance of the fluid, using SPHINI in *SECTION_SPH. The fluid-structure interaction is calculated entirely based on the SPH interaction of fluid and structure particles, and no *CONTACT keyword is necessary. Note that this formulation is inherently implicit and does not use the structural implicit solver; thus, it is supported for explicit calculations.

3. **Erosion.** The erosion criteria, which trigger particle deactivation when IEROD = 1, 2, or 3, may come from the material model, from *MAT_ADD_EROSION, or from the ERODE field in *CONTROL_TIMESTEP.

   a) For IEROD = 0, SPH particles remain active. This option is generally not recommended when materials with erosion are used, as many material models will still reset the stress field to zero periodically. If an unaltered stress field is desired, simply remove the erosion criteria in the material model parameters.

   b) For IEROD = 1, SPH particles are partially deactivated; that is, the stress states of the deactivated SPH particles will be set to zero, but these particles still remain in the domain integration for more stable results.

   c) For IEROD = 2, SPH particles are totally deactivated, so the stress states will be set to 0, and the deactivated particles do not remain in the domain integration.

   d) For IEROD = 3, SPH particles remain active. The deviatoric stress is set to zero. If an equation of state is used, the volumetric response remains unaltered; otherwise the volumetric stress is set to zero as well.

   e) Deactivated particles can be distinguished from active particles by setting ISHOW = 1.

   f) To disable contact for deactivated particles, set ICONT = 1.

4. **Artificial viscosity.** The artificial viscosity for standard solid elements, which is active when AVIS = 1, is given by:

$$q = \rho l\left(Q_1 l\dot{\varepsilon}_{kk}^2 - Q_2 a\dot{\varepsilon}_{kk}\right) \quad \dot{\varepsilon}_{kk} < 0$$
$$q = 0 \quad \dot{\varepsilon}_{kk} \geq 0$$

   where Q_1 and Q_2 are dimensionless input constants, which default to 1.5 and .06, respectively (see *CONTROL_BULK_VISCOSITY); l is a characteristic length given as the square root of the area in two dimensions and as the cube root of the volume in three; and a is the local sound speed. This formulation, which is consistent with solid artificial viscosity, has better energy balance for SPH elements.

   For general applications, Monaghan-type artificial viscosity is recommended as it is specifically designed for SPH particles. The Monaghan type artificial viscosity, which is active when AVIS = 0, is defined as follows:

$$q = \begin{cases} \dfrac{-Q_2\bar{c}_{ij}\phi_{ij} + Q_1\phi_{ij}^2}{\bar{\rho}_{ij}} & v_{ij}x_{ij} < 0 \\ 0 & v_{ij}x_{ij} \geq 0 \end{cases}$$

where

$$\phi_{ij} = \frac{h_{ij} v_{ij} x_{ij}}{|x_{ij}|^2 + \varphi^2}$$
$$\bar{c}_{ij} = 0.5(c_i + c_j)$$
$$\bar{\rho}_{ij} = 0.5(\rho_i + \rho_j)$$
$$h_{ij} = 0.5(h_i + h_j)$$
$$\varphi = 0.1 h_{ij}$$

and Q_1 and Q_2 are input constants. When using Monaghan-type artificial viscosity, it is recommended that the user set both Q_1 and Q_2 to 1.0 on either the *CONTROL_BULK_VISCOSITY or *HOURGLASS keywords; see for example G. R. Liu.

5. **Quasi-linear coefficient.** The moving least-squares-based formulation contains a quasi-linear approximation term to combine accuracy with stability in extremely large deformation simulations. The default value QL = 0.01 gives a good compromise between accuracy and stability in most cases. For greater accuracy, its value can be reduced to QL = 0.001 in simulations with small deformations, or increased to QL = 0.1 for extreme deformations, if instabilities are present.

*CONTROL_SPH_INCOMPRESSIBLE

Purpose: Provide controls relating to incompressible SPH (FORM = 13; see *CONTROL_SPH for more information).

Card 1	1	2	3	4	5	6	7	8
Variable	IBNDP	TAVG	TMAX	ROL	IHTC	IMAT	IRMV	ACMP
Type	I	F	F	F	I	I	I	F
Default	0	10^{-2}	10^{20}	10^{20}	0	0	0	0.0

VARIABLE **DESCRIPTION**

IBNDP Pressure treatment of boundary particles:

EQ.0: Pressure on boundary particles is extrapolated from fluid particles.

EQ.1: Pressure on boundary particles is explicitly calculated.

TAVG Tolerance criteria for convergence. If the average relative density (ρ/ρ_0) of particles under compression is below TAVG, this condition is satisfied.

TMAX Tolerance criteria for convergence. If the maximum relative density (ρ/ρ_0) of particles under compression is below TMAX, this condition is satisfied.

ROL Stuck particle detection criteria. In certain scenarios, some fluid particles can end up stuck between moving structures and as a result accumulate very large pressure values. If a particle's relative density contribution from boundaries is above ROL, it is deactivated. Its displacement is prescribed from the interpolated structure motion until it is sufficiently away from the structure at which point it is activated again.

IHTC Flag for Heat Transfer Coefficient (HTC) calculation:

EQ.0: HTCs are not calculated.

EQ.1: HTCs are calculated based on fluid properties given in *MAT_SPH_INCOMPRESSIBLE_FLUID input.

LT.0: |IHTC| is a function ID defining HTCs based on seven

VARIABLE	DESCRIPTION
	arguments: heat capacity, thermal conductivity, dynamic viscosity, density, tangential velocity relative to the closest segment, distance traveled along the surface, and the ratio of the distance from the closest segment to particle radius. (See *DEFINE_FUNCTION.)
IMAT	Flag for *MAT_SPH_INCOMPRESSIBLE_FLUID formulations: EQ.0: Surface tension and surface adhesion forces are calculated based on numerical parameters given in the material cards. EQ.1: Surface tension and surface adhesion forces are calculated based on the physical properties given in the material cards.
IRMV	Remove initially interpenetrated particles. Fluid particles that are too close to structural particles will be deactivated on initialization: EQ.0: Do not remove interpenetrated particles. EQ.1: Remove interpenetrated particles.
ACMP	Artificial compressibility term. For an analysis with a time-varying time step, we strongly recommend adding artificial compressibility to avoid oscillations and instabilities. We typically recommend a small value of 1.0×10^{-8} Pa^{-1}. Note that this parameter is unit sensitive. Thus, the recommended value needs to be converted to the unit system used in the model.

*CONTROL_SPOTWELD_BEAM

Purpose: Provides factors for scaling the failure force resultants of beam spot welds as a function of their parametric location on the contact segment and the size of the segment. Also, this keyword can optionally replace beam welds with solid hexahedron element clusters.

Card 1	1	2	3	4	5	6	7	8
Variable	LCT	LCS	T_ORT	PRTFLG	T_ORS	RPBHX	BMSID	ID_OFF
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
LCT	Load curve ID for scaling the response in tension based on the shell element size.
LCS	Load curve ID for scaling the response in shear based on the shell element size.
T_ORT	Table ID for scaling the tension response (and shear response if T_ORS = 0) based on the location of the beam node relative to the centroid of the shell.
PRTFLG	Set this flag to 1 to print for each spot weld attachment: the beam, node, and shell IDs, the parametric coordinates that define the constraint location, the angle used in the table lookup, and the three scale factors obtained from the load curves and table lookup. See Figure 12-90.
T_ORS	Optional table ID for scaling the shear response based on the location of the beam node relative to the centroid of the shell.
RPBHX	Replace each spot weld beam element with a cluster of RPBHX solid elements. The net cross-section of the cluster of elements is dimensioned to have the same area as the replaced beam. RPBHX may be set to 1, 4, or 8. When RPBHX is set to 4 or 8, a table is generated to output the force and moment resultants into the SWFORC file if this file is active. *DEFINE_HEX_SPOTWELD_ASSEMBLY describes this table. LS-DYNA uses the IDs of the beam elements as the cluster spot weld IDs, so the IDs in the

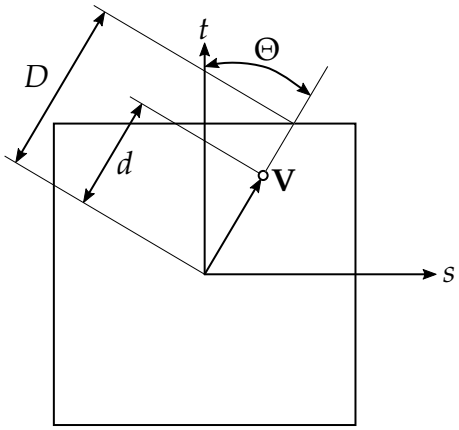


Figure 12-90. Definition of parameters for table definition.

VARIABLE	DESCRIPTION
	SWFORC file are unchanged. The beam elements are automatically deleted from the calculation, and the section and material data are automatically changed to be used with solid elements. See Figure 17-56.
BMSID	Optional beam set ID defining the beam element IDs that are to be converted to hex assemblies. If zero, all spot weld beam elements are converted to hex assemblies.
ID_OFF	This optional ID offset applies if and only if BMSID is nonzero. Beams, which share part IDs with beams that are converted to hex assemblies, will be assigned new part IDs by adding to the original part ID the value of ID_OFF. If ID_OFF is zero, the new part ID for such beams will be assigned to be larger than the largest part ID in the model.

Remarks:

The load curves and table provide a means of scaling the response of the beam spot welds to reduce any mesh dependencies for failure model 6 in *MAT_SPOTWELD. Figure 12-91 shows such dependencies that can lead to premature spot weld failure. Separate scale factors are calculated for each of the beam's nodes. The scale factors s_T, s_s, s_{OT}, and s_{OS} are calculated using LCT, LCS, T_ORT, and T_ORS, respectively, and are introduced in the failure criteria,

$$\left[\frac{s_T s_{OT}\sigma_{rr}}{\sigma_{rr}^F(\dot{\varepsilon}_{\text{eff}})}\right]^2 + \left[\frac{s_S s_{OS}\tau}{\tau^F(\dot{\varepsilon}_{\text{eff}})}\right]^2 - 1 = 0 \ .$$

If a curve or table is given an ID of 0, its scale factor is set to 1.0. The load curves LCT and LCS are functions of the characteristic size of the shell element used in the time step calculation at the start of the calculation. The orientation table is a function of the spot weld's

isoparametric coordinate location on the shell element. We define a vector **V** from the centroid of the shell to the contact point of the beam's node. The arguments for the orientation table are the angle:

$$\Theta = \tan^{-1}\left[\frac{\min(|s|,|t|)}{\max(|s|,|t|)}\right],$$

and the normalized distance $\bar{d} = d/D = \max(|s|,|t|)$. See Figure 12-90. The table is periodic over a range of 0 (**V** aligned with either the s or t axis) to 45 degrees (**V** is along the diagonal of the element). The angle of **V** in degrees, ranging from 0 to 45, specifies the table, and the individual curves give the scale factor as a function of the normalized distance of the beam node, $\bar{d}$, for a constant angle.

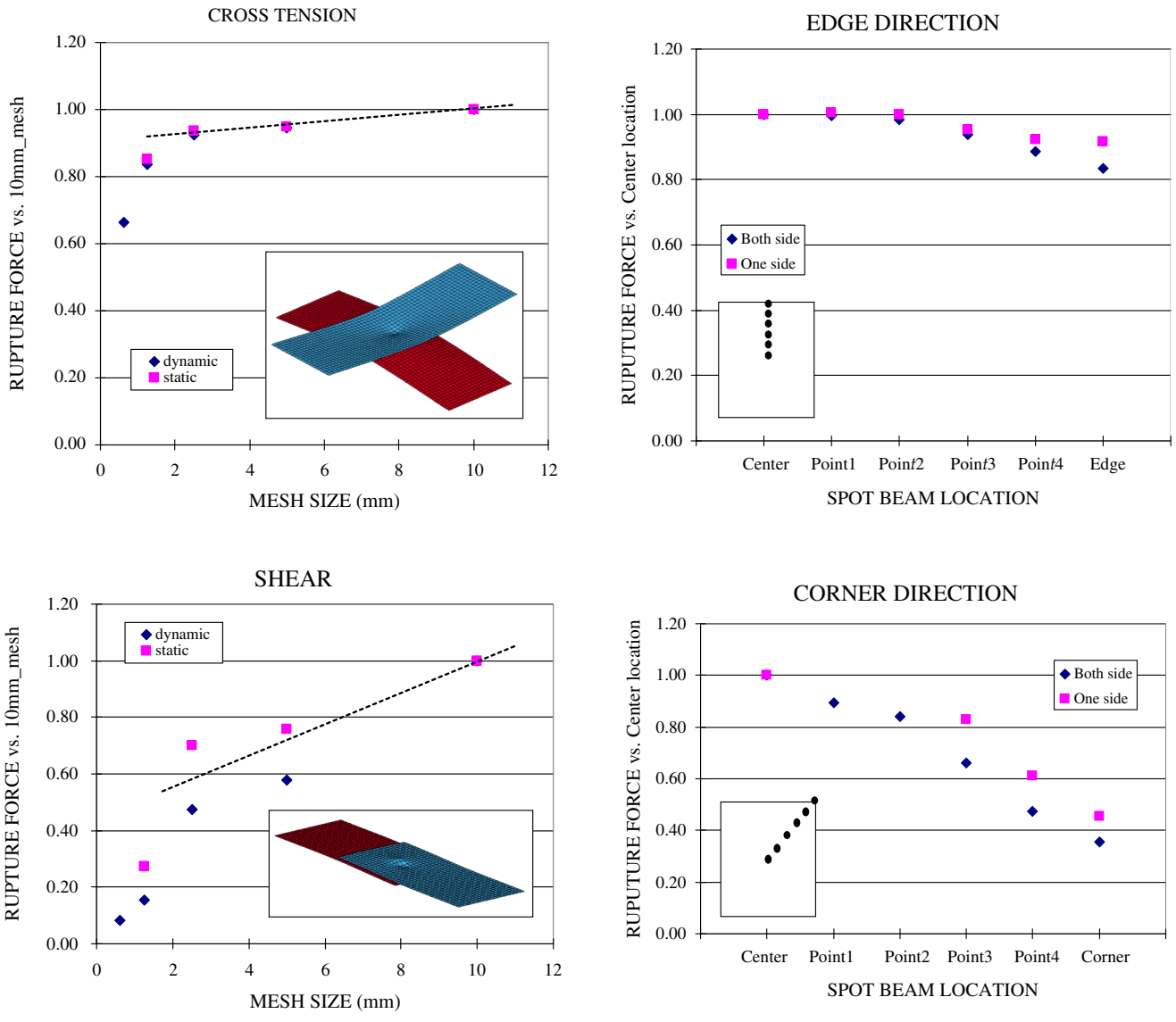


Figure 12-91 The failure force resultants can depend both on mesh size and the location of weld relative to the center of the contact segment

*CONTROL_STAGED_CONSTRUCTION

Purpose: Help break down analyses of construction processes into stages.

Card 1	1	2	3	4	5	6	7	8
Variable	TSTART	STGS	STGE	ACCEL	FACT	blank	DORDEL	NOPDEL
Type	F	I	I	F	F		I	I
Default	0.	0	0	0.0	10^{-6}		0	0

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	ITIME	blank	IDYNAIN					
Type	I		I					
Default	0		0					

VARIABLE	DESCRIPTION
TSTART	Time at start of analysis (normally leave blank). See Remark 5.
STGS	Construction stage at start of analysis
STGE	Construction stage at end of analysis
ACCEL	Default acceleration for gravity loading
FACT	Default stiffness and gravity factor for parts before they are added
DORDEL	Dormant part treatment in d3plot file (see Remark 6): EQ.0: Parts not shown when dormant (flagged as "deleted"). EQ.1: Parts shown normally when dormant.
NOPDEL	Treatment of pressure loads on deleted elements (see Remark 7): EQ.0: Pressure loads automatically deleted.

VARIABLE	DESCRIPTION
	EQ.1: No automatic deletion.
ITIME	Treatment of "Real Time" on *DEFINE_CONSTRUCTION_STAGES (see Remark 8): EQ.0: Real Time is ignored. EQ.1: Time in output files (d3plot, d3thdt, binout…) is converted to Real Time.
IDYNAIN	Flag to control output of dynain file at the end of every stage (see Remark 9): EQ.0: Write dynain file. EQ.1: Do not write dynain file.

Remarks:

1. **Related keywords.** See also *DEFINE_CONSTRUCTION_STAGES and *DEFINE_STAGED_CONSTRUCTION_PART.

2. **Re-running analysis.** The staged construction options offer flexibility to carry out the whole construction simulation in one analysis or to run it stage by stage. Provided that at least one construction stage is defined (*DEFINE_CONSTRUCTION_STAGES), a dynain file will be written at the end of each stage (file names are end_stage001_dynain, etc.). These contain node and element definitions and the stress state; the individual stages can then be re-run without re-running the whole analysis. To do this, make a new input file as follows:

   - Copy the original input file, containing *DEFINE_CONSTRUCTION_STAGES and *DEFINE_STAGED_CONSTRUCTION_PART.
   - Delete node and element definitions as these will be present in the dynain file (*NODE, *ELEMENT_SOLID, *ELEMENT_SHELL, and *ELEMENT_BEAM).
   - Delete any *INITIAL cards; the initial stresses in the new analysis will be taken from the dynain file.
   - On *CONTROL_STAGED_CONSTRUCTION set STGS to start at the desired stage.
   - Add an *INCLUDE statement referencing, for example, end_stage002_dynain if starting the new analysis from Stage 3.

- Move or copy the dynain file into the same directory as the new input file.

3. **Starting and ending stages.** When STGS is > 1 the analysis starts at a non-zero time (the start of stage STGS). In this case a dynain file must be included to start the analysis from the stress state at the end of the previous stage. The end time for stage STGE overrides the termination time on *CONTROL_TERMINATION. A new dynain file will be written at the end of all stages from STGS to STGE.

   If STGS > 1 and elements have been deleted in a previous stage, these elements will be absent from the new analysis and should not be referred to (for example, *DATABASE_HISTORY_SOLID) in the new input file.

4. **ACCEL and FACT.** ACCEL is the gravity loading applied to parts referenced by *DEFINE_STAGED_CONSTRUCTION_PART. FACT, which should be a very small number such as 10^{-6}, is the factor by which stresses and gravity load are scaled when parts referenced by *DEFINE_STAGED_CONSTRUCTION_-PART are inactive.

5. **Start time.** TSTART can be used to set a non-zero start time (again, assuming a compatible dynain file is included). This option is used only if construction stages have not been defined.

6. **Dormant parts.** By default, parts for which *DEFINE_STAGED_CONSTRUCTION_PART is defined are flagged as "deleted" in the d3plot file at time-states for which the part is not active (that is, stage STGA has not yet been reached). Parts that are deleted because STGR has been reached are also flagged as "deleted." When animating the results, the parts become visible when they become active, as if they had been added to the model at that time; and they disappear as they are deleted. If DORDEL is non-zero, inactive parts (before STGA) are shown normally. The parts are still shown as deleted after STGR is reached.

7. **Pressure load segments.** By default, LS-DYNA automatically deletes pressure load "segments" if they share all four nodes with a deleted solid or shell element. In staged construction, you may want to apply a pressure load to the surface of an element, A, that is initially shared with an element, B, where B is deleted during the calculation. For example, B may be in a layer of soil that is excavated, leaving A as the new top surface. The default scheme would delete the pressure segment when B is removed, despite A still being present. NOPDEL instructs LS-DYNA to skip the automatic deletion of pressure segments, irrespective of whether the elements have been deleted due to staged construction or material failure. You must then ensure that pressure loads are not applied to nodes no longer supported by an active element.

8. **Real time.** Construction processes that in real life take days, weeks, or months may be modelled in seconds of analysis time using a "time acceleration" method. For example, one second of analysis time might represent ten days of

real time. The speed-up factor might differ at different stages of the analysis. You may, however, want to present plots and graphs with real time rather than the analysis time used in LS-DYNA's calculations. To do this, set ITIMES to 1, and set RTS and RTE (see optional fields RTS and RTE on *DEFINE_CONSTRUCTION_STAGES) to the real time at the start and end of each stage. This feature has no effect on the calculations as it changes only the times shown in the d3plot, d3thdt and binout files.

9. **IDYNAIN.** Output of the dynain files can be suppressed by setting IDYNAIN to 1. If dynain files are required for some stages but not others, set IDYNAIN on *DEFINE_CONSTRUCTION_STAGES instead.

*CONTROL_START

Purpose: Define the start time of analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	BEGTIM							
Type	F							

VARIABLE	DESCRIPTION
BEGTIM	Start time of analysis (default = 0.0). *Load curves are not shifted to compensate for the time offset.* Therefore, this keyword will change the results of any calculation involving time-dependent load curves.

*CONTROL_STEADY_STATE_ROLLING

Card 1	1	2	3	4	5	6	7	8
Variable	IMASS	LCDMU	LCDMUR	IVEL	SCL_K			
Type	I	I	I	I	I			
Default	0	0	0	0	↓			

VARIABLE	DESCRIPTION
IMASS	Inertia switching flag: EQ.0: include inertia during an implicit dynamic simulation. EQ.1: treat steady state rolling subsystems as quasi-static during implicit dynamic simulations.
LCDMU	Optional load curve for scaling the friction forces in contact.
LCDMUR	Optional load curve for scaling the friction forces in contact during dynamic relaxation. If LCDMUR isn't specified, LCDMU is used.
IVEL	Velocity switching flag: EQ.0: eliminate the steady state rolling body forces and set the velocities of the nodes after dynamic relaxation. EQ.1: keep the steady state rolling body forces after dynamic relaxation instead of setting the velocities.
SCL_K	Scale factor for the friction stiffness during contact loading and unloading. The default values are 1.0 and 0.01 for explicit and implicit, respectively. Any scaling applied here applies only to contact involving the subsystem of parts defined for steady state rolling.

Remarks:

1. **Quasi-Static.** Treating the steady state rolling subsystems as quasi-static during an implicit simulation may eliminate vibrations in the system that are not of interest and is generally recommended.

2. **Scaling Friction and Convergence.** Ramping up the friction by scaling it with LCDMU and LCDMUR may improve the convergence behavior of implicit calculations. The values of the load curves should be 0.0 at initial contact and ramp up smoothly to a value of 1.0.

3. **Steady State Rolling Body Forces.** After dynamic relaxation, the default behavior is to initialize the nodes with the velocities required to generate the body forces on elements and remove the body forces. This initialization is skipped, and the body forces retained, after dynamic relaxation if IVEL = 1.

4. **Friction Stiffness Scale Factor.** The friction model in contact is similar to plasticity, where there is an elastic region during the loading and unloading of the friction during contact. The elastic stiffness is scaled from the normal contact stiffness. For implicit calculations, the default scale factor is 0.01, which results in long periods of time being required to build the friction force, and, in some cases, oscillations in the contact forces. A value between 10 and 100 produces smoother solutions and a faster build-up and decay of the friction force as the tire velocity or slip angle is varied, allowing a parameter study to be performed in a single run.

***CONTROL_STRUCTURED**_*{OPTION}*

Available options include:

<BLANK>

TERM

Purpose: Write out a structured input deck that is largely or wholly equivalent to the keyword input deck. This option may be useful in debugging errors that occur during processing of the input file, particularly if error messages of the type "*** ERROR ##### (STR + ###)" are written. The name of the structured input deck is "dyna.str".

Not all features are supported in structured input format. Some data, such as load curve numbers, will be output in an internal numbering system.

If the TERM option is activated, termination will occur after the structured input deck is written.

Adding "outdeck=s" to the LS-DYNA execution line serves the same purpose as including *CONTROL_STRUCTURED in the keyword input deck.

***CONTROL_SUBCYCLE**_{K}_{L} or
***CONTROL_SUBCYCLE**_{OPTION}*

Available options for subcycling first form with K and L

$$K, L \in \{\text{<BLANK>}, 1, 2, 4, 8, 16, 32, 64\}$$

Available options for multiscale (*OPTION*) include:

<BLANK>

MASS_SCALED_PART

MASS_SCALED_PART_SET

Purpose: This keyword is used to activate subcycling or mass scaling (multiscale). The common characteristic of both methods is that the time step varies from element to element, thereby eliminating unnecessary stepping on more slowly evolving portions of the model. These techniques are suited for reducing the computational cost for models involving large spatial variation in mesh density and/or material characteristics.

Subcycling is described in the LS-DYNA Theory Manual and in detail in Borrvall et.al. [2014] and may be seen as an alternative to using selective mass scaling, see the keyword *CONTROL_TIMESTEP.

This keyword comes in two variations:

1. **Subcycling.** Plain subcycling is activated by the *CONTROL_SUBCYCLE_{K}_{L} variant of this keyword. This form of the card should *not* be included more than once. It may be used in conjunction with mass scaling to limit the time step characteristics.

   For subcycling, time steps for integration are determined automatically from the characteristic properties of the elements in the model, with the restriction that the ratio between the largest and smallest time step is limited by K. Furthermore, L determines the relative time step at which external forces such as contacts and loads are calculated

   For example, *CONTROL_SUBCYCLE_16_4 limits the largest explicit integration time step to at most 16 times the smallest. Contact forces are evaluated every 4 time steps. The defaults are $K = 16$ and $L = 1$, and L cannot be specified larger than K. This option may be used without mass scaling activated but internally elements may still be slightly mass scaled to maintain computational efficiency.

2. **Mass Scaling/Multiscale.** For a multiscale simulation, mass scaling is mandatory and the time steps are directly specified in the input. The specified parts (see the PID field) or part sets (see the PSID field) run at the time step specified

in the TS field. All other elements evolve with a time step set by |DT2MS|, which is set on *CONTROL_TIMESTEP card.

This feature was motivated by automotive crash simulation, wherein it is common for a small subset of *solid* elements to limit the time step size. With this card the finely meshed parts (consisting of solid elements) can be made to run with a smaller time step through mass scaling so that the rest of the vehicle can run with a time step size of |DT2MS|.

Part Card. Additional card for the MASS_SCALED_PART and MASS_SCALED_-PART_SET keyword options. Provide as many cards as necessary. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	TS						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
PID/PSID	Part ID or part set ID if the SET option is specified.
TS	Time step size at which mass scaling is invoked for the PID or PSID

*CONTROL_TERMINATION

Purpose: Stop the job.

Card 1	1	2	3	4	5	6	7	8
Variable	ENDTIM	ENDCYC	DTMIN	ENDENG	ENDMAS	NOSOL		
Type	F	I	F	F	F	I		
Default	0.0	0	0.0	0.0	10^8	0		
Remarks	1		2					

VARIABLE	DESCRIPTION
ENDTIM	Termination time. Mandatory.
ENDCYC	Termination cycle. The termination cycle is optional and will be used if the specified cycle is reached before the termination time. Cycle number is identical with the time step number.
DTMIN	Reduction (or scale) factor to determine minimum time step, tsmin, where tsmin = dtstart × DTMIN and dtstart is the initial step size determined by LS-DYNA. When the time step drops to tsmin, LS-DYNA terminates with a restart dump. See the exception described in Remark 2. Also note that dtstart is the initial step size regardless of whether the first step is explicit or implicit; this can have implications if an analysis starts as implicit and later switches to explicit, resulting in a large drop in step size.
ENDENG	Percent change in energy ratio for termination of calculation. If undefined, this option is inactive.
ENDMAS	Percent change in the total mass for termination of calculation. This option is relevant if and only if mass scaling is used to limit the minimum time step size; see *CONTROL_TIMESTEP field DT2MS. LT.0.0: |ENDMAS| is the load curve ID defining the percent change in the total mass as a function of the total mass.
NOSOL	Flag for a non-solution run, that is, normal termination directly after initialization.

VARIABLE	DESCRIPTION
	EQ.0: Off (default),
	EQ.1: On.

Remarks:

1. **Displacement Termination.** Termination by displacement may be defined in the *TERMINATION section.

2. **Erosion.** If the erosion flag on *CONTROL_TIMESTEP is set (ERODE = 1), then solid elements and thick shell elements whose time step falls below tsmin will be eroded and the analysis will continue. This time-step-based failure option is not recommended when solid formulations 11 or 12 are included in the model. Furthermore, when PSFAIL in *CONTROL_SOLID is nonzero, regardless of the value of ERODE, then all solid elements excepting those with formulation 11 or 12, whose time step falls below tsmin will be eroded and the analysis will continue. This time-step-based erosion of solids due to a nonzero PSFAIL is not limited to solids in part set PSFAIL. Only the negative-volume-based erosion criterion is limited to solids in part PSFAIL.

*CONTROL_THERMAL_EIGENVALUE

Purpose: Compute eigenvalues of thermal conductance matrix for model evaluation purposes.

Card 1	1	2	3	4	5	6	7	8
Variable	NEIG							
Type	I							
Default	0						.	.

VARIABLE	DESCRIPTION
NEIG	Number of eigenvalues to compute: EQ.0: No eigenvalues are computed. GT.0: Compute NEIG eigenvalues of each thermal conductance matrix.

Remarks:

1. **Feature Description.** This feature causes LS-DYNA to compute NEIG eigenvalues for each thermal conductance matrix. This tool is for model evaluation, so only a small number of thermal time steps, such as 1, should be used with this feature.

*CONTROL_THERMAL_FORMING

Purpose: Simplify keyword input deck by defining key control parameters for hot stamping simulations. A hot stamping simulation is in general a thermo-mechanical-coupled analysis, which requires adding a set of thermal control keywords to fully define the simulation job. However, for certain situations this keyword can simplify input by reducing the number of keywords required.

Card Summary:

Card 1. This card is required.

ITS	PTYPE	TSF	THSHEL	ITHOFF	SOLVER	FWORK	

Card 2.1. The card set, Cards 2.1 and 2.2, is optional. When present Card 2.1 sets the default values for the fields of "THRM 1" on *CONTACT and "THRM 1" becomes optional.

K	FRAD	H0	LMIN	LMAX	FTOSA	BC_FLG	ALGO

Card 2.2. The card set, Cards 2.1 and 2.2, is optional. When present Card 2.2 sets the default values for the fields of "THRM 2" on *CONTACT and "THRM 2" becomes optional.

LCFST	LCFDT	FORMULA	A	B	C	D	LCH

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ITS	PTYPE	TSF	THSHEL	ITHOFF	SOLVER	FWORK	
Type	F	I	F	I	I	I	F	
Default	none	0	1.	0	0	3	1.0	

VARIABLE	DESCRIPTION
ITS	Initial thermal time step size
PTYPE	Thermal problem type (see Remark 1 for determining the type of problem):

VARIABLE	DESCRIPTION
	EQ.0: Linear problem EQ.1: Nonlinear problem with material properties evaluated at the temperature of the gauss point EQ.2: Nonlinear problem with material properties evaluated at the average temperature of the element
TSF	Thermal Speedup Factor. This factor multiplies all thermal parameters with units of time in the denominator (such as thermal conductivity and convection heat transfer coefficients). It is used to artificially scale the problem in time. For example, if the velocity of the stamping punch is artificially increased by 1000, then set TSF = 1000 to scale the thermal parameters.
THSHEL	Thermal shell option: EQ.0: No temperature gradient is considered through the shell thickness. EQ.1: A temperature gradient is calculated through the shell thickness.
ITHOFF	Flag for offsetting thermal contact surfaces for thick thermal shells: EQ.0: No offset; if thickness is not included in the contact, the heat will be transferred between the mid-surfaces of the corresponding contact segments (shells). EQ.1: Offsets are applied so that contact heat transfer is always between the outer surfaces of the contact segments (shells).
SOLVER	Thermal analysis solver type (see *CONTROL_THERMAL_SOLVER). For SMP only: EQ.1: Using solver 11 (enter -1 to use the old ACTCOL solver) EQ.2: Nonsymmetric direct solver EQ.3: Diagonal scaled conjugate gradient iterative (default) EQ.4: Incomplete choleski conjugate gradient iterative EQ.5: Nonsymmetric diagonal scaled bi-conjugate gradient For SMP or MPP: EQ.11: Direct solver

VARIABLE	DESCRIPTION
	EQ.12: Diagonal scaling conjugate gradient iterative (default) EQ.13: Symmetric Gauss-Seidel conjugate gradient iterative EQ.14: SSOR conjugate gradient iterative EQ.15: ILDLT0 (incomplete factorization) conjugate gradient iterative EQ.16: Modified ILDLT0 (incomplete factorization) conjugate gradient iterative For Conjugate Heat transfer problems in SMP or MPP: EQ.17: GMRES solver
FWORK	Fraction of mechanical work converted into heat EQ.0.0: Use default value 1.0.

Card 2.1	1	2	3	4	5	6	7	8
Variable	K	FRAD	H0	LMIN	LMAX	FTOSA	BC_FLG	ALGO
Type	F	F	F	F	F	F	I	I
Default	none	none	none	none	none	0.5	0	0

VARIABLE	DESCRIPTION
K	Thermal conductivity of fluid between the contact surfaces. If a gap with a thickness l_{gap} exists between the contact surfaces, then the conductance due to thermal conductivity between the contact surfaces is $$h_{cond} = \frac{K}{l_{gap}}$$ Note that LS- DYNA calculates l_{gap} based on deformation.
FRAD	Radiation factor between the contact surfaces. $$f_{rad} = \frac{\sigma}{\frac{1}{\varepsilon_1} + \frac{1}{\varepsilon_2} - 1},$$ where

VARIABLE	DESCRIPTION
	σ = Stefan-Boltman constant ε_1 = emissivity of SURFB surface ε_2 = emissivity of SURFA surface LS-DYNA calculates a radiant heat transfer conductance $h_{\text{rad}} = f_{\text{rad}}(T_{\text{SURFA}} + T_{\text{SURFB}})(T_{\text{SURFA}}^2 + T_{\text{SURFB}}^2)$
H0	Heat transfer conductance for closed gaps. Use this heat transfer conductance for gaps in the range $0 \le l_{\text{gap}} \le l_{\text{min}}$
LMIN	Minimum gap, l_{min}; use the heat transfer conductance defined (H0) for gap thicknesses less than this value. LT.0.0: -LMIN is a load curve ID defining l_{min} as a function of time.
LMAX	No thermal contact if gap is greater than this value (l_{max}).
FTOSA	Fraction, f, of sliding friction energy partitioned to the SURFA surface. Energy partitioned to the SURFB surface is $(1-f)$. EQ.0: Default set to 0.5. The sliding friction energy is partitioned 50% - 50% to the SURFA and SURFB surfaces in contact. $f = \dfrac{1}{1 + \dfrac{\sqrt{(\rho C_p k)_{\text{SURFB side material}}}}{\sqrt{(\rho C_p k)_{\text{SURFA side material}}}}}$
BC_FLAG	Thermal boundary condition flag: EQ.0: Thermal boundary conditions are on when parts are in contact. EQ.1: Thermal boundary conditions are off when parts are in contact.
ALGO	Contact algorithm type. EQ.0: Two-way contact; both surfaces change temperature due to contact. EQ.1: One-way contact; SURFB surface does not change temperature due to contact. SURFA surface does change temperature.

Card 2.2	1	2	3	4	5	6	7	8
Variable	LCFST	LCFDT	FORMULA	A	B	C	D	LCH
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
LCFST	Load curve number for static coefficient of friction as a function of temperature. The load curve value multiplies the coefficient value FS.
LCFDT	Load curve number for dynamic coefficient of friction as a function of temperature. The load curve value multiplies the coefficient value FD.
FORMULA	Formula that defines the contact heat conductance as a function of temperature and pressure.

EQ.1: $h(P)$ is defined by load curve A, which contains data for contact conductance as a function of pressure.

EQ.2: $h(P)$ is given by the following where A, B, C and D although defined by load curves are typically constants for use in this formula. The load curves are given as functions of temperature.

$$h(P) = a + bP + cP^2 + dP^3$$

EQ.3: $h(P)$ is given by the following formula from [Shvets and Dyban 1964].

$$h(P) = \frac{\pi k_{\text{gas}}}{4\lambda}\left[1. + 85\left(\frac{P}{\sigma}\right)^{0.8}\right] = \frac{a}{b}\left[1. + 85\left(\frac{P}{c}\right)^{0.8}\right] \ ,$$

where

a: is evaluated from the load curve, A, for the thermal conductivity, k_{gas}, of the gas in the gap as a function of temperature.

b: is evaluated from the load curve, B, for the parameter grouping $\pi/4\lambda$. Therefore, this load curve should be set to a constant value. λ is the surface roughness.

c: is evaluated from the load curve, C, which specifies a stress metric for deformation (e.g., yield) as a function of

VARIABLE	DESCRIPTION
	temperature. EQ.4: $h(P)$ is given by the following formula from [Li and Sellars 1996]. $$h(P) = a\left[1 - \exp\left(-b\frac{P}{c}\right)\right]^d \quad ,$$ where *a*: is evaluated from the load curve, A, which defines a load curve as a function of temperature. *b*: is evaluated from the load curve, B, which defines a load curve as a function of temperature. *c*: is evaluated from the load curve, C, which defines a stress metric for deformation (e.g., yield) as a function of temperature. *d*: is evaluated from the load curve D, which is a function of temperature. EQ.5: $h(\text{gap})$ is defined by load curve A, which contains data for contact conductance as a function of interface gap. LT.0: This is equivalent to defining the keyword *USER_INTERFACE_CONDUCTIVITY. The user subroutine `usrhcon` will be called for this contact interface to define the contact heat transfer coefficient.
A	Load curve ID for the *a* coefficient used in the formula.
B	Load curve ID for the *b* coefficient used in the formula.
C	Load curve ID for the *c* coefficient used in the formula.
D	Load curve ID for the *d* coefficient used in the formula.
LCH	Load curve ID for *h*. This parameter can refer to a curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION). When LCH is a curve ID (and a function ID) it is interpreted as follows: GT.0: The heat transfer coefficient is defined as a function of time, t, by a curve consisting of $(t, h(t))$ data pairs. LT.0: The heat transfer coefficient is defined as a function of temperature, T, by a curve consisting of $(T, h(T))$ data pairs. When the reference is to a function it is prototyped as follows $h = h(t, T_{\text{avg}}, T_{\text{slv}}, T_{\text{msr}}, P, g)$ where:

VARIABLE	DESCRIPTION

t = solution time
T_{avg} = average interface temperature
T_{SURFA} = SURFA segment temperature
T_{SURFB} = SURFB segment temperature
P = interface pressure
g = gap distance between SURFA and SURFB segments

Remarks:

1. **Problem Type.** The thermal problem becomes nonlinear when any of the following applies: (1) the material properties are temperature-dependent, (2) the thermal loads are temperature-dependent (for example, the convection coefficient is a function of temperature), (3) radiation exists.

2. **Heat Transfer at the Contact Interface between the Die and Blank.** The heat transfer coefficient h at the contact interface between the die and blank can be summarized as follows:

$$h = \begin{cases} h_0 & 0 \le l_{gap} \le l_{min} \\ h_{cond} + h_{rad} & l_{min} < l_{gap} \le l_{max} \\ 0 & l_{gap} > l_{max} \end{cases}$$

Examples:

The following input defines the thermal control parameters for a hot stamping simulation using *CONTROL_THERMAL_FORMING:

```
*CONTROL_THERMAL_FORMING
$      its     ptype       tsf    THSHEL    ITHOFF    solver     fwork
    1.0e-5         1       10.                            11
```

This is equivalent to defining the following set of keywords:

```
*CONTROL_SOLUTION
$#    soln       nlq     isnan     lcint     lcacc
         2
*CONTROL_THERMAL_SOLVER
$#   atype     ptype    solver     cgtol       gpt    eqheat     fwork       sbc
         1         1        11                 0  1.000000  1.00e-10  5.67e-11
$#  MSGLVL     MAXITR   ABSTOL    RELTOL     OMEGA                            TSF
                                                                             10.
*CONTROL_THERMAL_TIMESTEP
$#      ts       tip       its      tmin      tmax     dtemp      tscp      lcts
         0  1.000000    1.0e-5
*CONTROL_THERMAL_NONLINEAR
$#  refmax       tol
        10    1.0e-4
```

As shown above, *CONTROL_THERMAL_FORMING simplifies the user input by setting default values to less-frequently-used parameters implicitly. However, the user can always use a thermal control keyword to overwrite the default values of these hidden parameters. For instance, *CONTROL_THERMAL_FORMING sets FWORK (Fraction of mechanical Work converted to heat) to 10^{-10} implicitly. If FWORK needs to be increased to 0.8, add the following to the input deck to override the default value:

```
*CONTROL_THERMAL_SOLVER
$#   atype     ptype    solver     cgtol       gpt    eqheat     fwork       sbc
         1         1        11                   0  1.000000       0.8  5.67e-11
```

The thick shell option from Card 2 of *CONTROL_SHELL can also be turned on with just *CONTROL_THERMAL_FORMING with the THSHEL field as follows:

```
*CONTROL_THERMAL_FORMING
$       its     ptype       tsf    THSHEL    ITHOFF    solver     fwork
    1.0e-5         1       10.         1                 11
```

The ITHOFF field for this keyword changes the offset for thermal contact for shells which normally requires the inclusion of optional Card 5 in *CONTROL_CONTACT.

```
*CONTROL_THERMAL_FORMING
$       its     ptype       tsf     TSHEL    ITHOFF    solver     fwork
    1.0e-5         1       10.         1         1        11
```

A standard stamping simulation usually has multiple blank/die contact pairs with uniform or similar thermal/friction properties at the contact interface. With *CONTROL_-THERMAL_FORMING a default set of thermal/friction parameters can be defined by including the optional Cards 2.1 and 2.2. Input values in Cards 2.1 and 2.2 will be automatically applied to thermal/friction parameters of all contact pairs defined in THRM 1 and THRM 2 of *CONTACT. Therefore, the user does not need to repeat input of these parameters for every contact pair. An example is shown below:

```
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$#     cid                                                              title
         1 Die
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         1         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$#     cid                                                              title
         2 Punch
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         2         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$#     cid                                                              title
```

```
         3 Binder
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         3         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
*CONTROL_THERMAL_FORMING
$      its     ptype       tsf     TSHEL    ITHOFF    solver     fwork
    1.0e-5         1
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  4.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
       115
```

The above example is equivalent to the following input where the same set of thermal/friction parameters are defined three times, each corresponding to one single contact pair.

```
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_THERMAL_FRICTION_ID
$#     cid                                                                   title
         1 Die
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         1         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  4.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
       115
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_THERMAL_FRICTION_ID
$#     cid                                                                   title
         2 Punch
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         2         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  4.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
       115
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_THERMAL_FRICTION_ID
$#     cid                                                                   title
         3 Binder
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         3         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  4.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
       115
*CONTROL_THERMAL_FORMING
$      its     ptype       tsf     TSHEL    ITHOFF    solver     fwork
    1.0e-5         1
```

Although Cards 2.1 and 2.2 of *CONTROL_THERMAL_FORMING provide a convenient way to define/update thermal/friction properties at multiple contact interfaces, some contact pairs require different values for the properties. In that case, simply add the THERMAL or THERMAL_FRICTION keyword option to this particular contact pair and input the thermal/friction parameters in the THRM 1 and THRM 2 cards which will override the default values set by *CONTROL_THERMAL_FORMING. The following examples shows how we use *CONTACT_THERMAL_FORMING to set HTC (Heat Transfer Coefficient) = 4.0 for all contact pairs except contact pair 2, which has HTC = 3.0.

```
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$#     cid                                                                title
         1 Die
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         1         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_THERMAL_FRICTION_ID
$#     cid                                                                title
         2 Punch
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         2         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  3.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
       115
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ID
$#     cid                                                                title
         3 Binder
$#   surfa     surfb  surfatyp  surfbtyp   saboxid   sbboxid      sapr      sbpr
         4         3         2         2
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
  0.400000     0.000     0.000     0.000 20.000000         0     0.0001.0000E+20
$#    sfsa      sfsb      sast      sbst     sfsat     sfsbt       fsf       vsf
     0.000     0.000     0.000     0.000  1.000000  1.000000  1.000000  1.000000
*CONTROL_THERMAL_FORMING
$      its     ptype       tsf     TSHEL    ITHOFF    solver     fwork
    1.0e-5         1
$#       k      frad        h0      lmin      lmax     ftosa    bc_flg      algo
     0.000     0.000  4.000000  0.050000  0.050000                              1
$#   lcfst     lcfdt   formula         a         b         c         d       lch
115
```

*CONTROL_THERMAL_NONLINEAR

Purpose: Set parameters for a nonlinear thermal or coupled structural/thermal analysis. The control card, *CONTROL_SOLUTION, is also required.

Card 1	1	2	3	4	5	6	7	8
Variable	REFMAX	TOL	DCP	LUMPBC	THLSTL	NLTHPR	PHCHPN	
Type	I	F	F	I	F	I	F	
Default	10	10^{-4}	1.0 or 0.5	0	0.	0	100.	

VARIABLE	DESCRIPTION
REFMAX	Maximum number of matrix reformations per time step. EQ.0: Set to 10 reformations.
TOL	Convergence tolerance for temperature. EQ.0.0: Set to 1000 × machine roundoff.
DCP	Divergence control parameter: steady state problems $0.3 \le DCP \le 1.0$ default 1.0 transient problems $0.0 < DCP \le 1.0$ default 0.5
LUMPBC	Lump thermal boundary conditions. LUMPBC = 1 activates a numerical method to damp out anomalous temperature oscillations resulting from very large step function boundary conditions. This option is not generally recommended. EQ.0: Off (default) EQ.1: On
THLSTL	Line search convergence tolerance: EQ.0.0: No line search GT.0.0: Line search convergence tolerance
NLTHPR	Thermal nonlinear print out level: EQ.0: No print out

VARIABLE	DESCRIPTION
	EQ.1: Print convergence parameters during solution of nonlinear system.
PHCHPN	Phase change penalty parameter: EQ.0.0: Set to default value 100. GT.0.0: Penalty to enforce constant phase change temperature

*CONTROL_THERMAL_SOLVER

Purpose: Set options for the thermal solution in a thermal only or coupled structural-thermal analysis. The control card, *CONTROL_SOLUTION, is also required.

Card Summary:

Card 1. This card is required.

ATYPE	PTYPE	SOLVER		GPT	EQHEAT	FWORK	SBC

Card 2a. This card is optional. It is only read when SOLVER ≠ 17.

MSGLVL	MAXITR	ABSTOL	RELTOL	OMEGA			TSF

Card 2b. This card is optional. It is only read when SOLVER = 17.

MSGLVL	NINNER	ABSTOL	RELTOL	NOUTER			

Card 3. This card is optional.

MXDMP	DTVF	VARDEN		NCYCL			

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ATYPE	PTYPE	SOLVER		GPT	EQHEAT	FWORK	SBC
Type	I	I	I		I	F	F	F
Default	0	0	↓		8	1.	1.	0.
Remark				11				

VARIABLE	DESCRIPTION
ATYPE	Thermal analysis type: EQ.0: Steady state analysis (see Remark 4) EQ.1: Transient analysis (see Remark 3)

VARIABLE	DESCRIPTION
PTYPE	Thermal problem type: (see *CONTROL_THERMAL_NONLINEAR if nonzero) EQ.0: Linear problem, EQ.1: Nonlinear problem with material properties evaluated at gauss point temperature. EQ.2: Nonlinear problem with material properties evaluated at element average temperature.
SOLVER	Thermal analysis solver type (see Remarks 1 and 2): EQ.11: Direct solver EQ.12: Diagonal scaling (default for MPP) conjugate gradient iterative EQ.13: Symmetric Gauss-Seidel conjugate gradient iterative EQ.14: SSOR conjugate gradient iterative EQ.15: ILDLT0 (incomplete factorization) conjugate gradient iterative EQ.16: Modified ILDLT0 (incomplete factorization) conjugate gradient iterative EQ.17: GMRES solver for conjugate heat transfer problems EQ.18: ILDLT(T) (incomplete factorization with threshold pivoting) EQ.19: Preconditioned conjugate gradient with MUMPS (see Remark 2 in *CONTROL_IMPlICIT_SOLVER) EQ.30: Direct nonsymmetric factorization EQ.90: User-supplied linear equation (see Remark 3 of *CONTROL_IMPLICIT_SOLVER).
GPT	Number of Gauss points to be used in the solid elements: EQ.0: Use default value 8, EQ.1: One point quadrature is used.
EQHEAT	Mechanical equivalent of heat (see Remark 10). EQ.0.0: Default value 1.0, LT.0.0: Designates a load curve number for EQHEAT as a function of time.

VARIABLE	DESCRIPTION
FWORK	Fraction of mechanical work converted into heat. EQ.0.0: Use default value 1.0.
SBC	Stefan Boltzmann constant. Value is used with enclosure radiation surfaces; see *BOUNDARY_RADIATION_… LT.0.0: Use a smoothing algorithm when calculating view factors to force the row sum = 1.

Card 2a	1	2	3	4	5	6	7	8
Variable	MSGLVL	MAXITR	ABSTOL	RELTOL	OMEGA			TSF
Type	I	I	F	F	F			F
Default	0	500	10^{-10}	10^{-6}	1.0 or 0.			1.

VARIABLE	DESCRIPTION
MSGLVL	Output message level (For SOLVER > 10) EQ.0: No output (default), EQ.1: Summary information, EQ.2: Detailed information, use only for debugging.
MAXITR	Maximum number of iterations. For SOLVER > 11. EQ.0: Use default value 500,
ABSTOL	Absolute convergence tolerance. For SOLVER > 11. EQ.0.0: Use default value 10^{-10}
RELTOL	Relative convergence tolerance. For SOLVER > 11. EQ.0.0: Use default value 10^{-6}
OMEGA	Relaxation parameter omega for SOLVER 14 and 16. EQ.0.0: Use default value 1.0 for Solver 14, use default value 0.0 for Solver 16.

VARIABLE	DESCRIPTION
TSF	Thermal Speedup Factor. This factor multiplies all thermal parameters with units of time in the denominator (e.g., thermal conductivity, convection heat transfer coefficients). It is used to artificially time scale the problem. EQ.0.0: Default value 1.0, LT.0.0: |TSF| is a load curve ID. Curve defines speedup factor as a function of time. Its main use is in metal stamping. If the velocity of the stamping punch is artificially increased by 1000, then set TSF = 1000 to scale the thermal parameters.

Card 2b	1	2	3	4	5	6	7	8
Variable	MSGLVL	NINNER	ABSTOL	RELTOL	NOUTER			
Type	I	I	F	F	I			
Default	0	100	10^{-10}	10^{-6}	100			

VARIABLE	DESCRIPTION
MSGLVL	Output message level (For SOLVER > 10) EQ.0: No output (default), EQ.1: Summary information, EQ.2: Detailed information, use only for debugging.
NINNER	Number of inner iterations for GMRES solve
ABSTOL	Absolute convergence tolerance. For SOLVER > 11. EQ.0.0: Use default value 10^{-10}
RELTOL	Relative convergence tolerance. For SOLVER > 11. EQ.0.0: Use default value 10^{-6}
NOUTER	Number of outer iterations for GMRES solve

Card 3	1	2	3	4	5	6	7	8
Variable	MXDMP	DTVF	VARDEN		NCYCL			
Type	I	F	I		I			
Default	0	0.	0		1			.

VARIABLE	DESCRIPTION
MXDMP	Matrix Dumping for SOLVER > 11 EQ.0: No dumping GT.0: Dump using ASCII format every MXDMP time steps. LT.0: Dump using binary format every |MXDMP| time steps.
DTVF	Time interval between view factor updates.
VARDEN	Variable thermal density for solid elements in a coupled thermal-structural analysis. Setting VARDEN to 1 or 2 will adjust the material density in the thermal solver to account for changes in element volume due to, for example, material compaction, thermal expansion, etc. In applications where volume changes are small, the default is recommended. EQ.0: Thermal density remains constant and equal to TRO as given in *MAT_THERMAL_*OPTION* (default). EQ.1: Thermal density varies to account for change in volume. If an equation of state (*EOS) is used, the initial internal energy specified therein is taken into account. EQ.2: Thermal density varies to account for change in volume. The initial internal energy is not considered.
NCYCL	Thermal matrix reassembly frequency. Default is at every thermal cycle.

Remarks:

1. **Recommended direct solver.** Solver 11 is the preferred direct solver. Solver type 30 is a nonsymmetric direct solver.

2. **Direct versus iterative solver.** Even though using a direct solver is usually less efficient than using an iterative solver (SOLVER = 12, 13, 14, 15, 16, 17, 18 and 19), we recommend it for thermal coupling analysis. If you are running a thermal only analysis, then consider using an iterative solver to decrease execution time (particularly for large models); otherwise we recommend a direct solver.

3. **Transient problems.** For transient problems, diagonal scaling conjugate gradient (SOLVER = 12) should be adequate.

4. **Steady state problems.** For steady-state problems, convergence may be slow or unacceptable, so consider using direct solver (SOLVER = 11) or a more powerful preconditioner (SOLVER = 13, 14, 15, 16, 18, and 19).

5. **Solvers 13 & 14.** Solver 13 (symmetric Gauss-Seidel) and solver 14 (SSOR) are related. When OMEGA = 1.0, solver 14 is equivalent to solver 13. The optimal omega value for SSOR is problem dependent but lies between 1.0 and 2.0.

6. **Solvers 15 & 16.** Solver 15 (incomplete LDLT0) and solver 16 (modified incomplete LDLT0) are related. Both are no-fill factorizations that require one extra n-vector of storage. The sparsity pattern of the preconditioner is exactly the same as that of the thermal stiffness matrix. Solver 16 uses the relaxation parameter OMEGA. The optimal OMEGA value is problem dependent, but lies between 0.0 and 1.0.

7. **Solver 17.** The GMRES solver has been developed as an alternative to the direct solvers in cases where the structural thermal problem is coupled with the fluid thermal problem in a monolithic approach using the ICFD solver. A significant gain of calculation time can be observed when the problem reaches 1M elements.

8. **Completion conditions for solvers 12 – 15.** Solvers 12, 13, 14, 15 and 16 terminate the iterative solution process when (1) the number of iterations exceeds MAXITR or (2) the 2-norm of the residual drops below:

$$\text{ABSTOL} + \text{RELTOL} \times \text{2-norm of the initial residual.}$$

9. **Debug data.** Solvers 11 and up can dump the thermal conductance matrix and right-hand-side using the same formats as documented under *CONTROL_IMPLICIT_SOLVER. If this option is used, files beginning with T_will be generated.

10. **Unit conversion factor.** EQHEAT is a unit conversion factor. EQHEAT converts the mechanical unit for work into the thermal unit for energy according to,

$$\text{EQHEAT} \times [\text{work}] = [\text{thermal energy}]$$

However, it is recommended that a consistent set of units be used with EQHEAT set to 1.0. For example, when using SI,

$$[\text{work}] = 1\text{Nm} = [\text{thermal energy}] = 1\text{J} \Rightarrow \text{EQHEAT} = 1.$$

11. **Obsolete variable CGTOL.** Starting with LS-DYNA R12, the variable CGTOL is obsolete; use RELTOL for non-default values of the relative convergence tolerance.

*CONTROL_THERMAL_TIMESTEP

Purpose: Set time step controls for the thermal solution in a thermal-only or coupled structural/thermal analysis. This card requires that the input deck also include *CONTROL_SOLUTION and *CONTROL_THERMAL_SOLVER.

Card 1	1	2	3	4	5	6	7	8
Variable	TS	TIP	ITS	TMIN	TMAX	DTEMP	TSCP	LCTS
Type	I	F	F	F	F	F	F	I
Default	0	1.0	none	↓	↓	1.0	0.5	0

VARIABLE	DESCRIPTION
TS	Time step control: EQ.0: Fully prescribed time step with section-wise data from LCTS (if defined) EQ.1: Automatic variable time step (may increase or decrease) EQ.2: Fully prescribed time step using interpolated data from LCTS (if defined) EQ.3: Same as 0, but ensuring that maximum temperature change per time step does not exceed DTEMP EQ.4: Same as 2, but ensuring that maximum temperature change per time step does not exceed DTEMP See Remark 2 for further details.
TIP	Time integration parameter. See Remark 1 for details. EQ.0.5: Crank-Nicolson scheme EQ.1.0: Fully implicit (default)
ITS	Initial/fixed thermal time step size if LCTS is not defined. If LCTS is defined, this field is ignored. For TS = 1, ITS provides the initial time step, while for TS ≠ 1, it sets the time step size.
TMIN	Minimum thermal time step: EQ.0.0: Set to structural explicit time step (default). LT.0.0: Curve ID = (-TMIN) gives minimum thermal time step

VARIABLE	DESCRIPTION
	size as a function of time. The load curve defines pairs (thermal time breakpoint, new minimum time step). See Remark 3.
TMAX	Maximum thermal time step: EQ.0.0: Set to 100 × structural explicit time step (default). LT.0.0: Curve ID = (-TMAX) gives maximum thermal time step size as a function of time. The load curve defines pairs (thermal time breakpoint, new minimum time step). The time step will be adjusted to hit the time breakpoints exactly. See Remark 3.
DTEMP	Maximum temperature change in each time step above which the thermal time step will be decreased: EQ.0.0: Set to a temperature change of 1.0. LT.0.0: Curve ID = (-DTEMP) gives maximum temperature change as a function of time. The load curve defines pairs (thermal time breakpoint, new temperature change). See Remark 3.
TSCP	Time step control parameter (0. < TSCP < 1.0). The thermal time step is decreased by this factor, if the temperature change in the time step exceeds DTEMP or if the nonlinear solver fails to reach convergence within the given maximum number of matrix reformations. If the nonlinear solution algorithm is actually diverging, the thermal time step is decreased by DCP defined in *CONTROL_THERMAL_NONLINEAR. EQ.0.0: Set to a factor of 0.5.
LCTS	Load curve ID controlling the time step size variation over time. If defined, the value of ITS is ignored. Handling of curve data depends on the choice of TS. See Remark 2 for details.

Remarks:

1. **Time integration parameter.** TIP is relevant only to a transient analysis and is ignored for a steady state analysis. The only two accepted values that make mathematical sense are TIP = 0.5 or TIP = 1. TIP = 0.5 is the classical Crank-Nicolson method. It is mathematically more accurate than TIP = 1 and is unconditionally stable in time. However, a plot of nodal temperature as a function of

time can be oscillatory, especially in a nonlinear problem. The oscillations decrease with time. TIP = 1.0 is the classical implicit method. It is also unconditionally stable. For this method, the plot of nodal temperature as a function of time does not have oscillations.

2. **Time step control and load curve handling.** The value of time step control parameter TS determines the interpretation of the load curve data in LCTS. For values TS = 0, 1, or 3, the load curve defines data pairs. A pair consists of a thermal breakpoint and the new time step. The time step will be adjusted to hit the time breakpoints exactly. After the time breakpoint, the time step will be set to the 'new time step' ordinate value in the load curve. For the options TS = 2 and 4, however, interpolated load curve data is used to determine time step size as function of simulation time.

3. **Automatic time step size control.** If automatic time step size control is used (TS = 1) with load curve definitions of the limits (TMIN, TMAX, and DTEMP), the time step is adjusted to hit the breakpoints in the curves exactly. The time stepping algorithm ensures that the step size of the last time step before the breakpoint is not smaller than 50% of the previous one. If necessary, the sizes of the last two time steps before the breakpoint are averaged.

*CONTROL_TIMESTEP

Purpose: Set structural time step size control using different options.

Card 1	1	2	3	4	5	6	7	8
Variable	DTINIT	TSSFAC	ISDO	TSLIMT	DT2MS	LCTM	ERODE	MS1ST
Type	F	F	I	F	F	I	I	I
Default	↓	↓	0	0.0	0.0	0	0	0

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	DT2MSF	DT2MSLC	IMSCL			RMSCL	EMSCL	IHDO
Type	F	I	I			F	F	I
Default	not used	not used	0			0.0	0.0	0

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	RMAVG	IGADO	DTUSR	DTDYNV				
Type	F	I	F	I				
Default	0.0	0	0.0	0				

VARIABLE	DESCRIPTION
DTINIT	Initial time step size: EQ.0.0: LS-DYNA determines initial step size.

VARIABLE	DESCRIPTION
TSSFAC	Scale factor for computed time step (old name SCFT). See Remark 2 below.

LT.0.0: |TSSFAC| is the load curve or function ID defining the scale factor as a function of time.

ISDO: Basis of time size calculation for 4-node shell elements. 3-node shells use the shortest altitude for options 0,1 and the shortest side for option 2. This option has no relevance to solid elements, which use a length based on the element volume divided by the largest surface area.

EQ.0: Characteristic length is given by

$$\frac{\text{area}}{\min(\text{longest side},\ \text{longest diagonal})}.$$

EQ.1: Characteristic length is given by

$$\frac{\text{area}}{\text{longest diagonal}}.$$

EQ.2: Based on bar wave speed and,

$$\max\left[\text{shortest side}, \frac{\text{area}}{\min(\text{longest side},\ \text{longest diagonal})}\right].$$

> WARNING: Option 2 can give a much larger time step size that can lead to instabilities in some applications, especially when triangular elements are used.

EQ.3: This feature is currently unavailable. Time step size is based on the maximum eigenvalue. This option is okay for structural applications where the material sound speed changes slowly. The cost related to determining the maximum eigenvalue is significant, but the increase in the time step size often allows for significantly shorter run times without using mass scaling.

TSLIMT: Shell element minimum time step assignment, TSLIMT. When a shell controls the time step, element material properties (moduli *not* masses) will be modified such that the time step does not fall below the assigned step size. This option is applicable only to shell elements using material models:

*MAT_PLASTIC_KINEMATIC,

VARIABLE	DESCRIPTION

*MAT_POWER_LAW_PLASTICITY,

*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY,

*MAT_PIECE-WISE_LINEAR_PLASTICITY.

> WARNING: This so-called stiffness scaling option is NOT recommended. The DT2MS option below applies to all materials and element classes and is preferred.

If both TSLIMT and DT2MS below are active and if TSLIMT is input as a positive number, then TSLIMT defaults to 10^{-18}, thereby disabling it.

If TSLIMT is negative and less than |DT2MS|, then |TSLIMT| is applied prior to the mass being scaled. If |DT2MS| exceeds the magnitude of TSLIMT, then TSLIMT is set to 10^{-18}.

DT2MS — Time step size for mass-scaled solutions. (Default = 0.0)

- GT.0.0: Positive values are for quasi-static analyses or time history analyses where the inertial effects are insignificant.
- LT.0.0: TSSFAC × |DT2MS| is the minimum time step size permitted, and mass scaling is done if and only if it is necessary to meet the Courant time step size criterion. This option can be used in transient analyses if the mass increases remain insignificant. See also the variable MS1ST below and the *CONTROL_TERMINATION variable ENDMAS.

> WARNING: Superelements from *ELEMENT_DIRECT_MATRIX_INPUT are not mass-scaled; consequently, DT2MS does not affect their time step size. In this case, an error termination will occur, and DT2MS will need to be reset to a smaller value.

LCTM — Load curve ID that limits the maximum time step size (optional). This load curve defines the maximum time step size permitted as a function of time. If the solution time exceeds the final time value defined by the curve, the computed step size is used. If the time

VARIABLE	DESCRIPTION
	step size from the load curve is exactly zero, the computed time step size is also used.
ERODE	Erosion flag for elements with small time step. See Remark 6. EQ.0: Calculation will terminate if the solution time step drops to tsmin (see *CONTROL_TERMINATION). EQ.1: Solid elements or thick shell elements that cause the time step to drop to tsmin will erode; similarly, SPH particles that cause the time step to drop will be deactivated. EQ.10: Shell elements with a time step below tsmin will erode. EQ.11: Same as ERODE = 1, but shell elements will also erode EQ.100: Beam elements with a time step below tsmin will erode. EQ.101: Same as ERODE = 1, but beam elements will also erode EQ.110: Beam and shell elements will erode. EQ.111: Same as ERODE = 1, but beam and shell elements will also erode
MS1ST	Option for mass scaling that applies when DT2MS < 0. EQ.0: Mass scaling is considered throughout the analysis to ensure that the minimum time step cannot drop below TSSFAC × |DT2MS|. Added mass may increase with time, but it will never decrease. (default) EQ.1: Added mass is calculated at the first time step and remains unchanged thereafter. The initial time step will not be less than TSSFAC × |DT2MS|, but the time step may subsequently decrease, depending on how the mesh deforms and the element characteristic lengths change.
DT2MSF	Reduction (or scale) factor for initial time step size to determine the minimum time step size permitted. Mass scaling is done if it is necessary to meet the Courant time step size criterion. If this option is used, DT2MS effectively becomes –DT2MSF multiplied by the initial time step size, Δt, before scaling Δt by TSSFAC. This option is active only if DT2MS = 0 above.
DT2MSLC	Load curve for determining the magnitude of DT2MS as a function of time, $f_{\mathrm{DT2MS}}(t)$, during the explicit solutions phase. Time zero must be in the abscissa range of this curve. At a given simulation

VARIABLE	DESCRIPTION
	time t, $f_{\text{DT2MS}}(t) \times \text{sign}(\text{DT2MS})$ plays the role of DT2MS according to the description for DT2MS above. *All* negative ordinate values may be used in the curve, but the ordinate values *may not* change sign during the simulation. In the negative ordinate values case, $f_{\text{DT2MS}}(t)$ itself (sign and magnitude) determines how mass scaling is performed and DT2MS is neglected. GT.0: The magnitude of the added mass is the maximum of the current value and the value from the load curve. Hence, the amount of added mass will never decrease, although the time step size may be reduced as desired using DT2MSLC. LT.0: Uses load curve |DT2MSLC|. Allows the amount of added mass to decrease with time.
IMSCL	Flag for selective mass scaling if and only if mass scaling active. Selective mass scaling does not scale the rigid body mass and is therefore more accurate. Since it is memory and CPU intensive, it should be applied only to small, finely meshed parts. EQ.0: No selective mass scaling EQ.1: All parts undergo selective mass scaling. LT.0: Recommended; |IMSCL| is the part set ID of the parts that undergo selective mass scaling. All other parts are mass scaled the usual way.
RMSCL	Flag for using rotational option in selective mass scaling EQ.0.: Only translational inertia is selectively mass scaled. NE.0.: Both translational and rotational inertia are selectively mass scaled.
EMSCL	Fraction of added mass from mass scaling that contributes to gravity loads, in addition to the physical mass. See also *LOAD_BODY. This number should be between 0 and 1.
IHDO	Method for calculating solid element time steps (see Remark 7): EQ.0: Default method EQ.1: Modified method to improve time step continuity
RMAVG	Global value for the average rotational mass. For finite element nodes without rotational mass (such as those belonging to solid

VARIABLE	DESCRIPTION
	elements), an average value is calculated internally (using the existing rotational masses of all shells and beams in the model) and assigned to each of these nodes, such as for penalty tied contacts. This value is written to the d3hsp file during initialization ("... get an average value of RMAVG = ..."). Sometimes this value is too small, leading to instabilities in the tied contact. In these cases, the value should be chosen accordingly higher. Alternatively, RINRT = 1 can be used on *CONTROL_SOLID.
IGADO	Method for calculating time steps for IGA elements: EQ.0: Default method (conservative) EQ.1: Account for interelement continuity (usually leads to larger time steps)
DTUSR	User-defined time step for explicit analysis. A nonzero value invokes a call to user subroutine `utimestep` in dyn21.F. This feature can be used to synchronize time steps between coupled codes.
DTDYNV	Flag to consider the effects of dynamic viscosity on the critical time step. Like bulk viscosity, dynamic viscosity affects the critical time step. This flag applies only to solid elements with *MAT_NULL. EQ.0: Do not account for the effect of dynamic viscosity. This setting can lead to instabilities. EQ.1: Account for the effect of dynamic viscosity

Remarks:

1. **Time step scale factor TSSFAC.** During the solution, we loop through the elements and determine a new time step size by taking the minimum value over all elements:

$$\Delta t^{n+1} = \text{TSSFAC} \times \min\{\Delta t_1, \Delta t_2, \ldots, \Delta t_N\}$$

where N is the number of elements. The time step size roughly corresponds to the transient time of an acoustic wave through an element using the shortest characteristic distance. For stability reasons, the scale factor TSSFAC should be set to a value less than 1.0. The default value of TSSFAC is as follows:

a) If contacts are present, and SLSFAC is specified in CONTROL_CONTACT, then

$$\text{TSSFAC} = \begin{cases} 0.333 & \text{for} \quad \text{SLSFAC} \geq 9. \\ 0.667 & \text{for} \quad 1. \leq \text{SLSFAC} < 9. \end{cases}$$

b) TSSFAC = 0.667 if either of the following cards is used:

*INITIAL_DETONATION

*BOUNDARY_NON_REFLECTING

c) TSSFAC = 0.90 otherwise

Values larger than .90 will often lead to instabilities

2. **Sound speed and element size.** The sound speed in steel and aluminum is approximately 5 mm per microsecond; therefore, if a steel structure is modeled with element sizes of 5 mm, the computed time step size would be 1 microsecond. Elements made from materials with lower sound speeds, such as foams, will give larger time step sizes. Avoid excessively small elements and be aware of the effect of rotational inertia on the time step size in the Belytschko beam element. Sound speeds differ for each material, for example, consider:

Air	331 m/s
Water	1478
Steel	5240
Titanium	5220
Plexiglass	2598

3. **Use rigid bodies when possible.** It is recommended that stiff components be modeled by using rigid bodies. Do not scale the Young's modulus, as that can substantially reduce the time step size.

4. **Triangular elements.** The altitude of the triangular element should be used to compute the time step size. Using the shortest side is okay only if the calculation is closely examined for possible instabilities. This is controlled by parameter ISDO.

5. **Selective mass scaling.** In the explicit time integration context and in contrast to conventional mass scaling, selective mass scaling (SMS) is a well thought out scheme that not only reduces the number of simulation cycles but also does not significantly affect the dynamic response of the system under consideration. The drawback is that a linear system of equations must be solved in each time step for the accelerations. In this implementation, a preconditioned conjugate gradient method (PCG) is used.

   An unfortunate consequence of this choice of solver is that the efficiency will worsen when attempting large time steps since the condition number of the assembled mass matrix increases with the added mass. Therefore, caution should be taken when choosing the desired time step size. For large models, it is also recommended to only use SMS on critical parts since it is otherwise likely to

slow down execution; the bottleneck being the solution step for the system of linear system of equations.

While some constraints and boundary conditions available in LS-DYNA are not supported for SMS, they can be implemented upon request from a user.

A partial list of constraints and boundary conditions supported with SMS:

- Pointwise nodal constraints in global and local directions
- Prescribed motion in global and local directions
- Adaptivity
- Rigid walls
- Deformable elements merged with rigid bodies
- Constraint contacts and spotwelds
- Beam release constraints

By default, only the translational dynamic properties are treated. This means that only rigid body translation will be unaffected by the mass scaling imposed. There is an option to also properly treat rigid body rotation in this way, this is invoked by flagging the parameter RMSCL. A penalty in computational expense is incurred, but the results could be improved if rotations are dominating the simulation.

Selective mass scaling is supported for most element formulations. For beam elements, only types 1 (Hughes-Liu) and 9 (Spotweld) are supported at the moment.

6. **ERODE.** If DTMIN > 0 on *CONTROL_TERMINATION, then tsmin is calculated by tsmin = dtstart × DTMIN where dtstart is the initial step size internally determined.

   If tsmin > 0 and ERODE > 0, and if the solution time step drops below tsmin, then those elements that cause the time step to drop will be deleted so that the solution can continue with a time step that stays at or above tsmin. Valid values for ERODE are 0, 1, 10, 11, 100, 101, 110, and 111. Each digit in ERODE switches on or off checking and deletion for a type of element where a one turns on checking and a zero turns it off. The ones place controls solid and thick shell elements as well as SPH particles. The tens place controls shell elements, and the hundreds place controls beam elements. Set ERODE = 111 to check all element types.

   In addition to maintaining a reasonable time step, setting ERODE = 1, 11, or 111 will also prevent error terminations due to negative volume of solid elements. However, the solid elements checked for negative volume can be limited to those contained in a part set by setting part set PSFAIL on *CONTROL_SOLID.

7. **IHDO.** The explicit time-step calculation for solid elements depends not only on the highest element frequency but also on the volumetric strain rate and the linear coefficient of bulk viscosity. The default method for including the viscous effects causes a discontinuity in the equation when the volumetric strain rate changes sign. As a result, some explicit dynamic solutions are observed to have a solution time step that fluctuates between two values that differ by a few percent. An alternative method that eliminates the discontinuity in the viscous effects is invoked by setting IHDO = 1.

*CONTROL_UNITS

Purpose: Specify the user units for the current keyword input deck. This does not provide any mechanism for automatic conversion of units of any entry in the keyword input deck. It is intended to be used for several purposes, but currently only for the situation where an external database in another set of units will be loaded and used in the simulation. In this case, *CONTROL_UNITS provides the information necessary to convert the external data into internal units (see *CHEMISTRY_CONTROL for such external databases).

If the needed unit is not one of the predefined ones listed for use on the first card, then the second optional card is used to define that unit. Any non-zero scales that are entered on optional Card 2 override what is specified on the first card. These scales are given in terms of the default units on Card 1. For instance, if 3600.0 is given in the second 20 character field on the optional second card (TIME_SCALE), then 'hour' is the time unit (3600 seconds).

Card 1	1	2	3	4	5	6	7	8
Variable	LENGTH	TIME	MASS	TEMP				
Type	A	A	A	A				
Default	m	sec	kg	K				

User Defined Units Card. Optional card only used when a new unit needs to be defined.

Card 2	1	2	3	4	5	6	7	8
Variable	LENGTH_SCALE		TIME_SCALE		MASS_SCALE			
Type	F		F		F			
Default	1.0		1.0		1.0			

VARIABLE	DESCRIPTION
LENGTH	Length units: EQ.m: meter (default) EQ.mm: millimeter EQ.cm: centimeter EQ.in: inch EQ.ft: foot
TIME	Time units: EQ.sec: second (default) EQ.ms: millisecond EQ.micro_s: microsecond
MASS	Mass units: EQ.kg: kilogram (default) EQ.g: gram EQ.mg: milligram EQ.lb: pound EQ.slug: pound × second^2 / foot EQ.slinch: pound × second^2 / inch EQ.mtrc_ton: metric ton
TEMP	Temperature units: EQ.K: Kelvin (default) EQ.C: Celsius EQ.F: Fahrenheit EQ.R: Rankine
LENGTH_SCALE	Number of meters in the length unit for the input deck
TIME_SCALE	Number of seconds in the time unit for the input deck
MASS_SCALE	Number of kilograms in the mass unit for the input deck

*CONTROL_2D_REMESHING_REGION

Purpose: Specify a region of a 2D mesh to remesh. The keyword should be used with ADPTYP = 8 on *CONTROL_ADAPTIVE.

Note that each definition of *CONTROL_2D_REMESHING_REGION specifies one region. To include more than one region, you need to include multiple *CONTROL_2D_REMESHING_REGION keywords in the input deck.

Include as many cards as necessary. This input ends at the next keyword ("*") card. Each line for LTYP = 1, 2, 3, 7, or 8 defines a region to remesh. If there are several of these lines, the intersection of these regions is remeshed.

Card 1	1	2	3	4	5	6	7	8
Variable	LTYP	PAR1	PAR2	PAR3	PAR4	PAR5	PAR6	PAR7
Type	I	F	F	F	F	F	F	F
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
LTYP	Type of regions defined by the parameters PAR*i*: EQ.1: Box. PAR1 is the ID of *DEFINE_BOX that defines the region to remesh. The other parameters are not used. EQ.2: Part set. PAR1 is the ID of *SET_PART that selects the parts to remesh. The other parameters are not used. EQ.3: Region around elements in contact from a specified contact. To specify the desired contact, set PAR1 to its order of appearance in the input deck. For instance, PAR1 = 5 if the desired contact is the 5th contact keyword to appear in the input deck. A box is created around each element in contact for the remeshing. PAR2 through PAR4 add padding around the box to increase the remeshing region. PAR2 > 0 is the length subtracted from the *x*-coordinate of the box's lower corner. PAR3 > 0 is the length added to the *x*-coordinate of the box's upper corner. PAR4 > 0 is the length subtracted from the *y*-coordinate of the box's lower corner. PAR5 > 0 is the length added to the *y*-coordinate of the box's upper corner. The last 2 parameters are not used.

VARIABLE	DESCRIPTION
	EQ.4: PAR1 is the ID of *DEFINE_BOX that selects mesh boundaries (edges) of remeshing regions along which nodes added after remeshing (hanging nodes between edge ends) keep their initial parametric positions between the boundary corner nodes (edge-end constraining nodes). The other parameters are not used. By default, when nodes are added to the edges of elements in the region and the edges are not on the mesh boundary, the positions and velocities of the hanging nodes are interpolated from the positions and velocities of the original constraining nodes along these edges (constraining nodes are nodes that existed before remeshing). If the edge of a remeshed element is on the mesh boundaries, the positions and velocities of the hanging nodes on the edge are, by default, not interpolated because they are likely to be subject to boundary conditions. With LTYP = 4, the hanging nodes on the boundary of the mesh are interpolated.
	EQ.5: PAR1 is the ID of *SET_NODE that selects nodes along shell edges. After remeshing, the node set is recreated with nodes along the same shell edges. PAR2 > 0 is a thickness for the shell edges to select the new nodes after remeshing. The other parameters are not used.
	EQ.6: PAR1 is the ID of *SET_PART that selects the parts for which the total displacements are output to d3plot after remeshings.
	EQ.7: PAR1 is the ID of *SET_SHELL that selects the shells to remesh. The other parameters are not used.
	EQ.8: PAR1 is the ID of a 2D *SET_SEGMENT that selects the shells to remesh. A 2D *SET_SEGMENT is a list of segments with 2 nodes (the 2nd node is repeated for the 3rd and 4th node IDs). A 2D segment normal points to the right when going from the 1st to the 2nd node. A shell is in the region if nominally all the segment normal vectors point towards it (see Figure 12-92 for illustration). The other parameters are not used.
	EQ.9: PAR1 is the ID of a *SET_PART that selects the shell parts for which an area-weighted mapping of the element data is applied. The other parameters are not used. If PAR1 is 0, all parts are selected to be mapped.
	EQ.10: PAR1 is the ID of *DEFINE_CURVE that lists coordinate

VARIABLE	DESCRIPTION
	couples (x,y) to make a closed curve (the last point in this list is the same as the first one). The region of shells to remesh is on the right when moving along the curve from its 1st to last point (see Figure 12-93).
PAR*i*	Parameters defined by LTYP

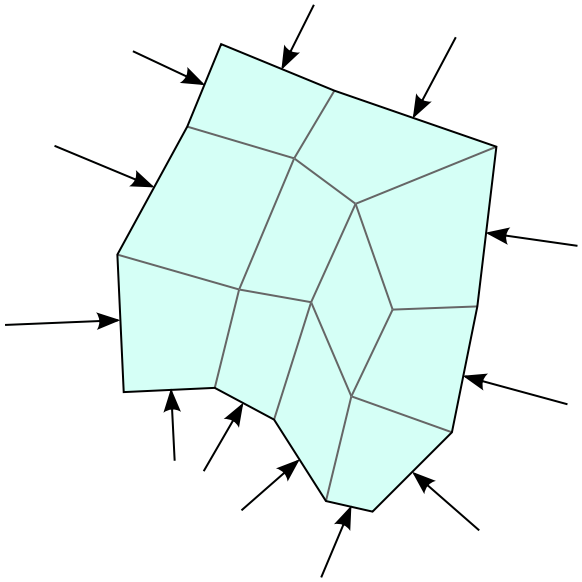


Figure 12-92. Example of a region specified with LTYP = 8. The arrows indicate the normal vectors for the segments.

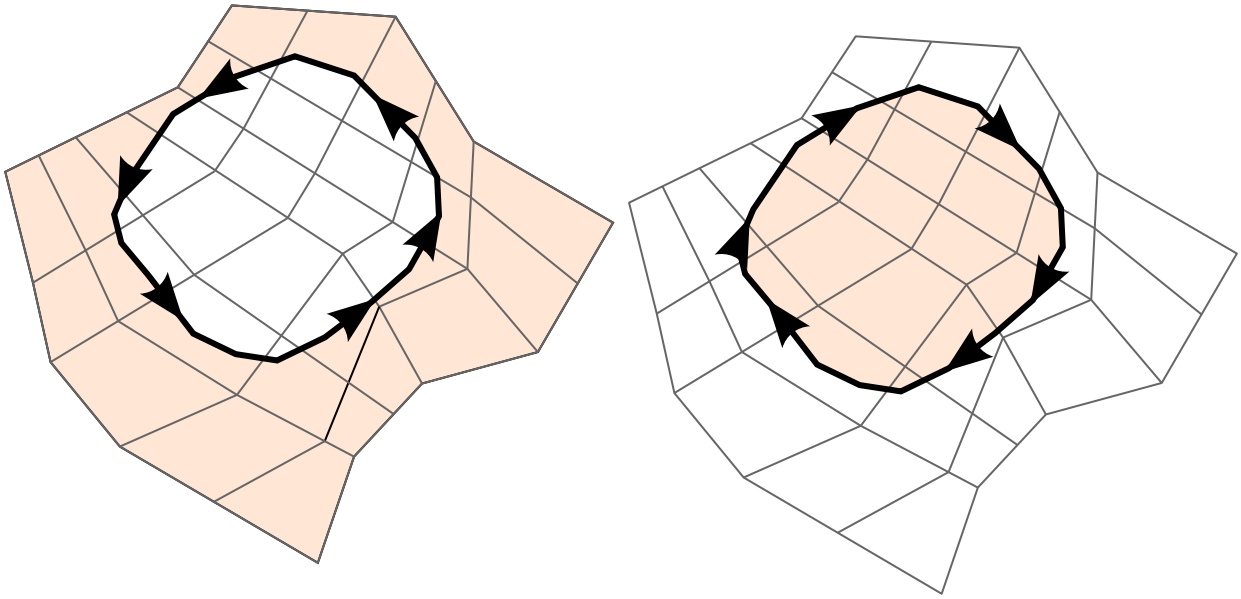


Figure 12-93. Example of regions specified with LTYP = 10. The arrows indicate the direction of the curve as input. The region depends on this direction.

*CONTROLLER

Purpose: The keyword *CONTROLLER provides capability related to the control application, such as controller design, model order reduction, etc.

*CONTROLLER_PLANT

*CONTROLLER_PLANT

Purpose: Perform model order reduction for linear systems and export the reduced matrices into the state space format $\dot{X} = AX + Bu, Y = CX + Du$. Matrices A, B, C, D can be written in specified file format, such as SCILAB and MATLAB.

This feature currently works for the SMP, double precision versions. We implemented two methods for model order reduction: modal truncation and Krylov subspace method. The modal truncation method is based on the truncated modes, meaning eigenvectors, which carry more physical meaning. The Krylov subspace method matches moments of transfer functions to ensure the reduced system has a similar response to the original one. For both methods, the input deck requires necessary *CONTROL_IMPLICIT cards.

ID card. Define plant ID, specify input/output channel numbers and method.

Card 1	1	2	3	4	5	6	7	8
Variable	PLNTID	NIN	NOUT	NMODE	MTXQ	MTXR	MOPT	
Type	I	I	I	I	I	I	I	
Default	none	none	none	0	0	0	0	

File card. Specify file names in SCILAB and MATLAB format.

Card 2	1	2	3	4	5	6	7	8
Variable	FSCILAB				FMATLAB			
Type	A				A			

Input DOFs card. Specify the input node/set and its DOFs. Repeat this card if necessary.

Card 3	1	2	3	4	5	6	7	8
Variable	NODI1	DOFI1	NODI2	DOFI2	NODI3	DOFI3	NODI4	DOFI4
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Output DOFs card. Specify the output node/set and its DOFs. Repeat this card if necessary.

Card 4	1	2	3	4	5	6	7	8
Variable	NODO1	DOFO1	NODO2	DOFO2	NODO3	DOFO3	NODO4	DOFO4
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Settings card. Set frequency number and tolerance for the Krylov subspace method.

Card 5	1	2	3	4	5	6	7	8
Variable	NFEQ	DEFTOL						
Type	I	F						
Default	1	10^{-9}						

Mode/Frequency card. Specify the mode and frequency index. Repeat this card if necessary.

Card 6	1	2	3	4	5	6	7	8
Variable	MOD1	MOD2	MOD3	MOD4	MOD5	MOD6	MOD7	MOD8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
PLNTID	Plant ID
NIN	Number of input DOFs, such as nodal force or voltage. If all nodes within a set share a single input variable, together they account for one DOF. For example, all nodes within a set share a single input voltage for a piezo actuator.
NOUT	Number of output DOFs, such as nodal displacement or voltage. Note that the same node velocity will be automatically exported as well.
NMODE	Number of modes for the modal truncation, or number of base vectors for the Krylov method. If zero, all active DOFs will be used (not recommended). The reduced system will have a dimension of 2NMODE for the modal truncation method, and NMODE for the Krylov method.
MTXQ	Q matrix for linear-quadratic-regular (LQR) method (unused currently)
MTXR	R matrix for linear-quadratic-regular (LQR) method (unused currently)
MOPT	Modal order reduction method (see Remark 1): EQ.0: Modal truncation method EQ.1: Krylov subspace method
FSCILAB	File name in SCILAB format .sci. If specified, the reduced matrices will be written accordingly. If left blank, no such file will be generated.

VARIABLE	DESCRIPTION
FMATLAB	File name in MATLAB format .m. If specified, the reduced matrices will be written accordingly. If left blank, no such file will be generated.
NODI*x*	Node or node set index for the input channel. GT.0: Node index LT.0: Node set index within which all nodes share the same input variable, such as force, voltage.
DOFI*x*	Degree-of-freedom for input: EQ.1: Nodal force in the x-direction, f_x EQ.2: Nodal force in the y-direction, f_y EQ.3: Nodal force in the z-direction, f_z EQ.7: Voltage if piezoelectric materials are defined. See Remark 2.
NODO*x*	Node index for output
DOFO*x*	Degree-of-freedom for output: EQ.1: Displacement along the x-direction EQ.2: Displacement along the y-direction EQ.3: Displacement along the z-direction EQ.7: Voltage output if piezoelectric materials are defined.
NFEQ	Number of shifted frequencies to generate the Krylov base vectors. In most cases, a single frequency at zero rad/s works. For the modal truncation method, just leave as it is.
DEFTOL	Deflation tolerance for the Krylov method. The default value of 10^{-9} works in most cases. For the modal truncation method, just leave as it is.
MOD*x*	List all NMODE mode indexes for the modal truncation method, or NFEQ shifting frequencies (unit: rad/s) for the Krylov method. The default setting of a single frequency at 0 rad/s works in most Krylov cases. For the modal truncation method, a negative MOD*x* triggers mode generation between MOD_{x-1} and $-MOD_x$, meaning all modes between MOD_{x-1} and $-MOD_x$ will be considered.

Remarks:

1. **MOPT.** If MOPT = 0, the modal truncation method is selected to perform the model order reduction based on the NMODE eigenvectors. Users need to define *CONTROL_IMPLICIT_GENERAL and *CONTROL_IMPLICIT_EIGENVALUE to generate the required eigenvectors, and the eigenvalue number *neig* should not be less than NMODE. If MOPT = 1, the Krylov subspace method will be selected. No eigenvalues are calculated, but *CONTROL_IMPLICIT_GENERAL is still needed to acquire the system matrices. The default setting of NFEQ = 1, DEFTOL = 10^{-9}, and 0 frequency works in most cases for the Krylov method.

2. **Voltage Input.** DOFI*x* = 7 indicates a voltage input. The material of that respective index should be piezoelectric, that is, *MAT_ADD_PZELECTRIC should be used.

*COSIM

Purpose: The keyword *COSIM allows LS-DYNA to co-simulate with other software. Currently, co-simulation works through either the functional mock-up interface (FMI) or Ansys System Coupling™ (SYC).

*COSIM_FMI_CONTROL

*COSIM_FMI_INTERFACE

*COSIM_SYC_CONTROL

*COSIM_SYC_INTERFACE

*COSIM_FMI_CONTROL

Purpose: Define the functional mock-up interface (FMI) co-simulation settings. With this keyword, you will define the co-simulation role of LS-DYNA, such as primary or secondary. You can also specify additional settings, such as IP address/port, functional mock-up unit (FMU) location, and co-simulation time step.

This keyword should be included in the input deck along with *COSIM_FMI_INTERFACE. You also need to download a free plugin called "FMU Manager" to enable this feature. The latest zip file that includes this plugin is located at the Ansys Download Center (https://download.ansys.com). The zip file also contains detailed documentation and illustrative sample decks.

Card 1	1	2	3	4	5	6	7	8
Variable	APPID		OPT	MODE	FMI			
Type	A20		A10	A10	I			
Default	none		none	none	0			

Settings card. Define additional settings. Repeat this card if necessary. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	SETTING							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
APPID	FMU identification. Each FMU must have a unique APPID.
OPT	LS-DYNA's role (see Remark 1): EQ.G: Generation mode. LS-DYNA will generate a new FMU. EQ.C: Co-simulation mode. LS-DYNA will co-simulate with an existing FMU.

VARIABLE	DESCRIPTION
MODE	LS-DYNA's mode (see Remark 1): EQ.P: LS-DYNA is primary, and another software is secondary. EQ.S: LS-DYNA is secondary, and another software is primary.
FMI	Flag to select FMI standard: EQ.1: The generated or co-simulated FMU is based on FMI1.0 standard. EQ.0.OR.2: The generated or co-simulated FMU is based on FMI2.0 standard. This is the default.
SETTINGS	Settings for the generating or using the FMU. Up to 80 characters per line. Multiple lines are allowed for each setting and multiple settings can be defined for each FMU. See Remark 2.

Remarks:

1. **How Co-Simulation Works.** The secondary software generates an FMU, which is imported into the primary software for co-simulation. The primary controls the co-simulation time step, Δt_1, which is different from the explicit time step, Δt_2, in LS-DYNA. Generally, the former should be greater than the latter by several magnitudes, depending on your application ($\Delta t_1 \leq \Delta t_2$ is also allowed). If LS-DYNA is secondary, Δt_1 is set in the primary software. If LS-DYNA is the primary, Δt_1 is set in LS-DYNA by "<time> fmudt=*xxx*" through Card 2 of this keyword as discussed in Remark 2.

   The process of setting up the FMU and running the co-simulation depends on whether LS-DYNA is the primary or secondary:

   a) *LS-DYNA is the secondary.* In prior versions of LS-DYNA this would correspond to LS-DYNA being the *slave*. If you want to run LS-DYNA as the secondary during a co-simulation, usually you need to set OPT = G and run LS-DYNA to generate an FMU. After importing the FMU into another primary software, switch OPT from G to C to run LS-DYNA in the co-simulation mode to co-simulate with the 3rd party software. In this case, you run LS-DYNA twice.

   b) *LS-DYNA is the primary.* In prior versions of LS-DYNA this would correspond to LS-DYNA being the *master*. If you want to run LS-DYNA as the primary during a co-simulation, you usually need to generate an FMU with another software and import the FMU into LS-DYNA for co-simulation. In

this case, set OPT = C and run LS-DYNA only once to connect to the other software for co-simulation.

2. **SETTINGS.** This field is used to setup general settings for the FMU, such as the FMU directory or the co-simulation time. Each setting must start on a new line and begins with <*setting name here*> followed by parameters for the setting. A setting can span multiple lines with each line having a maximum of 80 characters.

   See the table below for a full list of settings in alphabetical order. A description of each setting is below in the Overview of Settings section which is followed by some examples.

Setting Name	Description	OPT	MODE
<fmudir>	Specify FMU directory	C, G	P, S
<fmu_input>	Generate FMU with additional input variables	G	P
<fmu_output>	Generate FMU with additional output variables	G	P
<home>	Specify the direction of the FMU manager	C, G	P, S
<hopsan>	Generate FMU / co-simulate with Hopsan	C, G	S
<name>	Match variables in LS-DYNA and FMU	C, G	P, S
<message>	Specify how often to print out the co-sim messages		
<parameter>	Generate FMU with additional parameter variables	G	P
<socket>	Generate FMU with specified socket properties	G	S
<tcp>	Generate FMU with specified IP and bind LS-DYNA with specified port	C, G	S
<time>	Specify time step and duration of co-simulation	C	P, S

Overview of Settings:

The settings are described in greater detail below. Parameters in italics are for illustration only and indicate the parameters that you will replace for your co-simulation.

<fmudir> ***path***

The setting specifies either where the FMU is located during co-simulation mode or where the FMU is generated in the generation mode. If not included, the directory is assumed to be the folder of the input keyword file. Note the difference of \ and / in Windows or Linux. If the directory is relative to the input file, you should start with ..\ or .\ for Windows, ../ or ./ for Linux. Please avoid any space characters in or after the directory.

<fmu_input> name = ***current*****, type =** ***real*****, value =** ***0***

With this setting, you can define additional FMU input variables during generation mode. Currently only variables of **type=*real*** are supported. The initial value is set with **value**. LS-DYNA must be in the primary mode for this setting to work, meaning MODE = P and OPT = G.

<fmu_output> name=*voltage***, type=***real***, value=***0***

With this setting, you can define additional FMU output variables during generation mode. Currently only variables of **type=*real*** are supported. The initial value is set with **value**. LS-DYNA must be in the primary mode for this setting to work, meaning MODE = P and OPT = G.

<home> ***path***

This setting specifies the directory of the FMU manager kit. If not included, the directory is assumed to be the parent folder of the keyword input file, meaning ..\. You can also specify a directory relative to the input file, starting with ..\ or .\ for Windows and ../ or ./ for Linux. Please avoid any space character in or after the directory.

<hopsan> name = ***port1*****, type =** ***hydraulic_Q*****, segment =** ***1*****, node =** ***1867*****, dir = +***z*

You specify this setting when you want to co-simulate with piston models in Hopsan. Here, **name** is the port number, **type** is the piston type, **segment** is the piston segment set ID to which the hydraulic force is applied, and **node** can be any node on the piston where piston velocity is picked up. The parameter **dir** gives the direction of the piston motion. The positive direction is the extension while the negative direction is the retraction.

<message> step = ***100***

This setting specifies how often co-simulation messages are output to fmu_msg. They are output every ***step*** co-simulation time steps. If not specified, LS-DYNA will print out the following message every 100 co-simulation time steps:

FMU time: 0.000000e+00 cosim-dt: 6.644e-05 cosim-step: 1

FMU time: 6.471417e-03 cosim-dt: 6.258e-05 cosim-step: 101

<name> *DX500* = *In1*

This setting is used to match variables in LS-DYNA and the FMU during the generation or co-simulation mode. The above example tells LS-DYNA that *In1* in FMU is equivalent to the *X* displacement of node 500 (additional cases are documented in the co-simulation examples manual). During co-simulation, LS-DYNA will match the variables on both sides of "=" to send out or receive data from the FMU. The variable on the right side can be found inside the modelDescription.xml of the FMU. The variable on the left side corresponds to the LS-DYNA local variables with its name being the combination of the value of FIELD with the node number; see *COSIM_FMI_INTERFACE.

<parameter> name=*PCOEF1*, type=*real*, value=*1*

This setting is used to define parameter variables for an FMU to be generated. Currently, only variables of **type=*real*** are supported. The parameter **value** defines the initial value for the parameter. LS-DYNA must be in the primary mode for this setting to work, that is, MODE = P and OPT = G. Usually, you need to modify function `getReal()` in template/fmi_udf.c for such application to generate the needed FMU.

<socket> nconnect=*20*, tdelay=*1000*, recvtimeout=*0*

This setting is used to define the socket properties of an FMU to be generated. LS-DYNA must be in the secondary mode for this item to work, meaning MODE = S and OPT= G. At the beginning of the co-simulation, the other software will try **nconnect** times at most to reach the LS-DYNA computer for connection. If the socket communication to the IP designated with **<tcp>** is not successful, the co-simulation fails. **tdelay** is the time delay in milliseconds for the other software to reconnect to LS-DYNA if the previous connection fails. **recvtimeout** is the max time in milliseconds that FMU waits for data from LS-DYNA, and beyond this time limit, the co-simulation fails due to lost data. If **recvtimeout=0**, the FMU will wait infinitely for LS-DYNA.

<tcp> ip=*127.0.0.1*, port=*39400*

This setting is used to define the FMU properties to be generated. LS-DYNA must be in secondary mode (MODE = S) for this setting to work. The parameter **ip** is the IP address of the LS-DYNA computer in the co-simulation. If the co-simulation is local, you can always set **ip** to be 127.0.0.1. You can also specify a .txt file that includes the IP address in the first line instead of the IP address with **ip=*ip_addr.txt***. If this setting is not included, the default IP address is 127.0.0.1 with port 39400.

<time> tstart = *0*, tend = *1.2*, option = *0*, fmudt = *0.0005*, *tscale = 10*

This setting defines the LS-DYNA properties during co-simulation. When tstart < t < tend, the co-simulation is occurring. The parameter **option** controls the value of exported/imported variables in LS-DYNA when t > tend. When option = 0, the exported/imported variables are set to 0 when t > tend; when option = 1, the exported/imported variables for t > tend are held constant with the value from t = tend. When t < tstart, the imported variables are set to 0 and the exported variables are set to their initial values, which are specified by VINIT in *COSIM_FMI_INTERFACE.

The parameter **fmudt** defines the co-simulation time step between LS-DYNA and the FMU, and it only applies when LS-DYNA is the primary during the co-simulation. The co-simulation time step is different from the explicit time step of LS-DYNA. Generally, the former is usually greater by several magnitudes, however, values of smaller or comparable magnitude are also allowed. Between co-simulation time steps, the FMU variables are held constant. If this setting is not defined, tstart = 0, tend = 10^6, option = 0, fmudt = 100 × tstep, where tstep is the LS-DYNA explicit time step.

For cases when the FMU and LS-DYNA have different time scales, use **tscale** to specify the scale factor between the time scales for FMU and LS-DYNA. The default value is 1 without specification. For instance, if the MATLAB side is based on seconds and LS-DYNA is based on milliseconds, set **tscale** = 1000. In this case, when MATLAB proceeds by 1 s in co-simulation, LS-DYNA will proceed by 1000 ms. Note that the values of tstart, tend and fmudt are based on the LS-DYNA time unit.

Examples:

The following examples only serve as an illustration of syntax.

Example 1

In this example, LS-DYNA is the secondary and generates the FMU. The FMU is then imported into another primary software. You can switch G to C and run LS-DYNA to connect to the primary software.

```
*COSIMULATION_FMI_CONTROL
                MOTOR         G         S
<home> C:\DYNA_bin\FMU_Manager_v4_deliver
<fmudir> C:\DYNA_bin\FMU_Manager_v4_deliver\dyna_key
<tcp> ip=127.0.0.1,port=39400
<socket> nconnect=20,tdelay=1000,recvtimeout=0
<name> DX500=input1
<time> tstart=0, tend=0.2, option=1
```

Example 2

Here LS-DYNA is the primary during co-simulation. You generate an FMU with another secondary software and import it into LS-DYNA for co-simulation. In the settings, you set the co-simulation time step and match variables in LS-DYNA and FMU.

```
*COSIMULATION_FMI_CONTROL
                PID1         C         P
<home> ..\
<fmudir> .\
<time> fmudt=0.005
<name> DX500=disp_in1, VX500=veloc_in2
<name> FX500=force_out1
```

*COSIM_FMI_INTERFACE

Purpose: Define the interface variables during co-simulation. For instance, you may need LS-DYNA to export certain node displacements to another software and to import certain nodal forces.

This keyword should be included in an input deck with *COSIM_FMI_CONTROL. *You also need to download a free plugin called "FMU Manager" to enable this feature.*

This feature is available as of R12.

ID card. Specify the FMU, where the interface variables are imported or exported.

Card 1	1	2	3	4	5	6	7	8
Variable	APPID							
Type	A20							
Default	none							

Variable card. Specify the interface variables to be imported and exported. Repeat this card if necessary. This input terminates with the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	IMPEXP	REGTYP	REGID	FIELD	VINIT	RATIO	CID	REF
Type	A	A	I	A	F	F	I	I
Default	none	none	none	none	0	1.0	0	0

VARIABLE	DESCRIPTION
APPID	FMU (functional mock-up unit) identification. Each FMU must have a unique APPID.
IMPEXP	Import/export flag: EQ.IMP: Variables are to be imported into LS-DYNA. EQ.EXP: Variables are to be exported from LS-DYNA.

VARIABLE	DESCRIPTION
REGTYP	Type of interface region: EQ.NODE: Single node EQ.NSET: Node set EQ.SSET: Segment set EQ.PART: Rigid part EQ.FUNC: User defined curve function to be exported only. See Remark 1 and Example 2. EQ.CURV: User defined curve to be imported only. If the curve is to be imported for *ICFD (e.g., boundary condition), its field must start with 'ICFD_'. See Remark 2 and Example 3. EQ.SESW: Sense switch to be imported only. REGID is neglected with the sense switch specified in FIELD. See Remark 3 and Example 4. EQ.BAG: Control-volume airbag.
REGID	ID of the corresponding region type in LS-DYNA. If region ID is negative for a node set and FIELD = FX, FY or FZ, all nodes within this set share the same imported force value from the FMU. Otherwise, a distinct force value will be imported for each node.
FIELD	Field data to be imported or exported, depending on the region type. See Remark 4 for a full list of fields.
VINIT	Initial value. For variables to be exported, VINIT is the initial value for the FMU. Therefore, its units must be the same as the FMU's units.
RATIO	Scale factor applied during co-simulation (not FMU generation). When LS-DYNA exports variables, the actual value to be sent is RATIO × LS-DYNA value. When LS-DYNA imports data from FMU, the actual value received is the FMU value / RATIO.
CID	Coordinate system ID (see *DEFINE_COORDINATE). EQ.0: Global (default)
REF	Control how the coordinate system is used for the variable output when CID > 0 (see Remark 5):

VARIABLE	DESCRIPTION
	EQ.0: Variable output is in the local system fixed from the beginning. Note that you should set FLAG = 0 in *DEFINE_COORDINATE_NODES.
	EQ.1: Variable output is projected onto the moving local system.
	EQ.2: Variable output is the projection of the nodal translation motion relative to node N1 of the local coordinate system, COOR. Double precision LS-DYNA is recommended with REF = 2.

Remarks:

1. **REGTYP = "FUNC".** You can export variables from *DEFINE_CURVE_FUNCTION with the corresponding curve function ID. Note that it is for export only. For example, if you want to export the value of SENSOR(*n*) through the FMU, you must do the following three steps (see Example 2):

   a) Set up sensors through *SENSOR.

   b) Include the sensor in *DEFINE_CURVE_FUNCTION.

   c) Add the curve function to *COSIM_FMI_INTERFACE.

2. **REGTYP = "CURV".** You can import general variables through *DEFINE_CURVE. During co-simulation, the curve value in LS-DYNA will be dynamically updated. Note that it is for import only. For instance, if you want to import velocity from the controller in MATLAB and impose it onto a node in LS-DYNA, you must do the following three steps (see Example 3):

   a) Define a curve with initial values of all zero and the time range greater than the termination time of the co-simulation

   b) Set up *BOUNDARY with the defined curve.

   c) Add the curve to *COSIM_FMI_INTERFACE.

3. **REGTYP = "SESW".** If you want to import a sense switch (see Sense Switch Controls in Getting Started) ranging from "SW1~SW4" and "SWA~SWD" to control the status of LS-DYNA, you must specify the switch name in FIELD. All other fields, namely, REGID, VINIT, RATIO, CID, and REF, are neglected. The rising edge of the imported signal (the transition from a low to high value) will trigger the corresponding sense switch in LS-DYNA. If multiple switches are defined, they cannot be triggered at the same time. See Example 4.

4. **FIELD.** See the table below for a detailed list of possible values for FIELD. Not that PART must be for a rigid part.

FIELD	Description	REGTYP	Export	Import	REF
CX, CY, CZ	Position	NODE, NSET, PART	✓		0, 1
DX, DY, DZ	Translational displacement in *X*, *Y*, and *Z*, respectively	NODE, NSET, PART	✓		0, 1, 2
VX, VY, VZ	Translational velocity in *X*, *Y*, and *Z*, respectively	NODE, NSET, PART	✓		0, 1, 2
ACCX, ACCY, ACCZ	Translational acceleration in *X*, *Y*, and *Z*, respectively	NODE, NSET, PART	✓		0, 1, 2
AX, AY, AZ	Angular displacement in *X*, *Y*, and *Z*, respectively	PART	✓		0, 1
TMPT	Temperature	NODE, NSET	✓		0, 1
WX, WY, WZ	Angular velocity in *X*, *Y*, and *Z*, respectively	PART	✓		0, 1
WDTX, WDTY, WDTZ	Angular acceleration in *X*, *Y*, and *Z*, respectively	PART	✓		0, 1
BAGP, BAGV, BAGT	Airbag pressure, volume and temperature	BAG	✓		N/A
FX, FY, FZ	Force in *X*, *Y*, and *Z*, respectively	NODE, NSET, PART		✓	0, 1
P	Pressure	SSET		✓	N/A
TX, TY, TZ	Torque	PART		✓	0, 1
Variable name in FMU	Defined curve function	FUNC	✓		N/A
Variable name in FMU	Defined curve	CURV		✓	N/A

FIELD	Description	REGTYP	Export	Import	REF
SW1-SW4, SWA-SWD	Sense switch	SESW		✓	N/A

5. **REF.** The following are general comments about REF:

   a) REF = 1 and 2 differ in that variables with REF = 2 are measured relative to the changing local system.

   b) You cannot set REF to a different value for the same coordinate. For instance, you cannot export DX for node 1 based on CID = 1 with REF = 1, and then export DY for node 2 based on CID = 1 with REF = 2. To avoid this, you can define a new coordinate with a different REF.

   c) If you also define *DATABASE_HISTORY_NODE_LOCAL, please make sure the REF of the corresponding coordinate system is consistent, meaning you cannot set REF = 1 for CID = 1 in *COSIM_FMI_INTERFACE and REF = 2 for the same CID in *DATABASE_HISTORY_NODE_LOCAL.

Examples:

The following examples only serve as an illustration of syntax.

Example 1

This example illustrates exporting the velocity in the *X*-direction for node 100, exporting the displacements in the *Y*-direction for all nodes within set 3, importing the forces along the *X*-direction for all nodes in set 3, importing the same force along the *Y*-direction for all nodes in set 4, and importing the pressure in segment set 1.

```
*COSIMULATION_FMI_INTERFACE
              MOTOR
       EXP      NODE       100        VX         0         1         0         0
       EXP      NSET         3        DY         0         1         0         0
       IMP      NSET         3        FX         0         1         0         0
       IMP      NSET        -4        FY         0         1         0         0
       IMP      SSET         1         P         0         1         0         0
```

Example 2

Export the value of sensor 1 through a defined curve function. The corresponding variable is named “fyball” in FMU during co-simulation.

```
*SENSOR_DEFINE_NODE
         1       681      1107Y                  0COORD
*DEFINE_CURVE_FUNCTION_TITLE
diff_Y
       100         0
SENSORD(1)
*COSIMULATION_FMI_INTERFACE
```

```
                ball
       EXP      FUNC       100    fyball          0          1          0          0
```

Example 3

Import the Y-direction velocity value through a defined curve which is then prescribed to a node in LS-DYNA during co-simulation.

```
*DEFINE_CURVE_TITLE
vy
       200          0       1.0       1.0       0.0       0.0          0          0
                  0.0                 0.0
                100.0                 0.0
*BOUNDARY_PRESCRIBED_MOTION_NODE_ID
         2ball_vy
       681          2          0       200       1.0         01.00000E28       0.0
*COSIMULATION_FMI_INTERFACE
                ball
       IMP      CURV       200    vyball          0          1          0          0
```

Example 4

Import a sense switch during co-simulation.

```
*COSIMULATION_FMI_INTERFACE
$#             appid
              switch
$#  impexp    regtyp     regid     field      init     ratio      coor       ref
       IMP      SESW                 SW1          0          1          0          0
```

*COSIM_SYC_CONTROL

Purpose: Enable coupling with other Ansys solvers using Ansys System Coupling™. This keyword should be included in the input deck along with *COSIM_SYC_INTERFACE. See Appendix Z for a discussion of this feature and for examples.

Card 1	1	2	3	4	5	6	7	8
Variable	SYCID		OPT					
Type	A20		A10					
Default	none		none					

VARIABLE	DESCRIPTION
SYCID	LS-DYNA's System Coupling identification. System Coupling uses this ID to identify and connect the participants.
OPT	Purpose of the LS-DYNA simulation: EQ.C: Co-simulation mode. LS-DYNA will co-simulate with other Ansys solvers.

*COSIM_SYC_INTERFACE

Purpose: Define the interface variables during a co-simulation using System Coupling. For instance, you may need LS-DYNA to export particular nodal displacements to another software and to import certain nodal forces.

Card Summary:

Card 1. This card is required.

SYCID	NREG					

Card 2. Include NREG sets of this card with Card 2.1. Each set includes one Card 2 and NIMP + NEXP instances of Card 2.1. Each set provides the information needed for one coupling region. See Example below.

REGNAME	REGTYP	REGID	NEXP	NIMP		

Card 2.1. Include NIMP + NEXP of this card following its parent Card 2.

VARNAME	VARTYP	SCTYP	SCEXT	SCLOC	SCVS	

Data Card Definitions:

System Coupling ID Card. Specify the System Coupling ID and the number of regions where the interface variables are imported or exported. Following this ID card, define NREG coupling region card sets that include the card specifying the region, followed by cards specifying the interface variables.

Card 1	1	2	3	4	5	6	7	8
Variable	SYCID		NREG					
Type	A20		I					
Default	LSDYNA		none					

VARIABLE	DESCRIPTION
SYCID	LS-DYNA's System Coupling identification. System Coupling uses this ID to identify and connect the participants.
NREG	Number of interface regions to be defined

Region Card. Define a coupling region and its associated interface variables to be exchanged by LS-DYNA and other coupling solvers. Following each region definition card, include NEXP + NIMP instances of Card 2.1 to specify interface variables for export and import.

Card 2	1	2	3	4	5	6	7	8
Variable	REGNAME		REGTYP	REGID	NEXP	NIMP		
Type	A		A	I	I	I		
Default	none		none	none	none	none		

Interface Variables Card. Specify the interface variables to be imported and exported. Repeat this card NEXP times for exported variables and then NIMP times for imported variables after each region definition card (Card 2).

Card 2.1	1	2	3	4	5	6	7	8
Variable	VARNAME		VARTYP	SCTYP	SCEXT	SCLOC	SCVS	
Type	A		A	A	A	A	A	
Default	none		none	UNSPEC	FALSE	NODE	none	

VARIABLE	DESCRIPTION
REGNAME	Name of this region
REGTYP	Type of interface region: EQ.SETSEG: Set of segments defined in *SET_SEGMENT_N8.
REGID	ID of the corresponding LS-DYNA region type
NEXP	Number of variables to be exported by LS-DYNA for this region
NIMP	Number of variables to be imported into LS-DYNA for this region
VARNAME	Name of the variable to be used in the .scp and .py files

VARIABLE	DESCRIPTION
VARTYP	Field data type, for LS-DYNA, to describe the characteristics of the variable. EQ.FORCE: Nodal force. EQ.INCD: Incremental nodal displacement from the last System Coupling time step. EQ.PRESS: Pressure for segments, import only EQ.VEL: Nodal velocity, export only The following data types are additionally available when co-simulating with an LS-DYNA structural-thermal analysis (meaning SOLN of *CONTROL_SOLUTION is set to 2): EQ.CONVH: Nodal heat transfer coefficient for convection, import only EQ.CONVTR: Nodal reference temperature for convection, import only EQ.HTRATE: Nodal heat flow rate, import only EQ.TEMP: Nodal temperature, export only Note that these thermal data types are available only in R18 and later.
SCTYP	Field data type defined at the System Coupling level; see the System Coupling Settings Reference for details. EQ.CONVH: For nodal heat transfer coefficient for convection, use when VARTYP = CONVH (available for R18 and later) EQ.CONVTR: For nodal reference temperature for convection, use when VARTYP = CONVTR (available for R18 and later) EQ.FORCE: For force or pressure EQ.HTRATE: For nodal heat flow rate, use when VARTYP = HTRATE (available for R18 and later) EQ.INCD: For incremental displacement (available for R18 and later) EQ.TEMP: For nodal temperature, use when VARTYP = TEMP (available for R18 and later) EQ.UNSPEC: Unspecified

VARIABLE	DESCRIPTION
SCEXT	Flag to specify if the variable to be transferred is extensive. This flag is only needed when SCTYP is set to UNSPEC. A variable is extensive if its value depends on the system size. EQ.FALSE: Variable is not extensive. EQ.TRUE: Variable is extensive.
SCLOC	Location of the variable being transferred. It can be set to NODE or ELEMENT. This field is only needed when SCTYP is set to UNSPEC.
SCVS	Tensor type of the variable. It can be set to SCALAR or VECTOR. This field is only needed when SCTYP is set to UNSPEC.

Example:

The following is an example of input in the case of multiple regions:

```
*COSIM_SYC_INTERFACE
$               sycid      nreg
              lsdyna         3
$ 1st region: bottom
$             regname    regtyp     regid      nexp      nimp
               FSIN_B    SETSEG       100         1         1
$             varname   dynatyp    syctyp
                 INCD      INCD      INCD
                 FORC     FORCE     FORCE
$ 2nd region: side
$             regname    regtyp     regid      nexp      nimp
               FSIN_S    SETSEG       200         1         1
$             varname   dynatyp    syctyp
                 INCD      INCD      INCD
                 FORC     FORCE     FORCE
$ 3rd region: bottom
$             regname    regtyp     regid      nexp      nimp
               FSIN_T    SETSEG       300         1         1
$             varname   dynatyp    syctyp
                 INCD      INCD      INCD
                 FORC     FORCE     FORCE
```

*DAMPING

The keyword options in this section in alphabetical order are:

*DAMPING_FREQUENCY_RANGE

*DAMPING_GLOBAL

*DAMPING_PART_MASS

*DAMPING_PART_STIFFNESS

*DAMPING_PART_STRUCTURAL

*DAMPING_RELATIVE

*DAMPING_STRUCTURAL

*DAMPING_FREQUENCY_RANGE_{OPTION1}_{OPTION2}

Purpose: Provide approximately constant damping (that is, frequency-independent) over a range of frequencies. It applies to explicit analysis and to time-integrated implicit dynamic analysis (IMASS = 1 on *CONTROL_IMPLICIT_DYNAMICS) but not to any of the implicit analysis types that use modal superposition or those that work in the frequency domain.

Available settings of *OPTION1* are:

<BLANK> Applies damping to global motion of nodes

DEFORM Applies damping to element stresses/forces (see Remarks 2 and 4)

OPTION2 is available only when *OPTION1* is DEFORM. Available settings for *OPTION2* are:

<BLANK> Applies to solid, beam, shell, thick shell and discrete elements

DMIG Applies damping to superelements (see *ELEMENT_DIRECT_MATRIX_INPUT)

Card 1	1	2	3	4	5	6	7	8
Variable	CDAMP	FLOW	FHIGH	PSID		PIDREL	IFLG	ICARD2
Type	F	F	F	I		I	I	I
Default	0.0	none	none	0		0	0	0

Include this card only if ICARD2 = 1 and *OPTION1* is DEFORM.

Card 2	1	2	3	4	5	6	7	8
Variable	CDAMPV	IPWP						
Type	F	I						
Default	CDAMP	1						

VARIABLE	DESCRIPTION
CDAMP	Damping in fraction of critical. Accurate application of this damping depends on the time step being small compared to the period of interest.
FLOW	Lowest frequency in the range of interest (cycles per unit time, such as Hz if the time unit is seconds). Must be greater than zero.
FHIGH	Highest frequency in the range of interest (cycles per unit time, such as Hz if time unit is seconds). Must be greater than FLOW.
PSID	Meaning depends on *OPTION1* and *OPTION2*. If *OPTION1* or *OPTION2* is blank, PSID is a part set ID (see *SET_PART). The requested damping is applied only to the parts in the set. If PSID = 0, the damping is applied to all parts except those referred to by other *DAMPING_FREQUENCY_RANGE cards. If *OPTION1* is DEFORM and *OPTION2* is DMIG, PSID is the ID of a superelement (see EID on *ELEMENT_DIRECT_MATRIX_INPUT). If PSID = 0, the damping is applied to all superelements.
PIDREL	Optional part ID of a rigid body. Damping is then applied to the motion relative to the rigid body motion. This input does not apply to the DEFORM keyword option. See Remark 6.
IFLG	Method used for internal calculation of damping constants: EQ.0: Iterative (more accurate). See Remark 5. EQ.1: Approximate (same as R9 and previous versions)
ICARD2	Flag to include Card 2 (only available if *OPTION1* is DEFORM): EQ.0: Do not include Card 2. EQ.1: Card 2 must be included.
CDAMPV	Damping in fraction of critical applied to pressure/volume response of solid elements. See Remark 7.
IPWP	Flag to determine whether the damping is applied to excess pore pressure in parts referenced by *BOUNDARY_PORE_FLUID (see Remark 8): EQ.0: Same as EQ.1 EQ.1: Excess pore pressure is subjected to damping ratio CDAMPV

VARIABLE	DESCRIPTION
	EQ.2: Damping is not applied to pore pressure, only to the effective stress.

Remarks:

1. **General.** This feature is for explicit analysis and implicit time-stepping dynamic analysis (IMASS = 1 on *CONTROL_IMPLICIT_DYNAMICS). It provides an approximately constant (i.e., frequency-independent) damping ratio over a range of frequencies given by $F_{low} < F < F_{high}$. It is intended for small damping ratios (< 0.05) and frequency ranges such that F_{high}/F_{low} is in the range of 10 to 300. In contrast, other types of damping are not frequency-independent: mass-weighted damping (see *DAMPING_GLOBAL and *DAMPING_PART_MASS) gives a damping ratio that is inversely proportional to frequency, while stiffness-proportional damping (see *DAMPING_PART_STIFFNESS) gives a damping ratio that is proportional to frequency.

   At frequencies outside of the range $F_{low} < F < F_{high}$, some damping is applied, but the damping ratio becomes progressively lower as the frequency reduces below F_{low} or increases above F_{high}.

2. **Choice of OPTION1.** When *OPTION1* is blank, this card applies damping based on the velocities of the nodes of the parts in PSID relative to the global coordinate system. Thus, rigid body motion attracts damping forces. When *OPTION1* is DEFORM, damping is applied based on the rate of change of element stresses or forces. Therefore, rigid body motion is not damped when using the DEFORM keyword option. Thus, *we recommend DEFORM over the standard option.* The dependence of the damping on the rate of change of stresses or forces means that the damping forces depend on the current tangent stiffness. This method is believed to be more appropriate for a nonlinear analysis, which could be overdamped if a strain-rate-proportional or viscous damping scheme were used.

3. **Dynamic stiffness.** A drawback of *DAMPING_FREQUENCY_RANGE is that it alters the dynamic stiffness of the damped parts, and hence it also alters the natural frequencies of the structure. Both settings of *OPTION1* do this but in different ways. When *OPTION1* is blank, the damping method reduces the dynamic stiffness, while for the DEFORM option, the method increases dynamic stiffness. In both cases, the change of dynamic stiffness is proportional to the damping ratio CDAMP and is greater when the ratio F_{high}/F_{low} is greater. The degree to which the natural frequency of a particular mode increases or decreases depends on where the frequency sits within the band F_{low} to F_{high} and on *OPTION1*. When *OPTION1* is DEFORM, the maximum increase occurs for

natural frequencies close to F_{high}, while the increase for natural frequencies close to F_{low} is typically less than a third of the maximum increase. When *OPTION1* is blank, the reverse situation occurs: maximum decrease of natural frequency occurs for frequencies close to F_{low}, with decreases less than one-third of the maximum occurring for frequencies close to F_{high}. The maximum increase or reduction can be estimated from the following table, which is applicable to both settings of *OPTION1*.

Table 15-1. Approximate error in natural frequency for a given damping ratio

	% error in frequency for F_{high}/F_{low} =		
Damping Ratio	3 to 30	30 to 300	300 to 3000
0.01	3%	4.5%	6%
0.02	6%	9%	12%
0.04	12%	18%	24%

Modifying the elastic stiffnesses to compensate can workaround the above issue to some extent. For example, for 1% damping across a frequency range of 30 to 600 Hz ($F_{high}/F_{low} = 20$), the maximum error is 3% while the average error across the frequency range is about 2%. For the DEFORM option, it would therefore be appropriate to reduce the elastic stiffness by $(1.02)^2$, that is, by 4%.

4. **Further information about the DEFORM option.** The DEFORM keyword option works with the following element formulations:

   - Solids – types -1, -2, 1, 2, 3, 4, 9, 10, 13, 15, 16, 17, 99
   - Beams – types 1, 2, 3, 4, 5, 9, 13 (note: not type 6)
   - Shells – types 1-5, 7-17, 20, 21, 23-27, 99
   - Thick shells – all types
   - Discrete elements (see *ELEMENT_DISCRETE)
   - Superelements (see *ELEMENT_DIRECT_MATRIX_INPUT) if *OPTION2* is DMIG

   Table 15-2 lists how the DEFORM option differs from the standard option.

Table 15-2. Comparison of standard damping to deformation damping.

	Keyword Option	
Characteristic Property	<BLANK>	DEFORM
Damping on	Node velocities	Element responses
Rigid body motion	Can be damped	Never damped

	Keyword Option	
Characteristic Property	<BLANK>	DEFORM
Natural frequencies	Reduced (maximum percentages shown in the above table)	Increased (maximum percentages shown in the above table)
Recommended compensation	Increase elastic stiffness	Reduce elastic stiffness
Effect on time step	None	Small reduction applied automatically, the same percentage as in the frequency change
Element types damped	All	See list above
Damping energy output	Included in "system damping energy"	Included in Internal Energy only if RYLEN = 2 on *CONTROL_ENERGY

5. **Iterative method**. Starting with version R10, the internal calculation of the damping constants uses an iterative method by default (see input variable IFLG). This iterative method is available both with and without the DEFORM keyword option. The iterative method results in the actual damping matching the user-input damping ratio CDAMP more closely across the frequency range FLOW to FHIGH. As an example, for CDAMP = 0.01, FLOW = 1 Hz, and FHIGH = 30 Hz, the actual damping achieved by the previous approximate method varied between 0.008 and 0.012 (different values at different frequencies), meaning there were errors of up to 20% of the target CDAMP. The iterative algorithm reduces the errors to 1% of the target CDAMP.

6. **PIDREL**. This is applicable only when *OPTION1* is blank. It is intended to reduce the extent to which rigid body motion is damped (but note that damping of rigid body motion can be avoided completely by using the DEFORM option). PIDREL is the ID of a part with *MAT_RIGID as the material. The part could represent, for example, the foundation or base of a structure. In this case, instead of resisting the global motion of the structure, the damping forces resist motion relative to the base and are reacted onto the rigid part PIDREL. "Relative motion" here means the difference between the velocity of the node being damped and the velocity of a point rigidly connected to PIDREL at the same coordinates as the node being damped.

7. **CDAMPV.** CDAMPV applies only with the DEFORM option and only affects solid elements; see *ELEMENT_SOLID. By default, the same damping ratio CDAMP is applied to all stress components of solid elements. If CDAMPV is

nonzero, it affects the volumetric response of solid elements, while CDAMP applies only to the deviatoric response.

8. **Damping of parts with pore pressure.** This remark and the input parameter IPWP are relevant only to models containing *CONTROL_PORE_FLUID. IPWP has no effect unless the damping card references a part containing pore pressure (meaning a part that *BOUNDARY_PORE_FLUID references). The remarks section of *CONTROL_PORE_FLUID explains the terminology used in this remark. In LS-DYNA versions through R14, *DAMPING_FREQUENCY_RANGE_DEFORM was ineffective when applied to solid elements in models containing pore pressure. Starting from R15, the default is for the damping ratio CDAMP to be applied to both the excess pore pressure and the effective stress. A nonzero CDAMPV affects the excess pore pressure and the effective pressure, while CDAMP applies to the deviatoric response. IPWP set to 2 switches off damping of the pore pressure, leaving only the effective stress subjected to damping.

*DAMPING_GLOBAL

Purpose: Define mass weighted nodal damping that applies globally to the nodes of deformable bodies and to the mass center of the rigid bodies. For specification of mass damping by part ID or part set ID, use *DAMPING_PART_MASS.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	VALDMP	STX	STY	STZ	SRX	SRY	SRZ
Type	I	F	F	F	F	F	F	F
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Remarks	1		2	2	2	2	2	2

VARIABLE	DESCRIPTION
LCID	Load curve ID (see *DEFINE_CURVE) which specifies the system damping constant vs. time: EQ.0: a constant damping factor as defined by VALDMP is used, GT.0: system damping is given by load curve LCID (which must be an integer). The damping force applied to each node is $f = -d(t)mv$, where $d(t)$ is defined by load curve LCID.
VALDMP	System damping constant, D_s (this option is bypassed if the load curve number defined above is non zero).
STX	Scale factor on global x translational damping forces.
STY	Scale factor on global y translational damping forces.
STZ	Scale factor on global z translational damping forces.
SRX	Scale factor on global x rotational damping moments.
SRY	Scale factor on global y rotational damping moments.
SRZ	Scale factor on global z rotational damping moments.

Remarks:

1. **Restart.** This keyword is also used for the restart, see *RESTART.

2. **Defaults for Scale Factors.** If STX = STY = STZ = SRX = SRY = SRZ = 0.0 in the input above, all six values are defaulted to unity.

3. **Damping Exceptions.** Mass damping will not be applied to deformable nodes with prescribed motion or to nodes tied with CONSTRAINED_NODE_SET.

4. **Formulation.** With mass proportional system damping the acceleration is computed as:

$$\mathbf{a}^n = \mathbf{M}^{-1}\left(\mathbf{P}^n - \mathbf{F}^n - \mathbf{F}^n_{\text{damp}}\right)$$

where, $\mathbf{M}$ is the diagonal mass matrix, $\mathbf{P^n}$ is the external load vector, $\mathbf{F}^n$ is the internal load vector, and $\mathbf{F}^n_{\text{damp}}$ is the force vector due to system damping. This latter vector is defined as:

$$\mathbf{F}^n_{\text{damp}} = D_s m \mathbf{v}$$

The best damping constant for the system is usually some value approaching the critical damping factor for the lowest frequency mode of interest.

$$(D_s)_{\text{critical}} = 2\omega_{\text{min}}$$

The natural frequency ω_{min} (given in radians per unit time) is generally taken as the fundamental frequency of the structure. This frequency can be determined from an eigenvalue analysis or from an undamped transient analysis. Note that this damping applies to both translational and rotational degrees of freedom. Also note that mass proportional damping will damp rigid body motion as well as vibration.

Energy dissipated by through mass weighted damping is reported as system damping energy in the ASCII file glstat. This energy is computed whenever system damping is active.

5. **Filtering.** When global damping is used, IACCOP = 1 is automatically set in *CONTROL_OUTPUT, causing nodal accelerations to be averaged in the nodout and d3thdt files.

***DAMPING_PART_MASS**_*{OPTION}*

OPTION specifies that a part set ID is given with the single option:

<BLANK>

SET

If not used, a part ID is assumed.

Purpose: Define mass-weighted damping by part ID. Parts may be either rigid or deformable. In rigid bodies, the damping forces and moments act at the center of mass. This command may appear multiple times in an input deck but cannot be combined with *DAMPING_GLOBAL.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	LCID	SF	FLAG	STRUCT			
Type	I	I	F	I	I			
Default	0	0	1.0	0	0			

Scale Factor Card. Additional Card for FLAG = 1.

Card 2	1	2	3	4	5	6	7	8
Variable	STX	STY	STZ	SRX	SRY	SRZ		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
PID/PSID	Part ID, see *PART or part set ID, see *SET_PART.
LCID	Load curve ID (see *DEFINE_CURVE) that specifies the damping constant as a function of time applied to the part(s) specified in PID/PSID.

VARIABLE	DESCRIPTION
SF	Scale factor for load curve. This allows a simple modification of the load curve values.
FLAG	Set this flag to unity if the global components of the damping forces require separate scale factors.
STRUCT	Flag for damping only the structural mass. This field only applies to deformable bodies: EQ.0: Damp all mass in the part or part set. EQ.1: Damp only structural mass and exclude mass added with *ELEMENT_MASS(_...) from damping.
STX	Scale factor on global x translational damping forces.
STY	Scale factor on global y translational damping forces.
STZ	Scale factor on global z translational damping forces.
SRX	Scale factor on global x rotational damping moments.
SRY	Scale factor on global y rotational damping moments.
SRZ	Scale factor on global z rotational damping moments.

Remarks:

Mass-weighted damping damps all motions, including rigid body motions. For high-frequency oscillatory motion, stiffness-weighted damping may be preferred. With mass-proportional system damping, the acceleration is computed as:

$$\boldsymbol{\alpha}^n = \mathbf{M}^{-1}(\mathbf{P}^n - \mathbf{F}^n - \mathbf{F}^n_{\text{damp}})$$

where, $\mathbf{M}$ is the diagonal mass matrix, $\mathbf{P}^n$ is the external load vector, $\mathbf{F}^n$ is the internal load vector, and $\mathbf{F}^n_{\text{damp}}$ is the force vector due to system damping. This latter vector is defined as:

$$\mathbf{F}^n_{\text{damp}} = D_s m \boldsymbol{\nu}$$

The critical damping constant for the lowest frequency mode of interest is

$$D_s = 2\omega_{\text{min}}$$

where ω_{min} is that lowest frequency in units of radians per unit time. The damping constant specified as the ordinate of curve LCID is typically less than the critical damping constant. The damping is applied to both translational and rotational degrees of freedom.

The component scale factors can be used to limit which global components see damping forces.

Energy dissipated by mass-weighted damping is reported as system damping energy in the ASCII file glstat. This energy is computed whenever system damping is active.

Mass damping will not be applied to deformable nodes with prescribed motion or to nodes tied with CONSTRAINED_NODE_SET.

*DAMPING_PART_STIFFNESS_{OPTION}

OPTION specifies that a part set ID is given with the single option:

<BLANK>

SET

If the SET option is not used, a part ID goes in the first field of Card 1.

Purpose: Assign stiffness damping coefficient by part ID or part set ID. This damping command does not apply to parts comprised of discrete elements (*ELEMENT_DISCRETE) or discrete beams (*ELEMENT_BEAM with ELFORM = 6).

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	COEF						
Type	I	F						
Default	none	0.0						

VARIABLE	DESCRIPTION
PID/PSID	Part ID (see *PART) or part set ID (see *SET_PART).
COEF	Rayleigh damping coefficient. Two methods are now available: LT.0.0: Rayleigh damping coefficient in units of time, set based on a given frequency and applied uniformly to each element in the specified part or part set. This method is typically used for implicit dynamic analysis. See remarks below. EQ.0.0: Inactive. GT.0.0: Unitless damping coefficient for stiffness weighted damping. This non-classical method is typically used for explicit analyses as it does not require assembly of a stiffness matrix. Values between 0.01 and 0.25 are recommended. Higher values are strongly discouraged, and values less than 0.01 may have little effect. The damping coefficient is uniquely calculated internally for each element of the part ID.

Remarks:

The damping matrix in classical Rayleigh damping is defined as:

$$\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$$

where **C**, **M**, and **K** are the damping, mass, and stiffness matrices, respectively. The constants α and β are the mass and stiffness proportional damping constants. The mass proportional damping can be treated by system damping (see keywords *DAMPING_GLOBAL and *DAMPING_PART_MASS). Transforming **C** with the i^{th} eigenvector $\boldsymbol{\phi}_i$ gives:

$$\boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}\boldsymbol{\phi}_i = \boldsymbol{\phi}_i^{\mathrm{T}}(\alpha\mathbf{M} + \beta\mathbf{K})\boldsymbol{\phi}_i = \alpha + \beta\omega_i^2 = 2\omega_i\xi_i\delta_{ij}$$

where ω_i is the i^{th} frequency (radians/unit time) and ξ_i is the corresponding modal damping parameter.

Generally, the stiffness proportional damping is effective for high frequencies and is orthogonal to rigid body motion. Mass proportional damping is more effective for low frequencies and damps rigid body motion. *If a large value of the stiffness-based damping coefficient is used, it may be necessary to lower the time step size significantly.* This must be done manually by reducing the time step scale factor on the *CONTROL_TIMESTEP control card. A good rule of thumb is to scale TSSFAC by $\left(\sqrt{\text{COEF}^2 + 1} - \text{COEF}\right)$; for example, TSSFAC = 0.856 (instead of 0.9) for COEF = 0.05.

The critical stiffness damping coefficient for mode i is equal to 2 divided by ω_i. For example, 10% of critical damping in the i^{th} mode corresponds to

$$\beta = \frac{0.20}{\omega_i}$$

If COEF < 0.0 is input, COEF is -β. For implicit dynamic analysis, classical Rayleigh stiffness damping is performed. For an explicit analysis, a negative COEF is generally impractical because solution stability requires that β be significantly smaller than the explicit time step.

If COEF > 0.0 is input, COEF represents, in only a very approximate sense, a fraction of critical damping in the high-frequency domain. This approach is *not* classical Rayleigh damping and is only recommended for explicit analysis.

The energy dissipated by Rayleigh damping is computed only if the RYLEN on *CONTROL_ENERGY is set to 2. This energy is accumulated as an element's internal energy and is included in the energy balance. In the glstat file this energy will be lumped in with the internal energy.

NOTE: Type 2 beam elements are a special case in which COEF is internally scaled by 0.1. Thus, there is a factor of 10 less damping than stated above. This scaling applies to both negative and positive values of COEF.

*DAMPING_PART_STRUCTURAL_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Assign a structural damping coefficient by part ID or part set ID for frequency domain analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	G						
Type	I	F						
Default	0	0.0						

VARIABLE	DESCRIPTION
PID/PSID	Part ID (see *PART) with no keyword option or part set ID (see *SET_PART) with the SET keyword option
G	Constant structural damping coefficient

Remarks:

1. **Damping force.** The structural damping force F_j (for part or par set j) is calculated as follows:

$$F_j = iG_jK_ju_j$$

where i is the imaginary unit, G_j is structural damping coefficient for part or part set j, K_j is stiffness matrix for part or part set j, and u_j is nodal displacements for part or part set j. The damping force F_j and the nodal displacements u_j are both complex variables.

*DAMPING_RELATIVE

Purpose: Apply damping relative to the motion of a rigid body. For example, it could damp the deformation of a rotating tire relative to the wheel without damping the rotating motion.

Card 1	1	2	3	4	5	6	7	8
Variable	CDAMP	FREQ	PIDRB	PSID	DV2	LCID		
Type	F	F	F	I	F	I		
Default	0	0	0	0	0.0	0		

VARIABLE	DESCRIPTION
CDAMP	Fraction of critical damping.
FREQ	Frequency at which CDAMP is to apply (cycles per unit time; for example, Hz if time unit is seconds).
PIDRB	Part ID of rigid body; see *PART. Motion relative to this rigid body will be damped.
PSID	Part set ID. The requested damping is applied only to the parts in the set.
DV2	Optional constant for velocity-squared term. See Remark 3.
LCID	ID of curve that defines fraction of critical damping as a function of time. CDAMP will be ignored if LCID is non-zero.

Remarks:

1. **Model Description.** This feature provides damping of vibrations for objects that are moving through space. The vibrations are damped, but not the rigid body motion. This is achieved by calculating the velocity of each node relative to that of a rigid body, and applying a damping force proportional to that velocity. The forces are reacted onto the rigid body such that overall momentum is conserved. It is intended that the rigid body is embedded within the moving object.

2. **Damping.** Vibrations at frequencies below FREQ are damped by more than CDAMP, while those at frequencies above FREQ are damped by less than CDAMP. It is recommended that FREQ be set to the frequency of the lowest mode of vibration.

3. **Damping Force.** The damping force of each node is calculated as follows:

$$F = -(Dmv) - \left(\text{DV2} \times mv^2\right),$$

where $D = 4\pi \times \text{CDAMP} \times \text{FREQ}$, m is the mass of the node, and v is the velocity of the node relative to the velocity of a point on the rigid body at the same coordinates as the node.

*DAMPING_STRUCTURAL

Purpose: Define structural damping coefficients to be used in frequency domain analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	G	LCID	LCTYP					
Type	F	I	I					
Default	0.0	0	0					

VARIABLE	DESCRIPTION
G	Constant structural damping coefficient.
LCID	Curve ID to define frequency dependent structural damping coefficients.
LCTYP	Type of load curve defining structural damping coefficients: EQ.0: abscissa value defines frequency. EQ.1: abscissa value defines mode number.

Remarks:

1. **Damping Force.** The structural damping force F is calculated as follows:

$$F = iGKu$$

where i is imaginary unit, G is structural damping coefficient, K is global stiffness matrix, and u is nodal displacements. The damping force F and the nodal displacements u are both complex variables.

*DATABASE

The database definitions are optional, but they are necessary to obtain output files containing results information. The ordering of the database definition cards in the input file is completely arbitrary. In this section the database keywords are defined in alphabetical order:

DATABASE_*OPTION1_{OPTION2}

*DATABASE_ACEOUT

*DATABASE_ADAMS

*DATABASE_ALE

*DATABASE_ALE_MAT

*DATABASE_ALE_OPERATION

DATABASE_BINARY_*OPTION1_{OPTION2}

*DATABASE_BINARY_D3PROP

*DATABASE_CPM_SENSOR

DATABASE_CROSS_SECTION_*OPTION1_{OPTION2}

*DATABASE_DEFRAGMENT

*DATABASE_D3FTG

*DATABASE_D3MAX

*DATABASE_ENVELOPE_BEAM

*DATABASE_EXTENT_AVS

*DATABASE_EXTENT_BINARY

*DATABASE_EXTENT_D3PART

*DATABASE_EXTENT_INTFOR

*DATABASE_EXTENT_MOVIE

*DATABASE_EXTENT_MPGS

*DATABASE_EXTENT_SSSTAT

*DATABASE_FATIGUE_RAINFLOW_MATRIX

*DATABASE_FATIGUE_STRESS_CYCLE

*DATABASE_FATXML

*DATABASE_FORMAT

DATABASE_FREQUENCY_ASCII_*OPTION

DATABASE_FREQUENCY_BINARY_*OPTION

*DATABASE_FSI

*DATABASE_FSI_SENSOR

DATABASE_HISTORY_*OPTION

*DATABASE_HISTORY_ACOUSTIC

*DATABASE_ISPHHTC

*DATABASE_MASSOUT

DATABASE_MAX_*OPTION

*DATABASE_NODAL_FORCE_GROUP

*DATABASE_PAP_OUTPUT

*DATABASE_PBLAST_SENSOR

*DATABASE_PROFILE

*DATABASE_PWP_FLOW

*DATABASE_PWP_OUTPUT

*DATABASE_RCFORC_MOMENT

*DATABASE_RECOVER_NODE

*DATABASE_RVE

*DATABASE_SALE

*DATABASE_SPRING_FORWARD

*DATABASE_SUPERPLASTIC_FORMING

*DATABASE_TRACER

*DATABASE_TRACER_ALE

*DATABASE_TRACER_GENERAL

*DATABASE_TRACER_GENERATE

***DATABASE**_OPTION1_{OPTION2}

OPTION1 specifies the type of database. LS-DYNA will *not* create an ASCII database unless the corresponding *DATABASE_*OPTION1* card is included in the input deck. *OPTION1* may be any of the items in the following list:

ABSTAT Airbag statistics. These statistics are output to abstat for CV airbags, abstat_cpg for CPG airbags, and abstat_cpm for CPM airbags. See *AIRBAG_*OPTION*, *AIRBAG_CPG, and *AIRBAG_PARTICLE.

ATDOUT Automatic tiebreak damage statistics for *CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE_TIEBREAK with OPTION set to 7, 9, 10, or 11 (only SMP at the moment).

AVSFLT AVS database. See *DATABASE_EXTENT_AVS.

BEARING *ELEMENT_BEARING force output.

BNDOUT Boundary condition forces and energy.

CABLE Cross-sectional output from *DEFINE_CABLE

CURVOUT Output from *DEFINE_CURVE_FUNCTION.

DCFAIL Failure function data for *MAT_SPOTWELD_DAIMLERCHRYSLER

DEBOND Mode of bond breakage between discrete element spheres. See *DATABASE_HISTORY_DES_*OPTION*.

DEFGEO Deformed geometry file (Note that to output this file in Chrysler format insert the following line in your .cshrc file: "setenv LSTC_DEFGEO chrysler"). The nasbdf file (NASTRAN Bulk Data) is created whenever the DEFGEO file is requested.

DEFORC Discrete spring and discrete damper (*ELEMENT_DISCRETE) data. If the user wishes to be selective about which discrete elements are output in deforc, use *DATABASE_HISTORY_DISCRETE_*OPTION* to select elements for output (but only if BEAM = 0 in *DATABASE_BINARY_D3PLOT) or set PF = 1 in *ELEMENT_DISCRETE to turn off output for particular elements; otherwise all discrete elements are output.

DEMASSFLOW Mass flow rate across a plane as specified by *DEFINE_DE_MASSFLOW_PLANE.

DESTAT Frequency distribution and mean value of stress energies ratio for DES, including translational normal, translational shear, translational rolling, rotational normal, rotational shear, and rotational rolling energy. The stress energies of both DES-DES contact and DES-Structure contact (if *DEFINE_DE_TO_SURFACE_COUPLING is used) are included. The distribution is based on translational normal stress energy.

DISBOUT Discrete beam element, type 6, relative displacements, rotations, and force resultants, all in the local coordinate system, which is also output. Use with *DATABASE_HISTORY_BEAM.

ELOUT Element data. See *DATABASE_HISTORY_OPTION. Also, see the fields INTOUT and NODOUT on Card 3 of the *DATABASE_EXTENT_BINARY keyword. This latter option will output all integration point data or extrapolated data to the connectivity nodes in a file call eloutdet.

GCEOUT Geometric contact entity forces. See *CONTACT_ENTITY.

GLSTAT Global statistics and energies (recommended). See *CONTROL_ENERGY.

GLSTAT_DR Global statistics and energies output during the dynamic relaxation phase. See *CONTROL_ENERGY.

H3OUT Joint forces and moments for Hybrid III rigid body dummies. See *COMPONENT_HYBRIDIII.

ICVOUT Incompressible control volume output file. See *DEFINE_CONTROL_VOLUME.

JNTFORC Joint forces. See *CONSTRAINED_JOINT_*OPTION*.

MATSUM Part energies. See Remarks 1 and 2 below.

MATSUM_DR Part energies output during the dynamic relaxation phase. See Remarks 1 and 2 below.

MOVIE See *DATABASE_EXTENT_MOVIE.

MPGS See *DATABASE_EXTENT_MPGS.

NCFORC Nodal contact forces. See *CONTACT, Card 1 (SAPR and SBPR).

NODFOR Nodal force groups. See *DATABASE_NODAL_FORCE_GROUP.

NODOUT Nodal translational motion and, where applicable, nodal rotational motion. See *DATABASE_HISTORY_NODE_*OPTION*.

PBSTAT Particle blast data. This data is written only to the binout binary database, under the branch name abstat_pbm. See *DEFINE_PARTICLE_BLAST.

PLLYOUT Pulley element data for *ELEMENT_BEAM_PULLEY.

PRTUBE Pressure tube data for *DEFINE_PRESSURE_TUBE.

PYRO Pyro actuator data for *LOAD_PYRO_ACTUATOR.

RBDOUT Motion of rigid bodies in global and local coordinate systems. See Remark 2.

RCFORC Resultant contact interface forces. To output in a local coordinate system, see *CONTACT, Optional Card C.

RCFORC_DR Resultant contact interface forces output during the dynamic relaxation phase. To output in a local coordinate system, see *CONTACT, Optional Card C.

RWFORC Rigidwall forces. See *RIGIDWALL_*OPTION*.

SBTOUT Seatbelt output file. See *ELEMENT_SEATBELT_*OPTION*. See Remark 1 of *SENSOR_DEFINE_MISC.

SECFORC Cross-sectional forces. See *DATABASE_CROSS_SECTION_*OPTION*.

SLEOUT Contact interface energies. See *CONTACT_*OPTION*.

SLEOUT_DR Contact interface energies output during the dynamic relaxation phase. See *CONTACT_*OPTION*.

SNSROUT Sensed value or status of general sensors. See *SENSOR_*OPTION*.

SPCFORC SPC reaction forces. See *BOUNDARY_SPC_*OPTION*.

SPGCPL SPG to surface coupling forces. The ASCII data file is named spgcplforc. See *DEFINE_SPG_TO_SURFACE_COUPLING.

SPHMASSFLOW Mass flow across a plane as specified by *DEFINE_SPH_MASSFLOW_PLANE. The ASCII data file is named sphflow.

SPHOUT SPH data. See *DATABASE_HISTORY_*OPTION*.

SPHVICINITY SPH vicinity sensor. See *DEFINE_SPH_VICINITY_SENSOR.

SSSTAT Subsystem data. See *DATABASE_EXTENT_SSSTAT.

SWFORC Spot weld forces (spot welds and rivets).

TPRINT Thermal output from a coupled structural/thermal or thermal-only analysis. Includes all nodes unless *DATABASE_HISTORY_NODE_*OPTION* is also provided in the keyword input.

TRHIST Tracer particle history information. See *DATABASE_TRACER.

Available options for *OPTION2* are:

<BLANK>

FILTER

MASS_PROPERTIES

FILTER can only be used when *OPTION1* is NCFORC or SECFORC. With the FILTER option, this keyword requires an additional data card; see Card 2 below. See Remark 12.

To include global and subsystem mass and inertial properties in the glstat and ssstat files, add the keyword option MASS_PROPERTIES, as shown below. Activating this option leads to the output of the current mass and inertia properties, including the principle inertia and their axes. The calculated properties do not include the mass of deleted nodes and rigid bodies.

GLSTAT_MASS_PROPERTIES This is an option for the glstat file to include mass and inertial properties.

SSSTAT_MASS_PROPERTIES This is an option for the ssstat file to include mass and inertial properties for the subsystems.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY	LCUR	IOOPT	OPTION1	OPTION2	OPTION3	OPTION4
Type	F	I	I	I	F/I	I	I	I
Default	0.	1 or 2	none	0.	0	0	0	0

VARIABLE **DESCRIPTION**

DT Time interval between outputs. If DT is zero, no output is printed. If DT < 0.0, the result will be output every -DT time steps.

BINARY Flag for binary output. See remarks under "Output Files and Post-Processing" in Appendix O, "LS-DYNA MPP User Guide."

EQ.1: ASCII file is written. This is the default for shared memory parallel (SMP) LS-DYNA executables.

EQ.2: Data written to a binary database binout, which contains data that would otherwise be output to the ASCII file. The ASCII file in this case is not created. This is the default for MPP LS-DYNA executables.

EQ.3: ASCII file is written, and the data is also written to the binary database (NOTE: MPP LS-DYNA executables will only produce the binary database).

LCUR Optional curve ID specifying time interval between outputs. Use *DEFINE_CURVE to define the curve. The abscissa is time, and the ordinate is the time interval between dumps.

IOOPT Flag to govern behavior of the output frequency load curve defined

VARIABLE	DESCRIPTION

by LCUR:

EQ.1: When output is generated at time t_n, the next output time t_{n+1} is computed as

$$t_{n+1} = t_n + \mathrm{LCUR}(t_n) \ .$$

This is the default behavior.

EQ.2: When output is generated at time t_n, the next output time t_{n+1} is computed as

$$t_{n+1} = t_n + \mathrm{LCUR}(t_{n+1}) \ .$$

EQ.3: Output is generated for each abscissa point in the load curve definition. The actual value of the load curve is ignored.

OPTION1

OPTION1 applies to the bndout, nodout, and elout files. For the nodout file OPTION1 is a *real* variable that defines the time interval between outputs for the high frequency file, nodouthf. If OPTION1 is zero, no output is printed. Nodal points that are to be output at a higher frequency are flagged using HFO in the input for *DATABASE_HISTORY_NODE_LOCAL.

For the elout file OPTION1 is an *integer* variable that gives the number of additional history variables written into the elout file for each integration point in the solid elements. See Remark 7 below for the elout file.

See Remark 9 for the bndout file.

OPTION2

OPTION2 applies to the bndout, nodouthf and elout files. For the nodouthf OPTION2 defines the binary file flag for the high frequency nodouthf file. See the field BINARY above.

For the elout file OPTION2 is an *integer* variable that gives the number of additional history variables written into the elout file for each integration point in the shell elements. See Remark 7 below for the elout file.

See Remark 9 for the bndout file.

OPTION3

OPTION3 applies to the bndout and elout files only. For the elout file OPTION3 is an *integer* variable that gives the number of additional history variables written into the elout file for each integration point in the thick shell elements. See Remark 7 below for the elout file and Remark 9 for the bndout file.

VARIABLE	DESCRIPTION
OPTION4	OPTION4 applies to the bndout and elout files only. For the elout file OPTION4 is an *integer* variable that gives the number of additional history variables written into the elout file for each integration point in the beam elements. See Remark 7 below for the elout file and Remark 9 for the bndout file.

The following Card 2 applies only to *DATABASE_NCFORC_FILTER and *DATABASE_SECFORC_FILTER.

Card 2	1	2	3	4	5	6	7	8
Variable	RATE	CUTOFF	WINDOW	TYPE				
Type	F	F	F	I				
Default	none	none	none	0	0	0	0	0

VARIABLE	DESCRIPTION
RATE	Time interval *T* between filter sampling.
CUTOFF	Frequency cut-off *C* in Hz.
WINDOW	The width of the window *W* in units of time for storing the single, forward filtering required for the TYPE = 2 filter option. Increasing the width of the window will increase the memory required for the analysis. A window that is too narrow will reduce the amplitude of the filtered result significantly, and values below 15 are not recommended for that reason. In general, the results for the TYPE = 2 option are sensitive to the width of the window and experimentation is required.
TYPE	Flag for filtering options: EQ.0: No filtering (default) EQ.1: Single pass, forward Butterworth filtering EQ.2: Two-pass filtering over the specified time window. Backward Butterworth filtering is applied to the forward Butterworth results that have been stored. This option improves the phase accuracy significantly at the expense of memory.

The file names and corresponding unit numbers are:

Description	I/O Unit #	File Name
Airbag statistics	43	abstat
Automatic tiebreak damage	92	atdout
ASCII database	44	avsflt
Boundary conditions	46	bndout (nodal forces and energies)
Cable output	422	cableout
Discrete element sphere bonds	498	debond
Smug animator database	40	defgeo
Discrete elements	36	deforc
Discrete elements mass flow	219	demflow
Discrete beam elements	215	disbout
Element data	34	elout
Contact entities	48	gceout
Global data	35	glstat
Global data	35	glstat_dr (for dynamic relaxation)
Joint forces	53	jntforc
Material energies	37	matsum
Material energies	37	matsum_dr (for dynamic relaxation)
MOVIE file family	50	movie*nnn*.xxx where.*nnn*=001-999
MPGS file family	50	mpgs*nnn*.xxx where *nnn* = 001-999
Nastran/BDF file	49	nasbdf (see comment below)
Nodal interface forces	38	ncforc
Nodal force group	45	nodfor
Nodal point data	33	nodout
Pulley element data	216	pllyout
Pressure tube data	421	prtube
Pyro actuator data	442	pyro
Rigid body data	47	rbdout
Resultant interface forces	39	rcforc
Resultant interface forces	39	rcforc_dr (for dynamic relaxation)

Description	I/O Unit #	File Name
Rigidwall forces	32	rwforc
Seat belts	52	sbtout
Cross-section forces	31	secforc
Interface energies	51	sleout
Interface energies	51	sleout_dr (for dynamic relaxation)
SPC reaction forces	41	spcforc
SPG surface coupling forces	124	spgcplforc
SPH mass flow data	224	sphflow
SPH element data	68	sphout
Subsystems statistics	58	ssstat
Nodal constraint resultants	42	swforc (spot welds/rivets)
Thermal output	73	tprint
Tracer particles	70	trhist

Output Components for ASCII Files.

ABSTAT
volume
pressure
internal energy
input mass flow rate
output mass flow rate (total)
mass
temperature
density
surface area of airbag
reaction
output mass flow rate (porosity)
output mass flow rate (venting)

ABSTAT_PBM
internal energy
translational energy
part pressure
part area
part x, y, z force

ABSTAT_CPG	
Airbag results	**Part results**
surface area	surface area
average density	part forces (x, y, z)
average specific energy	heat convection
average kinetic energy	energy leakage
average pressure	heat convection rate
average temperature	energy leakage rate
volume	mass leakage
atmospheric work	mass flow rate
useful work	area-weighted pressure average
number of particles	area-weighted temperature average
inflator mass	area-weighted total temperature average†
inflator energy	volumetric flow rate
inflator mass flow rate	mass flow rate per gas species
inflator energy rate	

† Total temperature is the temperature a flowing fluid would reach if it stopped moving adiabatically.

ABSTAT_CPM		
Airbag/chamber results	**Part results**	**Species results**
damping energy	total area	mass flow rate
translational kinetic energy	unblocked area	
density	enhanced venting††	
mass flow rate in	heat convection	
mass flow rate out	leaked energy	
gas temperature	part temperature	
inflator internal energy	porous mass leakage	
internal energy	porous volume leakage	
particle pressure†	pressure+†††	
pressure†	pressure-†††	
surface area	smoothed part pressure	
total mass	vent leakage	
volume		

† Pressure is the thermodynamic pressure from the gas internal energy in the airbag. Particle pressure is the pressure from the total particle translational kinetic energy.

†† Enhanced venting is a flag with a value of 0 for off and 1 for on. It is active if the vent area is open and the vent is created using many elements. The normal area of the vent also has to be more than 80% of the total area.

††† Pressure+ is the absolute pressure on the side of the shell whose normal vector is in the direction of the shell normal vector. Pressure- is the absolute pressure on the other side of the shell.

BNDOUT
x, y, z force
x, y, z, moment
energies

CABLEOUT
cross-sectional area
contact force
stress

DCFAIL
failure function
normal term
bending term
shear term
weld area
effective strain rate
axial force
shear force
torsional moment
bending moment

DEBOND
normal break
shear break
total break

DEFORC
x, y, z force / moment
relative rotation / displacement

ELOUT		
Beams	**(t)Shells**	**Solids**
axial force resultant	xx, yy, zz stress	xx, yy, zz stress
s shear resultant	xy, yz, zx stress	xy, yz, zx stress
t shear resultant	plastic strain	effective stress
s moment resultant	xx, yy, zz strain†	yield function††
t moment resultant	xy, yz, zx strain†	xx, yy, zz strain†
torsional resultant		xy, yz, zx strain†
for ELFORM = 1,7,8,9,11,14		
sig11, sig12, sig31		
plastic strain		

† Strains written for solids and for lower and upper integration points of shells and tshells if STRFLG = 1 in *DATABASE_EXTENT_BINARY.

†† The "yield function" is whatever is written as the 7th history variable. For example, this would be effective plastic strain for *MAT_024.

GCEOUT	
x, y, z force	x, y, z moment

GLSTAT/GLSTAT_DR	
time step	total energy
kinetic energy	external work
internal energy	total and initial energy
sprint and damper energy	energy ratio without eroded energy
hourglass energy	element & part ID controlling time step
system damping energy	global x, y, z velocity
sliding interface energy	time per zone cycle
eroded kinetic energy	joint internal energy
eroded internal energy	stonewall energy
eroded hourglass energy	rigid body stopper energy
added mass	percentage [mass] increase
drilling energy	dissipation energy†

† For implicit, integration errors due to large time steps will result in inaccurate estimates of kinetic and internal energies, regardless of how tight the convergence tolerances are. The "lost" energy arising from this discretization error is accumulated in the dissipated kinetic and internal energies, respectively, to render energy balance in the sense described in the introduction. Energy balance in implicit is an implication of having solved the implicit problem to sufficient degree of accuracy, and even though the opposite may not be true it can still be used as an indicator; energy balance *likely* implies a good solution while poor energy balance *definitely* implies a less accurate one.

JNTFORC	
x, y, z force	x, y, z moment

MATSUM/MATSUM_DR	
kinetic energy	x, y, z rigid body velocity
internal energy	eroded internal energy
hourglass energy	eroded kinetic energy
x, y, z momentum	added mass

NCFORC
x force
y force
z force

NODOUT
x, y, z displacement
x, y, z velocity
x, y, z acceleration
x, y, z rotation
x, y, z rotational velocity
x, y, z rotation acceleration

NODFOR
x, y, z force
x, y, z moment

PRTUBE
cross-sectional area
pressure
velocity
density

PYRO
pressure
volume
massflow
mass
temperature
energy
density

PLLYOUT
adjacent beam IDs
slip
slip rate
resultant force
wrap angle

RBDOUT
x, y, z displacement
x, y, z velocity
x, y, z acceleration

RCFORC/RCFORC_DR
x, y, z force
mass of nodes in contact

RWFORC
normal
x, y, z force

SECFORC
x, y, z force
x, y, z moment
x, y, z center
area
resultant force

SLEOUT/SLEOUT_DR
surfa energy
surfb energy
frictional energy

SPCFORC
x, y, z force
x, y, z moment

SWFORC
axial force
shear force
failure function
weld length
resultant moment
torsion

SPHFLOW
mass flow†
mass flow +†
mass flow - †

† Mass flow + is the cumulative mass flow across the plane in the direciton of the surface normal vectors, while mass flow - is the cumulative mass flow across the plane in the direction opposite to the surface normal vectors. Mass flow is the net cumulative mass flow, that is, (mass flow +) – (mass flow -).

SPHOUT	
xx, yy, zz stress	plastic strain
xy, yz, zx stress	particle active state
density	effective stress
number of neighbors	temperature
xx, yy, zz strain	xx, yy, zz strain rate
xy, yz, zx strain	xy, yz, zx strain rate
half of smoothing length	SPH to SPH coupling forces

Remarks:

1. **Discrepancies between "matsum" and "glstat" output.** The kinetic energy quantities in the matsum and glstat files may differ slightly in values for several reasons. First, the energy associated with added mass (from mass scaling) is included in the glstat calculation but is not included in matsum. Secondly, the energies are computed element by element in matsum for the deformable materials and, consequently, nodes that are merged with rigid bodies will also have their kinetic energy included in the rigid body total. Furthermore, kinetic energy is computed from nodal velocities in glstat and from element midpoint velocities in matsum.

2. **PRINT keyword option on *PART.** The PRINT option in the part definition allows some control over the extent of the data that is written into the matsum and rbdout files. If the print option is used, the variable PRBF can be defined such that the following numbers take on the meanings:

   EQ.0: Default is taken from the keyword *CONTROL_OUTPUT.

   EQ.1: Write data into rbdout file only.

EQ.2: Write data into matsum file only.

EQ.3: Do not write data into rbdout and matsum.

Also, see *CONTROL_OUTPUT and *PART_PRINT.

3. **The restart feature.** This keyword is also used in the restart phase; see Restart Input Data. Thus, the output interval can be changed when restarting.

4. **LS-PrePost.** All information in the files except in AVSFLT, MOVIE, and MPGS can also be plotted using LS-PrePost. Arbitrary cross-plotting of results between ASCII files is easily handled.

5. **The "rcforc" file.** Resultant contact forces reported in rcforc are averaged over the preceding output interval.

6. **Spring and damper energy.** "Spring and damper energy" reported in glstat is a subset of "internal energy." The "spring and damper energy" includes the internal energy of discrete elements, seatbelt elements, and that associated with joint stiffness (see *CONSTRAINED_JOINT_STIFFNESS_…).

7. **OPTION*n* field for "elout."** OPTION1, OPTION2, OPTION3, and OPTION4 give the number of additional history variables output for the integrated solids, shells, thick shells, and beams, respectively. Within this special option, each integration point is printed with its corresponding history data. No integration points are averaged. This is different than the default output where the stress data within a shell ply of a fully integrated shell, for example, are averaged and then written as output. The primary purpose of this database extension is to allow the actual integration point stress data and history variable data to be checked. There are no transformations applied to either the output stresses or history data; for example, EOCS in *CONTROL_OUTPUT and CMPFLG in *DATABASE_EXTENT_BINARY do not apply to elout when OPTION2 is nonzero.

8. **The failure function.** The failure function reported to the DCFAIL database is set to zero when the weld fails. If damage is active, then it is set to the negative of the damage scale factor which goes from 1 to 0 as damage grows.

9. **OPTION*n* field for "bndout."** For the bndout file, OPTION1 controls the nodal force group output, OPTION2 controls the concentrated force output, OPTION3 controls the pressure boundary condition output, and OPTION4 controls the velocity/displacement/acceleration nodal boundary conditions. If the value is 0 or left blank, the category is included (the default), and if it is 1, the category is not included in the bndout file.

10. **Contents of "glstat."** The glstat table above includes all items that *may* appear in the glstat data. The items that are actually written depend on the contents of

the input deck. For example, hourglass energy appears only if HGEN = 2 in *CONTROL_ENERGY, and added mass only appears if DT2MS < 0 in *CONTROL_TIMESTEP.

11. **Element ID controlling the time step.** The element ID controlling the time step is included in the glstat data but is not read by LS-PrePost. If the element ID is of interest to the user, the ASCII version of the glstat file can be opened with a text editor.

12. **The FILTER option.** The FILTER option uses a single-pass or double-pass Butterworth filter. For the single-pass filter, the data is forward filtered, while for the double-pass filter, forward filtered results undergo an additional backward filter. The forward filtered output $Y(n)$ at sampling interval n is obtained from the solution value $X(n)$ using the formula

$$Y(n) = a_0 X(n) + a_1 X(n-1) + a_2 X(n-2) + b_1 Y(n-1) + b_2 Y(n-2) ,$$

where the coefficients are

$$\begin{aligned}
\omega_d &= 2\pi \left(\frac{C}{0.6}\right) 1.25 \\
\omega_a &= \tan(\omega_a T/2) \\
a_0 &= \omega_a^2/(1+\sqrt{2}\omega_a+\omega_a^2) \\
a_1 &= 2a_0 \\
a_2 &= a_0 \\
b_1 &= 2\left(1-\omega_a^2\right)/\left(1+\sqrt{2}\omega_a+\omega_a^2\right) \\
b_2 &= \left(-1+\sqrt{2}\omega_a-\omega_a^2\right)/\left(1+\sqrt{2}\omega_a+\omega_a^2\right)
\end{aligned}$$

The two previous solution values and filtered values at $n-1$ and $n-2$ are stored.

Backward filtering improves the phase response of the filtered output. It is performed according to the formula

$$Z(n) = a_0 Y(n) + a_1 Y(n+1) + a_2 Y(n+2) + b_1 Z(n+1) + b_2 Z(n+2) ,$$

where $Z(n)$ is the backward filtered value at sample time n. This implies that all the forward filtered values $Y(n)$ are stored during the analysis which requires a prohibitive amount of memory. To limit the amount of memory required, the forward filtered values are stored for the time interval W, where the number of stored states is W/T, and the backward filtering is applied starting at the last saved value of the forward filtered values. As the window width increases, the filtered values approach the values that would be obtained from storing all of the forward filtered values.

The results of the backward filtering are sensitive to the window width, and experimentation with the width is necessary to obtain good results with the minimum window width. A window width of at least 10 to 15 times the sample rate

T should be used as a starting point. Some applications may require a window width that is much larger. The required window width decreases as the cut-off frequency increases. Or, to put it another way, the window width must be increased to make the filtered output smoother.

As an example, a random series of numbers between 0 and 1 was generated and filtered at intervals of 0.1 milliseconds with cut-off frequencies from 60 Hz to 420 Hz. The reverse filtering was applied with various window widths to determine how many forward-filtered states must be saved to achieve fixed levels of accuracy compared to complete reverse filtering from the last state to the first state. The results are shown in the table below. Note that the error is calculated *only* for the first state and the numbers being filtered are random. *This example should only be used as a very rough guide that indicates the overall trends and not as a recommendation for specific problems.*

Cut-off Frequency	No. of States 50% Error	No. of States 25% Error	No. of States 10% Error	No. of States 5% Error	No. of States 1% Error
60 Hz	26	33	55	68	87
120 Hz	13	16	30	37	44
180 Hz	8	10	22	26	30
240 Hz	6	8	17	19	23
300 Hz	5	6	12	15	18
360 Hz	5	6	10	12	16
420 Hz	4	5	9	10	15

*DATABASE_ACEOUT

Purpose: Time interval for nodal output of acoustic solution results of SSD and spectral analyses invoked with *CONTROL_IMPLICIT_SSD_DIRECT and *CONTROL_ACOUSTIC_SPECTRAL.

Card 1	1	2	3	4	5	6	7	8
Variable	DT							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
DTOUT	Time interval between the outputs

*DATABASE_ADAMS

Purpose: Request output of an MDI Modal Neutral File, d3mnf, for later use in Hexagon's Adams software. It is available for MPP only.

Card 1	1	2	3	4	5	6	7	8
Variable	IFLAG	M_UNITS	L_UNITS	T_UNITS				
Type	I	I	I	I				
Default	0	none	none	none				

VARIABLE	DESCRIPTION
IFLAG	Flag controlling write of d3mnf after eigenvalue analysis: EQ.0: Do not write (default). EQ.1: Write to d3mnf.
M_UNITS	Mass units of measure used in the model: EQ.-1: Kilogram EQ.-2: Gram EQ.-3: Megagram (metric ton) EQ.-4: lbf × s2/in (psi-compatible) EQ.-5: Slug EQ.-6: Pound-mass
L_UNITS	Length units of measure used in the model: EQ.-1: Meter EQ.-2: Centimeter EQ.-3: Millimeter EQ.-4: Inch EQ.-5: Foot
T_UNITS	Time units of measure used in the model: EQ.-1: Second

VARIABLE	DESCRIPTION
	EQ.-2: Millisecond
	EQ.-3: Minute
	EQ.-4: Hour

Remarks:

1. **Units.** Models must be created using a combination of the above units.

2. **Restrict to node set.** To restrict d3mnf to a node set, set IPARM3 on *CONTROL_IMPLICIT_EIGENVALUE. This ability is available for EIGMTH = 2, 3, 5, 6, 102, and 103.

*DATABASE_ALE

Purpose: For each ALE group (or material), this card controls the output for element time-history variables (in a tabular format that can be plotted in LS-PrePost by using the XYPlot button). It can also control d3plot output for ALE elements (see Remark 3).

Card 1	1	2	3	4	5	6	7	8
Variable	DTOUT	SETID	IAVER					
Type	F	I	I					
Default	none	none	0					

Variable Cards. Optional cards that can be used to add more variables to the database (the volume fractions are always output). Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
DTOUT	Time interval between the outputs (see Remark 3)
SETID	ALE element set ID (see Remark 3). If the model is 1D (*SECTION_ALE1D), the set should be *SET_-BEAM. If the model is 2D (*SECTION_ALE2D), the set should be *SET_-SHELL. If the model is 3D (*SECTION_SOLID), the set should be *SET_-SOLID.
IAVER	Average flag: EQ.0: Do not output average plot files EQ.1: Average the variable values selected by VAR*i* over all the

VARIABLE	DESCRIPTION
	elements in SETID and output the results in files with names ending with *average*.xy
VAR*i*	Variable to be output:
	LT.0: |VAR*i*| refers to the auxiliary variable to be replaced in d3plot (see Remark 3)
	EQ.1: *xx*-stress
	EQ.2: *yy*-stress
	EQ.3: *zz*-stress
	EQ.4: *xy*-stress
	EQ.5: *yz*-stress
	EQ.6: *zx*-stress
	EQ.7: plastic strain
	EQ.8: internal energy
	EQ.9: bulk viscosity
	EQ.10: previous volume
	EQ.11: pressure
	EQ.12: mass
	EQ.13: volume
	EQ.14: density
	EQ.15: kinetic energy
	EQ.16: internal energy density
	EQ.17: impulse (pressure integrated over time)
	If there is a blank column between 2 VAR*i* values, the list between these 2 values is selected. For example, if the card is input as follows:
	`1, ,6`
	then the 6 stresses are added to the database.

Remarks:

1. **Creating .xy Files.** The .xy files are created when the termination time is reached or if one of the following switches (after pressing the keys Ctrl - C) stops the job: sw1, stop, quit. During the run, they can be created with the switch sw2.

2. **Curve output.** The .xy files are created by element. There is a curve for each ALE group (or material). A curve can be added for volume-averaged variables.

3. **Modification of auxiliary variables in d3plot.** If VAR1 and VAR2 are input with VAR1 > 0 and VAR2 < 0, |VAR2| is a location in the auxiliary array (array listing for each element and ALE group: 6 stresses, plastic strain, internal energy, bulk viscosity, previous volume, history variable #1, history variable #2,…). Before outputting data to d3plot, the variable at location |VAR2| is replaced by the variable at location VAR1 in the variable list. For instance, if VAR1 = 10 and VAR2 = -1, then the previous volume will be output in the *xx*-stress spot in d3plot. More than one variable can be output using this method by setting additional VAR*i*. This is a way to control output to d3plot. If this method is used, the .xy file is not output. Therefore, DTOUT is not used (see *DATABASE_BINARY_D3PLOT). If SETID is not provided, all the ALE elements are considered.

*DATABASE_ALE_MAT

Purpose: For each ALE group (or material), this card activates extra output for:

1. material volume: alematvol.xy,
2. material mass: alematmas.xy,
3. internal energy: alematEint.xy,
4. kinetic energy: alematEkin.xy,
5. and kinetic energy loss during the advection: alematEkinlos.xy.

These files are written in the ".xy" format, which LS-PrePost can plot with its "XYPlot" button.

Card	1	2	3	4	5	6	7	8
Variable	DTOUT	BOXLOW	BOXUP	DTXY				
Type	F	I	I	F				
Default	none	0	0	0				

VARIABLE	DESCRIPTION
DTOUT	Time interval between the outputs
BOXLOW, BOXUP	Range of *DEFINE_BOX ids. BOXLOW is the lower bound for the range while BOXUP is the upper bound. The series of volumes covered by the specified range of *DEFINE_BOX determines the mesh regions for which ALE material data are to be output.
DTXY	Time interval between the extractions of ".xy" files from datalemat.tmp.

Remarks:

The ".xy" files are created at termination or if one of the following switches (Ctrl-C) is encountered: sw2, sw1, stop, quit.

*DATABASE_ALE_OPERATION

Purpose: Output values from a function defined by *DEFINE_FUNCTION to certain databases, namely d3plot and .xy history files. The function arguments are defined by the variable cards in this keyword as well as predefined variables (see Remark 1). The value computed by the function for each ALE element can replace a history variable in d3plot at a frequency defined in *DATABASE_BINARY_D3PLOT. Also, the function can output values for a set of elements to .xy history files (in a tabular format that can be plotted in LS-PrePost by using the XYPlot button) at a frequency defined by a time interval set in the input for this keyword. This keyword is only supported for ALE and ALE 2D.

Card Summary:

Card 1. This card is required.

FCT	HISVN	WRT					

Card 2. This card is included if WRT ≥ 10.

DT	SETID						

Card 3. This card is optional. Include as many of this card as needed. The next keyword ("*") card terminates this input.

VAR	VAR	VAR	VAR	VAR	VAR	VAR	VAR

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FCT	HISVN	WRT					
Type	I	I	I					
Default	none	none	1					

VARIABLE	DESCRIPTION
FCT	*DEFINE_FUNCTION ID; see Remark 1.
HISVN	Number of the history variable replaced in d3plot (see Remark 2.)

VARIABLE	DESCRIPTION
WRT	File output flag. WRT must be a two digit number: $$WRT = L + M \times 10$$ The 1's digit controls the replacement of the history variable number HISVN in d3plot: L.EQ.1: For each ALE element in the mesh, replace the values of the history variable with values computed by the function FCT. (See Remark 2.) L.EQ.0: Do not modify d3plot. The 10's digit controls the history output of values computed by the function FCT: M.EQ.1: For each ALE element in the set SETID, write .xy file that stores values computed by FCT at a frequency DT. (See Remarks 3 and 4.) M.EQ.0: Do not output this history file.

History Card. This card is included if WRT $\geq$ 10.

Card 2	1	2	3	4	5	6	7	8
Variable	DT	SETID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
DT	Time interval between computed function values included in the .xy file. (See Remarks 3 and 4.)
SETID	ALE element set ID. See Remarks 3 and 4. If the model is 2D (*SECTION_ALE2D), the set should be a shell set (see *SET_SHELL). If the model is 3D (*SECTION_SOLID), the set should be a solid set (see *SET_SOLID).

Function Argument Cards. Optional cards used to add more arguments to the user-defined function (time, time step, cycle number and volume fractions are always included as the first arguments to the function). Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	VAR	VAR	VAR	VAR	VAR	VAR	VAR	VAR
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
VAR	Arguments that can be included in function FCT (see Remark 1): EQ.1: *xx*-stress EQ.2: *yy*-stress EQ.3: *zz*-stress EQ.4: *xy*-stress EQ.5: *yz*-stress EQ.6: *zx*-stress EQ.7: Plastic strain EQ.8: Internal energy EQ.9: Bulk viscosity EQ.10: Previous volume EQ.11: Mass EQ.12: Volume EQ.13: Nodal *x*-positions EQ.14: Nodal *y*-positions EQ.15: Nodal *z*-positions EQ.16: Nodal *x*-velocities EQ.17: Nodal *y*-velocities EQ.18: Nodal *z*-velocities EQ.19: Nodal *x*-accelerations

VARIABLE	DESCRIPTION
	EQ.20: Nodal y-accelerations
	EQ.21: Nodal z-accelerations

Remarks:

1. **Function Argument Specification.** The argument values are presented to the function FCT as floating point data (see the example below). The time, time step, cycle number and volume fractions are the first arguments of the function by default. Then the order of the following arguments appearing in *DEFINE_-FUNCTION should match the order of the input in Card 3. For example, if the model has 2 ALE groups (so 2 volume fractions) and Card 3 has input "7, 8, 11", the arguments should be as follows:

   - Time,
   - Time step,
   - Cycle number,
   - Volume fraction of the 1st group,
   - Volume fraction of the 2nd group,
   - Plastic strain of the 1st group,
   - Plastic strain of the 2nd group,
   - Internal energy of the 1st group,
   - Internal energy of the 2nd group,
   - Mass of the 1st group,
   - Mass of the 2nd group.

   If there is a blank column between 2 inputs, all the variables between the two values are also included. For example, if the card contains "1, ,6", then the 6 stresses (1 through 6) are selected as arguments.

   For the nodal variables (VAR $\geq$ 13), the number of arguments in the function depends on the element type and its number of nodes. For a solid, 8 nodal variables are expected as arguments each time VAR is greater than or equal to 13., one for each node. For a shell, 4 nodal variables are expected as arguments. For example, VAR = 17 in a 3D ALE model would mean 8 y-velocities are arguments of the function FCT.

2. **Modification of History Variables in d3plot.** The history variables appear in d3plot after the first 7 volume averaged auxiliary variables: 6 stresses and plastic strain. The number of additional history variables output to d3plot is controlled by NEIPH (for solids) or NEIPS (for shells) in *DATABASE_EXTENT_BINARY. The history variables are listed in the MISC menu of LSPREPOST as history variable #1, history variable #2, … (the exact number in the code depends on *MAT

and *EOS). Before outputting data in d3plot, the history variable at history variable number HISVN is replaced by the value computed using FCT.

3. **Creating .xy Files.** The .xy files are created when the termination time is reached or if one of the following switches (after pressing the keys Ctrl - C) stops the job: sw1, stop, quit. During the run, they can be created with switch sw2. The file names start with "dataleop" followed by the Element ID and end with the extension .xy

4. **Curve Output.** Values computed by the function FCT are included in the .xy file for each element listed in the shell (2D) or solid (3D) set SETID at a frequency defined by the time interval DT.

Example:

In the example below, LS-DYNA computes the volume averaged pressure:

- for each ALE element so that the pressure can be fringed in d3plot in the place of history variable #2
- for each ALE element included in SETID so that the pressure history for these elements can be output to a .xy file

```
*DATABASE_ALE_OPERATION
$#     fct     hisvn       wrt
         1         2        11
$#      dt     setid
      1e-9         1
$#     var       var       var
         1         2         3
*DEFINE_FUNCTION
$#     fct
         1
float pressure(float     time, float timestep, float cycle
              ,float volfrac1, float volfrac2
              ,float xstress1, float xstress2
              ,float ystress1, float ystress2
              ,float zstress1, float zstress2)
{
  float pres,pres1,pres2;
  pres1 = -(xstress1+ystress1+zstress1)/3.0;
  pres2 = -(xstress2+ystress2+zstress2)/3.0;
  pres = volfrac1*pres1+volfrac2*pres2;
  return pres;
}
```

DATABASE_BINARY_*OPTION1_{OPTION2}

Purpose: Request binary output. See also *DATABASE_EXTENT_BINARY.

Choices for *OPTION1* are:

BLSTFOR Blast pressure database. See also *LOAD_BLAST_ENHANCED and Remark 3.

CPMFOR Corpuscular Particle Method interface force database. See Remark 2.

D3CRACK Option to control output interval for ASCII aea_crack file for the Winfrith concrete model (*MAT_084/085). Oddly, this command does not control the output of the binary crack database for the Winfrith concrete model. The binary crack database is written when "q = " appears on the execution line, and its output interval is taken from *DATABASE_BINARY_D3PLOT. LS-PrePost displays cracks in deformed Winfrith concrete materials with the combination of this file and d3plot.

D3DAT Database for CPG model data.

D3DRLF Dynamic relaxation database.

D3DUMP Database for restarts. Define output frequency in cycles.

D3PART Database for subset of parts. See also *DATABASE_EXTENT_BINARY and *DATABASE_EXTENT_D3PART.

D3PLOT Database for entire model. See also *DATABASE_EXTENT_BINARY.

D3PROP Database containing property data. See *DATABASE_BINARY_D3PROP.

D3THDT Database containing time histories for subsets of elements and nodes. See *DATABASE_HISTORY. This database does not include geometry.

DEMFOR DEM interface force database. See Remark 5.

FSIFOR ALE interface force database. The fsifor database does *not* have a default filename, so it *must* be given a filename using "h=" on the execution line. See Remark 1.

FSILNK ALE interface linking database. The force is in the normal direction only. Alternatively, FSILNK_FULL records and applies FSI forces in all directions. See Remark 4. Note that if an input deck includes both FSILNK and FSILNK_FULL only FSILNK_FULL will be used.

FSILNK_FULL ALE interface linking database. The forces are in all directions. See Remark 4. Note that if an input deck includes both FSILNK and FSILNK_FULL only FSILNK_FULL will be used.

INTFOR	Contact interface database. The intfor database does *not* have a default filename, so it *must* either be given a filename using the FILE option or using "S=" on the execution line. See *CONTACT fields SAPR and SBPR on Card 1. See Remark 9 for restrictions.
ISPHFOR	Incompressible SPH interface force database. See Remark 8.
PBMFOR	Particle Blast Method interface force database. See Remark 6.
RUNRSF	Database for restarts. Define output frequency in cycles.

The only choice for *OPTION2* is FILE. When *OPTION2* is set to FILE, the LS-DYNA keyword reader requires one extra data card indicating the file name for the database selected in *OPTION1*. *Presently, this is only implemented for *DATABASE_BINARY_INTFOR and *DATABASE_BINARY_ISPHFOR.*

The D3DUMP and the RUNRSF options create complete databases that are necessary for restarts. See Restart Input Data. When RUNRSF is specified, the same file is overwritten after each interval, unless the field NR is set, which causes a series of restart files to be overwritten in a cyclic order. When D3DUMP is specified, a new d3dump file is created after each interval. After the first interval, LS-DYNA writes d3dump01, after the second d3dump02, and so on. The default file names are runrsf and d3dump unless other names are specified on the execution line, see EXECUTION SYNTAX in the GETTING STARTED section. Since all data held in memory is written to the restart files, these files can be quite large. Care should be taken with the d3dump files not to create too many.

If *DATABASE_BINARY_D3PLOT is not specified in the keyword deck, then the output interval for d3plot is automatically set to one-twentieth of the termination time.

The d3plot, d3part, d3drlf, d3dat, and intfor databases contain histories of geometry and of state variables. With the LS-PrePost program, these databases can be used to, for example, animate deformed geometry and plot time histories of element stresses and nodal displacements. The d3thdt database contains time history data for element subsets and global information, but it does not include information about the geometry. See *DATABASE_HISTORY. This data can be plotted with the LS-PrePost program. For the contents of the d3plot, d3part, and d3thdt databases, see the *DATABASE_EXTENT_BINARY keyword. The size of the databases can be reduced by restricting the information that is dumped. The contents of the d3thdt database are also specified with the *DATABASE_HISTORY definition. Note that, in particular, the databases can be considerably reduced for models with rigid bodies containing many elements.

The fsifor database, like the intfor database, does not have a default filename. For this database a unique filename must be specified on the execution line with h=filename; see EXECUTION SYNTAX in the GETTING STARTED section. The file structure is such that each file contains the full geometry at the beginning, followed by the analysis-generated output data at the specified time intervals.

Card Summary:

Card 0. Additional card for the FILE keyword option (*OPTION2*). Presently, this is only implemented for *DATABASE_BINARY_INTFOR and *DATABASE_BINARY_ISPHFOR.

FNAME

Card 1. This card is required.

DT/CYCL	LCDT/NR	BEAM	NPLTC	PSETID	CID		

Card 2a. This card is read when *OPTION1* is D3PLOT. It is optional.

IOOPT	RATE	CUTOFF	WINDOW	TYPE	PSET		

Card 2b. This card is read when *OPTION1* is D3PART. It is optional.

HSETID	BSETID	SSETID	TSETID				

Card 2c. This card is read when *OPTION1* is INTFOR, D3DUMP, or D3DAT. It is optional.

IOOPT							

Data Card Definitions:

FILE Card. When *OPTION2* is set to FILE, include this card. Presently, this is only implemented for *DATABASE_BINARY_INTFOR and *DATABASE_BINARY_ISPHFOR.

Card 0	1	2	3	4	5	6	7	8
Variable	FNAME							
Type	A80							

VARIABLE	DESCRIPTION
FNAME	Name of the database. S=filename on the execution line overrides FNAME for *DATABASE_BINARY_INTFOR, while isph=filename on the execution line overrides FNAME for *DATABASE_BINARY_ISPHFOR

Card 1	1	2	3	4	5	6	7	8
Variable	DT/CYCL	LCDT/NR	BEAM	NPLTC	PSETID	CID	NSKIP	
Type	F	I	I	I	I	I	I	
Default	none	0/1	0	optional	optional	0	1	

VARIABLE	DESCRIPTION
DT	This field defines the time interval between output states, DT, for all options except D3DUMP, RUNRSF, and D3DRLF.
CYCL	For D3DUMP and RUNRSF options, this field is the number of time steps between output states. For the D3DRLF option, the value, n, inputted in this field causes an output state to be written every n^{th} convergence check during the explicit dynamic relaxation phase. Note that the keyword reader rounds CYCL to the nearest integer.
LCDT	Optional load curve ID specifying the output time interval as a function of time. This variable is only available for options D3DAT, D3DUMP, D3PART, D3PLOT, D3THDT, INTFOR, and BLSTFOR.
NR	Number of RUNning ReStart Files, runrsf, written in a cyclical fashion. The default is 1, that is, only one runrsf file is created, and the data therein is overwritten each time data is output.
BEAM	Discrete element option flag (*DATABASE_BINARY_D3PLOT only): EQ.0: Discrete spring and damper elements are added to the d3plot database where they are displayed as beam elements. The discrete elements' global x, global y, global z and resultant forces (moments) and change in length (rotation) are written to the database where LS-PrePost (incorrectly) labels them as though they were beam quantities, such as axial force, S-shear resultant, T-shear resultant, etc. EQ.1: No discrete spring, damper. and seatbelt elements are added to the d3plot database. This option is useful when translating old LS-DYNA input decks to KEYWORD input. In older input decks, there is no requirement that beam and spring elements have unique IDs, and beam elements may be created for the spring and dampers with identical IDs to existing beam elements, causing a fatal error. However, this option

VARIABLE	DESCRIPTION
	comes with some limitations and, therefore, should be used with caution. 1. Contact interfaces which are based on part IDs of seatbelt elements will not be properly generated if this option is used. 2. DEFORMABLE_TO_RIGID will not work if PID refers to discrete, damper, or seatbelt elements. EQ.2: Discrete spring and damper elements are added to the d3plot database where they are displayed as beam elements (similar to option 0). In this option, the element resultant force is written to its first database position, allowing beam axial forces and spring resultant forces to be plotted at the same time. This can be useful during some post-processing applications. This flag, set in *DATABASE_BINARY_D3PLOT, also affects the display of discrete elements in several other databases, such as d3drlf and d3part.
NPLTC	DT = ENDTIM/NPLTC. Applies to D3PLOT, D3PART, D3DUMP, D3DAT, DEMFOR, INTFOR, FSILNK, and FSILNK_FULL options only. This overrides the DT specified in the first field. ENDTIM is specified in *CONTROL_TERMINATION.
PSETID	Part set ID for D3PART and D3PLOT options only. See *SET_PART. Parts in PSETID are excluded from the d3plot database. The d3dat database also honors excluding the parts excluded from d3plot, but this option may not be provided to d3dat directly. Only parts in PSETID are included in the d3part database.
CID	Coordinate system ID for FSIFOR, FSILNK, and FSILNK_FULL. See *DEFINE_COORDINATE_SYSTEM.
NSKIP	For options D3DUMP and RUNRSF, MPP/HYBRID executables output both local and global restart files. Each processor creates its own local file for a small or simple restart. Then, processor 0 collects data from all processors and outputs the full deck restart file, but this output requires lots of data communication and disk space. This option provides a scale factor for CYCL (number of time steps between output states) to reduce the number of full deck restart output states to save CPU time. Thus, a full deck restart file is output every NSKIP × NCYCL time steps. The last full deck restart is always output with the regular restart file at the end of the job. The default value of

VARIABLE	DESCRIPTION
	NSKIP is 1.

D3PLOT Card. Additional card for the D3PLOT option. It is optional.

Card 2a	1	2	3	4	5	6	7	8
Variable	IOOPT	RATE	CUTOFF	WINDOW	TYPE	PSET		
Type	I	F	F	F	I	I		
Default	0	none	none	none	0	0		

VARIABLE	DESCRIPTION
IOOPT	This input field governs how the plot state frequency is determined from curve LCDT: EQ.1: When a plot is generated at time t_n, the next plot time t_{n+1} is computed as $$t_{n+1} = t_n + \text{LCDT}(t_n)\ .$$ This is the default behavior. EQ.2: When a plot is generated at time t_n, the next plot time t_{n+1} is computed as $$t_{n+1} = t_n + \text{LCDT}(t_{n+1})\ .$$ EQ.3: A plot is generated for each abscissa point in the load curve definition. The actual value of the load curve is ignored.
RATE	Time interval T between filter sampling. See Remark 7.
CUTOFF	Frequency cut-off. See Remark 7.
WINDOW	The width of the window W in units of time for storing the single, forward filtering required for the TYPE = 2 filter option. Increasing the width of the window will increase the memory required for the analysis. A window that is too narrow will reduce the amplitude of the filtered result significantly, and values below 15 are not recommended for that reason. In general, the results for the TYPE = 2 option are sensitive to the width of the window and experimentation is required. See Remark 7.

VARIABLE	DESCRIPTION
TYPE	Flag for filtering options. See Remark 7. EQ.0: No filtering (default). EQ.1: Single pass, forward Butterworth filtering. EQ.2: Two pass filtering over the specified time window. Backward Butterworth filtering is applied to the forward Butterworth results that have been stored. This option improves the phase accuracy significantly at the expense of memory.
PSET	Part set ID for filtering. If no set is specified, all parts are included. For each element integration point in the d3plot file, 24 words of memory are required in LS-DYNA for the single-pass filtering and more for the two-pass filtering. Specifying PSET is recommended to minimize the memory requirements. See Remark 7.

Optional D3PART Card. This card is read for the D3PART option. It is meant to specify element sets for which data is to be output and works in addition to PSETID. It is optional.

Card 2b	1	2	3	4	5	6	7	8
Variable	HSETID	BSETID	SSETID	TSETID				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
HSETID	Set ID of a *SET_SOLID for which data will be output to d3part
BSETID	Set ID of a *SET_BEAM for which data will be output to d3part
SSETID	Set ID of *SET_SHELL for which data will be output to d3part
TSETID	Set ID of *SET_TSHELL for which data will be output to d3part

Optional D3DAT, D3DUMP, and INTFOR Card. Additional card for the D3DAT, D3-DUMP, and INTFOR options. This card is optional.

Card 2c	1	2	3	4	5	6	7	8
Variable	IOOPT							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
IOOPT	This input field governs how the plot state frequency is determined from curve LCDT: EQ.1: When a plot is generated at time t_n, then next plot time t_{n+1} is computed as $t_{n+1} = t_n + \text{LCDT}(t_n)$. This is the default behavior. EQ.2: When a plot is generated at time t_n, then next plot time t_{n+1} is computed as $t_{n+1} = t_n + \text{LCDT}(t_{n+1})$. EQ.3: A plot is generated for each abscissa point in the load curve definition. The actual value of the load curve is ignored.

Remarks:

1. **FSIFOR.** *DATABASE_BINARY_FSIFOR only applies to models having penalty-based coupling between Lagrangian and ALE materials CTYPE = 4 or 5 in the coupling card, *CONSTRAINED_LAGRANGE_IN_SOLID, or in case of S-ALE, *ALE_STRUCTURED_FSI. When *DATABASE_FSI is defined, a few pieces of coupling information of some Lagrangian surface entities interacting with the ALE materials may be output as history parameters into a file called "dbfsi". Coupling pressure is one of the output variables. However, this coupling pressure is averaged over the whole surface entity being monitored. To obtain the coupling pressure contour plot as a function of time over the coupled surface, define *DATABASE_BINARY_FSIFOR. To use it, three things must be done:

a) The INTFORC parameter (*CONSTRAINED_LAGRANGE_IN_SOLID, 4th row, 3rd column) must be turned ON (INTFORC = 1). This is not required for *ALE_STRUCTURED_FSI. In that case, it is always ON.

b) A *DATABASE_BINARY_FSIFOR card is defined, controlling the output interval. The parameter DT in this card defines the time interval between outputs.

c) Executing LS-DYNA as follows activates writing this interface force file:

lsdyna i=**inputfilename.k** ... h=**interfaceforcefilename**

This binary interface force file contains the segment coupling pressure and forces over the whole simulation interval.

d) Forces from a 2D axisymmetric simulation are reported per radian, regardless of the element formulation used. Multiply the 2D force by 2π to get the 3D force.

2. **CPMFOR.** *DATABASE_BINARY_CPMFOR applies to models using *AIRBAG_PARTICLE feature. It controls the output interval of the CPM interface force file. This file does not have a default name. Thus, activating the writing of this file requires the cpm command line option (cpm=):

lsdyna i=**inputfilename.k** … cpm=**interfaceforce_filename**

The CPM interface force file stores each segment's coupling pressure and forces. The coupling pressure is averaged over each segment without considering the effect of ambient pressure, P_{atm}.

3. **BLSTFOR.** The BLSTFOR database is available for two-dimensional axisymmetric analysis. While line plotting is possible, fringing is not, at this time.

4. **FSILNK and FSILNK_FULL.** The *DATABASE_BINARY_FSILNK and *DATABASE_BINARY_FSILNK_FULL variants write the selected *CONSTRAINED_LAGRANGE_IN_SOLID interface's segment pressure and forces, respectively, to the fsilink file for the next analysis without ALE meshes. To select the interface, set INTFORC = 1 on *CONSTRAINED_LAGRANGE_IN_SOLID. This file does not have a default name. Thus, activating the writing of this file requires the fsilnk command line option (fsilnk=):

lsdyna i=**inputfilename.k** … fsilnk=**filename**

The output binary file can be used to load segments in a subsequent run with *LOAD_SEGMENT_FSILNK. It also can be converted into a keyword file in which the data is converted into *LOAD_SEGMENTs and *DEFINE_CURVEs. To create this keyword file, run LS-DYNA with only the fsilnk command option:

lsdyna fsilnk=**filename**

5. **DEMFOR.** *DATABASE_BINARY_DEMFOR applies to models using DEM coupling option *DEFINE_DE_TO_SURFACE_COUPLING. This keyword controls the output interval of DEM interface force file. This file does not have a default name. Thus, activating the writing of this file requires the dem command line option (dem=):

lsdyna i=**inputfilename.k** … dem=**interfaceforce_filename**

The DEM interface force file stores the coupling pressure and forces at each segment.

6. **PBMFOR.** *DATABASE_BINARY_PBMFOR applies to models using *DEFINE_PARTICLE_BLAST. It controls the output interval of PBM interface force file. This file does not have a default name. Thus, activating the writing of this file requires the pbm command line option (pbm=):

lsdyna i=**inputfilename.k** … pbm=**interfaceforce_filename**

The PBM interface force file stores the pressure and forces applied to the segments of the structure.

7. **D3PLOT filtering.** Filtered stress data replaces the stress data in the d3plot file when filtering is enabled.

8. **ISPHFOR.** *DATABASE_BINARY_ISPHFOR applies to models using incompressible SPH (FORM = 13 in *CONROL_SPH) with *DEFINE_SPH_MESH_SURFACE. This keyword controls the output interval of the ISPH interface force file. This file does not have a default name. Thus, activating the writing of this file requires either giving the file name with the FILE keyword option or using the isph command line option (isph=):

lsdyna i=**inputfilename.k** … isph=**interfaceforce_filename**

If both are provided, the file name given on the command line is used. The incompressible SPH interface force file stores the pressures and forces applied to the segments of the mesh surface by the SPH fluid. It also reports whether a given segment is wet or not (instantaneous wetness flag) at each output time step as well as how long a segment has been wet (time-accumulated wetness).

9. **INTFOR and eroding contacts.** The intfor database is not supported for eroding contacts as it was not designed to handle the creation and deletion of segments. Using this database for eroding contacts will trigger a warning and, in some cases (such as MPP or hybrid with 8 CPUs or more), can cause the analysis to freeze.

*DATABASE_BINARY_D3PROP

Purpose: Add part, material, equation of state, section, and hourglass data to the first d3plot file, or else write the data to a separate database d3prop. Rigidwall data can also be included.

Card 1	1	2	3	4	5	6	7	8
Variable	IFILE	IMATL	IWALL	ISETS				
Type	I	I	I	I				
Default	1	0	0	0				

VARIABLE	DESCRIPTION
IFILE	Specify file for d3prop output (This can also be defined on the command line by adding d3prop = 1 or d3prop = 2, which also sets IMATL = IWALL = 1): EQ.1: Output data at the end of the first d3plot file. EQ.2: Output data to the file d3prop.
IMATL	Output *EOS, *HOURGLASS, *MAT, *PART and *SECTION data: EQ.0: No EQ.1: Yes
IWALL	Output *RIGIDWALL data: EQ.0: No EQ.1: Yes
ISETS	Output all *SET data: EQ.0: No EQ.1: Yes

*DATABASE_CPM_SENSOR

Purpose: This card activates an ASCII file cpm_sensor. Its input defines the sensors' locations based on the positions of some Lagrangian segments. The output gives the history of the velocity, temperature, density and pressure averaged on the number of particles contained in the sensors. This card is activated only when the *AIRBAG_PARTICLE card is used.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY						
Type	F	I						

Sensor Definition Cards. Each card defines one sensor. This card may be repeated to define multiple sensors. Input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SEGID	OFFSET	LX	LY	LZ			
Type	I	F	F	F	F			

VARIABLE	DESCRIPTION
DT	Output interval
BINARY	Flag for the binary file EQ.1: ASCII file is written, EQ.2: Data is written to the binary file binout, EQ.3: ASCII file is written and the data is written to the binary file binout.
SEGID	Segment set ID. See *SET_SEGMENT.
OFFSET	Offset distance, d, between sensor and the segment center. See Remark 2.
LX	Radius(sphere and cylinder)/length in local x-direction(rectangular) of the sensor. See Remarks 1 and 2.

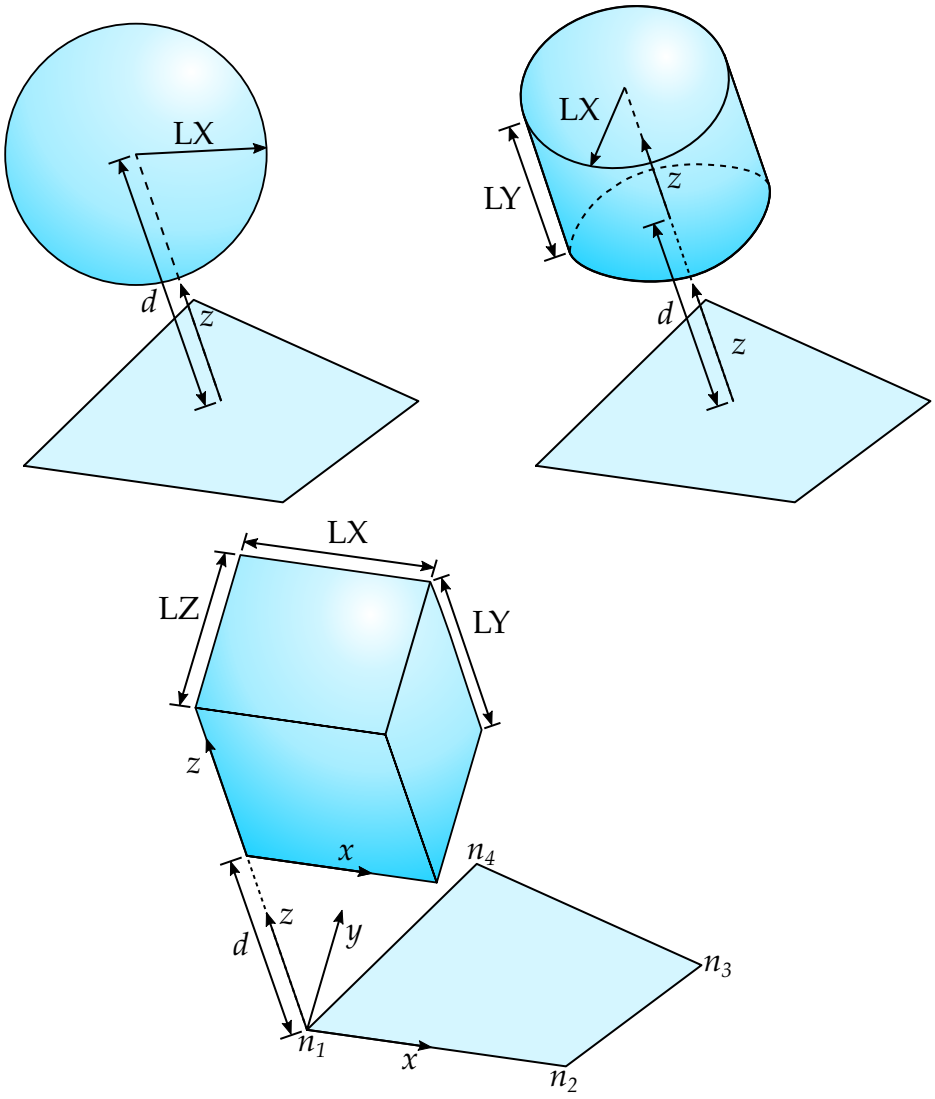


Figure 16-1. Sensor shapes

VARIABLE	DESCRIPTION
LY	Length(cylinder)/length in local *y*-direction(rectangular) of the sensor. See Remarks 1 and 2.
LZ	Length in local *z*-direction(rectangular) of the sensor. See Remarks 1 and 2.

Remarks:

1. **Sensor Shape.** The sensor can be three different shapes, a sphere, cylinder, or rectangle, depending on the length fields defined (LX, LY, and LZ). See Figure 16-1.

   a) If only LX is defined, then the sensor is sphere with radius LX.

b) If LX and LY are defined, then the sensor is a cylinder with radius LX and length LY.

c) If LX, LY, and LZ are defined, then the sensor is a rectangular prism with side lengths LX, LY, and LZ.

2. **OFFSET.** Each segment has a sensor. The distance between the segment and the sensor is defined by OFFSET (d in Figure 16-1) in the normal direction of the segment. This distance is constant, that is, the sensor moves along with the segment.

a) For a spherical sensor, OFFSET is the distance between the sphere center and the segment center. OFFSET should be larger than the radius of the sensor to prevent the segment from cutting the sphere.

b) For a cylindrical sensor, OFFSET is the distance from the segment center to the center of the base of the cylinder. The cylinder is extruded in the normal direction of the segment from the base.

c) For a rectangular sensor, OFFSET is the distance from the segment to the sensor. The sensor is defined using the segment's coordinates system. The base point for the coordinate system is n_1 from the segment definition (see *SET_SEGMENT). The local x-direction is along the vector $n_2 - n_1$. The local z-direction is the segment normal direction, and the local y-direction is constructed by the cross product of the local z- and x-directions.

3. **Output.** The output parameters in the cpm_sensor file are:

velx	=	x-velocity
vely	=	y-velocity
velz	=	z-velocity
velr	=	velocity
temp	=	temperature
dens	=	density
pres	=	pressure

These values are averaged on the number of particles in the sensor. The sensor should be large enough to contain a reasonable number of particles for the averages.

Example:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|.
$ INPUT:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|.
*DATABASE_CPM_SENSOR
      0.01
```

```
$   SEGSID    OFFSET        LX        LY
       123       5.0       5.0
       124      -0.2       0.1
       125       0.7       0.6       1.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|..
$ The segment set id: 123 has 1 segment.
$ The segment set id: 124 has 1 segment.
$ The segment set id: 125 has 11 segments.

$ Each segment has an ID defined in D3HSP
$ The D3HSP file looks like the following:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|..
Segments for sensor          1
  Sensor id        n1        n2        n3        n4
         1      3842      3843      3848      3847
 Segments for sensor          2
  Sensor id        n1        n2        n3        n4
         2      3947      3948      3953      3952

 Segments for sensor          3
  Sensor id        n1        n2        n3        n4
         3      3867      3868      2146      2145
         4      3862      3863      3868      3867
         5      3857      3858      3863      3862
         6      3852      3853      3858      3857
         7      3847      3848      3853      3852
         8      3837      3838      3843      3842
         9      3842      3843      3848      3847
        10      3832      3833      3838      3837
        11      3827      3828      3833      3832
        12      3822      3823      3828      3827
        13      1125      1126      3823      3822
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|..
```

DATABASE_CROSS_SECTION_*OPTION1_{OPTION2}

OPTION1 includes:

PLANE

SET

To define an ID and heading for the database cross section, set *OPTION2* to:

ID

Purpose: Define a cross section for resultant forces written to ASCII file secforc.

1. For the PLANE option, a set of two cards defines a cutting plane, see Figure 16-2. Based on this cutting plane, a node set and element set(s) which comprise the cross section are internally generated. Using the variable ICRFILE in *CONTROL_OUTPUT, those sets may be output for the purpose of displaying the nodes and elements of the cross section using LS-PrePost.

2. If the SET option is used, just one card is needed which identifies a node set and at least one element set. In this case the node set(s) defines the cross section, and the forces from the elements belonging to the element set(s) are summed up to calculate the section forces. Thus, the element set(s) should include elements on only one side (not both sides) of the cross section.

The cross-section should cut through deformable elements only, not rigid bodies. Cutting through reference segments for deformable solid element spot welds can lead to incorrect section forces since the constraint forces are not accounted for in the force and moment summations. Beam element modeling of welds do *not* require any special precautions.

Card Summary:

Card ID. This card is included if the ID keyword option is used.

CSID	HEADING

Card 1a.1. This card is included if the PLANE keyword option is used.

PSID	XCT	YCT	ZCT	XCH	YCH	ZCH	RADIUS

Card 1a.2. This card is included if the PLANE keyword option is used.

XHEV	YHEV	ZHEV	LENL	LENM	ID	ITYPE	

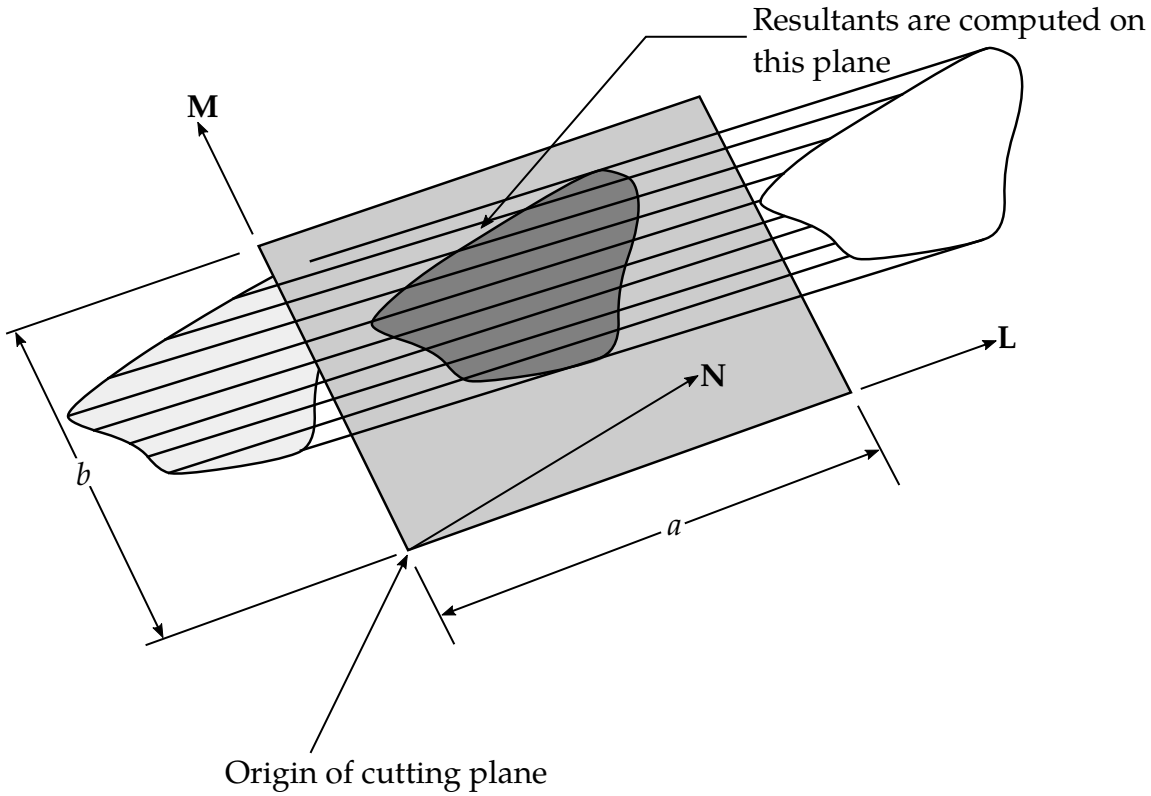


Figure 16-2. Definition of cutting plane for automatic definition of interface for cross-sectional forces. The automatic definition does not check for springs and dampers in the section. For best results the cutting plane should cleanly pass through the middle of the elements, distributing them equally on either side. Elements that intersect the edges of the cutting plane are deleted from the cross-section.

Card 1b. This card is included if the SET keyword option is used.

NSID	HSID	BSID	SSID	TSID	DSID	ID	ITYPE

Data Card Definitions:

ID Card. Additional card for ID keyword option. The heading is picked up by some of the peripheral LS-DYNA codes to aid in post-processing.

Card ID	1	2	3	4	5	6	7	8
Variable	CSID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
CSID	Cross section ID. This must be a unique number.
HEADING	Cross section descriptor. It is suggested that unique descriptions be used.

Plane Card 1. First additional card for PLANE keyword option.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	PSID	XCT	YCT	ZCT	XCH	YCH	ZCH	RADIUS
Type	I	F	F	F	F	F	F	F
Default	0	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
PSID	Part set ID. If zero, all parts are included.
XCT	x-coordinate of tail of any outward drawn normal vector, **N**, originating on wall (tail) and terminating in space (head), see Figure 16-2.
YCT	y-coordinate of tail of normal vector, **N**.
ZCT	z-coordinate of tail of normal vector, **N**.
XCH	x-coordinate of head of normal vector, **N**.
YCH	y-coordinate of head of normal vector, **N**.
ZCH	z-coordinate of head of normal vector, **N**.
RADIUS	Optional radius: EQ.0.0: Not used. GT.0.0: A circular cut plane will be created that is centered at (XCT,YCT,ZCT) with radius = RADIUS and has a normal vector originating at (XCT,YCT,ZCT) and pointing towards (XCH,YCH,ZCH). LT.0.0: The radius will be the absolute value of RADIUS and XCT and XCH will be nodes IDs. The node with ID XCT is the center of the circular cut plane. The normal vector

VARIABLE	DESCRIPTION
	of the plane is the vector pointing from the node with ID XCT to the node with ID XCH. YCT, ZCT, YCH, and ZCH are ignored.
	If RADIUS ≠ 0.0, the variables XHEV, YHEV, ZHEV, LENL, and LENM, which are specified on Card 1a.2, will be ignored.

Plane Card 2. Second additional card for PLANE keyword option.

Card 1a.2	1	2	3	4	5	6	7	8
Variable	XHEV	YHEV	ZHEV	LENL	LENM	ID	ITYPE	
Type	F	F	F	F	F	I	I	
Default	0.	0.	0.	infinity	infinity	global	0	

VARIABLE	DESCRIPTION
XHEV	x-coordinate of head of edge vector, **L**.
YHEV	y-coordinate of head of edge vector, **L**.
ZHEV	z-coordinate of head of edge vector, **L**.
LENL	Length of edge a, in **L** direction.
LENM	Length of edge b, in **M** direction.
ID	Rigid body (see *MAT_RIGID, type 20), accelerometer ID (see *ELEMENT_SEATBELT_ACCELEROMETER), or coordinate ID (see *DEFINE_COORDINATE_NODES). The force resultants are output in the *updated* local system of the rigid body or accelerometer. For ITYPE = 2, the force resultants are output in the updated local coordinate system if FLAG = 1 in *DEFINE_COORDINATE_NODES or if NID is nonzero in *DEFINE_COORDINATE_VECTOR.
ITYPE	Flag that specifies whether ID above pertains to a rigid body, an accelerometer, or a coordinate system. EQ.0: Rigid body, EQ.1: Accelerometer,

VARIABLE	DESCRIPTION
	EQ.2: Coordinate system.

Set Card. Additional Card for the SET keyword option. The set option requires that the equivalent of the automatically generated input by the cutting plane capability be identified manually and defined in sets. All nodes in the cross-section and their related elements that contribute to the cross-sectional force resultants must be defined.

Card 1b	1	2	3	4	5	6	7	8
Variable	NSID	HSID	BSID	SSID	TSID	DSID	ID	ITYPE
Type	I	I	I	I	I	I	I	I
Default	required	0	0	0	0	0	global	0

VARIABLE	DESCRIPTION
NSID	Nodal set ID, see *SET_NODE_*OPTION*.
HSID	Solid element set ID, see *SET_SOLID.
BSID	Beam element set ID, see *SET_BEAM.
SSID	Shell element set ID, see *SET_SHELL_*OPTION*.
TSID	Thick shell element set ID, see *SET_TSHELL.
DSID	Discrete element set ID, see *SET_DISCRETE.
ID	Rigid body (see *MAT_RIGID, type 20), accelerometer ID (see *ELEMENT_SEATBELT_ACCELEROMETER), or coordinate ID (see *DEFINE_COORDINATE_NODES). The force resultants are output in the *updated* local system of the rigid body or accelerometer. For ITYPE = 2, the force resultants are output in the updated local coordinate system if FLAG = 1 in *DEFINE_COORDINATE_NODES or if NID is nonzero in *DEFINE_COORDINATE_VECTOR.
ITYPE	Flag that specifies whether ID above pertains to a rigid body, an accelerometer, or a coordinate system. EQ.0: Rigid body, EQ.1: Accelerometer,

VARIABLE	DESCRIPTION
	EQ.2: Coordinate system.

*DATABASE_DEFRAGMENT

Purpose: Output the fragment volume, radius, and maximum size at a specified time interval for the bonded DEM. Note that *DATABASE_DEFRAGMENT is applicable only to bonded DEM (*DEFINE_DE_BOND).

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY						
Type	F	I						
Default	none	0						

VARIABLE	DESCRIPTION
DTOUT	Time interval between outputs. If DT is zero, no output is printed.
BINARY	Flag for binary output: EQ.0: Do not write the file. EQ.1: Write data to an ASCII file. EQ.2: Write data to binary database binout. EQ.3: Write data to an ASCII file and to binary database binout.

*DATABASE_D3FTG

Purpose: Activate writing binary plot database d3ftg, which saves fatigue analysis results.

Card 1	1	2	3	4	5	6	7	8
Variable	BINARY	DT	MSFTG					
Type	I	F	I					
Default	1	0.0	0					

VARIABLE	DESCRIPTION
BINARY	Flag for writing the binary plot file: EQ.0: Do not write binary plot file d3ftg. EQ.1: Write the binary plot file d3ftg.
DT	Time interval between output states in time-domain fatigue analysis (see *FATIGUE_*{OPTION}* and Remark 2) EQ.0.0: Only fatigue results at the end of the analysis are output.
MSFTG	Multi-step fatigue damage calculation method for time-domain fatigue analysis (see Remark 3): EQ.0: Fatigue damage is calculated incrementally. EQ.1: Fatigue damage is calculated using the entire stress/strain history from time 0 to the current output time point.

Remarks:

1. **Frequency-domain fatigue.** To activate writing the d3ftg database for frequency-domain fatigue analysis, including random vibration fatigue (*FREQUENCY_DOMAIN_RANDOM_VIBRATOIN_FATIGUE) and steady-state vibration fatigue (*FREQUENCY_DOMAIN_SSD_FATIGUE), either this keyword or *DATABASE_FREQUENCY_BINARY_D3FTG can be used. More details about the d3ftg database can be found in *DATABASE_FREQUENCY_BINARY_*OPTION*.

2. **Time-domain fatigue.** For time-domain fatigue (*FATIGUE_*{OPTION}*), if DT is blank or is defined as 0.0, only the fatigue results at the end of the analysis are saved to d3ftg, so d3ftg includes only one state. If DT is defined as a nonzero value, the fatigue results at every DT are saved in d3ftg, so d3ftg includes multiple states. With this option, you can observe the evolution of the fatigue results (such as the cumulative damage ratio) in the model over time.

3. **MSFTG.** MSFTG = 0 invokes rainflow counting and fatigue damage calculations for each time interval (e.g., 0-DT, DT-2DT, 2DT-3DT, etc., where DT is the time interval between output states). The fatigue damage accumulated in each interval is then summed to obtain the total damage reported at the current output time in d3ftg. In other words, the overall damage is obtained by adding the contributions from many small rainflow-counting samples. MSFTG = 1 causes rainflow counting and fatigue damage calculations that use the entire stress/strain history from time 0 to the current output time. Thus, a single rainflow-counting operation is performed for each output point. As a result, the reported damage ratio may differ slightly between MSFTG = 0 and MSFTG = 1 because rainflow counting is not strictly additive when cycle histories are partitioned into smaller segments. In terms of computational cost, MSFTG = 0 is generally less expensive because it processes only the newly added time interval at each step. MSFTG = 1 is more computationally expensive since the complete history must be reprocessed for every output point, but it can provide higher accuracy because it preserves the full cycle history during rainflow counting.

*DATABASE_D3MAX

Purpose: Activate writing binary plot database d3max (accessible with LS-PrePost). d3max contains the maximum or minimum (depending on OUTPUT in Card 1) values of stresses or strains for each element or certain elements during a transient analysis. The maximum (or minimum) stresses/strains can be dumped to d3part database if needed. *DATABASE_MAX_*OPTION* restricts which elements have their maximum (or minimum) values extracted (see Remark 1). Note that when this keyword is enabled, stresses are always output. The output of the strains, however, is controlled by STRFLG on *DATABASE_EXTENT_BINARY.

Card 1	1	2	3	4	5	6	7	8
Variable	DTCHECK	ME	PSTRS	PSTRN	IFILT	OUTPUT	FCUTOUT	NFAIL
Type	F	I	I	I	I	I	F	I
Default	none	1	0	0	0	0	0.	0

VARIABLE	DESCRIPTION
DTCHECK	Time step for checking and updating maximum (or minimum) values. For instance, if DTCHECK = 10^{-6}, LS-DYNA will check and update the maximum values every 10^{-6} seconds (assuming for this example the time units are seconds). It will compare the current values (stress or strain) with the maximum values up to now. If the current values are larger, the maximum values will be replaced by the current values. Otherwise, the maximum values will remain unchanged. If OUTPUT > 2, the minimum values are output instead of the maximum. The same algorithm applies to the minimum values, except that the minimum values are replaced if the current values are smaller.
ME	Method for extracting stresses/strains: EQ.1: Extract maximum (or minimum) stress/strain during transient analysis. EQ.2: Extract maximum (or minimum) stress/strain after transient analysis (not used).
PSTRS	Output maximum or minimum principal stress in place of maximum or minimum normal stresses in the global coordinate system

VARIABLE	DESCRIPTION
	(see Remark 2): EQ.0: No EQ.1: Yes
PSTRN	Output maximum or minimum principal strain in place of the maximum or minimum normal strains in the global coordinate system (see Remark 2): EQ.0: No EQ.1: Yes
IFILT	Use filter: EQ.0: No EQ.1: Use low pass 2nd order Butterworth filter
OUTPUT	Output format and flag to determine whether minimum or maximum values are output (see Remarks 4 and 5): EQ.0: Write maximum stress/strain to d3max EQ.1: Append the maximum stress/strain results to d3part EQ.2: Write the maximum stress/strain results to d3part instead of the normal data that goes into d3part (negative time stamps are used in d3part to distinguish when this is done from the normal d3part output, which saves time history results for selected parts) EQ.3: Write minimum stress/strain to d3max EQ.4: Append the minimum stress/strain results to d3part EQ.5: Write the minimum stress/strain results to d3part instead of the normal data that goes into d3part (negative time stamps are used in d3part to distinguish when this is done from the normal d3part output, which saves time history results for selected parts EQ.6: Same as OUTPUT = 0, but map the results onto the original geometry instead of using the deformed geometry in d3max EQ.7: Same as OUTPUT = 3, but map the results onto the original geometry instead of using the deformed geometry in d3max EQ.8: Write the maximum stress/strain results, the minimum

VARIABLE	DESCRIPTION
	stress/strain results, and the absolute maximum stress/strain results to d3max (state 1: maximum stress/strain; state 2: minimum stress/strain; state 3: absolute maximum stress/strain). EQ.9: Same as OUTPUT = 8, but map the results onto the original geometry instead of using the deformed geometry in d3max
FCUTOUT	Cutout frequency for the Butterworth filter
NFAIL	Flag to determine whether failed elements are kept in d3max: EQ.0: Failed elements are removed from d3max (default). EQ.1: Failed elements are kept in d3max so that their maximum stress/strain are included in the fringe plot.

Remarks:

1. **D3MAX.** This keyword activates saving the maximum stress and/or strain values of elements in a structure during a transient simulation to the d3max or d3part database. The time step, DTCHECK, determines how often the maximum values are updated. Only elements or sets of elements specified with *DATABASE_MAX_*OPTION* have nonzero values for the stress/strain in d3max. Stresses/strains for all elements not included in a *DATABASE_MAX_*OPTION* are also output to d3max but with zero values.

2. **Principal stress and strain.** By default, the maximum or minimum stress / strain components in the global coordinate system are individually updated and output. When PSTRS = 1, the maximum or the minimum of the principal stress components are stored at the location of the normal stresses in the global coordinate system in the output file. Similarly, when PSTRN = 1, the maximum or the minimum of the principal strain components are stored at the location of normal strain components in the global coordinate system. The maximum or minimum shear stress / strain components in the global coordinate system are output regardless of the setting of PSTRS / PSTRN.

3. **Save d3max data to d3part.** With OUTPUT = 2, the maximum stress / strain data for each state is output to the d3part database. The normal data in d3part is not output in this case. The time stamps of the output data to d3part are changed to negative values. Only the elements in the parts specified with PSETID on *DATABASE_BINARY_D3PART are saved to d3part. Elements specified with *DATABASE_MAX_*OPTION* that are not in parts in PSETID are

not in the output. However, if *DATABASE_MAX_*OPTION* is used and does not include some elements that are in the parts of PSETID, the elements that are in the parts but not in *DATABASE_MAX_*OPTION* will have values of zero in the output to d3part.

4. **OUTPUT.** OUTPUT sets whether the maximum or minimum stress/strain is output to d3max or d3part. The maximum (OUTPUT = 0, 1, 2, or 6) should be used if the stress values are mostly positive (tension), and the minimum (OUTPUT = 3, 4, 5, or 7) should be used if the stress values are mostly negative (compression). To obtain the maximum, minimum, and absolute maximum in a single run, set OUTPUT to 8 or 9. For OUTPUT = 0 through 5 and 8, deformed or updated geometry is used in d3max or d3part. For OUTPUT = 6, 7, and 9, the original geometry is used in d3max. The original geometry can be useful if the model undergoes large deformations, and it is difficult to review the stress plot on the deformed geometry.

5. **Maximum, minimum, and absolute maximum.** The stress may vary from compressive (negative stress) to tensile (positive stress). The maximum and minimum stress/strain referred to in d3max (or d3part) are based on their arithmetic values. For example, if a stress component increases monotonically from -100 MPa to 90 MPa, then in a d3max file with three states, the maximum stress (state 1) is 90 MPa, the minimum stress (state 2) is -100 MPa, and the absolute maximum stress (state 3) is 100 MPa.

6. **von Mises stress.** d3max automatically contains the maximum von Mises stress. However, only versions 4.12 and later of LS-PrePost read the value output to d3max. Older versions of LS-PrePost calculated the value from the stress components output to d3max, leading to an incorrect von Mises stress.

*DATABASE_ENVELOPE_BEAM

Purpose: Write a text file containing the maximum values of force, moment and other beam element data that have occurred at any time during the analysis. The purpose of this keyword is to avoid writing very large time-history files in cases where results might vary rapidly and the maximum values need to be recorded, such as in a seismic analysis of a building.

Card 1	1	2	3	4	5	6	7	8
Variable	TCHECK	TBACK			BSETID			OUTPUT
Type	F	F			I			I
Default	0.0	0.0			0			2

VARIABLE	DESCRIPTION
TCHECK	Time step for checking beam data against the previous maxima (minima) and updating maximum (or minimum) values (see Remark 2) EQ.0.0: Data is checked every time step.
TBACK	Time interval for writing output files EQ.0.0: Output files are written only at termination.
BSETID	ID of a set containing the selected beam elements EQ.0: All beam elements in the model
OUTPUT	Output file control for the details file (see Remark 4): EQ.0: Same as 2 EQ.1: Human-readable text format EQ.2: CSV format EQ.3: No details file

Remarks:

1. **Output files.** This keyword activates output of a summary file and a details file. By default, both are written only when the analysis terminates. If TBACK is

nonzero, or if one or more restarts have been performed, a series of files will be written with filenames containing 0001, 0002, 0003, etc.

2. **Summary file.** The summary file tabulates the most positive and most negative values of each data component (defined in Remark 3) that have occurred at any time during the analysis in each of the selected beam elements. These maxima/minima are referred to as "peaks" in these remarks. Note that, for a given element, the peaks recorded in the different data components will likely have occurred at different times. The summary file is always written in CSV format.

3. **Data components.** The data components for which peaks are written to the summary file include:

   a) All six forces and moments that are written to the d3plot file.

   b) For Belytschko beam elements (ELFORM = 2), the moments about the local *s*- and *t*-axes at Node 2 as well as those at Node 1.

   c) For discrete beam elements (ELFORM = 6), the translational displacements (only if *DATABASE_DISBOUT is activated).

   d) For integrated beams (ELFORM = 1) the maximum axial stress, shear stress, and plastic strain across all of the element's integration points.

4. **Details file.** In addition to the summary file, a further file can be written containing the time at which each force and moment peak occurred, together with the full force/moment state in the element at the time of that peak. For example, if the maximum positive axial force in element 1 occurred at time = 0.001 while the most negative axial force occurred at time = 0.0015, then the details file will contain a row of data giving all the forces and moments that existed in element 1 at time 0.001 and another row of data giving all the forces and moments that existed in element 1 at time 0.0015. In total, 16 rows of data will be output for each selected beam element. Two formats are available for the details file, or the details file may be de-activated altogether (see OUTPUT).

5. **Default values.** In the summary and details files, a value of zero means that the data component was checked and its value never exceeded zero (for positive peaks) or fell below zero (for negative peaks). A value of 1.0E20 or -1.0E20 means that the data component is not available. For example, discrete beam displacements are not available for beams with ELFORM = 2.

*DATABASE_EXTENT

This family of keywords control to some extent the content of specific output databases. They are listed below in alphabetical order:

*DATABASE_EXTENT_AVS

*DATABASE_EXTENT_BINARY

*DATABASE_EXTENT_D3PART

*DATABASE_EXTENT_INTFOR

*DATABASE_EXTENT_MOVIE

*DATABASE_EXTENT_MPGS

*DATABASE_EXTENT_SSSTAT

The BINARY option of *DATABASE_EXTENT applies to the binary databases d3plot, d3thdt, and d3part. In the case of the d3part database, variables set using the D3PART option will override the corresponding variables of the BINARY option. See also *DATABASE_BINARY_*OPTION*.

The AVS, MOVIE, and MPGS databases will be familiar to users that have a use for those databases.

*DATABASE_EXTENT_AVS

Purpose: Control the content written to the avsflt database. See AVSFLT option to *DATABASE card.

Variable Cards. Define as many cards as needed. Input ends at next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	VTYPE	COMP						
Type	I	I						

VARIABLE	DESCRIPTION
VTYPE	Variable type: EQ.0: node, EQ.1: brick, EQ.2: beam, EQ.3: shell, EQ.4: thick shell.
COMP	Component ID. For the corresponding VTYPE, integer components from the following tables can be chosen: VTYPE.EQ.0: Table 16-1, VTYPE.EQ.1: Table 16-2, VTYPE.EQ.2: not supported, VTYPE.EQ.3: Table 16-3, VTYPE.EQ.4: not supported.

Remarks:

The AVS database consists of a title card, then a control card defining the number of nodes, brick-like elements, beam elements, shell elements, and the number of nodal vectors, NV, written for each output interval. The next NV lines consist of character strings that describe the nodal vectors. Nodal coordinates and element connectivity follow. For each state, the solution time is written, followed by the data requested below. The last word in the file is the number of states. We recommend creating this file and examining its contents, since the organization is relatively transparent.

Table 16-1. Nodal Quantities

Component ID	Quantity
1	x, y, z-displacements
2	x, y, z-velocities
3	x, y, z-accelerations

Table 16-2. Brick Element Quantities

Component ID	Quantity
1	x-stress
2	y-stress
3	z-stress
4	xy-stress
5	yz-stress
6	zx-stress
7	effective plastic strain

Table 16-3. Shell and Thick Shell Element Quantities

Component ID	Quantity
1	mid-surface x-stress
2	mid-surface y-stress
3	mid-surface z-stress
4	mid-surface xy-stress
5	mid-surface yz-stress
6	mid-surface xz-stress
7	mid-surface effective plastic strain
8	inner surface x-stress
9	inner surface y-stress
10	inner surface z-stress
11	inner surface xy-stress
12	inner surface yz-stress

Component ID	Quantity
13	inner surface zx-stress
14	inner surface effective plastic strain
15	outer surface x-stress
16	outer surface y-stress
17	outer surface z-stress
18	outer surface xy-stress
19	outer surface yz-stress
20	outer surface zx-stress
21	outer surface effective plastic strain
22	bending moment M_{xx} (4-node shell)
23	bending moment M_{yy} (4-node shell)
24	bending moment M_{xy} (4-node shell)
25	shear resultant Q_{xx} (4-node shell)
26	shear resultant Q_{yy} (4-node shell)
27	normal resultant N_{xx} (4-node shell)
28	normal resultant N_{yy} (4-node shell)
29	normal resultant N_{xy} (4-node shell)
30	thickness (4-node shell)
31	element dependent variable
32	element dependent variable
33	inner surface x-strain
34	inner surface y-strain
35	inner surface z-strain
36	inner surface xy-strain
37	inner surface yz-strain
38	inner surface zx-strain
39	outer surface x-strain
40	outer surface y-strain
41	outer surface z-strain
42	outer surface xy-strain

Component ID	Quantity
43	outer surface *yz*-strain
44	outer surface *zx*-strain
45	internal energy
46	mid-surface effective stress
47	inner surface effective stress
48	outer surface effective stress
49	mid-surface max. principal strain
50	through thickness strain
51	mid-surface min. principal strain
52	lower surface effective strain
53	lower surface max. principal strain
54	through thickness strain
55	lower surface min. principal strain
56	lower surface effective strain
57	upper surface max. principal strain
58	through thickness strain
59	upper surface min. principal strain
60	upper surface effective strain

Table 16-4. Beam Element Quantities

Component ID	Quantity
1	*x*-force resultant
2	*y*-force resultant
3	*z*-force resultant
4	*x*-moment resultant
5	*y*-moment resultant
6	*z*-moment resultant

*DATABASE_EXTENT_BINARY_{OPTION}

Purpose: Control to some extent the content of binary output databases d3plot, d3thdt, and d3part. See also *DATABASE_BINARY_*OPTION* and *DATABASE_EXTENT_D3PART. The content of the binary output database intfor may be modified using *DATABASE_EXTENT_INTFOR.

Available options include:

<BLANK>

COMP

The *DATABASE_EXTENT_BINARY_COMP reduces the content of binary output databases d3plot and d3eigv to a maximum of only seven categories (see the seven variables of Card 1a) and therefore overrides most of the settings in *DATABASE_EXTENT_BINARY. Furthermore, a maximum of three through-thickness shell or thick shell integration points are output to all binary databases (d3plot, d3eigv, d3part, d3drlf, etc.) when *DATABASE_EXTENT_BINARY_COMP is used, that is, if MAXINT in *DATABASE_EXTENT_BINARY is set greater than 3, that variable is automatically reset to 3.

Card Summary:

Card 1a. This card is included if the COMP keyword option is used

IGLB	IXYZ	IVEL	IACC	ISTRS	ISTRA	ISED	

Card 1b. This card is included if no keyword option is used.

NEIPH	NEIPS	MAXINT	STRFLG	SIGFLG	EPSFLG	RLTFLG	ENGFLG

Card 2. This card is included if no keyword option is used.

CMPFLG	IEVERP	BEAMIP	DCOMP	SHGE	STSSZ	N3THDT	IALEMAT

Card 3. This card may be included if no keyword option is used. This card is optional.

NINTSLD	PKP_SEN	SCLP	HYDRO	MSSCL	THERM	INTOUT	NODOUT

Card 4. This card may be included if no keyword option is used. This card is optional.

DTDT	RESPLT	NEIPB	QUADSLD	CUBSLD	DELERES		

Data Card Definitions:

COMP Card. For COMP option, use this card (no Cards 2-4).

Card 1a	1	2	3	4	5	6	7	8
Variable	IGLB	IXYZ	IVEL	IACC	ISTRS	ISTRA	ISED	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

VARIABLE	DESCRIPTION
IGLB	Output flag for global and part history variables such as internal energy, kinetic energy, rigid body velocity, etc., that is, the sort of data that can also be output to glstat and matsum. EQ.0: No EQ.1: Yes
IXYZ	Output flag for nodal coordinates: EQ.0: No EQ.1: Yes
IVEL	Output flag for nodal velocities: EQ.0: No EQ.1: Yes
IACC	Output flag for nodal accelerations: EQ.0: No EQ.1: Yes
ISTRS	Output flag for stress tensor and “plastic strain”: EQ.0: No EQ.1: Yes
ISTRA	Output flag for strain tensor: EQ.0: No

VARIABLE	DESCRIPTION
	EQ.1: Yes
ISED	Output flag for strain energy density: EQ.0: No EQ.1: Yes

Card 1b	1	2	3	4	5	6	7	8
Variable	NEIPH	NEIPS	MAXINT	STRFLG	SIGFLG	EPSFLG	RLTFLG	ENGFLG
Type	I	I	I	I	I	I	I	I
Default	0	0	3	0	1	1	1	1
Remarks			1	10				

VARIABLE	DESCRIPTION
NEIPH	Number of additional integration point history variables written to the binary databases (d3plot, d3part, d3drlf) for solid elements and SPH particles. The integration point data is written in the same order that it is stored in memory; each material model has its own history variables that are stored. For user-defined materials the history data that is needed for plotting should be stored before the data which is not of interest. See also *DEFINE_MATERIAL_HISTORIES. For output of additional integration point history variables for solid elements to the elout database, see the field OPTION1 for *DATABASE_ELOUT.
NEIPS	Number of additional integration point history variables written to the binary databases (d3plot, d3part, d3drlf) for both shell and thick shell elements for each integration point; see NEIPH above and *DEFINE_MATERIAL_HISTORIES. For output of additional integration point history variables for shell and thick shell elements to the elout database, see the fields OPTION2 and OPTION3, respectively, in *DATABASE_ELOUT.

VARIABLE	DESCRIPTION

MAXINT

Number of shell and thick shell through-thickness integration points for which output is written to d3plot. This does not apply to the strain tensor output flagged by STRFLG.

MAXINT (def = 3)	Number of Integration Points	Description
3	> 3 (even & odd)	Results are output for the outermost (top) and innermost (bottom) integration points together with results for the neutral axis.
3	1	All three results are identical.
> 3	≤ MAXINT	Results for the first MAXINT integration points in the element will be output.
≠ 3	Even	See above. This will *exclude* mid-surface results, whereas when MAXINT = 3 mid-surface results are calculated and reported.
< 0	Any	MAXINT integration points are output for each in plane integration point location and no averaging is used. This can greatly increase the size of the binary databases d3plot, d3thdt, and d3part.

See Remark 1 for more information.

STRFLG

Flag for output of strain tensors. STRFLG is interpreted digit-wise STRFLG = [*NML*],

$$\text{STRFLG} = L + M \times 10 + N \times 100$$

L.EQ.1: Write strain tensor data to d3plot, elout, dynain, d3rms, and d3psd. For shell and thick shell elements, two tensors are written, one at the innermost and one at the outermost integration point. For solid elements, a single strain tensor is written. For cohesive elements, the cohesive separations are output in the location of the strain tensor data.

M.EQ.1: Write plastic strain data to d3plot.

N.EQ.1: Write thermal strain data to d3plot.

VARIABLE	DESCRIPTION
	Examples. For STRFLG = 11 (011) LS-DYNA will write both strain and plastic strain tensors, but no thermal strain tensors. Whereas for STRFLG = 110, LS-DYNA will write plastic and thermal strain tensors but no strain tensors. For more information and supported elements and materials, see Remark 10.
SIGFLG	Flag for including the stress tensor for shells and solids. EQ.1: Include (default), EQ.2: Exclude for shells, include for solids. EQ.3: Exclude for shells and solids.
EPSFLG	Flag for including the effective plastic strains for shells and solids: EQ.1: Include (default), EQ.2: Exclude for shells, include for solids. EQ.3: Exclude for shells and solids.
RLTFLG	Flag for including stress resultants in the shell LS-DYNA database: EQ.1: Include (default), EQ.2: Exclude.
ENGFLG	Flag for including shell, tshell, and beam internal energy density and shell thickness: EQ.1: Include (default), EQ.2: Exclude.

Card 2	1	2	3	4	5	6	7	8
Variable	CMPFLG	IEVERP	BEAMIP	DCOMP	SHGE	STSSZ	N3THDT	IALEMAT
Type	I	I	I	I	I	I	I	I
Default	0	0	0	1	1	1	2	1

VARIABLE	DESCRIPTION
CMPFLG	Flag to indicate the coordinate system for output of stress and strain of solids, shells, and thick shells comprised of orthotropic or

VARIABLE	DESCRIPTION
	anisotropic materials. CMPFLG affects d3plot, d3part, eloutdet, and elout, with exceptions as noted below. EQ.-1: Same as 1, but for *MAT_FABRIC (FORM = 14 or -14) and *MAT_FABRIC_MAP the stress and strain are in engineering quantities instead of Green-Lagrange strain and 2nd Piola-Kirchhoff stress. EQ.0: Global coordinate system with the exception of elout for shells (see EOCS in *CONTROL_OUTPUT). EQ.1: Local material coordinate system (as defined by AOPT and associated parameters in the *MAT input, and if applicable, by angles B1, B2, etc. in *SECTION_SHELL, *SECTION_TSHELL, or *PART_COMPOSITE, and by optional input in the *ELEMENT data). The effect of CMPFLG = 1 on shell output in elout is overridden by EOCS = 1 or 2 in *CONTROL_OUTPUT or by OPTION2 > 0 in *DATABASE_ELOUT. These overriding conditions do not apply to eloutdet.
IEVERP	Every output state for the d3plot database is written to a separate file. EQ.0: More than one state can be on each plot file, EQ.1: One state only on each plot file.
BEAMIP	Number of beam integration points for output. This option applies to integrated beams but also generates certain limited output for resultant beams. See Remark 2.
DCOMP	Data compression to eliminate rigid body data: EQ.1: Off (default), no rigid body data compression, EQ.2: On, rigid body data compression active, EQ.3: Off, no rigid body data compression, but all nodal velocities and accelerations are eliminated from the database. EQ.4: On, rigid body data compression active and all nodal velocities and accelerations are eliminated from the database. EQ.5: On, rigid body data compression active and rigid nodal data are eliminated from the database. Only 6 DOF rigid body motion is written. EQ.6: On, rigid body data compression active, rigid nodal data, and all nodal velocities and accelerations are eliminated

VARIABLE	DESCRIPTION
	from the database. Only 6 DOF rigid body motion is written.
SHGE	Flag for including shell hourglass energy density: EQ.1: Off (default), no hourglass energy written, EQ.2: On.
STSSZ	Flag for including shell element time step, mass, or added mass: EQ.1: Off (default), EQ.2: Output time step size, EQ.3: Output mass, added mass, or time step size. See Remark 3 below.
N3THDT	Flag for including material energy in d3thdt database: EQ.1: Off, energy is *not* written to d3thdt database, EQ.2: On (default), energy is written to d3thdt database.
IALEMAT	Output solid part ID list containing ALE materials. EQ.1: On (default)

Card 3	1	2	3	4	5	6	7	8
Variable	NINTSLD	PKP_SEN	SCLP	HYDRO	MSSCL	THERM	INTOUT	NODOUT
Type	I	I	F	I	I	I	A	A
Default	1	0	1.0	0	0	1 or 2	none	none

VARIABLE	DESCRIPTION
NINTSLD	Number of solid element integration points written to the LS-DYNA database. When NINTSLD is set to 1 (default) or to any value other than 8, integration point values are averaged and only those averages are written output. To obtain values for individual integration points, set NINTSLD to 8, even if the multi-integration point solid has fewer than 8 integration points.

VARIABLE	DESCRIPTION
PKP_SEN	Flag to output the peak pressure and surface energy (energy per unit area) computed by each contact interface into the interface force database. To obtain the surface energy, FRCENG, must be sent to 1 on the control contact card. When PKP_SEN = 1, it is possible to identify the energies generated on the upper and lower shell surfaces, which is important in metal forming applications. This data is mapped after each h-adaptive remeshing. EQ.0: No data is written. EQ.1: Output the peak pressures and surface energy by contact interface.
SCLP	Weighting or smoothing factor used to apply exponential smoothing to the calculation of peak pressure. It must be a value between 0.0 and 1.0. When set to a value less than 1.0, the calculation uses previous values of the pressure.
HYDRO	Either 3, 5 or 7 additional history variables useful to shock physics are output as the last history variables to d3plot (does not apply to elout). For HYDRO = 1, the internal energy per reference volume, the reference volume, and the pressure from bulk viscosity are added to the database; for HYDRO = 2, the relative volume and current density are also added; and for HYDRO = 4, volumetric strain (defined as relative volume – 1.0) and hourglass energy per unit initial volume are additionally added. These history variables are not valid for ALE elements.
MSSCL	Output nodal and part information related to mass scaling into the d3plot database. This option can be activated only if DT2MS < 0.0. See Remark 3. Also, see *CONTROL_TIMESTEP. MSSCL is interpreted digit-wise, MSSCL = [*ML*], $\text{MSSCL} = L + M \times 10$ *L* sets the nodal mass output: L.EQ.0: No incremental nodal mass data is written. L.EQ.1: Output incremental nodal mass. L.EQ.2: Output percentage increase in nodal mass. *M* determines what mass is output for each part into d3plot: M.EQ.0: Output original mass for each part.

VARIABLE	DESCRIPTION
	M.EQ.1: Output effective mass for each part. Effective mass is the original mass plus the incremental mass.
THERM	Output of thermal data (if applicable) to d3plot. The effect of the parameter is different in thermal-only analyses compared to coupled simulations. For coupled simulations: EQ.1: Output temperature (default). EQ.2: Output temperature and flux. EQ.3: Output temperature, flux, and the temperatures at the shell's bottom and top surfaces. For thermal-only simulations: EQ.1: Output temperature and flux. EQ.2: Output temperature and flux (default). EQ.3: Output temperature, flux, and the temperatures at the shell's bottom and top surfaces.
INTOUT	Output stress/strain at all integration points for detailed element output in the ASCII file eloutdet. DT and BINARY of *DATABASE_ELOUT apply to eloutdet. See Remark 4. EQ.STRESS: When stress output is required EQ.STRAIN: When strain output is required EQ.ALL: When both stress and strain output are required
NODOUT	Output extrapolated stress/strain at connectivity nodes for detailed element output in the ASCII file eloutdet. DT and BINARY of *DATABASE_ELOUT apply to eloutdet. See Remark 4. EQ.STRESS: When stress output is required EQ.STRAIN: When strain output is required EQ.ALL: When both stress and strain output are required EQ.STRESS_GL: When nodal averaged stress output along the global coordinate system is required EQ.STRAIN_GL: When nodal averaged strain output along the global coordinate system is required EQ.ALL_GL: For global nodal averaged stress and strain output

Card 4	1	2	3	4	5	6	7	8
Variable	DTDT	RESPLT	NEIPB	QUADSLD	CUBSLD	DELERES		
Type	I	I	I	I	I	I		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
DTDT	Output of node point Δtemperature/Δtime data to d3plot. EQ.0: No output (default) EQ.1: Output $\Delta T/\Delta t$
RESPLT	Output of translational and rotational residual forces to d3plot and d3iter. EQ.0: No output EQ.1: Output residual
NEIPB	Number of additional element or integration point history variables written to the binary databases (d3plot, d3part, d3drlf) for beam elements; see NEIPH above, BEAMIP and *DEFINE_MATERIAL_HISTORIES. See also Remark 12. For output of additional integration point history variables for beam elements to the elout database, see the variable OPTION4 in *DATABASE_ELOUT.
QUADSLD	(Under development) Output flag for quadratic solid types 24, 25, and 26: EQ.0: Average stress and strain and rendered with 8 nodes EQ.1: Average stress and strain and rendered with all edge and face nodes EQ.2: All integration points written and rendered with all edge and face nodes
CUBSLD	(Under development) Output flag for cubic solids types 27, 28, and 29: EQ.0: Average stress and strain and rendered with 8 nodes

VARIABLE	DESCRIPTION
	EQ.1: Average stress and strain and rendered with all edge and face nodes
	EQ.2: All integration points written and rendered with all edge and face nodes
DELERES	Output flag for results of deleted elements:
	EQ.0: No results output (all zero)
	EQ.1: Last available results, such as stresses and history variables, are written to d3plot and d3part.

Remarks:

1. **MAXINT field.** If MAXINT is set to 3, then mid-surface, inner-surface, and outer-surface stresses are output at the center of the element. For an even number of integration points, the points closest to the center are averaged to obtain the midsurface values. If multiple integration points are used in the shell plane, the stresses at the center of the element are found by computing the average of these points. For MAXINT equal to 3, LS-DYNA assumes that the data for the user-defined integration rules are ordered from bottom to top even if this is not the case. If MAXINT is not equal to 3, then the stresses at the center of the element are output in the order that they are stored for the selected integration rule. If multiple points are used in plane, the stresses are first averaged.

2. **BEAMIP field.** BEAMIP causes the output of additional data to d3plot and d3thdt for beam elements. For integrated beams, such as the Hughes-Liu beam (ELFORM = 1), the additional data consists of stresses and strains at the first BEAMIP integration points. The order of the output data matches the order of the integration points. The data at each integration point consists of the following five values for elastic-plastic Hughes-Liu beams: the normal stress, σ_{rr}; the transverse shear stresses, σ_{rs} and σ_{tr}; the effective plastic strain; and the axial strain which is logarithmic. For beams that are not elastic-plastic, the first history variable, if any, is output instead of the plastic strain. For the beam elements of Belytschko and his co-workers (ELFORM = 4 and 5), the transverse shear stress components are not used in the formulation.

   No data is output for Belytschko-Schwer resultant beams (ELFORM = 2), discrete beams (ELFORM = 6), or linear beams (ELFORM = 12), except under these conditions:

   a) For ELFORM = 2 and 13, beam elements have a linear distribution of moment along their length. The default moment data output to d3plot, d3dtdt,

and elout are the moments at the Node 1 end of the beams only. This restriction presents an incomplete picture of the moment distribution in the beam and may miss the maximum moment value. However, the moments at both ends of the beam can be obtained by setting BEAMIP greater than or equal to 3 (for ELFORM = 13, this is available only in releases starting from R16). In this case, the moments at both ends of the beam are written to the d3plot and d3thdt files in the positions where post-processors expect to find the following data components:

Integration point	Post-processing data component	Actual meaning
2	ZX shear stress	Moment about s-axis at Node 1 end
2	Plastic strain	Moment about s-axis at Node 2 end
2	Axial strain	Moment about t-axis at Node 1 end
3	Axial stress	Moment about t-axis at Node 2 end

b) Belytschko-Schwer beams (ELFORM = 2) with material types 191 and 209 automatically increase BEAMIP and write additional data specific to those material models as well as the moments at both ends of the beam. See Remarks under these material types.

c) Discrete beams (ELFORM = 6) with material types 205 and 208 automatically increase BEAMIP and write additional data specific to those material models. See Remarks under these material types.

3. **Mass scaling.** If mass scaling is active, the output of the time step size reveals little information about the calculation. If global mass scaling is used for a constant time step, the total element mass is output; however, if the mass is increased so that a minimum time step size is maintained (DT2MS is negative), the added mass is output. Also, see the control card *CONTROL_TIMESTEP.

4. **Estimated stress and strain output to eloutdet for nodes.**

a) When NODOUT is set to STRESS, STRAIN, or ALL. Each node of the element nodal connectivity will be output. See Example 1.

b) When NODOUT is set to STRESS_GL, STRAIN_GL, or ALL_GL. Averaged nodal results are calculated by:

$$\frac{1}{n}\sum_{i=1}^{n} c_{\mathrm{ip}(i)}$$

where n is the number of elements sharing the node and $c_{\mathrm{ip}(i)}$ is the contribution from the closest integration of element i to the node. Averaged

nodal values are always output in the global coordinate system. See Example 2.

5. **Contents of eloutdet.** Available stress/strain components in eloutdet stress components include 6 stress components (sig-*xx*, sig-*yy*, sig-*zz*, sig-*xy*, sig-*yz*, sig-*zx*), yielding status, and effective plastic strain. Strain components include 6 strain components.

6. **Shell element output at integration points.** Stresses at all integration points can be output. The strain at the top and bottom integration layer can be output. At a connective node the extrapolated stress and strain at the top and bottom layer can be output.

7. **Thick shells.** Thick shell element output includes the six stress components at each integration point. Strain at the top and bottom layers can be output. At the element node, values at the bottom layer are extrapolated to yield the values of nodes 1 - 4, and values at the top layer are extrapolated to yield values of nodes 5 - 8.

8. **Integration point locations.** Stresses and strain at all integration points can be output. The integration point order is as follows:

   a) point #1 is the point closest to node #1 in the connectivity array

   b) point #2 is the closest point to node #2, etc

   c) For tetrahedral types 4, 16, and 17 with 5 integration points, point #5 is the midpoint.

   d) For the nodal points, values at the integration points are extrapolated.

9. **Reporting residual forces and moments.** The output of residual forces and moments is supported for implicit and double precision only. With this option, the forces and moments appear under the *Ndv* button in the fringe menu in LS-PrePost. The residual for rigid bodies is distributed to the constrained nodes for the body without scaling for the purpose of capturing the complete residual vector.

10. **Calculation of strains (STRFLG).** The strain tensors $\boldsymbol{\varepsilon}$ that are output to the d3plot database are calculated using proper time integration of the rate-of-deformation tensor $\mathbf{D}$. More specifically, to assert the objectivity of the resulting strain, it uses the Jaumann rate of strain for solids, whereas for shells it uses the co-rotational strain rate. In mathematical terms, the integration uses the following strain rates

$$\dot{\boldsymbol{\varepsilon}} = \mathbf{D} - \boldsymbol{\varepsilon}\mathbf{W} + \mathbf{W}\boldsymbol{\varepsilon} \quad \text{(solids)}$$

$$\dot{\boldsymbol{\varepsilon}} = \mathbf{D} - \boldsymbol{\varepsilon}\boldsymbol{\Omega} + \boldsymbol{\Omega}\boldsymbol{\varepsilon} \quad \text{(shells)}$$

where $\mathbf{W}$ is the spin tensor and $\mathbf{\Omega} = \dot{\mathbf{Q}}\mathbf{Q}^T$ is the rotational velocity of the co-rotational system $\mathbf{Q}$ used for the shell element in question, taking into account invariant node numbering and such. This is to say that the resulting strains would be equal to the Cauchy stress for a hypo-elastic material (*MAT_ELASTIC) with a Young's modulus of 1 and a Poisson's ratio of 0. This should be considered when interpreting the results since they are not invariant to changes in element formulations and possibly nodal connectivities.

11. **Plastic and thermal strain (STRFLG).** The algorithm for writing plastic and thermal strains, which is also activated using STRFLG, is a modification of the algorithm used for mechanical strains (see Remark 10).

    a) For solids, the element average strain in the global system, which has 6 components, is written (local system if CMPFLG is set).

    b) For shells, both plastic and thermal strains have 6 components. The thermal strain is written as a single tensor, as in the solid case. The plastic strain output consists of 3 plane-averaged tensors: one for the bottom, one for the middle, and one for the top. For an even number of through-thickness integration points, the middle is taken to be the average of the two integration points closest to the mid-surface. Only the following element/material combinations are currently supported, but others *will* be added upon request.

Thermal strain tensors			*Plastic strain tensors*		
Shells	Solids	Materials	Shells	Solids	Materials
2, 16, 23	1, 2, 10, 13, 15	Add thermal expansion, 255	2, 16, 23	1, 2, 10, 13, 15	3, 10, 15, 18, 24, 81, 98, 103, 106, 107, 123, 124, 187, 199, 224, 251, 255

12. **History variables for beams (NEIPB).** In general, NEIPB follows the same conventions as NEIPH and NEIPS do for solid and shell elements and is supported in LS-PrePost v4.3 or later. Average, min, and max values for each element are output, including data for resultant elements. If BEAMIP is nonzero, then element data is complemented with BEAMIP integration point values that can be examined individually. Beam history data is post-processed similarly to that of solid and shell element history data.

Example 1:

Excerpt from eloutdet file for a shell element with two through-thickness integration points and four in-plane integration points, with INTOUT = STRESS and NODOUT = STRESS:

```
element  materl
     ipt  stress  sig-xx   sig-yy   sig-zz   sig-xy   sig0yz   sig-zx   yield         location
       1-      1
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  1
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  2
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  3
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  4
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       21
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       22
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       20
   1- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       19
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  1
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  2
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  3
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  int.  point  4
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       21
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       22
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       20
   2- 10 elastic  4.41E-2  2.51E-1  0.00E+0  7.76E-8  0.00E+0  0.00E+0  0.00E+0  node       19
```

Example 2:

Excerpt from eloutdet file for averaged nodal strain:

```
nodal strain calculations for time step   24  (at time 9.89479E+01 )
 node (global)
          strain   eps-xx     eps-yy         eps-zz   eps-xy       eps-yz      eps-zx
          1-
 lower surface     2.0262E-01 -2.6058E-02 -7.5669E-02 -5.1945E-03  0.0000E+00  0.0000E+00
 upper surface     2.0262E-01 -2.6058E-02 -7.5669E-02 -5.1945E-03  0.0000E+00  0.0000E+00
          2-
 lower surface     1.9347E-01  2.3728E-04 -8.3019E-02 -1.4484E-02  0.0000E+00  0.0000E+00
 upper surface     1.9347E-01  2.3728E-04 -8.3019E-02 -1.4484E-02  0.0000E+00  0.0000E+00
          3-
 lower surface     2.0541E-01 -5.7521E-02 -6.3383E-02 -1.7668E-03  0.0000E+00  0.0000E+00
 upper surface     2.0541E-01 -5.7521E-02 -6.3383E-02 -1.7668E-03  0.0000E+00  0.0000E+00
          ⋮           ⋮          ⋮           ⋮          ⋮           ⋮           ⋮
```

*DATABASE_EXTENT_D3PART

The following cards control content to the d3part binary database (Card 3 is optional). The parameters given here will supercede the corresponding parameters in *DATABASE_EXTENT_BINARY when writing the d3part binary database. See also *DATABASE_BINARY_D3PART which defines the output interval for d3part and the set of parts included in d3part.

Card 1	1	2	3	4	5	6	7	8
Variable	NEIPH	NEIPS	MAXINT	STRFLG	SIGFLG	EPSFLG	RLTFLG	ENGFLG
Type	I	I	I	I	I	I	I	I
Default	0	0	3	0	1	1	1	1

Card 2	1	2	3	4	5	6	7	8
Variable		IEVERP			SHGE	STSSZ		
Type		I			I	I		
Default		0			0	0		

Card 3	1	2	3	4	5	6	7	8
Variable	NINTSLD							
Type	I							
Default	1							

VARIABLE	DESCRIPTION
NEIPH	Number of additional integration point history variables written to the binary database for solid elements. The integration point data is written in the same order that it is stored in memory; each material model has its own history variables that are stored. For user

VARIABLE	DESCRIPTION
	defined materials the history data that is important for plotting should be stored before the data that is not of interest.
NEIPS	Number of additional integration point history variables written to the binary database for both shell and thick shell elements for each integration point; see NEIPH above.
MAXINT	Number of shell integration points written to the binary database; see also *INTEGRATION_SHELL. See Remark 1.
STRFLG	Set to 1 to dump strain tensors for solid, shell and thick shell elements for plotting by LS-PrePost and ASCII file elout. For shell and thick shell elements two tensors are written, one at the innermost and one at the outermost integration point. For solid elements a single strain tensor is written.
SIGFLG	Flag for including the stress tensor for shells: EQ.1: Include (default), EQ.2: Exclude.
EPSFLG	Flag for including the effective plastic strains for shells: EQ.1: Include (default), EQ.2: Exclude.
RLTFLG	Flag for including stress resultants for shells: EQ.1: Include (default), EQ.2: Exclude.
ENGFLG	Flag for including shell internal energy density and shell thickness: EQ.1: Include (default), EQ.2: Exclude.
IEVERP	Every plot state for the d3part database is written to a separate file. This option will limit the database to 1000 states: EQ.0: More than one state can be on each plot file, EQ.1: One state only on each plot file.
SHGE	Flag for including shell hourglass energy density: EQ.1: Off (default), no hourglass energy written,

VARIABLE	DESCRIPTION
	EQ.2: On.
STSSZ	Flag for including shell element time step, mass, or added mass: EQ.1: Off (default), EQ.2: Output time step size, EQ.3: Output mass, added mass, or time step size. See Remark 2 below.
NINTSLD	Number of solid element integration points written. The default value is 1. For solids with multiple integration points NINTSLD may be set to 8. Currently, no other values for NINTSLD are allowed. For solids with multiple integration points, an average value is output if NINTSLD is set to 1.

Remarks:

1. **MAXINT.** If the default value of 3 is used, then results are output for the outermost (top) and innermost (bottom) integration points together with results for the neutral axis. If MAXINT is set to 3 and the element has 1 integration point, then all three results will be the same. If a value other than 3 is used, then results for the first MAXINT integration points in the element will be output. Note that if the element has an even number of integration points and MAXINT is not set to 3, then you will not get mid-surface results. If MAXINT is set to a negative number, MAXINT integration points are output for each in-plane integration point location and no averaging is used. This can greatly increase the size of the binary d3part database.

2. **Mass scaling.** If mass scaling is active, the output of the time step size reveals little information about the calculation. If global mass scaling is used for a constant time step, the total element mass is output; however, if the mass is increased so that a minimum time step size is maintained (DT2MS is negative), the added mass is output. Also, see the control card *CONTROL_TIMESTEP.

*DATABASE_EXTENT_INTFOR

Purpose: The following card controls to some extent the content of the optional intfor binary database. See also *DATABASE_BINARY_INTFOR. The intfor database contains geometry and time history data pertaining to those contact surfaces which are flagged in *CONTACT with the variables SAPR and/or SBPR. The name of the intfor database may be given either on the execution line using "s=**filename**", or using the option FILE on *DATABASE_BINARY_INTFOR.

Card 1	1	2	3	4	5	6	7	8
Variable	NGLBV	NVELO	NPRESU	NSHEAR	NFORC	NGAPC	NFAIL	IEVERF
Type	I	I	I	I	I	I	I	I
Default	1	1	1	1	1	1	0	0

Optional Card.

Card 2	1	2	3	4	5	6	7	8
Variable	NWEAR	NWUSR	NHUF	NTIED	NENG	NPEN		
Type	I	I	I	I	I	I		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
NGLBV	Output global variables: EQ.-1: No, EQ.1: Yes (default).
NVELO	Output nodal velocity: EQ.-1: No, EQ.1: Yes (default).
NPRESU	Output pressures: EQ.-1: No,

VARIABLE	DESCRIPTION
	EQ.1: Normal interface pressure (default), EQ.2: Normal interface pressure and peak pressure, EQ.3: Normal interface pressure, peak pressure and time to peak.
NSHEAR	Output shear stresses: EQ.-1: No, EQ.1: Shear stress in *r*-direction and *s*-direction (default).
NFORC	Output forces: EQ.-1: No, EQ.1: *x*-, *y*-, *z*-force at all nodes (default).
NGAPC	Output contact gaps at all nodes and surface energy density. For Mortar contact, see Remark 1. EQ.-1: No, EQ.1: Yes (default).
NFAIL	Flag for display of deleted contact segments: EQ.0: All segments are displayed (default). EQ.1: Remove deleted contact segments from display.
IEVERF	Every interface force state for the intfor database is written to a separate file: EQ.0: More than one interface force state can be on each intfor file (default). EQ.1: One interface force output state only on each intfor file.
NWEAR	Output contact wear data; see *CONTACT_ADD_WEAR. EQ.0: No output (default). GE.1: Output wear depth. GE.2: Output sliding distance.
NWUSR	Number of user wear history variables to output from user defined wear routines; see *CONTACT_ADD_WEAR. See Remark 2.

VARIABLE	DESCRIPTION
NHUF	Number of user friction history variables to output from user defined friction routines; see *USER_INTERFACE_FRICTION (MPP only). See Remark 2.
NTIED	Output tied segments for Mortar contact. See Remark 3. EQ.0: No output EQ.1: Output
NENG	Output contact energy density for Mortar contact and SOFT = 2 contact (see also ENGOUT on *CONTROL_OUTPUT): EQ.0: No output EQ.1: Output
NPEN	Output penetration information for sliding contacts (non-TIED or TIEBREAK) when SOFT ≠ 2. A nodal field gives the penetration for each node (magnitude and direction) in the sliding interface; see also PENOUT on *CONTROL_OUTPUT. EQ.0: No output EQ.1: Output absolute penetration EQ.2: Output absolute and relative penetrations. Relative penetration is output as a percentage of the penetration at which the contact is released.

Remarks:

1. **Gaps in mortar contact.** Gaps in Mortar contact are measured with respect to the *nominal* contact surfaces of the two interacting segments. For instance, if IGNORE = 2 on *CONTACT_...MORTAR, then an initial penetration d will dislocate the tracked contact surface in the negative direction of the tracked surface normal $\mathbf{n}$. If NGAPC = 1, the gap g reported to the intfor file is still measured between the tracked and reference surface, neglecting this dislocation; thus, only *physical* gaps are reported.

2. **Wear outputs.** Wear outputs are governed by NWEAR and NWUSR and require the use of a wear model associated with the contact interface. For NWEAR the "wear depth" (NWEAR ≥ 1) and "sliding distance" (NWEAR ≥ 2) are listed under the Nodal fringe menu in LS-PrePost. Following this, NWUSR user defined history variables are listed, corresponding to user wear history variables

in a user wear routine. These are listed in the order that they are stored in the wear routine; see WTYPE ≤ 0 on *CONTACT_ADD_WEAR.

3. **Mortar tied contacts.** For all Mortar tied contacts, that is,

   *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED,

   *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR_TIED_WELD, and

   *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK_MORTAR.

   NTIED will activate the output of an indicator stating whether a tracked segment is tied (= 1) or not (= 0); nothing is shown on the reference side. When opening the intfor file in LS-PrePost, the fringe menu will have two entries titled "tied at top" and "tied at bottom," corresponding to the top and bottom of a shell segment, respectively. The top and bottom labels are with respect to the shell normal. For obvious reasons, solid segments will only have nonzero values for the "tied at top" field. For tiebreak contact, these variables represent the relative adhesive strength in the contact, that is, $1 - D$, where D is the damage, so fully damaged segments will show 0 and undamaged segments will show 1. For the tied weld contact the entries will give the segments that have been welded to the corresponding reference side of the contact and can thus have only 0 (not welded) or 1 (welded).

*DATABASE_EXTENT_MOVIE

Purpose: Control the content written to the MOVIE databases. See movie option on *DATABASE manual entry.

Variable Cards. Define as many cards as needed. Input ends at next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	VTYPE	COMP						
Type	I	I						

VARIABLE	DESCRIPTION
VTYPE	Variable type: EQ.0: node, EQ.1: brick, EQ.2: beam, EQ.3: shell, EQ.4: thick shell.
COMP	Component ID. For the corresponding VTYPE, integer components from the following tables can be chosen: VTYPE.EQ.0: Table 16-1 (see DATABASE_EXTENT_AVS), VTYPE.EQ.1: Table 16-2 (see DATABASE_EXTENT_AVS), VTYPE.EQ.2: not supported, VTYPE.EQ.3: Table 16-3 (see DATABASE_EXTENT_AVS), VTYPE.EQ.4: not supported.

*DATABASE_EXTENT_MPGS

Purpose: Control the content written to the MPGS databases. The created MPGS databases consist of a geometry file and one file for each output database. See MPGS option to *DATABASE keyword.

Variable Cards. Define as many cards as needed. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	VTYPE	COMP						
Type	I	I						

VARIABLE	DESCRIPTION
VTYPE	Variable type: EQ.0: node, EQ.1: brick, EQ.2: beam, EQ.3: shell, EQ.4: thick shell.
COMP	Component ID. For the corresponding VTYPE, integer components from the following tables can be chosen: VTYPE.EQ.0: Table 16-1 (see DATABASE_EXTENT_AVS), VTYPE.EQ.1: Table 16-2 (see DATABASE_EXTENT_AVS), VTYPE.EQ.2: not supported, VTYPE.EQ.3: Table 16-3 (see DATABASE_EXTENT_AVS), VTYPE.EQ.4: not supported.

DATABASE_EXTENT_SSSTAT_*OPTION

The only *OPTION* is:

ID

The ID option allows the definition of a heading which will be written at the beginning of the ASCII file ssstat.

Purpose: This command defines one or more subsystems. A subsystem is simply a set of parts, grouped for convenience. The ASCII output file ssstat provides histories of energy (kinetic, internal, hourglass) and momentum (x, y, and z) for each subsystem. The ssstat file is thus similar to glstat and matsum. But whereas glstat provides data for the whole model and matsum provides data for each individual part, ssstat provides data for each subsystem. The output interval for the ssstat file is given using *DATABASE_SSSTAT. To also include histories of subsystem mass properties in the ssstat file, use *DATABASE_SSSTAT_MASS_PROPERTIES.

For *DATABASE_EXTENT_SSSTAT without the ID option, the following card(s) applies. Define as many cards as necessary. Define one part set ID per subsystem, up to 8 subsystems per card.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID1	PSID2	PSID3	PSID4	PSID5	PSID6	PSID7	PSID8
Type	I	I	I	I	I	I	I	I

For *DATABASE_EXTENT_SSSTAT_ID, the following card(s) applies. Define as many cards as necessary. Define one part set ID per subsystem, 1 subsystem per card.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID1	HEADING1						
Type	I	A70						

VARIABLE	DESCRIPTION
PSID*n*	Part set ID for subsystem *n*; see *SET_PART.
HEADING*n*	Heading for subsystem *n*.

*DATABASE_FATIGUE_RAINFLOW_MATRIX

Purpose: Activate writing binary lsda database fatigue.rainflow. This file contains the rainflow counting information for stress cycles at integration points of elements. The elements are selected through set IDs.

Card 1	1	2	3	4	5	6	7	8
Variable	BINARY	SOLID	BEAM	SHELL	TSHELL			
Type	I	I	I	I	I			
Default	0	0	0	0	0			

VARIABLE	DESCRIPTION
BINARY	Flag for writing out the fatigue.rainflow lsda file: EQ.0: No fatigue.rainflow output. EQ.1: Write out fatigue.rainflow lsda file.
SOLID	Solid set ID
BEAM	Beam set ID
SHELL	Shell set ID
TSHELL	Thick shell set ID

Remarks:

1. **Output information.** This lsda database contains the rainflow cycle counting information (e.g., stress range, mean stress, and number of stress cycles), which can be plotted with LS-PrePost. This rainflow matrix plot can be helpful to see which type of cycles dominate and to compare different load histories.

*DATABASE_FATIGUE_STRESS_CYCLE

Purpose: Activate outputting number of stress cycles versus stress range in random vibration fatigue analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	BINARY							
Type	I							
Default	1							

VARIABLE	DESCRIPTION
BINARY	Flag for outputting ASCII files providing the number of stress cycles versus stress range: EQ.0: Off EQ.1: Output stress_cycle_[EID] ASCII files.

Remarks:

1. **Stress cycle versus stress range.** This keyword instructs LS-DYNA to write out ASCII files stress_cycle_[EID] which record the number of stress cycles versus stress range in random vibration fatigue analysis. EID is the element ID. The elements included in this output are defined by keywords *DATABASE_HISTORY_SOLID / SHELL / TSHELL_*{SET}*. The stress_cycle_[EID] file is in xyplot curve format and can be plotted using LS-PrePost.

.

*DATABASE_FATXML

Purpose: Process FATXML data. FATXML is an open, standardized data format based on the Extensible Markup Language (XML). It is developed by the German Research Association of Automotive Technology (Forschungsvereinigung Automobiltechnik - FAT). FATXML is designed for consistent data management in the overall CAE process chain. Schulte-Frankenfeld and Deiters [2016] give a comprehensive explanation of the FATXML data format specification.

LS-DYNA reads all lines between this keyword and the next keyword recognized by the star (*) sign, processes the data with respect to the include file structure and writes everything together in one output file called d3plot.xml.

Remarks:

We intend for a corresponding FE model to consist of one primary file with several associated include files. Each include file should contain a description with *DATABASE_FATXML data, usually at the end of the file. The primary file loads the include files via *INCLUDE_TRANSFORM with potential offset values for nodes, elements, parts, etc. (IDNOFF, IDEOFF, IDPOFF, …). All data from different include files with different offsets are collected and then summarized in d3plot.xml. Since the resulting data format is public domain, post-processors can read that data and correlate it with the associated CAE model.

Example:

```
...
*DATABASE_FATXML
<?xml version="1.0"?>
<CAE_META_DATA>
  < PART_ID NAME="TestCase">
    <PDM_DATA>
      ...
      <PDD_THICKNESS>
        < THICKNESS ID="123">1.0</THICKNESS >
        < THICKNESS ID="124">1.1</THICKNESS >
        ...
      </PDD_THICKNESS >
      ...
    </PDM_DATA >
  </PART_ID >
</CAE_META_DATA >
*END
```

DATABASE_FORCE_*OPTION

Available options include:

BEAM

BEAM_ID

BEAM_SET

SHELL

SHELL_ID

SHELL_SET

SOLID

SOLID_ID

SOLID_SET

TSHELL

TSHELL_ID

TSHELL_SET

Purpose: Control which elements have data output to the binary history directory elout_frc.

Card Summary:

Card 1a. Include this card if the keyword option is BEAM, BEAM_SET, SHELL, SHELL_SET, SOLID, SOLID_SET, TSHELL, or TSHELL_SET. Include as many as needed. This input terminates at the next keyword ("*") card.

ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8

Card 1b. Include this card if the keyword option is BEAM_ID, SHELL_ID, SOLID_ID, and TSHELL_ID. Include as many cards as needed. This input terminates at the next keyword ("*") card.

ID	HEADING

Data Card Definitions:

Element Cards for Case I (no "ID"). Cards for keyword options BEAM, BEAM_SET, SHELL, SHELL_SET, SOLID, SOLID_SET, TSHELL, and TSHELL_SET. Include as many as needed. This input terminates at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
ID*n*	Element/element set ID *n*. The elements may be beams, shells, solids, or thick shells depending on the keyword option.

Element Cards for Case II ("ID" option). Cards for keyword options BEAM_ID, SHELL_ID, SOLID_ID, and TSHELL_ID. Include as many as needed. This input terminates at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Element ID
HEADING	A description of the element. We suggest using unique descriptions.

Remarks:

1. **Forces extracted by this feature.** This feature extracts the internal forces from individual elements. These internal forces can then be used, for instance, to conduct fatigue analysis. Thus, its output differs from other nodal force outputs (such as *DATABASE_NODAL_FORCE_GROUP), which provide the resultant nodal dynamic forces and can therefore not be used as an indicator of how the surrounding elements are individually contributing to the force. The output from this feature isolates the contribution from the material. Note that using

stress as a basis for fatigue analysis is an equivalent alternative, and the chosen method for fatigue analysis a matter of preference.

*DATABASE_FORMAT

Purpose: Define the output format for binary files.

Card 1	1	2	3	4	5	6	7	8
Variable	IFORM	IBINARY						
Type	I	I						
Default	0	0						
Remarks	1	2						

VARIABLE	DESCRIPTION
IFORM	Output format for d3plot and d3thdt files: EQ.0: LS-DYNA database format (default), EQ.1: Ansys database format (generally disabled since R12; see Remark 1), EQ.2: Both LS-DYNA and Ansys database formats (generally disabled since R12; see Remark 1).
IBINARY	Flag to control the word size in the binary output files, such as d3plot and d3drlf. This variable applies only to double precision LS-DYNA executables. EQ.0: 64 bit format for binary output (default for double precision), EQ.1: 32 bit IEEE format for binary output. This reduces the volume of binary output from double precision executables by a factor of two.

Remarks:

1. **Availability restrictions.** The Ansys output option is not available in MPP and is not universally available in SMP. The LS-DYNA banner in d3hsp includes "ANSYS database format" under the list of "Features enabled" if the option is available. By default, our most recent executables do not have this feature enabled.

2. **32 bit IEEE format.** As an alternative to setting IBINARY = 1, the user may set the system environment variable LSTC_BINARY to 32ieee.

DATABASE_FREQUENCY_ASCII_*OPTION1_{OPTION2}

OPTION1 specifies one of the following frequency domain ASCII databases:

NODOUT_SSD ASCII database for nodal results for SSD (displacement, velocity, and acceleration). See also *FREQUENCY_DOMAIN_SSD.

ELOUT_SSD ASCII database for element results for SSD (stress and strain components). See also *FREQUENCY_DOMAIN_SSD.

NODFOR_SSD ASCII database for nodal force group for SSD (nodal forces for each node and reaction forces for the group). See also *FREQUENCY_DOMAIN_SSD.

SECFORC_SSD ASCII database for the cross-sectional force for SSD. See also *FREQUENCY_DOMAIN_SSD. See Remark 5.

NODOUT_PSD ASCII database for nodal PSD results (displacement, velocity, and acceleration). See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION.

ELOUT_PSD ASCII database for element PSD results (stress and strain components). See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION.

OPTION2:

MODAL_CONTRIBUTION (see Remark 7)

SUBCASE (see Remark 8)

<BLANK>

Purpose: Define output frequencies for the specified ASCII databases above. The frequencies can be different from the output frequencies for the binary databases D3SSD and D3PSD, which are defined by the keywords *DATABASE_FREQUENCY_BINARY_D3SSD and *DATABASE_FREQUENCY_BINARY_D3PSD, respectively.

Card 1	1	2	3	4	5	6	7	8
Variable	FMIN	FMAX	NFREQ	FSPACE	LCFREQ			
Type	F	F	I	I	I			
Default	0.0	0.0	0	0	0			

VARIABLE	DESCRIPTION
FMIN	Minimum frequency for output (cycles/time)
FMAX	Maximum frequency for output (cycles/time)
NFREQ	Number of frequencies for output
FSPACE	Frequency spacing option for output: EQ.0: Linear EQ.1: Logarithmic EQ.2: Biased EQ.3: Eigenfrequencies only
LCFREQ	Load curve ID defining the frequencies for output

Remarks:

1. **Output files.** The ASCII databases NODOUT_SSD, ELOUT_SSD, NODFOR_-SSD, NODOUT_PSD, and ELOUT_PSD are saved in binout files. LS-PrePost can read the binout files directly. The files can be converted to the ASCII format by feeding them to the l2a program like this:

   l2a binout*

2. **Nodal data.** The nodes to be output to NODOUT_SSD and NODOUT_PSD databases are specified by card *DATABASE_HISTORY_NODE.

3. **Element data.** The solid, beam, shell, and thick shell elements to be output to ELOUT_SSD and ELOUT_PSD are specified by the following cards:

   DATABASE_HISTORY_SOLID_*{OPTION}

   DATABASE_HISTORY_BEAM_*{OPTION}

   DATABASE_HISTORY_SHELL_*{OPTION}

   DATABASE_HISTORY_TSHELL_*{OPTION}

4. **Nodal set data.** The nodal sets to be output to NODFOR_SSD are specified by card *DATABASE_NODAL_FORCE_GROUP.

5. **Cross-sectional plane data.** Use *DATABASE_CROSS_SECTION_PLANE or *DATABASE_CROSS_SECTION_SET to specify the cross-sectional plane(s) output to SECFORC_SSD.

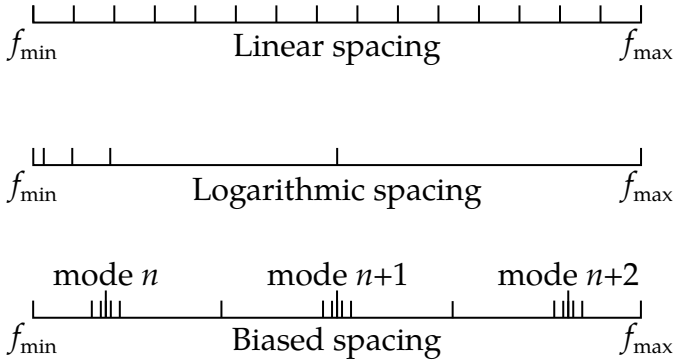


Figure 16-3. Spacing options of the frequency points.

6. **Output frequencies.** There are two methods to define the output frequencies.

   a) The first method is to define FMIN, FMAX, NFREQ, and FSPACE. FMIN and FMAX specify the frequency range of interest, and NFREQ specifies the number of frequencies at which results are required. FSPACE specifies the type of frequency spacing (linear, logarithmic, biased, or eigenfrequencies only) to be used. These frequency points for which results are required can be spaced equally along the frequency axis (on a linear or logarithmic scale), be biased toward the eigenfrequencies (the frequency points are placed closer together at eigenfrequencies in the frequency range) to obtain the detailed definition of the response close to resonance frequencies, or be just the eigenfrequencies in the frequency range defined by FMIN and FMAX. See Figure 16-3.

   b) The second method involves using a load curve (LCFREQ) to define the frequencies of interest.

7. **MODAL_CONTRIBUTION.** When the keyword option MODAL_CONTRIBUTION is present, LS-DYNA also computes the modal contribution fraction for each of the involved vibration modes. The modal contribution fraction shows the percentage of the response contributed by one particular mode over the total response. The modal contribution fraction values are saved in binout and can be processed using LS-PrePost. This keyword option is only valid for the SSD computation, that is, OPTION1 is NODOUT_SSD or ELOUT_SSD.

8. **SUBCASE.** The option SUBCASE only works when the keyword *FREQUENCY_DOMAIN_SSD_SUBCASE is present. It can only be used with *OPTION1* set to NODOUT_SSD, ELOUT_SSD, or NODFOR_SSD. With SUBCASE, Card 1 can be repeated to define individual output frequencies for each loading case defined in *FREQUENCY_DOMAIN_SSD_SUBCASE.

***DATABASE_FREQUENCY_BINARY**_*OPTION1_{OPTION2}*

Purpose: Options for frequency domain binary output files.

OPTION1 specifies one of the following output file names:

D3ACC Binary output file for BEM acoustics (element acoustic pressure contribution and contribution percentage). See also *FREQUENCY_DOMAIN_ACOUSTIC_BEM.

D3ACS Binary output file for FEM acoustics (acoustic pressure and sound pressure level in the acoustic volume) and BEM (collocation BEM) acoustics (acoustic pressure, sound pressure level, and normal velocity on the surface of the acoustic volume). See also *FREQUENCY_DOMAIN_ACOUSTIC_FEM and *FREQUENCY_DOMAIN_ACOUSTIC_BEM.

D3ATV Binary output file for acoustic transfer vectors given by BEM acoustic analysis. See also *FREQUENCY_DOMAIN_ACOUSTIC_BEM_ATV.

D3ERP Binary output file for ERP (Equivalent Radiated Power) density and some other data (e.g., real and imaginary parts of normal velocity at surface nodes). See also *FREQUENCY_DOMAIN_SSD_ERP.

D3FTG Binary output file for random vibration fatigue analysis. See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION_FATIGUE.

D3PSD Binary Power Spectral Density output file for random vibration analysis. See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION.

D3RMS Binary Root Mean Square output file for random vibration analysis. See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION.

D3SPCM Binary output file for response spectrum analysis. See also *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM.

D3SSD Binary output file for steady-state dynamics. See also *FREQUENCY_DOMAIN_SSD.

D3ZCF Binary Zero-Crossing Frequency output file for random vibration analysis. See also *FREQUENCY_DOMAIN_RANDOM_VIBRATION.

OPTION2 takes one of the following options:

<BLANK>

SUMMATION

SUBCASE

SUMMATION only works with *OPTION1* set to either D3PSD or D3RMS. With SUMMATION, the existing PSD and RMS databases from multiple random vibration analyses will be summed up to provide the total response of the same model subjected to multiple loading sources.

SUBCASE only works with D3SSD. With SUBCASE, an individual set of output frequencies is specified for each loading case included in *FREQUENCY_DOMAIN_SSD_SUBCASE.

The D3ACC, D3ACS, D3ATV, D3ERP, D3FTG, D3PSD, D3RMS, D3SPCM, D3SSD, and D3ZCF files contain plotting information to plot data over the three-dimensional geometry of the model. LS-PrePost can plot data from these databases.

- The D3ACC file contains each boundary element's acoustic pressure contribution (and contribution percentage) for a range of frequencies defined in *FREQUENCY_DOMAIN_ACOUSTIC_BEM.
- The D3ACS file contains acoustic results, including acoustic pressure and sound pressure level, for a range of frequencies defined in *FREQUENCY_DOMAIN_ACOUSTIC_FEM. Alternatively, it can be used to plot acoustic pressure, sound pressure level, and normal velocity on the surface of an acoustic volume for a range of frequencies defined in *FREQUENCY_DOMAIN_ACOUSTIC_BEM. Please note that only collocation BEM (METHOD = 3 or 4 in *FREQUENCY_DOMAIN_ACOUSTIC_BEM) can output the acoustic pressure, sound pressure level, and normal velocity on the surface of the acoustic volume to D3ACS.
- The D3ATV file contains NFIELD × NFREQ states, where NFIELD is the number of acoustic field points, and NFREQ is the number of output frequencies.
- The D3ERP file contains state data for a range of frequencies. These frequencies are the same as those used in D3SSD output.
- The D3FTG, D3RMS, and D3ZCF files contain only one state each as they are the data for cumulative fatigue damage ratio, root mean square of response for random vibration, peak response for response spectrum analysis, and zero-crossing frequency of response for random vibration separately.
- The D3PSD file contains PSD state data for a range of frequencies.
- When BINARY = 1, the D3SPCM file contains only one state: peak response for response spectrum analysis. When BINARY = 2, D3SPCM contains multiple states. The first state is peak response, like for BINARY = 1. The other states are the individual mode response sequentially for the normal modes involved in the mode combination.
- The D3SSD file contains state data for a range of frequencies. The data can be real or complex, depending on the variable BINARY defined below.

Card Summary:

Card 1a. Include this card if *OPTION2* is not used or *OPTION2* is SUBCASE.

BINARY				PSETID			

Card 1b. Include this card if *OPTION2* is SUMMATION. This card can be repeated if multiple binary databases exist.

FILENAME

Card 2a. Include this card if *OPTION1* is set to D3ACC. Include as many of this card as needed. Up to 10 NIDs are allowed.

NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8

Card 2b. Include this card if *OPTION1* is set to D3PSD or D3SSD. For D3SSD, when *OPTION2* is set to SUBCASE, this card can be repeated to define the output frequencies for each of the loading cases included in *FREQUENCY_DOMAIN_SSD_SUBCASE (in the same order).

FMIN	FMAX	NFREQ	FSPACE	LCFREQ			

Card 2c. Include this card if *OPTION1* is set to D3SPCM or D3RMS and BINARY = 3.

ISTATE	FILENAME

Data Card Definitions:

Include this card if *OPTION2* is unset or *OPTION2* is set to SUBCASE.

Card 1a	1	2	3	4	5	6	7	8
Variable	BINARY	SF			PSETID			
Type	I	F			I			
Default	0	3.0			0			

VARIABLE	DESCRIPTION
BINARY	Flag for writing the binary plot file. See Remark 1. EQ.0: Off

VARIABLE	DESCRIPTION
	EQ.1: Write the binary plot file. EQ.2: Write the complex variable binary plot file D3SSD (*OPTION1* = D3SSD) or include the individual mode response in the binary plot file D3SPCM (*OPTION1* = D3SPCM). EQ.3: Write the binary plot file, which combines response spectrum analysis or RMS results, depending on the choice of *OPTION1*, with other structural analysis results provided by the file specified with Card 2c It is available for *OPTION1* = D3SPCM or D3RMS. See Remarks 4 and 5. EQ.90: Write only the real part of the frequency response (D3SSD only). EQ.91: Write only the imaginary part of the frequency response (D3SSD only).
SF	Scale factor on the RMS response when using *OPTION1* = D3RMS with BINARY = 3. See Remark 5.
PSETID	Part set ID for the parts to be included in the database. If not defined, all the parts are included in the database. Currently it works for D3SSD only.

Summation Card. Card 1 when *OPTION1* = D3PSD or D3RMS and *OPTION2* = SUMMATION. This card can be repeated if multiple binary databases exist.

Card 1b	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	Path and file name of precomputed PSD or RMS binary databases (see Remark 3)

D3ACC Card. Additional card for D3ACC keyword option. Include as many of this card as needed. Up to 10 NIDs are allowed.

Card 2a	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
NID*i*	Field point node ID for writing D3ACC file

D3PSD and D3SSD Card. Additional card for D3PSD and D3SSD keyword options.

Card 2b	1	2	3	4	5	6	7	8
Variable	FMIN	FMAX	NFREQ	FSPACE	LCFREQ			
Type	F	F	I	I	I			
Default	0.0	0.0	0	0	0			

VARIABLE	DESCRIPTION
FMIN	Minimum frequency for output (cycles/time). See Remark 2.
FMAX	Maximum frequency for output (cycles/time). See Remark 2.
NFREQ	Number of frequencies for output. See Remark 2.
FSPACE	Frequency spacing option for output (See Remark 2): EQ.0: Linear EQ.1: Logarithmic EQ.2: Biased EQ.3: Eigenfrequencies only
LCFREQ	Load curve ID defining the frequencies for output. See Remark 2.

D3SPCM and D3RMS Card. Additional card for BINARY = 3.

Card 2c	1	2	3	4	5	6	7	8
Variable	ISTATE	FILENAME						
Type	I	C						

VARIABLE	DESCRIPTION
ISTATE	State number in a binary plot file with the name FILENAME. The structural analysis results at this state are combined with the results from the current run.
FILENAME	Path and file name of precomputed structural response binary plot file from which the structural response at a specific state will be combined with the current run's spectrum or RMS results (see Remarks 4 and 5)

Remarks:

1. **Binary data written for D3SSD.** For *OPTION1* = D3SSD, for BINARY = 1, the binary database d3ssd only contains the magnitude of the displacement, velocity, acceleration, and stress response. LS-PrePost 3.0 or older versions can access this data. For customers using LS-PrePost 3.0 or older versions, we suggest setting BINARY = 1. For BINARY = 2, d3ssd includes both the response's magnitude and phase angle so that LS-PrePost (3.1 or higher versions) can run modal expansion (to show the cyclic time history fringe plot) on each output frequency. If BINARY = 90 or 91, only the real or imaginary part of the response is written into d3ssd.

2. **Output frequencies.** There are two methods to define the output frequencies.

   a) The first method is to define FMIN, FMAX, NFREQ, and FSPACE. FMIN and FMAX specify the frequency range of interest, and NFREQ specifies the number of frequencies at which results are required. FSPACE specifies the type of frequency spacing (linear, logarithmic, biased, or eigenfrequencies only). These frequency points for which results are required can be spaced equally along the frequency axis (on a linear or logarithmic scale). Or they can be biased toward the eigenfrequencies (the frequency points are placed closer together at eigenfrequencies in the frequency range) to obtain a detailed response close resonance frequencies. Or they can be just the

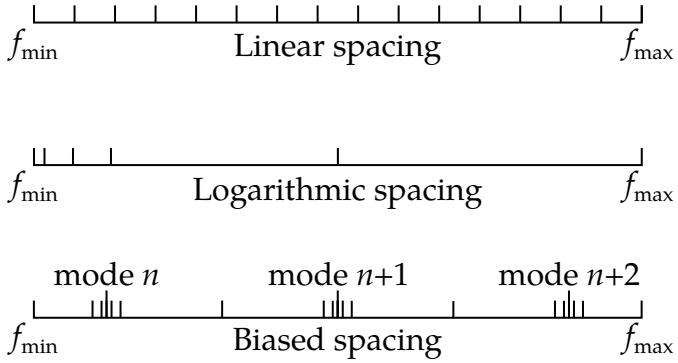


Figure 16-4. Spacing options of the frequency points.

eigenfrequencies in the frequency range defined by FMIN and FMAX. See Figure 16-4.

b) The second method uses a load curve (LCFREQ) to define the frequencies of interest.

3. **Summation for D3PSD and D3RMS.** For *OPTION1* = D3PSD and D3RMS, *OPTION2* = SUMMATION sums up and dumps to a new binary plot database d3psd and d3rms all the precomputed PSD and RMS results saved in existing binary databases defined in Card 1 (by default, d3psd and d3rms are under a different path or prefix). For d3psd, the new psd results are simply the summation of the psd results from each d3psd database (please note that the output frequencies of the existing d3psd databases must be the same, and the new d3psd provides psd results at the same set of frequencies). For d3rms, the new rms results are computed as the square root of the summation of squares of the rms values from each d3rms database. This provides the total PSD and RMS response for the same model, subjected to multiple loading resources. RESTRT = 2 in *FREQUENCY_DOMAIN_RANDOM_VIBRATION activates this operation.

4. **Combined results for response spectrum analysis with other structural analyses.** For *OPTION1* = D3SPCM, if BINARY = 3 in Card 1a, LS-DYNA combines the response spectrum analysis results with other structural analysis results. These other results come from a file specified in Card 2c and correspond to a specific state in that file. This feature allows obtaining the maximum value of the total response from multiple loadings.

d3spcm includes four states when BINARY = 3:

- u
- $r + u$
- $r - u$

- $|r| + |u|$

where u is the response spectrum analysis result and r is the other structural analysis result.

5. **Combined RMS results with other structural analyses.** For *OPTION1* = D3RMS with BINARY = 3 on Card 1a causes outputting the combination of RMS results with prestress from other structural analyses. These other results come from a file specified in Card 2c and correspond to a specific state in that file. This feature enables using stress from a prestressed random vibration in a failure analysis.

   d3rms includes three states when BINARY = 3:

   - u
   - $r + \text{SF} \times u$
   - $|r| + \text{SF} \times u$

   where u is the RMS response, r is the prestress from the structural analyses, and SF is the scale factor input on Card 1a.

*DATABASE_FSI_{OPTION}

Available options include:

<BLANK>

DR

Purpose: When a Lagrangian mesh overlaps with an Eulerian or ALE mesh, the fluid-structure (or ALE-Lagrangian) interaction is often modeled using the *CONSTRAINED_LAGRANGE_IN_SOLID or *ALE_STRUCTURED_FSI card. This keyword (*DATABASE_FSI) causes certain coupling information related to the flux through and load on selected Lagrangian surfaces defined in corresponding *CONSTRAINED_LAGRANGE_IN_SOLID or *ALE_STRUCTURED_FSI card to be written to the ASCII-based dbfsi file or in the case of MPP-DYNA the binout file.

A run involving *CONTROL_DYNAMIC_RELAXATION has two phases: a relaxation phase (to prestress a structure, for instance) and the actual run. The DR option enables saving the dbfsi data of the first phase if needed. If dbfsi is output as an ASCII file (BINARY = 1 or 3) during the first phase, the file is renamed dbfsi_dr before launching the second phase. If dbfsi is output in binout, the second phase data are concatenated to the first phase ones in binout.

NOTE: If used with *CONSTRAINED_LAGRANGE_IN_SOLID, penalty method coupling must be used (CTYPE = 4 or 5). This card is *not* compatible with constraint-based coupling.

Card 1	1	2	3	4	5	6	7	8
Variable	DTOUT	BINARY						
Type	F	I						

Surface Card. Add one card per surface. This input terminates at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	DBFSI_ID	SID	SIDTYPE	SWID	CONVID	NDSETID	CID	
Type	I	I	I	I	I	I	I	

VARIABLE	DESCRIPTION
DTOUT	Output interval time step
BINARY	Flag for binary output. See remarks under "Output Files and Post-Processing" in Appendix O, "LS-DYNA MPP User Guide." EQ.1: ASCII file is written. This is the default for shared memory parallel (SMP) LS-DYNA executables. EQ.2: Data written to binary database binout, which contains data that would otherwise be output to the ASCII file. The ASCII file, in this case, is not created. This is the default for MPP LS-DYNA executables. EQ.3: ASCII file is written, and the data is also written to the binary database. NOTE: MPP LS-DYNA executables only produce the binary database.
DBFSI_ID	Surface ID (for reference purposes only) or a *DATABASE_FSI entity ID. It consists of a geometric entity defined by the SID below.
SID	Set ID defining the geometrical surface(s) through which or upon which some data is to be tracked and output to a file called "dbfsi". This set ID can be a (1) PID or (2) PSID or (3) SGSID. This Lagrangian SID must be contained in a Lagrangian structure SID defined in a corresponding coupling card, *CONSTRAINED_LAGRANGE_IN_SOLID.
SIDTYPE	Set type: EQ.0: Part set EQ.1: Part EQ.2: Segment set
SWID	This is an ID from a corresponding *ALE_FSI_SWITCH_MMG_ID card. This card allows for the AMMG ID of an ALE material to be

VARIABLE	DESCRIPTION
	switched as it passes across a monitoring surface. If defined, the accumulative mass of the "switched" ALE multi-material group (AMMG) is written out under the "mout" parameter in the "dbfsi" file.
CONVID	This is used mostly for airbag application only. CONVID is an ID from a corresponding *LOAD_ALE_CONVECTION_ID card which computes the heat transfer between inflator gas (ALE material) and the inflator canister (Lagrangian part). If defined, the temperature of the Lagrangian part having heat transfer with the gas, and its change in temperature as function of time are output in the "dbfsi" file.
NDSETID	Set ID consisting of the nodes on which the moments of the forces applied on SID are computed. See Remark 3.
CID	Coordinate system ID, see *DEFINE_COORDINATE_SYSTEM.

Remarks:

1. **Overview of dbfsi File.** The dbfsi parameters output are enumerated below.

   pres = Averaged estimated coupling pressure over each surface entity being monitored. For example, if using SI base units for mass-length-time-temperature, this pressure would then be in Pascal.

   fx, fy, fz = Averaged total estimated coupling force components (N in metric units) along the global coordinate directions, over each surface entity defined, and acting at the centroid of each surface.

   mout = Accumulated mass (kg in metric units) passing through each DBFS_ID surface entity. See Remark 2 below. (This parameter used to be called "pleak").

   obsolete = (This parameter used to be called "mflux").

   gx, gy, gz = Average estimated leakage-control force component over the surface entity. This data is useful for debugging. Leakage control forces are too large (relative to the main coupling forces, fx, fy and fz) may indicate that alternate coupling approach should be considered since the main coupling force is putting out too little resistance to leakage. (These parameters used to be called fx-lc, fy-lc and fz-lc).

Ptmp = Lagrangian part Temperature (Activated only when the *LOAD_ALE_CONVECTION card is used).

PDt = Lagrangian part Temperature change (Activated only when the *LOAD_ALE_CONVECTION card is used).

2. **MOUT.** "mout" parameter in the "dbfsi" output from this keyword contains the accumulated mass passing through each DBFS_ID surface entity. For 4 different cases:

   a) When LCIDPOR is defined in the coupling card (CLIS), porous accumulated mass transport across a Lagrangian shell surface may be monitored and output in "mout".

   b) Porous flow across Lagrangian shell may also be defined via a load curve in the *MAT_FABRIC card, and similar result will be tracked and output. This is an alternate form of (a).

   c) When NVENT in the CLIS card is defined (isentropic venting), the venting mass transport across the isentropic vent hole surface may be output in "mout".

   d) When an *ALE_FSI_SWITCH_MMG_ID card is defined, and the SWID parameter specifies this ID to be tracked, then the amount of accumulated mass that has been switched when passing across a monitoring surface is output.

3. **Calculation of Moments for NDSETID.** A geometrical surface SID has a centroid where the coupling forces are averaged. The distances between this centroid and the nodes defined by the set NDSETID are the lever arms. The moments are the cross-products of these distances with the averaged coupling forces. For each node in the set NDSETID, a new line in the "dbfsi" file is inserted after each output for the corresponding coupling forces (see Remark 1). These additional lines have the format following the template established by the example in Remark 1 where the forces are replaced by the moments, and the node ID replaces the DBFSI_ID values.

4. **2D axisymmetric models.** The forces from a 2D axisymmetric simulation are reported per radian, regardless of the element formulation used. Multiply the 2D force by 2π to get the 3D force.

Example:

Consider a model with a Lagrangian mesh overlaps with an Eulerian or ALE mesh. On the Lagrangian mesh, there are 3 Lagrangian surface sets over which some data is to be written out.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ INPUT:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*DATABASE_FSI
$       dt
  2.97E-06
$ DBFSI_ID       SID     STYPE      swid    convid [STYPE: 0=PSID;1=PID;2=SGSID]
        11         1         2
        12         2         2
        13         3         1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ This reads:
$ DBFSI_ID 11 is defined by a SID=1: a SGSID = as specified by STYPE=2
$ DBFSI_ID 12 is defined by a SID=2: a SGSID = as specified by STYPE=2
$ DBFSI_ID 13 is defined by a SID=3: a PID   = as specified by STYPE=1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ An OUTPUT file called "dbfsi" looks like the following:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
     Fluid-structure interaction output
     Number of surfaces:       3

        id          pres            fx            fy            fz          mout
                 obsolete            gx            gy            gz          Ptmp
PDt
         time=  0.00000E+00
        11    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00

        12    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00

        13    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00

         time=  0.29709E-05
        11    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00

        12    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00

        13    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
              0.1832E-06    0.0000E+00    0.0000E+00    0.0000E+00    0.0000E+00
0.0000E+00
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*DATABASE_FSI_SENSOR

Purpose: Activate the output of an ASCII file called dbsensor. This keyword's input defines the pressure sensors' locations, which follow the positions of some Lagrangian segments during the simulation. The ASCII output file, dbsensor, contains the spatial position of the sensor and its recorded pressure from the ALE elements containing the sensors.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY						
Type	F	I						

Sensor Card. Include one card per sensor. This input terminates at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	DBFSI_ID	NID	SEGMID	OFFSET	ND1	ND2	ND3	
Type	I	I	I	F	I	I	I	

VARIABLE	DESCRIPTION
DT	Output interval
BINARY	Flag for binary output. See "Output Files and Post-Processing" in Appendix O, "LS-DYNA MPP User Guide." EQ.1: ASCII file is written. This is the default for SMP executables. EQ.2: Data written to binary database binout, which contains data that would otherwise be output to the ASCII file. The ASCII file in this case is not created. This is the default for MPP executables. EQ.3: ASCII file is written, and the data is also written to the binary database (NOTE: MPP executables will only produce the binary database).
DBFSI_ID	Pressure sensor ID

VARIABLE	DESCRIPTION
NID	An optional Lagrangian node ID defining an approximate pressure sensor location with respect to a Lagrangian shell element. This is not a required input.
SEGMID	Lagrangian element ID for locating the pressure sensor. If NID = 0 or is blank, the sensor will be automatically placed in the center of this SEGMID, accounting for the offset distance. If the model is three-dimensional, the Lagrangian element can be a shell or solid (for the solid element, ND1 and ND2 are required to define the face). If the model is two-dimensional, the Lagrangian element can be a beam or shell (for the shell element, ND1 and ND2 are required to define the side). If SEGMID = 0, ND1, ND2, and ND3 are required for an ALE 3D model, and ND1 and ND2 are needed for an ALE 2D model.
OFFSET	Offset distance between the pressure sensor and the Lagrangian segment surface. If it is positive, it is on the side pointed to by the segment normal vector; otherwise, vice versa.
ND1, ND2, ND3	Nodes defining the solid face for the solid element in the three-dimensional model or the shell side for the shell element in the two-dimensional model, from which the sensor is located. In three dimensions, if the solid face has four nodes, only the diagonal opposites ND1 and ND2 are required. If the solid face is triangular, a third node, ND3, should be provided. In two dimensions, only ND1 and ND2 are required to define the shell side.

Remarks:

1. The output parameters in the dbsensor ASCII file are:

   ID = Sensor ID.

   x, y, z = Sensor spatial location.

   P = Sensor recorded pressure (Pa) from the ALE fluid element containing the sensor.

   To plot the sensor pressure in LS-PrePost, select:

   ASCII → dbsensor → LOAD → (select sensor ID) → Pressure → PLOT

Example 1:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ INPUT:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*DATABASE_FSI_SENSOR
      0.01
$ DBFSI_ID       NID SEGMENTID    OFFSET
        10       360       355      -0.5
        20       396       388      -0.5
        30       324       332      -0.5
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ The 1st line reads:
$ SENSOR_ID 10 is located by segment-ID=355.  Node-ID=360 precisely locate this
$ sensor (if NID=0, then the sensor is located at the segment center).  This
$ sensor is located 0.5 length unit away from the segment surface.  Negative
$ sign indicates a direction opposite to the segment normal vector.
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ An OUTPUT file called "dbsensor" looks like the following:
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
     ALE sensors output
     Number of sensors:  3

        id          x           y           z           p
         time=  0.17861E-02
        10    0.0000E+00    0.0000E+00   -0.3900E+00    0.1085E-03
        20   -0.2250E+02    0.2250E+02   -0.3900E+00    0.1085E-03
        30    0.2250E+02   -0.2250E+02   -0.3900E+00    0.1085E-03
         time=  0.20081E-02
        10    0.0000E+00    0.0000E+00   -0.3900E+00    0.1066E-03
        20   -0.2250E+02    0.2250E+02   -0.3900E+00    0.1066E-03
        30    0.2250E+02   -0.2250E+02   -0.3900E+00    0.1066E-03
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ ID    = DBFSI_ID
$ x,y,z = Sensor location (defined based on a Lagrangian segment)
$ p     = Sensor pressure as taken from the fluid element containing the sensor.
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

DATABASE_HISTORY_*OPTION

Available options include:

BEAM

BEAM_SET

BEAM_ID

DES (see Remark 4)

DES_SET

DISCRETE

DISCRETE_ID

DISCRETE_SET

NODE

NODE_ID

NODE_LOCAL

NODE_LOCAL_ID

NODE_SET

NODE_SET_LOCAL

SEATBELT (see Remark 2)

SEATBELT_ID

SEATBELT_RETRACTOR

SEATBELT_RETRACTOR_ID (see Remark 3)

SEATBELTSLIPRING

SEATBELT_SLIPRING_ID (see Remark 3)

SHELL

SHELL_ID

SHELL_SET

SOLID

SOLID_ID

SOLID_SET

SPH

SPH_SET

TSHELL

TSHELL_ID

TSHELL_SET

Purpose: Control which nodes, elements, or bonds are output into the binary history file d3thdt, the ASCII file nodout, the ASCII file elout, the ASCII file sphout, and the ASCII file debond. See also *DATABASE_BINARY_*OPTION* and *DATABASE_*OPTION*.

Card Summary:

Card 1a. Include this card if the keyword option is BEAM, BEAM_SET, DES, DES_SET, DISCRETE, DISCRETE_SET, NODE, NODE_SET, SEATBELT, SEATBELT_RETRACTOR, SEATBELT_SLIPRING, SHELL, SHELL_SET, SOLID, SOLID_SET, SPH, SPH_SET, TSHELL, or TSHELL_SET. Include as many as needed. Input terminates at the next keyword ("*") card.

ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8

Card 1b. Include this card if the keyword option is BEAM_ID, NODE_ID, SEATBELT_ID, SEATBELT_RETRACTOR_ID, SEATBELT_SLIPRING_ID, SHELL_ID, SOLID_ID, and TSHELL_ID. Include as many as needed. Input terminates at the next keyword ("*") card.

ID	HEADING

Card 1c. Include this card if the keyword option is NODE_LOCAL, NODE_LOCAL_ID, and NODE_SET_LOCAL. If the keyword option is NODE_LOCAL_ID, see Card 1c.1 as well. Include as many cards as needed to specify all the nodes. This input terminates at the next keyword ("*") card.

ID	CID	REF	HFO				

Card 1c.1. This card is only used for the NODE_LOCAL_ID keyword option. When activated, each node is specified by a pair of cards consisting of Card 1c, and, secondly, this card. Include as many pairs as needed to specify all the nodes. This input terminates at the next keyword ("*") card.

HEADING	

Data Card Definitions:

Node/Element Cards for Case I (no "ID", and no "LOCAL"). Cards for keyword options BEAM, BEAM_SET, DES, DES_SET, DISCRETE, DISCRETE_SET, NODE, NODE_SET, SEATBELT, SEATBELT_RETRACTOR, SEATBELT_SLIPRING, SHELL, SHELL_SET, SOLID, SOLID_SET, SPH, SPH_SET, TSHELL, and TSHELL_SET. Include as many as needed. Input terminates at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
ID*n*	NODE/NODE_SET or element/element set ID *n*. Elements may be BEAM/BEAM_SET, DES/DES_SET, DISCRETE/DISCRETE_SET, SEATBELT, SEATBELT_RETRACTOR, SEATBELT_SLIPRING, SHELL/SHELL_SET, SOLID/SOLID_SET, SPH/SPH_SET or TSHELL/TSHELL_SET. ID*n* for NODE_SET, SPH_SET, and DES_SET refers to node set ID *n* defined using the *SET_NODE_*{OPTION}*. The contents of the files are given in Table 16-1 for nodes, Table 16-2 for solid elements, Table 16-3 for shells and thick shells, and Table 16-4 for beam elements. The contents of binary file d3thdt may be extended or reduced with *DATABASE_EXTENT_BINARY.

Node/Element Cards for Case II ("ID" option, but no "LOCAL"). Cards for keyword options BEAM_ID, NODE_ID, SEATBELT_ID, SEATBELT_RETRACTOR_ID, SEATBELT_SLIPRING_ID, SHELL_ID, SOLID_ID, and TSHELL_ID. Include as many as needed. Input terminates at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Node or element ID
HEADING	A description of the node or element. Using a unique description is recommended. This description is written into the d3hsp file and into the ASCII databases nodout and elout.

Node Cards for Case III ("LOCAL" option). Card 1 for keyword options NODE_LOCAL, NODE_LOCAL_ID, and NODE_SET_LOCAL. Include as many cards as needed to specify all the nodes. This input terminates at the next keyword ("*") card.

Card 1c	1	2	3	4	5	6	7	8
Variable	ID	CID	REF	HFO				
Type	I	I	I	I				

ID Card for Case III. Additional card for ID option. This card is only used for the NODE_LOCAL_ID keyword option. When activated, each node is specified by a pair of cards consisting of Card 1c and, secondly, this card. Include as many pairs as needed to specify all the nodes. This input terminates at the next keyword ("*") card.

Card 1c.1	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	A70							

VARIABLE	DESCRIPTION
ID	NODE/NODE_SET set ID. The contents of the files are given in Table 16-1 for nodes. See Remark 1 concerning accelerometer nodes.
CID	Coordinate system ID for nodal output. See *DEFINE_COORDINATE options.
REF	Output coordinate system for displacements, velocities, and accelerations. (Nodal coordinates are always in the global coordinate system.) EQ.0: Output is in the local system fixed for all time from the beginning of the calculation. If CID is nonzero, FLAG in the corresponding *DEFINE_COORDINATE_NODES command must be set to 0. FLAG has no bearing on results when REF is set to 1 or 2. EQ.1: Translational output is the projection of the node's absolute translational motion onto the local system. The local system is defined by the *DEFINE_COORDINATE_NODES command and can change orientation according to the movement of the three defining nodes. The defining nodes can belong to either deformable or rigid parts. EQ.2: Translational output is the motion of the node, expressed in the local system attached to node N1 of CID. The local system is defined as described in REF = 1 above. For displacements, the fixed reference location is defined to contain the initial coordinates of the history node in the local coordinate system. During the analysis, the coordinates of the history node are first expressed in the translating and rotating local coordinate system and then the local displacement vector is determined by subtracting the reference location from these coordinates. If dynamic relaxation is used, the reference location is reset when convergence is achieved. Rotational output is truly relative to the updated location coordinate system only if REF = 2.
HFO	Flag for high frequency output into nodouthf: EQ.0: Nodal data written to nodout file only. EQ.1: Nodal data also written nodouthf at the higher frequency.
HEADING	A description of the nodal point. It is suggested that unique

VARIABLE	DESCRIPTION
	description be used. This description is written into the d3hsp file and into the ASCII database nodout.

Remarks:

1. **Accelerometer.** If a node belongs to an accelerometer (see *ELEMENT_SEATBELT_ACCELEROMETER) and also appears as an active node in the NODE_LOCAL or NODE_SET_LOCAL keyword, the coordinate system transformations are skipped, and the LOCAL option has no effect.

2. **Seat belt.** The SEATBELT keyword option is only for 1D (bar-type) seat belts, not 2D (shell-type).

3. **Legend for slip ring/retractor data.** To define legends for slip ring and retractor data, append ID to the SEATBELT_SLIPRING and SEATBELT_RETRACTOR keyword options. The HEADING field gives the legend name.

4. **DES.** By default, the debond ASCII file (see *DATABASE_DEBOND) monitors all DES bonds for breakage. Defining *DATABASE_HISTORY_DES_*{OPTION}* causes debond to only monitor the bonds associated with the node IDs defined by ID*n* . For example, if a single node appears in *DATABASE_HISTORY_DES, all bonds that bind the discrete sphere defined by that node will be monitored for breakage type.

*DATABASE_HISTORY_ACOUSTIC

Purpose: Identify acoustic nodes for time history output.

Card 1	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
NID*n*	Node ID of the acoustic node.

Remarks:

1. **Results File Name.** In explicit, transient solutions, the histories are written to file ACEOUT. In implicit, direct SSD solutions, the histories are written to ACEOUT_SSD.

*DATABASE_ISPHHTC

Purpose: Create Heat Transfer Coefficient output for Incompressible SPH simulations. HTC values are time-averaged over the defined output time interval.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY	LCDT	IOOPT	PSET			
Type	F	I	I	I	I			

VARIABLE	DESCRIPTION
DT	Time interval between output states. If DT is zero, the output will be skipped even if LCDT is defined.
BINARY	Currently unused. The output format is currently a profile file that can be directly imported inside Ansys Fluent or Ansys Mechanical.
LCDT	Optional load curve ID specifying the output time interval as a function of time.
IOOPT	This input field governs how the plot state frequency is determined from curve LCDT: EQ.1: When a plot is generated at time t_n, the next plot time t_{n+1} is computed as $t_{n+1} = t_n + \mathrm{LCDT}(t_n)$. This is the default behavior. EQ.2: When a plot is generated at time t_n, the next plot time t_{n+1} is computed as $t_{n+1} = t_n + \mathrm{LCDT}(t_{n+1})$. EQ.3: A plot is generated for each abscissa point in the load curve definition. The actual value of the load curve is ignored.
PSET	Optional part set. If defined, only parts contained in this part set will be included in the HTC output file.

*DATABASE_MASSOUT

Purpose: Output nodal masses into ASCII file MASSOUT.

Card 1	1	2	3	4	5	6	7	8
Variable	SETID	NDFLG	RBFLG					
Type	I	I	I					
Default	0	1	0					

VARIABLE	DESCRIPTION
SETID	Optional set ID. EQ.0: mass output for all nodes, LT.0: no output, GT.0: set ID identifying nodes whose mass will be output.
NDFLG	Database extent: EQ.1: output translational mass for deformable nodes identified by SETID (default), EQ.2: output translational mass and rotary inertias for the deformable nodes identified by the SETID. EQ.3: output translational mass for deformable and rigid nodes identified by SETID (default), EQ.4: output translational mass and rotary inertias for the deformable and rigid nodes identified by the SETID.
RBFLG	Rigid body data: EQ.0: no output for rigid bodies, EQ.1: output rigid body mass and inertia.

Remarks:

1. **Massless Nodes and Rigid Bodies.** Nodes and rigid bodies with no mass are not output. By inference, when the set ID is zero and no output shows up for a node, then the mass of that node is zero.

***DATABASE_MAX**_OPTION_

Available options include:

BEAM

BEAM_ID

BEAM_SET

SHELL

SHELL_ID

SHELL_SET

SOLID

SOLID_ID

SOLID_SET

TSHELL

TSHELL_ID

TSHELL_SET

Purpose: Control which elements have data output to the binary history file d3max. See also *DATABASE_D3MAX.

Card Summary:

Card 1a. This card is included if the keyword option is BEAM, BEAM_SET, SHELL, SHELL_SET, SOLID, SOLID_SET, TSHELL, or TSHELL_SET. Include as many as needed. Input terminates at the next keyword ("*") card.

ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8

Card 1b. This card is included if the keyword option is BEAM_ID, SHELL_ID, SOLID_-ID, and TSHELL_ID. Include as many as needed. Input terminates at the next keyword ("*") card.

ID	HEADING

Data Card Definitions:

Element Cards for Case I (no "ID"). Cards for keyword options BEAM, BEAM_SET, SHELL, SHELL_SET, SOLID, SOLID_SET, TSHELL, and TSHELL_SET. Include as many as needed. Input terminates at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
ID*n*	Element/element set ID *n*. The elements may be beams, shells, solids, or tshells depending on the keyword option.

Element Cards for Case II ("ID" option"). Cards for keyword options BEAM_ID, SHELL_ID, SOLID_ID, and TSHELL_ID. Include as many as needed. Input terminates at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Element ID
HEADING	A description of the element. We suggest using unique descriptions.

*DATABASE_NODAL_FORCE_GROUP_{OPTION}

OPTION allows an optional ID to be given for the node set. If you define a heading, the ID and heading are included at the beginning of the ASCII file nodfor.

ID

Purpose: Define a nodal force group for output into the ASCII file nodfor. The output interval must be specified using *DATABASE_NODFOR (see *DATABASE_*OPTION*).

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	CID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ID	Group ID
HEADING	Optional descriptor for the given ID that will be written into the nodfor file
NSID	Nodal set ID, see *SET_NODE
CID	Coordinate system ID for output of data in the local system

Remarks:

1. **Reaction forces.** The reaction forces in the global x, y, and z directions (and local x, y, and z directions if CID is defined above) for the nodal force group are written to the nodfor file (see *DATABASE_NODFOR) along with the external

work done by these reaction forces. The reaction forces in the global x, y, and z directions for each node in the nodal force group are also written to nodfor. These forces can result from applied boundary forces, such as nodal point forces and pressure boundary conditions, body forces, and contact interface forces. In the absence of body forces, interior nodes always yield a null force resultant vector. In general, this option should be used for surface nodes.

*DATABASE_PAP_OUTPUT

Purpose: Set contents of output files for pore air pressure calculations.

Card 1	1	2	3	4	5	6	7	8
Variable	IVEL	IACCX	IACCY	IACCZ	NCYOUT	NDAIR		
Type	I	I	I	I	I	I		
Default	0	0	0	0	100	0		

VARIABLE	DESCRIPTION
IVEL	Meaning of "Velocity" in d3plot and d3thdt output files EQ.0: Nodal velocity vector EQ.1: Seepage velocity vector
IACCX, IACCY, IACCZ	Meaning of "X/Y/Z-Acceleration" in d3plot and d3thdt output files EQ.0: Not written EQ.21: Nodal air density EQ.22: Nodal pore air pressure EQ.24: Nodal air mass EQ.25: Nodal air mass flow rate
NCYOUT	Number of cycles between outputs of calculation status to d3hsp and log files
NDAIR	Nodes in the model for which the application of the above-defined meanings are reset in d3plot and d3thdt files: EQ.0: All nodes EQ.1: Only nodes involved in pore air pressure calculations

*DATABASE_PBLAST_SENSOR

Purpose: This keyword causes LS-DYNA to output a directory called pblast_sensor to the file, binout. The input for this keyword defines the sensors' locations based on the positions of some Lagrangian shell elements. With this keyword, LS-DYNA outputs the history of the position, temperature, density, and pressure averaged over the number of particles contained in the sensors to binout. This keyword is activated only when the *DEFINE_PARTICLE_BLAST card is used.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY						
Type	F	I						
Default	none	3						

Sensor Definition Cards. Each card defines one sensor. This card may be repeated to define multiple sensors. Input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	ID	ITYPE	OFFSET	RADIUS				
Type	I	I	F	F	F			

VARIABLE	DESCRIPTION
DT	Output interval
BINARY	Flag for the binary file: EQ.3: Data is written to the binary file binout.
ID	Set ID
ITYPE	Type for ID: EQ.0: *SET_SHELL ID EQ.1: Shell ID

VARIABLE	DESCRIPTION
OFFSET	Offset distance, d, between the sensor and the segment center. Here $d > 0$ is along the shell normal and $d < 0$ is against the shell normal.
RADIUS	Radius of the spherical sensor. Please see *DATABASE_CPM_-SENSOR for more details about spherical sensors.

Remarks:

1. **Output.** The output parameters to the pblast_sensor directory are:

x	=	x-coordinate
y	=	y-coordinate
z	=	z-coordinate
temp	=	temperature
dens	=	density
pres	=	pressure
counts	=	number of particles in the sensor

These values are averaged over the number of particles in the sensor. The sensor should be large enough to contain a reasonable number of particles for the averages.

*DATABASE_PROFILE

Purpose: Plot the distribution or profile of a data along x, y, or z-direction.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	ID	TYPE	DATA	DIR	UPDLOC	MMG	
Type	I	I	I	I	I	I	I	
Default	none	none	none	none	none	0	0	

VARIABLE	DESCRIPTION
DT	Interval time.
ID	Set ID.
TYPE	Set type: EQ.1: node set EQ.2: solid set EQ.3: shell set EQ.4: segment set EQ.5: beam set EQ.6: tshell set
DATA	Data type: EQ.1: x-velocity, EQ.2: y-velocity, EQ.3: z-velocity, EQ.4: velocity magnitude, EQ.5: x-acceleration, EQ.6: y-acceleration, EQ.7: z-acceleration, EQ.8: acceleration magnitude, EQ.9: pressure,

VARIABLE	DESCRIPTION
	EQ.10: *xx*-stress, EQ.11: *yy*-stress, EQ.12: *zz*-stress, EQ.13: *xy*-stress, EQ.14: *yz*-stress, EQ.15: *zx*-stress, EQ.16: temperature, EQ.17: volume fraction, EQ.18: kinetic energy, EQ.19: internal energy, EQ.20: density, EQ.21: *xx*-strain, EQ.22: *yy*-strain, EQ.23: *zz*-strain, EQ.24: *xy*-strain, EQ.25: *yz*-strain, EQ.26: *zx*-strain. EQ.27: effective plastic strain
DIR	Direction: EQ.1: *x*-direction EQ.2: *y*-direction EQ.3: *z*-direction EQ.4: curvilinear (relative distances between elements of set ID are added up in the order defined by the set)
UPDLOC	Flag to update the set location: EQ.0: only the initial position of set ID is considered. EQ.1: the positions of the elements composing the set are updated each DT.
MMG	Multi-Material ALE group ID. See Remark 2. GT.0: Multi-Material ALE group ID

VARIABLE	DESCRIPTION
LT.0:	|MMG| is the ID of a *SET_MULTI-MATERIAL_GROUP_-LIST that can list several Multi-Material ALE group IDs.

Remarks:

1. **File Description.** At a given time T the profile is written in a file named profile_DATA_DIR_time$T$.xy (DATA and DIR are replaced by the data and direction names respectively). The file has a xyplot format that LS-PrePost can read and plot. For example, DATA = 9, DIR = 2, and DT = 0.1 sec will save a pressure profile at $t = 0.0$ sec in profile_pressure_y_time0.0.xy, at $t = 0.1$ sec in profile_pressure_y_time0.1.xy, at $t = 0.2$ sec in profile_pressure_y_time0.2.xy. The stresses are output for each integration point. The shell strains are output for the innermost and outermost integration points if STRFLG = 1 in *DATABASE_-EXTENT_BINARY. For stresses and shell strains, the integration point rank is appended to the data names in the filename.

2. **Multi-Material ALE Model.** For the case of a multi-material ALE model (elform = 11 in *SECTION_SOLID or *SECTION_ALE2D or *SECTION_-ALE1D), an element can contain several materials with each material being associated with its own pressures and stresses. By default element data is volume averaged before being written out; however, when the multi-material group field, MMG, is set, then element data are output only for the specified materials.

*DATABASE_PWP_FLOW

Purpose: Request output containing the net inflow of fluid at a set of nodes.

Node Sets. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NSET							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
NSET	Node set ID

Remarks:

1. **Inflow.** Net inflow or outflow occurs when maintaining an applied PWP boundary condition causes the addition or removal of water. See *BOUNDARY_PWP.

2. **Output file.** Output is written to a file named database_pwp_flow.csv, a comma-separated ASCII file. Each line consists of (time, flow1, flow2, …) where flow1 is the total inflow at the node set for the first *DATABASE_PWP_FLOW request, flow2 is for the second, etc.

3. **Output time interval.** For versions up to and including R15, the output is written at the time interval DT on *DATABASE_BINARY_D3THDT. If this card is absent, no output is written.

*DATABASE_PWP_OUTPUT

Purpose: Set contents of output files for pore pressure calculations.

Card 1	1	2	3	4	5	6	7	8
Variable	IVEL	IACCX	IACCY	IACCZ	NCYOUT			
Type	I	I	I	I	I			
Default	0	0	0	0	100			

VARIABLE **DESCRIPTION**

IVEL Meaning of nodal "Velocity" in d3plot:

- EQ.0: Nodal velocity vector
- EQ.1: Seepage velocity vector

IACCX, Y, Z Meaning of nodal "X/Y/Z-Acceleration" in d3plot, d3thdt, and nodout:

- EQ.0: Not written
- EQ.1: Total pwp head
- EQ.2: Excess pwp head (this is also written as temperature)
- EQ.3: Target rate of volume change
- EQ.4: Actual rate of volume change
- EQ.5: Part whose pore pressure properties are used for this node (internal ID)
- EQ.7: Hydraulic pwp head
- EQ.8: Error in rate of volume change (calculated from seepage minus actual)
- EQ.9: Volume at node
- EQ.10: Rate of volume change calculated from seepage
- EQ.14: Void volume (generated at suction limit)
- EQ.17: Status flag (e.g: +4/-4 for nodes on suction limit)

VARIABLE	DESCRIPTION
NCYOUT	Number of cycles between outputs of calculation status to d3hsp, log, and tdc_control_output.csv files (time-dependent and steady-state analysis types).

*DATABASE_RCFORC_MOMENT

Purpose: Define contact ID and nodes for moment calculations. Moments are written to rcforc according to output interval given in *DATABASE_RCFORC. If *DATABASE_RCFORC_MOMENT is not used, the moments reported to rcforc are about the origin (0,0,0).

Card 1	1	2	3	4	5	6	7	8
Variable	CID	NODESA	NODESB					
Type	I	I	I					

VARIABLE	DESCRIPTION
CID	Contact ID
NODESA	Node about which moments are calculated due to contact forces on SURFA surface
NODESB	Node about which moments are calculated due to contact forces on SURFB surface

*DATABASE_RECOVER_NODE

Purpose: Recovers the stresses at nodal points of solid or thin shell elements by using either Zienkiewicz-Zhu's Superconvergent Patch Recovery (SPR) method or an elemental extrapolation method.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	IAX	IAY	IAZ	METHOD	IVX	IVY	IVZ
Type	I	A	A	A	I	A	A	A
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
PSID	Part set ID of solid or thin shell elements whose nodal stress will be recovered
IAX, IAY, IAZ, IVX, IVY, IVZ	Meaning of "$x/y/z$-Acceleration" or "$x/y/z$-Velocity" in d3plot and d3thdt output files EQ.SMNPD: the minimum principal deviator stress EQ.SMNPR: the minimum principal stress EQ.SMXPD: the maximum principal deviator stress EQ.SMXPR: the maximum principal stress EQ.SMXSH: the maximum shear stress EQ.SPR: nodal pressure EQ.SVM: nodal von Mises stress EQ.SXX: nodal normal stress along x direction EQ.SYY: nodal normal stress along y direction EQ.SZZ: nodal normal stress along z direction EQ.SXY: nodal shear stress along x-y direction EQ.SYZ: nodal shear stress along y-z direction EQ.SZX: nodal shear stress along z-x direction For shell elements append either "B" or "T" to the input string to recover nodal stresses at the bottom or top layer of shell elements. For example, SPRT recovers the nodal pressure at the top layer.

VARIABLE	DESCRIPTION
METHOD	Method used to recover the nodal stress: EQ.0: Zienkiewicz-Zhu's Superconvergent Patch Recovery method EQ.1: elemental extrapolation method

Remarks:

1. **Coordinate System.** Recovered stresses are in global coordinate system.
2. **SPR Method.** The SPR method is based on the extrapolation of an element patch, a group of elements surrounding a node. Therefore, it should be more accurate than the elemental extrapolation method which is based on the extrapolation of a single element. This is especially true for under-integrated elements where the integration points are the super convergent points. However, since SPR requires an element patch, it cannot recover the nodal stresses correctly when a mesh is too course to form a patch (for example, through the thickness direction of a beam, only a single layer of elements is used). Also, when used for shell elements, it can only provide a good solution in two-dimensional analysis or in small strain analysis where most elements remain on the same plane.
3. **Elemental Extrapolation Method.** The elemental extrapolation method is also used for detailed element out, eloutdet. It is more robust and provides acceptable solutions for elements of multiple integration points.

*DATABASE_RVE

Purpose: Output of the RVE homogenization results to the rveout file. Double precision SMP/MPP LS-DYNA version R13 and newer versions support this RVE analysis function.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	BINARY						
Type	F	I						
Default	0.0	0						

VARIABLE	DESCRIPTION
DT	Time interval for the output of RVE homogenization results to the rveout file.
BINARY	Type of the output file: EQ.0: ASCII database file named rveout

Remarks:

At each output time t, LS-DYNA records the macroscopic material responses to the rveout file. The responses include the macroscopic deformation gradient $\tilde{\mathbf{F}}^t$, Green strain $\tilde{\mathbf{E}}^t$, Cauchy stress $\tilde{\boldsymbol{\tau}}^t$, and the 1st-Piola-Kirchhoff (PK1) stress $\tilde{\mathbf{P}}^t$. These results are obtained through nonlinear computational homogenization of RVE (*Representative Volume Element*) for composite materials. Please refer to the keyword *RVE_ANALYSIS_FEM for more details.

*DATABASE_SALE

Purpose: Control how the S-ALE solver outputs the post-processing database and whether the generated S-ALE mesh is output or not. This keyword can also specify additional element history variable data for output.

Card 1	1	2	3	4	5	6	7	8
Variable	D3SALE	DMPMSH						NEHSV
Type	I	I						I
Default	0	0						0

Element History Variable Cards. Include ceil(NEHSV/8) instances of this card to specify the NEHSV history variables output based on the names of the history variables given in Table 16-5.

Card 2	1	2	3	4	5	6	7	8
Variable	HSVNM1	HSVNM2	HSVNM3	HSVNM4	HSVNM5	HSVNM6	HSVNM7	HSVNM8
Type	A	A	A	A	A	A	A	A

VARIABLE	DESCRIPTION
D3SALE	Flag for where to output the S-ALE part data (see Remarks 1 and 2): EQ.0: Output S-ALE part data to d3plot. EQ.1: Write S-ALE part data to binary, LSDA-format database d3sale.
DMPMSH	Flag for outputting the generated S-ALE mesh (see Remark 3): EQ.0: Do not output the S-ALE mesh. EQ.1: Output the S-ALE mesh into a file named salemsh.inc.
NEHSV	Number of element history variables to output to d3sale (depending on the setting of D3SALE). If not specified, only element pressure is output.

VARIABLE	DESCRIPTION
HSVNM*i*	Element history variable names for the history variables to be output. See Table 16-5.

Table 16-5. The following table lists the options for HSVNM*i*. The names are case insensitive.

HSVNM	DESCRIPTION
DFID	Dominant fluid (AMMG) ID
TLIT*ab*	Lighting time. This variable is nonzero only for AMMGs defined with *MAT_HIGH_EXPLOSIVE. *ab* is a double-digit integer specifying which AMMG. For example, TLIT01 means lighting time of the first AMMG. If *ab* = 00, the volumetric average value over all AMMGs (composite value) in that element will be calculated and output.
TEMP*ab*	Temperature. Please see Remark 4 for supported material models. *ab* is a double-digit integer specifying which AMMG. For example, TEMP01 means temperature of the first AMMG. If *ab* = 00, the volumetric average value over all AMMGs (composite value) in that element will be calculated and output.
EPS*ab*	Effective Plastic Strain. *ab* is a double-digit integer specifying which AMMG. For example, EPS02 means effective plastic strain of the second AMMG. If *ab* = 00, the volumetric average value over all AMMGs (composite value) in that element will be calculated and output.
PRES*ab*	Pressure. *ab* is a double-digit integer specifying which AMMG. For example, PRES01 means pressure of the first AMMG. Note: *ab* = 00 is not supported as the composite pressure is output by default.
SXX*ab*, SYY*ab*, SZZ*ab*, SXY*ab*, SYZ*ab*, SZX*ab*	Stress component. *ab* is a double-digit integer specifying which AMMG. For example, SXX12 means pressure of the twelfth AMMG. If *ab* = 00, the volumetric average value over all AMMGs (composite value) in the element is calculated and output.
VMS*ab*	von Mises stress. *ab* is a double-digit integer specifying which AMMG.

HSVNM	DESCRIPTION
XHV*cdab*	Extra element history variables. *ab* is a double-digit integer specifying which AMMG; and *cd* is a double-digit integer specifying which extra history variable. For example, XHV0201 retrieves the second extra history variable of the first AMMG. See Remark 5.

Remarks:

1. **S-ALE data output interval.** *DATABASE_BINARY_D3PLOT specifies the output interval of S-ALE data for either choice of D3SALE.

2. **D3SALE.** The d3sale file is an LSDA-formatted database for flexibility and file size reduction. When turned on, the output d3sale contains S-ALE post-processing data, while d3plot contains the data for Lagrange parts.

3. **Outputting S-ALE mesh.** The keyword input phase generates the S-ALE mesh (elements/nodes) based on information provided in *ALE_STRUCTURED_MESH and *ALE_STRUCTURED_MESH_CONTROL_POINTS. DMPMSH = 1 invokes outputting the generated mesh keywords into a file named salemsh.inc. The file can be used to validate the mesh in a pre-processor.

4. **TEMP element history variable.** AMMGs defined with the following material models are supported for the TEMP option: *MAT_ELASTIC_PLASTIC_THERMAL, *MAT_STEINBERG, *MAT_JOHNSON_COOK, and *MAT_GAS_MIXTURE.

5. **XHV element history variable.** The number of extra history variables depends on the combination of material model with equation of state model. For example, let AMMG #3 be defined with a material model and equation of state model such that the specific internal energy is the first extra history variable. To specify outputting that history variable with this keyword, set one of the history variable names to XHV0103, where 01 means the first extra history variable and 03 means the third AMMG.

*DATABASE_SPRING_FORWARD

Purpose: Create spring forward nodal force file. This option is to output resultant nodal force components of sheet metal at the end of the forming simulation into an ASCII file, SPRING-FORWARD, for spring forward and die corrective simulations.

Card 1	1	2	3	4	5	6	7	8
Variable	IFLAG							
Type	I							

VARIABLE	DESCRIPTION
IFLAG	Output type: EQ.0: off, EQ.1: output element nodal force vector for deformable nodes.

*DATABASE_SUPERPLASTIC_FORMING

Purpose: Specify the output intervals to the superplastic forming output files. The option *LOAD_SUPERPLASTIC_FORMING must be active.

Card 1	1	2	3	4	5	6	7	8
Variable	DTOUT							
Type	F							

VARIABLE	DESCRIPTION
DTOUT	Output time interval for output to "pressure," "curve1" and "curve2" files. The "pressure" file contains general information from the analysis and the files "curve1" and "curve2" contain pressure as a function of time from phases 1 and 2 of the analysis. The data in the pressure and curve files may be plotted using *ASCII* → *superpl* in LS-PrePost.

*DATABASE_TRACER_{OPTION}

Purpose: Tracer particles will save a history of either a material point or a spatial point into an ASCII file: trhist. This history includes positions, velocities, and stress components. The option *DATABASE_TRHIST must be active. This option applies to ALE, SPH, and DEM (Discrete Element Method) problems.

Available options are:

<BLANK>

DE

The DE option defines a tracer corresponding to discrete elements (*ELEMENT_DISCRETE_SPHERE). See Remarks 2 and 4.

Card	1	2	3	4	5	6	7	8
Variable	TIME	TRACK	X	Y	Z	AMMGID	NID	RADIUS
Type	F	I	F	F	F	I	I	F
Default	0.0	0	0.0	0.0	0.0	0	0	0.0

VARIABLE	DESCRIPTION
TIME	Start time for tracer particle
TRACK	Tracking option: EQ.0: Particle follows material (see Remark 5). EQ.1: Particle is fixed in space. EQ.2: Particle follows the mesh
X	Initial *x*-coordinate
Y	Initial *y*-coordinate
Z	Initial *z*-coordinate
AMMGID	The ALE multi-material referenced with its AMMGID for general ALE or with either its AMMGID or AMMG name (AMMGNM) for S-ALE, of the material being tracked in a multi-material ALE element. See Remark 1.

VARIABLE	DESCRIPTION
NID	An optional node ID defining the initial position of a tracer particle. If defined, its coordinates will overwrite the x, y, and z coordinates above. This feature is for TRACK = 0 only and can be applied to ALE tracers and DE tracers. See Remark 2.
RADIUS	Radius is used only for the DE option to indicate whether the tracer follows and monitors a single discrete element or multiple discrete elements. GT.0.0: The tracer takes the average results of all discrete elements located inside a sphere with radius = RADIUS. That sphere stays centered on the DE tracer. LT.0.0: The discrete element closest to the tracer is used. The magnitude of RADIUS, in this case, is unimportant.

Remarks:

1. **Multi-material groups.** ALE elements can contain multi-materials. Each material is referred to as an ALE multi-material group or AMMG. Each AMMG has its list of history variables that can be output. For example, if a tracer is in a mixed element consisting of 2 AMMGs, and the history variables of AMMG 1 are to be output or tracked, the AMMGID should be defined as AMMGID = 1. If AMMGID = 0, a volume-fraction-weighted-averaged pressure will be reported instead.

2. **NID description.** For ALE, NID is a massless dummy node. Its location will be updated according to the motion of the ALE material.

   For the DE option, NID is a discrete element node that defines the initial location of the tracer. The DE tracer continues to follow that node if RADIUS < 0. On the other hand, the DE tracer's location is updated according to the average motion of the group of DE nodes inside the sphere defined by RADIUS when RADIUS > 0.

3. **Tracer particles in ambient ALE elements.** Since the auxiliary variables (6 stresses, plastic strain, internal energy, …) for ambient elements are reset to their initial values before and after advection and tracer data are stored in trhist during the advection cycle, tracers in ambient elements show the initial stresses, not the current ones.

4. **Discrete elements.** If the DE keyword option is used, tracer particles will save a history of either a material point or a spatial point into an ASCII file: demtrh. This history includes positions, velocities components, stress components,

porosity, void ratio, and coordination number. *DATABASE_TRHIST must be active.

5. **Mapping.** Lagrange (TRACK = 0) tracers with mapping (map =) produce a keyword file tracers_final_positions.k, containing the final positions of the tracer particles. This file should be included in the mapped simulation to correctly locate the Lagrange tracers.

*DATABASE_TRACER_ALE

Purpose: Save a history of either a material point or a spatial point using tracer particles into an ASCII file: traleh. This history includes positions, velocities, and stress and other user-specified element history variables. *DATABASE_TRHIST must be active. This keyword is only to be used with ALE problems.

Card	1	2	3	4	5	6	7	8
Variable	NID	TRACK	AMMGID	HVBEG	HVEND	TIME		
Type	I	I	I	I	I	I		
Default	0	0	0	0	0	0.0		

VARIABLE	DESCRIPTION
NID	Node ID defining the initial position of a tracer particle. See Remark 1.
TRACK	Tracking option: EQ.0: Particle follows material EQ.1: Particle is fixed in space
AMMGID	ALE multi-material referenced with its AMMGID for general ALE or with either its AMMGID or AMMG name (AMMGNM) for S-ALE, of the material being tracked in a multi-material ALE element. See Remark 2.
HVBEG	The beginning index of element history variables to be output. See Remark 3.
HVEND	The ending index of element history variables to be output. The number of extra history variables must be no more than 15, meaning HVEND -HVBEG = 15.
TIME	Start time for tracer particle activation

Remarks:

1. **NID description.** NID is a massless dummy node. Its location will be updated according to the motion of the ALE material.

2. **Multi-material groups.** ALE elements can contain multi-materials. Each material is referred to as an ALE multi-material group or AMMG. Each AMMG has its list of history variables that can be output. For example, if a tracer is in a mixed element consisting of 2 AMMGs, and the history variables of AMMG 1 are to be output or tracked, the AMMGID should be defined as AMMGID = 1. If AMMGID = 0, a volume-fraction-weighted-averaged pressure will be reported instead.

3. **History variables.** This keyword provides a way to output specific element history variables. By default, there are 9 element properties output for each particle: 6 stresses, plastic strain, volume fraction, and density. For example, if you want to know the air temperature of AMMG 2 which is material *MAT_148, you set AMMGID to 2 and HVBEG = HVEND = 16 since history variable 16 of *MAT_148 is temperature. At most, data for 15 history variables can be output this way.

4. **Tracer particles in ambient ALE elements.** Since the auxiliary variables (6 stresses, plastic strain, internal energy, …) for ambient elements are reset to their initial values before and after advection, and tracer data are stored in trhist during the advection cycle, tracers in ambient elements show the initial stresses, not the current ones.

Table 16-6. Variables output to traleh

Variable	Description
elemID	ID of ALE element in which the tracer is currently located
x, y, z	Position
vx, vy, vz	Velocity
sx, sy, sz, sxy, syz, szx	Stress
efp	Effective plastic strain
rho	Density
rvol	Volume fraction
hisvar*i*	History variable *i*

Example File:

Below is an example traleh file. 7 and 30 in the second line tells the reader that there are 7 tracers and each tracer has 30 output variables. The next six lines list the output variables as they are output in the file. These output variables are described in Table 16-6.

```
ALE Tracer particle file
    7   30
          elemID
```

```
          x           y           z          vx          vy          vz
         sx          sy          sz         sxy         syz         szx
        efp         rho        rvol       hisv1       hisv2       hisv3
      hisv4       hisv5       hisv6       hisv7       hisv8       hisv9
     hisv10      hisv11      hisv12      hisv13      hisv14      hisv15
 0.00000E+00
       5002767
 5.01000E+02  1.00100E+03  1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
-9.99400E-02 -9.99400E-02 -9.99400E-02  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
       5002780
 1.00100E+03  1.00100E+03  1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
-9.99400E-02 -9.99400E-02 -9.99400E-02  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
       5002792
 1.50100E+03  1.00100E+03  1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
-9.99400E-02 -9.99400E-02 -9.99400E-02  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
 0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
```

*DATABASE_TRACER_GENERAL

Purpose: Output histories for any solids, beams, shells, or thick shells in which the tracer, which is a node, is located. The histories are saved into a binary file: trcrgal_binout. These histories include positions, velocities, and stress components. The data structure is identical to the one output by *DATABASE_TRACER_ALE into traleh file. Except for the positions and element ID specifying where the tracer is, the output can be controlled with the VARLOC and VAREPL fields.

Card 1	1	2	3	4	5	6	7	8
Variable	NODE	ELEM	TYPM	MOVE	SET	TYPS	COR	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Optional Variable Cards. Cards defining the times to output data in trcrgal_binout.

Card 2	1	2	3	4	5	6	7	8
Variable	DT	TBEG	TEND	FID				
Type	F	F	F	I				
Default	0.0	0.0	10^{20}	0				

Optional Variable Cards. Cards defining new variables to be output to trcrgal_binout instead of the default ones. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	VARLOC	VAREPL						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
NODE	Node ID that locates the tracer (see Remark 1)
ELEM	Element ID that controls the tracer motion (see Remarks 1 and 2) GT.0: Data are output for ELEM if the tracer is located inside ELEM. LT.0: Data are not output for ELEM if the tracer is located inside ELEM.
TYPM	ELEM type: EQ.1: Solid EQ.2: Beam EQ.3: Shell EQ.4: Tshell
MOVE	Flag to define how the tracer moves (see Remark 1): EQ.0: The tracer does not move with ELEM. EQ.1: The tracer velocity is interpolated from ELEM nodal velocities. EQ.2: The tracer position is interpolated from ELEM nodal positions.
SET	Element set for which the data are output by the tracer (see Remark 2)
TYPS	SET type: EQ.0: Part EQ.1: Solid EQ.2: Beam EQ.3: Shell EQ.4: Tshell
COR	Flag to correct the tracer position if it is found outside the tracked mesh during the initialization: EQ.0: No correction EQ.1: The tracer is moved to the center of the closest element.

VARIABLE	DESCRIPTION
	EQ.2: The tracer is moved to the nearest face center or node of the closest element.
DT	Time interval between outputs (see Remark 3)
TBEG	Time to start the output
TEND	Time to stop the output
FID	ID to be appended to trcrgal_binout (see Remark 3)
VARLOC	Variable location in trcrgal_binout to be replaced with the variable specified in the VAREPL field: EQ.4: *x*-velocity EQ.5: *y*-velocity EQ.6: *z*-velocity EQ.7: *xx*-stress EQ.8: *yy*-stress EQ.9: *zz*-stress EQ.10: *xy*-stress EQ.11: *yz*-stress EQ.12: *zx*-stress EQ.13: Plastic strain EQ.14: Nodal mass EQ.15: Undefined GE.16 and LE.30: Other auxiliary variables
VAREPL	Data to be output to the trcrgal_binout file instead of the variable located at VARLOC. The interpretation of VAREPL is enumerated in the following list: EQ.1: *x*-acceleration EQ.2: *y*- acceleration EQ.3: *z*- acceleration EQ.4: Nodal temperature EQ.5: Density EQ.6: Compression ratio

VARIABLE	DESCRIPTION
	EQ.7: Pressure

Remarks:

1. **NODE, ELEM, and MOVE.** A node represents the tracer. The initial location of the tracer is given by the nodal coordinates of the node defined in *NODE like for any node. If NODE = 0 and ELEM is defined, a node is added at the center of the element ELEM. If NODE is defined and ELEM = 0, the closest element to the tracer found during the initialization is used for ELEM. NODE and ELEM cannot both be zero. At least one of them should be provided to position the tracer. If MOVE > 0, the tracer will move with ELEM, even if NODE is not in ELEM.

2. **ELEM and SET.** If SET > 0, data will be output for the element in which the tracer is located if the element belongs to the set (if TYPS = 0, it will be the elements in the parts in SET). If SET = 0 and ELEM > 0, the elements of the same type (TYPM) as ELEM will be the output focus. If SET = 0 and ELEM ≤ 0, data will be output for any solid, beam, shell and thick shell.

3. **DT and FID.** DT is the time interval between data outputs to the file trcrgal_binout. If DT = 0.0 and *DATABASE_TRHIST is defined, the time step in this keyword will be used for DT. If DT = 0.0, data will be output every computational cycle. If there are several *DATABASE_TRACER_GENERAL keywords with different DT values, outputs from tracers with identical time steps will be grouped in the same file with a unique ID added to the end of trcrgen_binout. This ID can be provided by the user with FID. Two keywords with different DT but identical FID will terminate the job with an error.

4. **Post-processing.** The output of *DATABASE_TRACER_GENERATE is written to a file named trcrgal_binout. To access the output in LS-PrePost: *Post* (or *TAB 2*) → *Binout* → *trcrgal_binout* → *traleh* → *"Traleh Branch"* window contains a list of variables output for each tracer.

5. **Binary to ASCII file conversion.** The variables in traleh and trcrgal_binout are arranged in an identical order. Therefore, the traleh can be obtained from the trcrgal_binout file by using the l2a program.

*DATABASE_TRACER_GENERATE

Purpose: Generate tracer particles along an isosurface for a variable defined in the VALTYPE list. The tracer particles follow the motion of this surface and save data histories into a binary file called trcrgen_binout (see Remarks 4 and 5). These histories are identical to the ones output by *DATABASE_TRACER into the trhist file; they include positions, velocities, and stress components. Except for the positions and element ID specifying where the tracer is, the output can be controlled with the VARLOC and VALTYPE2 fields. This option applies to ALE problems.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	VALOW	VALUP	VALTYPE1	SET	SETYPE	MMGSET	UPDT
Type	F	F	F	I	I	I	I	F
Default	none	0.0	0.0	none	0	0	none	DT

Optional Variable Cards. Cards defining new variables to be output to trcrgen_binout instead of the default ones. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	VARLOC	VALTYPE2	MMGSET					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
DT	Interval time between each tracer generation and position update (See Remark 1).
VALOW, VALUP	Range of values between which the isosurface is defined. VALOW is the lower bound while VALUP is the upper bound. See Remark 2. The value at the isosurface is 0.5(VALOW + VALUP). The variable with this value is defined by VALTYPE.

VARIABLE	DESCRIPTION
VALTYPE1	The variable that will be used to generate the isosurfaces. See VALTYPE2 for enumeration of values.
VALTYPE2	Data to be output to the trcrgen_binout file. The interpretation of VALTYPE1 and VALTYPE2 is enumerated in the following list:

EQ.1:	*xx*-stress
EQ.2:	*yy*-stress
EQ.3:	*zz*-stress
EQ.4:	*xy*-stress
EQ.5:	*yz*-stress
EQ.6:	*zx*-stress
EQ.7:	plastic strain
EQ.8:	internal energy
EQ.9:	bulk viscosity
EQ.10:	relative volume
GE.11 and LE.19:	other auxiliary variables
EQ.20:	pressure
EQ.21:	density
EQ.22:	material volume
EQ.23:	compression ratio
EQ.24:	element volume fraction
EQ.25:	nodal volume fraction
EQ.26:	*x*-position
EQ.27:	*y*-position
EQ.28:	*z*-position
EQ.29:	*x*-velocity
EQ.30:	*y*-velocity
EQ.31:	*z*-velocity
EQ.31:	velocity
EQ.33:	*x*-acceleration
EQ.34:	*y*- acceleration

VARIABLE	DESCRIPTION
	EQ.35: z- acceleration
	EQ.36: acceleration
	EQ.37: nodal mass
	EQ.38: nodal temperature
SET	Set ID (See Remark 2)
SETYPE	Type of set (See Remark 2):
	EQ.0: solid set
	EQ.1: segment set
	EQ.2: node set
MMGSET	Multi-material group set (See Remark 3).
UPDT	Time interval between tracer position update (See Remark 1).
VARLOC	Variable location in trcrgen_binout to be replaced with the variable specified in the VALTYPE2 field:
	EQ.4: x-velocity
	EQ.5: y-velocity
	EQ.6: z-velocity
	EQ.7: xx-stress
	EQ.8: yy-stress
	EQ.9: zz-stress
	EQ.10: xy-stress
	EQ.11: yz-stress
	EQ.12: zx-stress
	EQ.13: plastic strain
	EQ.14: density
	EQ.15: relative volume

Remarks:

1. **DT.** The frequency to create tracers is defined by DT. The default value of UPDT, which is the time interval between updates to the tracer position, is also set

to DT. The default behavior, then, is to update tracer positions when a new tracer is created; however, by setting UPDT to a value less than DT tracer positions can be updated more frequently without creating new tracers.

2. **Tracing algorithm.** When LS-DYNA adds new tracer particles (see DT), tracers are created at element centers, segment centers, or nodes depending on the set type (SETYPE). A new tracer particle is created when the value at the element center, segment center, or node center is in the bounding interval [VALOW, VALUP], provided that there is *not* already a nearby tracer particle. The tracer particles follow the isosurface defined by the midpoint of the bounding interval (VALOW + VALUP)/2.

3. **Multi-material groups.** ALE elements can contain several materials. Each material is referred to as an ALE multi-material group. The volume fractions define how much of the element volume is occupied by the groups. Each group has their own variables for 0 < VALTYPE < 26. If VALTYPE < 21 or VALTYPE = 23, the variable is volume averaged over the groups defined by MMGSET.

4. **Post-processing.** The output of *DATABASE_TRACER_GENERATE is written to a file named trcrgen_binout. To access the output in LS-PrePost: *TAB 2* → *LOAD* → *trcrgen_binout* → *trhist* → *"Trhist Data"* window contains a list of variables output for each tracer.

5. **Binary to ASCII file conversion.** The variables in trhist and trcrgen_binout are arranged in an identical order. Therefore, the trhist can be obtained from the trcgen_binout file by using the l2a program packaged with the executable.

.

*DEFINE

The keyword *DEFINE provides a way of defining boxes, coordinate systems, load curves, functions, tables, transformations, and orientation vectors. Some of the *DEFINE keywords also assist with particle methods, airbags, failure, metal forming, welds, generalized elements, and contact. The keyword cards in this section are defined in alphabetical order:

*DEFINE_ADAPTIVE_SOLID_TO_DES

*DEFINE_ADAPTIVE_SOLID_TO_SPH

*DEFINE_BEAM_SOLID_COUPLING

DEFINE_BIRTH_*OPTION

*DEFINE_BOLTED_JOINTS

*DEFINE_BOX

*DEFINE_BOX_ADAPTIVE

*DEFINE_BOX_COARSEN

*DEFINE_BOX_DRAWBEAD

*DEFINE_BOX_NODES_ADAPTIVE

*DEFINE_BOX_SPH

*DEFINE_CABLE

*DEFINE_CONNECTION_PROPERTIES

*DEFINE_CONSTRUCTION_STAGES

*DEFINE_CONTACT_EXCLUSION

*DEFINE_CONTACT_VOLUME

*DEFINE_CONTROL_VOLUME

*DEFINE_CONTROL_VOLUME_FLOW_AREA

*DEFINE_CONTROL_VOLUME_INTERACTION

*DEFINE_COORDINATE_NODES

*DEFINE_COORDINATE_SYSTEM

*DEFINE_COORDINATE_VECTOR

*DEFINE_CPG_ADAPTIVE

*DEFINE_CPG_GAS_PROPERTIES

*DEFINE_CPG_REGION

*DEFINE_CPG_VENT

*DEFINE_CPM_BAG_INTERACTION

*DEFINE_CPM_CHAMBER

*DEFINE_CPM_GAS_PROPERTIES

*DEFINE_CPM_NPDATA

*DEFINE_CPM_SWITCH_REGION

*DEFINE_CPM_VENT

*DEFINE_CURVE

*DEFINE_CURVE_BOX_ADAPTIVITY

DEFINE_CURVE_COMPENSATION_CONSTRAINT_*OPTION

*DEFINE_CURVE_DRAWBEAD

*DEFINE_CURVE_DUPLICATE

*DEFINE_CURVE_ENTITY

*DEFINE_CURVE_FEEDBACK

*DEFINE_CURVE_FLC

*DEFINE_CURVE_FLD_FROM_TRIAXIAL_LIMIT

*DEFINE_CURVE_FUNCTION

*DEFINE_CURVE_SMOOTH

*DEFINE_CURVE_STRESS

*DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_FLD

*DEFINE_CURVE_TRIM

*DEFINE_DE_ACTIVE_REGION

*DEFINE_DE_BOND

*DEFINE_DE_BOND_OVERRIDE

*DEFINE_DE_BY_PART

*DEFINE_DE_COHESIVE

*DEFINE_DE_FLOW_DRAG

*DEFINE_DE_HBOND

*DEFINE_DE_INJECT_BONDED

*DEFINE_DE_INJECT_SHAPE

*DEFINE_DE_INJECTION

*DEFINE_DE_INTERNAL_SKIP

*DEFINE_DE_MASSFLOW_PLANE

*DEFINE_DE_MESH_BEAM

*DEFINE_DE_MESH_SURFACE

*DEFINE_DE_MESH_VOLUME

*DEFINE_DE_PARTS_INTERACTION

*DEFINE_DE_PATTERN_OUTPUT

*DEFINE_DE_TEMP

*DEFINE_DE_TO_BEAM_COUPLING

*DEFINE_DE_TO_CPG_COUPLING

*DEFINE_DE_TO_CPM_COUPLING

*DEFINE_DE_TO_SURFACE_COUPLING

*DEFINE_DE_TO_SURFACE_TIED

DEFINE_DEATH_TIMES_*OPTION

*DEFINE_DRIFT_REMOVE

DEFINE_ELEMENT_DEATH_*OPTION

DEFINE_ELEMENT_EROSION_*OPTION

*DEFINE_ELEMENT_GENERALIZED_SHELL

*DEFINE_ELEMENT_GENERALIZED_SOLID

*DEFINE_FABRIC_ASSEMBLIES

*DEFINE_FIBERS

*DEFINE_FIELD

*DEFINE_FILTER

*DEFINE_FORMING_ADAPTIVE_PART_INITIAL

*DEFINE_FORMING_BLANKMESH

*DEFINE_FORMING_CLAMP

*DEFINE_FORMING_CONTACT

*DEFINE_FORMING_ONESTEP_PRIMARY

*DEFINE_FP_TO_SURFACE_COUPLING

*DEFINE_FRICTION

*DEFINE_FRICTION_ORIENTATION

*DEFINE_FRICTION_SCALING

*DEFINE_FUNCTION

*DEFINE_FUNCTION_TABULATED

*DEFINE_GROUND_MOTION

*DEFINE_HAZ_PROPERTIES

*DEFINE_HAZ_TAILOR_WELDED_BLANK

*DEFINE_HEX_SPOTWELD_ASSEMBLY

*DEFINE_ISPG_TO_SURFACE_COUPLING

*DEFINE_LANCE_SEED_POINT_COORDINATES

*DEFINE_MATERIAL_HISTORIES

*DEFINE_MULTI_DRAWBEADS_IGES

*DEFINE_MULTI_SHEET_CONNECTORS

*DEFINE_MULTISCALE

*DEFINE_NURBS_CURVE

*DEFINE_PART_FROM_LAYER

*DEFINE_PARTICLE_BLAST

*DEFINE_PBLAST_AIRGEO

*DEFINE_PBLAST_GEOMETRY

*DEFINE_PLANE

*DEFINE_POINT_CLOUD

DEFINE_POROUS_*OPTION

*DEFINE_PRESSURE_TUBE

*DEFINE_QUASAR_COUPLING

*DEFINE_REGION

*DEFINE_SALEAB_PARAMETERS

*DEFINE_SD_ORIENTATION

*DEFINE_SET_ADAPTIVE

*DEFINE_SHELL_SWITCH

*DEFINE_SPG_TO_SPG_COUPLING

*DEFINE_SPG_TO_SURFACE_COUPLING

*DEFINE_SPH_ACTIVE_REGION

*DEFINE_SPH_AMBIENT_DRAG

*DEFINE_SPH_DE_COUPLING

*DEFINE_SPH_INJECTION

*DEFINE_SPH_INJECTION_SIMPLIFIED

*DEFINE_SPH_MASSFLOW_PLANE

*DEFINE_SPH_MESH_BOX

*DEFINE_SPH_MESH_OBJ

*DEFINE_SPH_MESH_SURFACE

*DEFINE_SPH_TO_SPH_COUPLING

*DEFINE_SPH_VICINITY_SENSOR

*DEFINE_SPOTWELD_FAILURE

*DEFINE_SPOTWELD_FAILURE_RESULTANTS

*DEFINE_SPOTWELD_MULTISCALE

*DEFINE_SPOTWELD_RUPTURE_PARAMETER

*DEFINE_SPOTWELD_RUPTURE_STRESS

*DEFINE_STAGED_CONSTRUCTION_PART

DEFINE_STOCHASTIC_ELEMENT_*OPTION

*DEFINE_STOCHASTIC_VARIATION

*DEFINE_STOCHASTIC_VARIATION_PROPERTIES

*DEFINE_TABLE

*DEFINE_TABLE_2D

*DEFINE_TABLE_3D

*DEFINE_TABLE_*{X}*D

*DEFINE_TABLE_MATRIX

*DEFINE_TARGET_BOUNDARY

*DEFINE_TRACER_PARTICLES_2D

*DEFINE_TRANSFORMATION

*DEFINE_TRIM_SEED_POINT_COORDINATES

*DEFINE_TWINBUILDER_COUPLING

*DEFINE_VECTOR

*DEFINE_VECTOR_NODES

Unless noted otherwise, an additional keyword option TITLE may be appended to the *DEFINE keywords. If this option is used, then an addition line is read for each section in 80a format which can be used to describe the defined curve, table, etc. At present, the title serves no purpose other than to perhaps lend clarity to input decks.

Examples for the *DEFINE keyword can be found at the end of this section.

*DEFINE_ADAPTIVE_SOLID_TO_DES_{OPTION}

Purpose: Adaptively transform a Lagrangian solid part or part set to DES (Discrete Element Sphere) particles (elements) when the Lagrangian solid elements comprising those parts fail. One or more DES particles will be generated for each failed element as debris. The DES particles replacing the failed element inherit the properties of the failed solid element, including mass and kinematical state.

The available options include:

<BLANK>

ID

ID Card. Additional card for the ID keyword option.

Optional	1	2	3	4	5	6	7	8
Variable	DID	HEADING						
Type	I	A70						
Default	none	none						

Card 1	1	2	3	4	5	6	7	8
Variable	IPID	ITYPE	NQ	IPDES	ISDES	RSF	OUTDES	IBOND
Type	I	I	I	I	I	F	I	I
Default	none	none	none	none	none	1.0	0	0

Bonds Card. This card is only included when IBOND = 1.

Card 2	1	2	3	4	5	6	7	8
Variable	PBN	PBS	PBN_S	PBS_S	SFA	ALPHA		
Type	F	F	F	F	F	F		
Default	none	none	none	none	1.0	0.0		

VARIABLE	DESCRIPTION
DID	Definition ID. This must be a unique number.
HEADING	Definition descriptor. It is suggested that unique descriptions be used.
IPID	ID of the solid part or part set to transform.
ITYPE	IPID type: EQ.0: Part ID NE.1: Part set ID
NQ	Adaptive option for hexahedral elements. For tetrahedral and pentahedral elements, see Remark 1. EQ.1: Adapt one solid element to one discrete element EQ.2: Adapt one solid element to 8 discrete elements EQ.3: Adapt one solid element to 27 discrete elements
IPDES	Part ID for newly generated discrete elements. See Remark 2.
ISDES	Section ID for discrete elements. See Remark 2.
RSF	DES radius scale down factor, which is the ratio of the radius of the generated DES to the calculated radius based on volume consistency.
OUTDES	Allow user output generated discrete element nodes and DES properties to a keyword file. EQ.0: No output (default). EQ.1: Write data under filename desvfill.inc.

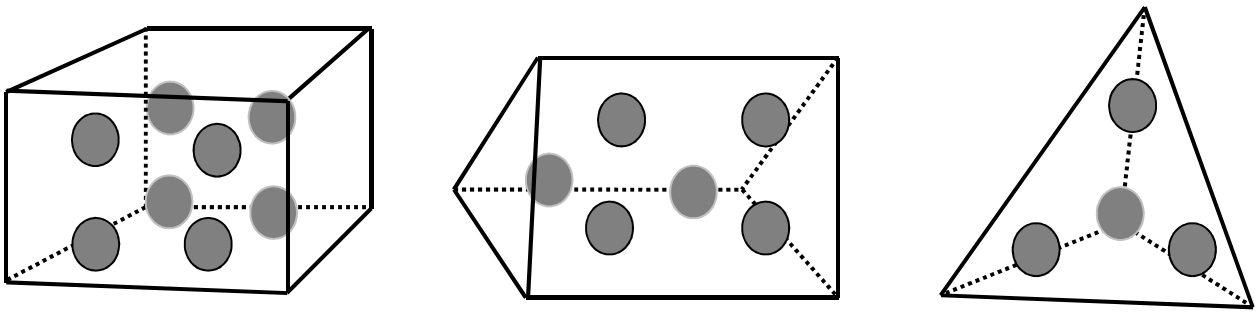


Figure 17-1. Left to right, illustration of conversion from solid to DES for NQ = 2 of hexahedron, pentahedron, and tetrahedron elements.

VARIABLE	DESCRIPTION
IBOND	Allow user define bonds between DES generated from the same solid element. EQ.0: No bonds (default). EQ.1: Bonds generated, need to define Card 2.
PBN	Parallel-bond modulus [Pa]. See Remark 3.
PBS	Parallel-bond stiffness ratio. Shear stiffness/normal stiffness. See Remark 3.
PBN_S	Parallel-bond maximum normal stress. A zero value defines an infinite maximum normal stress.
PBS_S	Parallel-bond maximum shear stress. A zero value defines an infinite maximum shear stress.
SFA	Bond radius multiplier. Default is 1.0.
ALPHA	Numerical damping

Remarks:

1. **DES Element to Solid Element Ratio.** The DES particles are evenly distributed within the solid element. For hexahedral elements the number of the generated DES particles is NQ × NQ × NQ. For pentahedral elements, the number of generated DES particles is 1, 6, and 18 for NQ = 1, 2, and 3, respectively. For tetrahedral elements, the number of generated DES particles is 1, 4, and 10 for NQ = 1, 2, and 3, respectively. See Figure 17-1.

2. **Part ID.** The part ID for newly generated DES particles can be either a new part ID or the ID of an existing DES part.

3. **Bond Forces.** The normal force between two bonded discrete elements with radii r_1 and r_2 is calculated as

$$\Delta f_{\mathrm{n}} = \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_n \ ,$$

where

$$A = \pi r_{\mathrm{eff}}^2$$
$$r_{\mathrm{eff}} = \min(r_1, r_2) \times \mathrm{SFA}$$

The shear force is calculated as

$$\Delta f_{\mathrm{s}} = \mathrm{PBS} \times \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_s \ .$$

*DEFINE_ADAPTIVE_SOLID_TO_SPH_{OPTION}

Purpose: Create SPH particles to either replace or supplement solid Lagrangian elements.

Applications of this feature include adaptively transforming a Lagrangian solid Part or Part Set to SPH particles when the Lagrangian solid elements comprising those parts fail. One or more SPH particles (elements) will be generated for each failed element. The SPH particles replacing the failed solid Lagrangian elements inherit all the Lagrange nodal quantities and all the Lagrange integration point quantities of these failed solid elements. Those properties are assigned to the newly activated SPH particles. The constitutive properties assigned to the new SPH part will correspond to the MID and EOSID referenced by the SPH *PART definition. This keyword with options ICPL = 0 and 1 has been extended to 2D cases too.

The available options include:

<BLANK>

ID

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	DID	HEADING						
Type	I	A70						
Default	none	none						

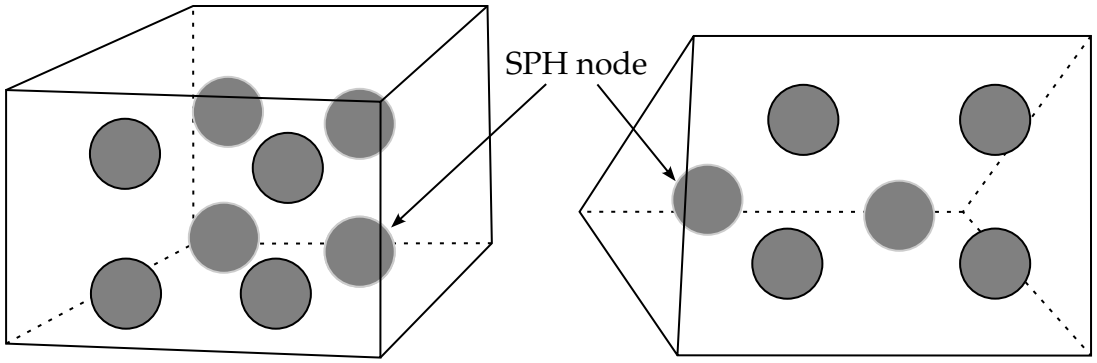
Card 1	1	2	3	4	5	6	7	8
Variable	IPID	ITYPE	NQ	IPSPH	ISSPH	ICPL	IOPT	CPCD
Type	I	I	I	I	I	I	I	F
Default	none	none	none	none	none	none	none	average

VARIABLE	DESCRIPTION
DID	Definition ID. This must be a unique number.

VARIABLE	DESCRIPTION
HEADING	Definition descriptor. We suggest using unique descriptions.
IPID	ID of the solid part or part set to transform
ITYPE	IPID type: EQ.0: Part ID NE.0: Part set ID
NQ	Adaptive option for hexahedral elements. For tetrahedral and pentahedral elements, see Remark 1: EQ.n: Adapt one 8-node solid element to $(n \times n \times n)$ SPH elements. The range of n is from 1 to 6.
IPSPH	Part ID for newly generated SPH elements. See Remark 2.
ISSPH	Section ID for SPH elements. See Remark 2.
ICPL	Coupling of newly generated SPH elements to the adjacent solid elements: EQ.0: Failure without coupling (debris simulation), EQ.1: Coupled to solid element, EQ.3: Provide only thermal coupling between SPH part and solid part (must be combined with IOPT = 0 option; see Remark 4).
IOPT	Coupling method: EQ.0: Coupling from beginning (used as constraint between SPH elements and solid elements), EQ.1: Coupling begins when Lagrangian solid element fails. See Remark 3.
CPCD	Thermal coupling conductivity between SPH part and solid part for ICPL = 3 option. The default value is set as the average value of the conductivity from SPH part and the conductivity from solid part.

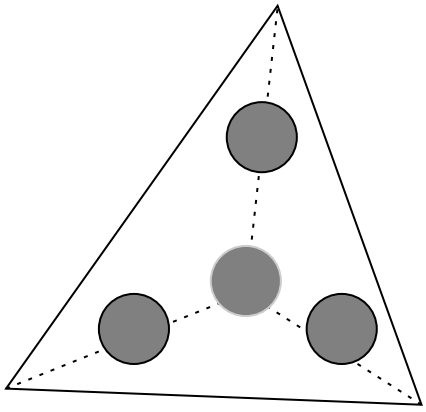
Remarks:

1. **Number of SPH particles per element.** The SPH particles are evenly distributed within the solid element. For hexahedral elements, the number of the generated SPH particles is NQ × NQ × NQ. For pentahedral elements, the number



Example of SPH nodes for a hexahedral element with NQ = 2

Example of SPH nodes for a pentahedral element with NQ = 2



Example of SPH nodes for a tetrahedral element with NQ = 2

Figure 17-2. Examples of distributions of SPH particles converted from solid Lagrangian elements.

of generated SPH particles is 1, 6, and 18 for NQ equal to 1, 2, and 3, respectively. For tetrahedral elements, the number of generated SPH particles is 1, 4, and 10 for NQ equal to 1, 2, and 3, respectively. See Figure 17-2.

2. **SPH part ID**. The Part ID for newly generated SPH particles can be either a new Part ID or the ID of an existing SPH Part. For constraint coupling, such as when ICPL = 1 and IOPT = 0, the newly generated SPH part ID should be different from the existing one.

3. **Mechanical failure coupling.** ICPL = 0 is used for debris simulation, so no coupling happens between newly generated SPH particles and solid elements; the user, therefore, needs to define node-to-surface contact for the interaction between those two parts. When ICPL = 1 and IOPT = 1, the newly generated SPH particles are bonded with solid elements as one part through the coupling, and

the new material ID with different failure criteria can be applied to the newly generated SPH particles.

4. **Thermal coupling.** ICPL = 3, which must be combined with IOPT = 0, is used to thermally couple an SPH part and solid part(s). There is no structural coupling provided. A thermal conductivity value may be defined using the variable CPCD.

5. **Output.** This keyword reports both active and inactive adaptive SPH in the fragment file sldsph_frag. This file contains nodal masses, coordinates, and velocities.

*DEFINE_BEAM_SOLID_COUPLING

Purpose: To define a coupling interface between embedded structure and a block defined by solid(s) without sharing nodes.

Card 1	1	2	3	4	5	6	7	8
Variable	LSTRID	SOLID	LSTRTYPE	SOLTYPE	FORM	PSF		
Type	I	I	I	I	I	F		
Default	none	none	0	0	0	1.		

VARIABLE	DESCRIPTION
LSTRID	Part set ID or part ID of the Lagrangian structure. LSTRTYPE below indicates the ID type specified by LSTRTYPE.
SOLID	Part set ID or part ID of the solid block. SOLTYPE below indicates the ID type specified by SOLID.
LSTRTYPE	Type of Lagrangian structure set: EQ.0: Part set EQ.1: Part
TYPE	Type of solid set: EQ.0: Part set EQ.1: Part
Form	Coupling type: EQ.0: Constrained acceleration and velocity EQ.1: Penalty tied in all directions
PSF	Scale factor for penalty stiffness

DEFINE_BIRTH_*OPTION

Purpose: Define the birth of a particular feature. For each unique birth time, include one line of data. The input ends at the following keyword ("*") card.

Available options include:

PART

Currently, only solid cohesive parts are supported.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	BIRTH	DRTIME					
Type	I	F	I					
Default	none	0.0	0					

VARIABLE	DESCRIPTION
PID	Part ID to which the birth time applies LT.0.0: |PID| is the ID for a part set containing the affected parts.
BIRTH	Birth time
DRTIME	Controls whether dynamic relaxation pseudo time is to be accounted for or not: EQ.0: The birth time is in relation to the transient time. In other words, the dynamic relaxation time is ignored. For example, if BIRTH = 0.0 and DRTIME = 0, the feature is activated at time zero of the transient run. This is the default. EQ.1: The birth time is in relation to dynamic relaxation pseudo time. If the dynamic relaxation ends before BIRTH is reached, the feature is activated at time zero of the transient run.

*DEFINE_BOLTED_JOINTS

Purpose: Define a group of bolted joints consisting of beam elements and hole nodes. Each bolted joint can contain many bolts and holes with different geometries (such as bolt diameter and length) but has the properties defined in the data cards below.

During initialization, the beams and nodes are sorted such that all connected beams are defined as one bolt and the closest nodes as their respective hole. A group of connected beams define a bolt shaft accompanied by an analytically defined nut and head at the end nodes. The nut side of each bolt is the end with the smallest end node ID.

Card Summary:

Card Sets. For each unique bolted joint, include one set of data cards. The input ends at the following keyword ("*") card.

Card 1. This card is required.

BJID	SLSFAC	FRICTION	VDC				

Card 2. This card is required.

BSID	HDANG	NTANG	HDDIAM	NTDIAM			

Card 3. This card is required.

NSID	HDDEPT	NTDEPT					

Card 1	1	2	3	4	5	6	7	8
Variable	BJID	SLSFAC	FRICTION	VDC				
Type	I	F	F	F				
Default	none	1.0	0.0	0.0				

VARIABLE	DESCRIPTION
BJID	Bolted joint ID. It should be unique from other bolted joints.
SLSFAC	Optional scale factor for the bolted joint sliding interface penalty stiffness.
FRICTION	Friction coefficient. The default is no friction.

VARIABLE	DESCRIPTION
VDC	Viscous damping coefficient as a fraction of the critical viscous damping coefficient. The applied damping coefficient is given by: $$\xi = \text{VDC}\ \xi_{\text{crit}}\ ,$$ where VDC is a value between 0.0 and 1.0. The formula for critical damping is: $$\xi_{\text{crit}} = 2m\omega,$$ where m is determined by nodal masses as $$m = \min(m_{\text{tracked}}, m_{\text{reference}})\ ,$$ and ω is determined from k, the interface stiffness, according to $$\omega = \sqrt{k\frac{m_{\text{tracked}} + m_{\text{reference}}}{m_{\text{tracked}} m_{\text{reference}}}}$$

Bolts Card. This card contains the beam set defining the bolts and the head and nut geometries.

Card 2	1	2	3	4	5	6	7	8
Variable	BSID	HDANG	NTANG	HDDIAM	NTDIAM			
Type	I	F	F	F	F			
Default	none	0.0	0.0	↓	↓			

VARIABLE	DESCRIPTION
BSID	Beam set ID containing bolt beams. Only those beam elements in this set will be considered as parts of the bolts in this bolted joint. See Remark 1.
HDANG	Head countersunk angle in degrees
NTANG	Nut countersunk angle in degrees
HDDIAM	Head outer diameter. The default is twice the diameter of the closest beam.
NTDIAM	Nut outer diameter. The default is twice the diameter of the closest beam.

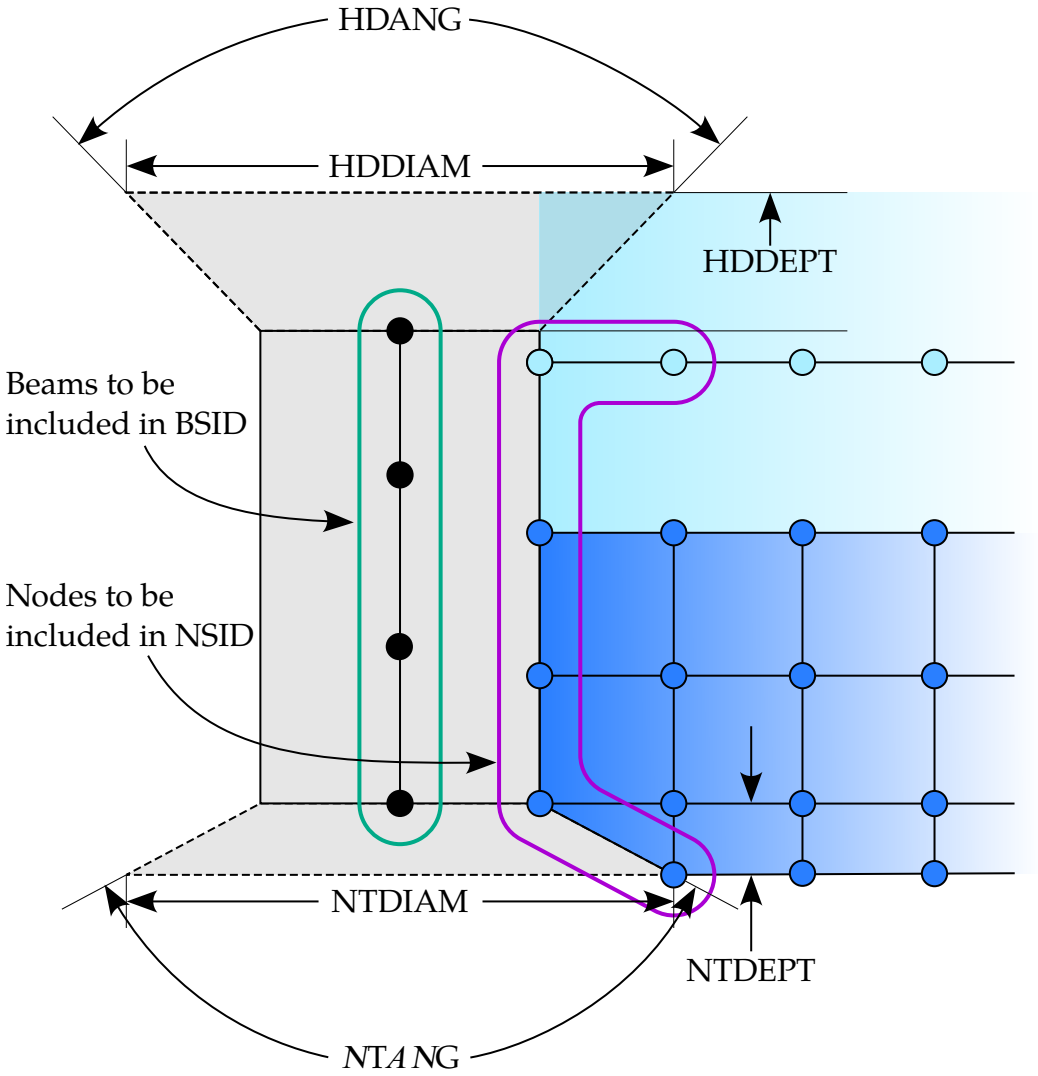


Figure 17-3. Definitions of bolt parameters. Note that in this example, only HDDEPT will have any effect since there are only shells at the head side.

Holes Card. This card contains the node set defining the holes and countersunk geometry. The countersunk angle is the same as the bolt head and nut, respectively.

Card 3a	1	2	3	4	5	6	7	8
Variable	NSID	HDDEPT	NTDEPT					
Type	I	F	F					
Default	none	0.0	0.0					

VARIABLE	DESCRIPTION
NSID	Node set ID containing nodes on the hole edges for this bolted joint. Only those nodes in this set will be used to define the hole in this bolted joint. See Remark 1.
HDDEPT	Countersunk depth at head side of hole. Note that this will only affect the shell elements that are closest to the head. Solid elements closest to the head must include the countersunk geometry. See Remark 2.
NTDEPT	Countersunk depth at nut side of hole. Note that this will only affect the shell elements that are closest to the nut. Solid elements closest to the nut must include the countersunk geometry. See Remark 2.

Remarks:

1. **Description of model.** As mentioned in the introduction, the beams and nodes of BSID and NSID are sorted such that all connected beams are defined as one bolt and the closest nodes as their respective hole. A group of connected beams defines a bolt shaft, accompanied by an analytically defined nut and head at the end nodes. The nut side of each bolt is the end with the smallest end node ID. To ensure a good contact definition of the deformed bolt, each bolt should consist of several beam elements.

   All nodes that can potentially interact with the bolt should be included in the NSID set. In other words, in the layers interacting with the nut or head, nodes within the radius of the nut and head should be included in the set.

2. **Countersunk depth and geometry.** The countersunk depth is measured at the edge of the hole, specifically at the node closest to the bolt axis. For shell nodes in the layers closest to a countersunk nut or head, the countersunk depth is automatically adjusted. Solid layers closest to a countersunk nut or head must include the countersunk geometry. Furthermore, we assume that:

   a) The surfaces closest to the nut and head are relatively flat (for shell elements, within 10% of shell thickness).

   b) When adjusting for countersunk geometry in shells, the countersunk depth is less than the thickness of the layer.

   c) Shell elements close to the bolt have the same orientation.

3. **Current limitations.** The following are current limitations in this model:

a) The bolt is assumed circular with uniform diameter.

b) Any initial penetrations will result in an initial contact force.

c) The set of nodes defining the hole is static. That is, it is not updated in case of eroding elements. If elements can erode, any nodes that could come into contact with the bolt must be included from the start.

*DEFINE_BOX_{OPTION}

Available options include:

<BLANK>

LOCAL

Purpose: Define a box-shaped volume. Two diagonally opposite corner points of a box are specified in global or local coordinates if the LOCAL option is active. The box volume is then used for various specifications for a variety of input options, such as velocities, contact, etc.

If the option, LOCAL, is active, a local coordinate system with two vectors (see Figure 17-14) is defined. The vector cross product, $\mathbf{z} = \mathbf{x} \times \mathbf{v}_{xy}$ where $\mathbf{v}_{xy}$ is a vector in the xy-plane, determines the local z-axis. The local y-axis is then given by $\mathbf{y} = \mathbf{z} \times \mathbf{x}$.

A point, P, in the global coordinate system is considered to lie within the volume of the box if the coordinate $P - C$, where C is the global coordinate offset vector defined on Card 3, lies within the box after transformation into the local system, $P_{\text{local}} = T \times (P - C)$. The local coordinate, P_{local}, is checked against the minimum and maximum coordinates defined on Card 1 in the local system.

For the *INCLUDE_TRANSFORM options that include translations, rotations and mirrors, all box options are automatically converted from *DEFINE_BOX_*xxxx* to *DEFINE_BOX_*xxxx*_LOCAL in the DYNA.INC file. Here, *xxxx* represents the box options: ADAPTIVE, COARSEN, and SPH, which are defined below. If the transformation matrix of *INCLUDE_TRANSFORM involves using negative values for SCALE, see *DEFINE_TRANSFORMATION, to mirror/reflect an object, the box might not be correctly transformed. Instead of SCALE, it is advised to use the MIRROR option with A7 = 1 for the purpose of mirroring/reflecting.

Card 1	1	2	3	4	5	6	7	8
Variable	BOXID	XMN	XMX	YMN	YMX	ZMN	ZMX	
Type	I	F	F	F	F	F	F	
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	

Local Card 1. First additional card for LOCAL keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	XX	YX	ZX	XV	YV	ZV		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Local Card 2. Second additional card for LOCAL keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	CX	CY	CZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
XMN	Minimum *x*-coordinate. Define in the local coordinate system if the option LOCAL is active.
XMX	Maximum *x*-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMN	Minimum *y*-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMX	Maximum *y*-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMN	Minimum *z*-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMX	Maximum *z*-coordinate. Define in the local coordinate system if the option LOCAL is active.

VARIABLE	DESCRIPTION
XX	X-coordinate on local x-axis. Origin lies at (0,0,0). Define if the LOCAL option is active.
YX	Y-coordinate on local x-axis. Define if the LOCAL option is active.
ZX	Z-coordinate on local x-axis. Define if the LOCAL option is active.
XV	X-coordinate of local xy-vector. Define if the LOCAL option is active.
YV	Y-coordinate of local xy-vector. Define if the LOCAL option is active.
ZV	Z-coordinate of local xy-vector. Define if the LOCAL option is active.
CX	X-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CY	Y-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CZ	Z-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.

*DEFINE_BOX_ADAPTIVE_{OPTION}

Available options include:

<BLANK>

LOCAL

Purpose: Define a box-shaped volume enclosing (1) the shells where the *h*-adaptive level is to be specified, or (2) the solids where the tetrahedral *r*-adaptive mesh size is to be specified. If the midpoint of the element falls within the box, the *h*-adaptive level is reset. The box can translate. It is also possible to define a fission box followed by a fusion box, in which case the mesh can refine when deformed and coarsen when flattened. Shells falling outside of this volume use the value, MAXLVL, on the *CONTROL_ADAPTIVE control cards. A related keyword is *DEFINE_CURVE_BOX_ADAPTIVITY. An alternative command in the case of shell *h*-adaptivity is *DEFINE_BOX_NODES_ADAPTIVE.

Card 1	1	2	3	4	5	6	7	8
Variable	BOXID	XMN	XMX	YMN	YMX	ZMN	ZMX	
Type	I	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	

Card 2	1	2	3	4	5	6	7	8
Variable	PID	LEVEL	LIDX/NDID	LIDY	LIDZ	BRMIN	BRMAX	
Type	I	I	I	I	I	F	F	
Default	0	1	0	0	0	0.0	0.0	

Local Card 1. First additional card for LOCAL keyword option. See *DEFINE_BOX for a description of the LOCAL option.

Card 3	1	2	3	4	5	6	7	8
Variable	XX	YX	ZX	XV	YV	ZV		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Local Card 2. Second additional card for LOCAL keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	CX	CY	CZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
XMN	Minimum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
XMX	Maximum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMN	Minimum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMX	Maximum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMN	Minimum z-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMX	Maximum z-coordinate. Define in the local coordinate system if the option LOCAL is active.

VARIABLE	DESCRIPTION
PID	Deformable part ID. EQ.0: all active elements within box are considered.
LEVEL	Maximum number of refinement levels for elements that are contained in the box. Values of 1, 2, 3, 4, ... allow for a maximum of 1, 4, 16, 64, ... elements, respectively, to be created for each original element.
LIDX/NDID	Load curve ID / Node ID (see Remark 1). GT.0: load curve ID. Define adaptive box movement (displacement as a function of time) in global X-axis. LT.0: absolute value is a node ID, whose translation will be followed by the moving adaptive box. The node ID can be on a moving rigid body. EQ.0: no movement
LIDY	Load curve ID (see Remark 1). GT.0: load curve ID. Define adaptive box movement (displacement as a function of time) in global Y-axis. EQ.0: no movement
LIDZ	Load curve ID (see Remark 1). GT.0: load curve ID. Define adaptive box movement (displacement as a function of time) in global Z-axis. EQ.0: no movement
BRMIN	Minimum mesh size in 3D tetrahedron adaptivity
BRMAX	Maximum mesh size in 3D tetrahedron adaptivity
XX	X-coordinate on local x-axis. Origin lies at (0,0,0). Define if the LOCAL option is active.
YX	Y-coordinate on local x-axis. Define if the LOCAL option is active.
ZX	Z-coordinate on local x-axis. Define if the LOCAL option is active.
XV	X-coordinate of local xy-plane vector. Define if the LOCAL option is active.

VARIABLE	DESCRIPTION
YV	*Y*-coordinate of local *xy*-plane vector. Define if the LOCAL option is active.
ZV	*Z*-coordinate of local *xy* -plane vector. Define if the LOCAL option is active.
CX	*X*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CY	*Y*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CZ	*Z*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.

Remarks:

1. **Moving Box.** The moving adaptive box is very useful and efficient in situations where deformations happens locally, such as roller hemming and incremental forming simulations. With the moving box feature, elements entering one box can be refined and fused together when they enter another box. Mesh fission outside of the moving box envelope is controlled by MAXLVL and other parameters under *CONTROL_ADAPTIVE. The fusion controls (NCFREQ, IADPCL) can be defined using *CONTROL_ADAPTIVE. Currently, only IADPCL = 1 is supported.

   Only when one of the LCIDX/NDID, LICDY, or LCIDZ is defined, the adaptive box will be moving; otherwise it will be stationary.

2. **3D *r*-Adaptivity.** For 3D tetrahedron *r*-adaptivity, the current implementation does not support the LOCAL option. In Card 2, LEVEL is not supported in 3D *r*-adaptivity.

Example:

Referring to a partial input deck below, and Figure 17-4, a strip of sheet metal is being roller hemmed. The process consists of pre- and final hemming. Each pre- and final roller is defined with a moving adaptive box with IDs 2 and 3, respectively. Figure 17-4 shows the box shapes. The first box, a fission box, has a LEVEL = 3 refinement, while the second box, a fusion box, was set at LEVEL = 1 refinement. Elements outside of the volume envelope made by the moving boxes undergo no fission and fusion (MAXLVL = 1). These settings cause mesh fission when the elements enter moving box 2 (LEVEL = 3) and mesh fusion only when the elements enter moving box 3 (LEVEL = 1). No fission/fusion

(MAXLVL = 1) occurs outside of the volume envelope created by the moving boxes. In the example, the boxes 2 and 3 move in the global *X*-direction for a distance of 398 mm as defined by load curve 11 and 450 mm as defined by load curve 12, respectively.

```
*CONTROL_TERMINATION
0.252
*CONTROL_ADAPTIVE
$  ADPFREQ    ADPTOL    ADPTYP    MAXLVL    TBIRTH    TDEATH     LCADP    IOFLAG
   8.05E-4  0.200000         2         1     0.0001.0000E+20         0         1
$  ADPSIZE    ADPASS    IREFLG    ADPENE     ADPTH    MEMORY    ORIENT     MAXEL
  0.300000         1         0       5.0
$  IADPN90              NCFREQ    IADPCL    ADPCTL    CBIRTH    CDEATH
        -1         0         1         1      10.0     0.000      10.30
*DEFINE_BOX_ADAPTIVE
$#   BOXID       XMN       XMX       YMN       YMX       ZMN       ZMX
         2 -10.00000 36.000000 -15.03000  3.991000  1.00E+00 48.758000
$#     PID     LEVEL LIDX/NDID      LIDY      LIDZ
         6         3        11
*DEFINE_BOX_ADAPTIVE
$#   BOXID       XMN       XMX       YMN       YMX       ZMN       ZMX
         3 -100.0000  -60.0000 -15.03000  3.991000  1.00E+00 48.758000
$#     PID     LEVEL LIDX/NDID      LIDY      LIDZ
         6         1        12
*DEFINE_CURVE
11
               0.000                 0.0
           0.00100000                1.0
           0.19900000              397.0
           0.20000000              398.0
                1.000              398.0
                   ⋮                   ⋮
*DEFINE_CURVE
12
                 0.0                 0.0
                0.05                 0.0
               0.051                 1.0
               0.251              401.0
               0.252              450.0
                   ⋮                   ⋮
```

A moving box can also follow the movement of a node, which can be on a moving rigid body. In this case, NDIDs for the motion of the boxes are defined instead of load curves. For example, in Figure 17-4 and a partial keyword example below, box 2 follows a node (ID: 33865) on the pre-roller, and box 3 follows another node (ID: 38265) on the final roller.

```
*DEFINE_BOX_ADAPTIVE
$#   BOXID       XMN       XMX       YMN       YMX       ZMN       ZMX
         2 -10.00000 36.000000 -15.03000  3.991000  1.00E+00 48.758000
$#     PID     LEVEL LIDX/NDID      LIDY       LID
         6         3    -33865
*DEFINE_BOX_ADAPTIVE
$#   BOXID       XMN       XMX       YMN       YMX       ZMN       ZMX
         3 -100.0000  -60.0000 -15.03000  3.991000  1.00E+00 48.758000
$#     PID     LEVEL LIDX/NDID      LIDY       LID
         6         3    -38265
```

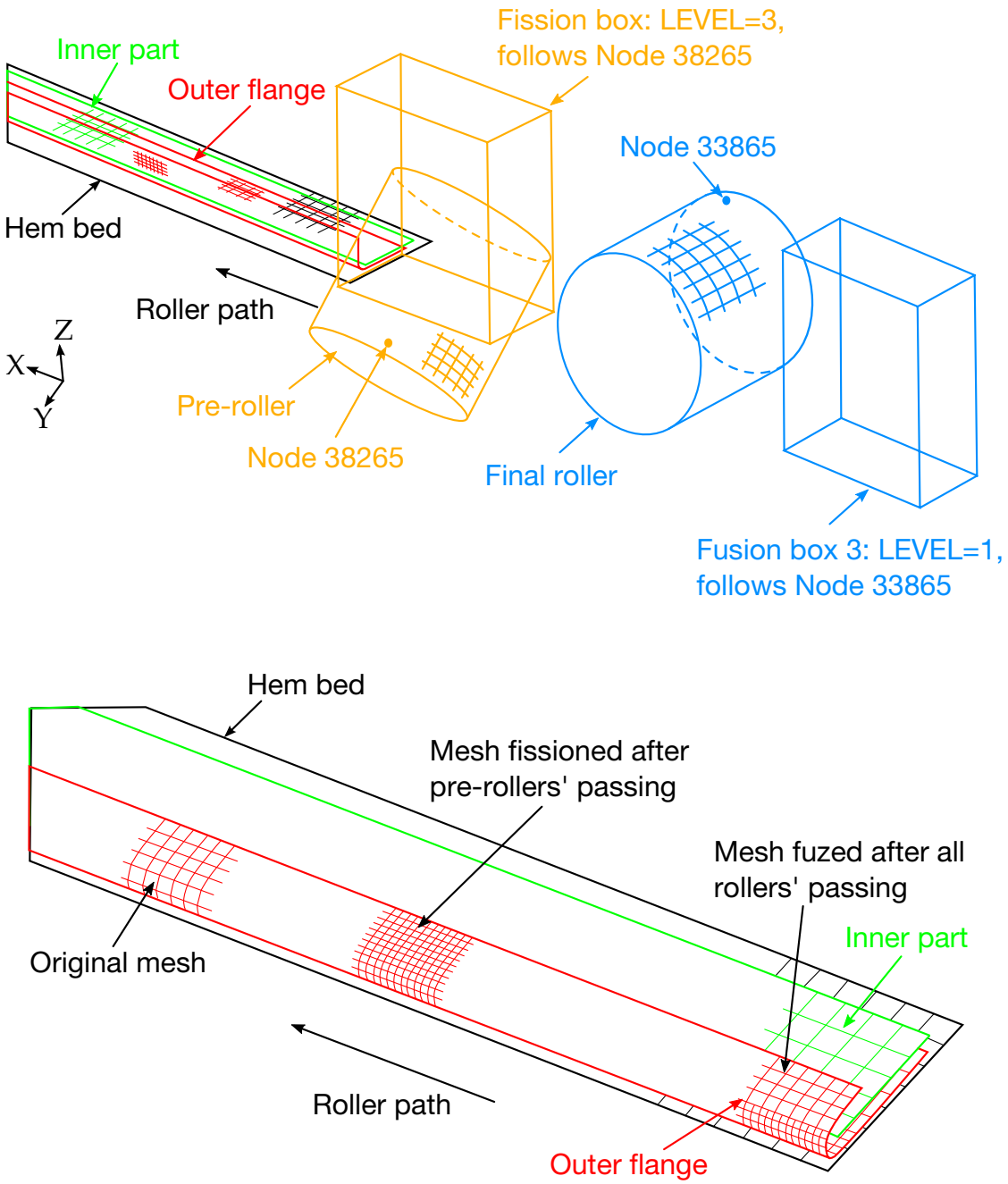


Figure 17-4. Defining mesh fission and fusion.

***DEFINE_BOX_COARSEN**_*{OPTION}*

Available options include:

<BLANK>

LOCAL

Purpose: Define a specific box-shaped volume indicating elements which are protected from mesh coarsening. See also *CONTROL_COARSEN.

Card	1	2	3	4	5	6	7	8
Variable	BOXID	XMN	XMX	YMN	YMX	ZMN	ZMX	IFLAG
Type	I	F	F	F	F	F	F	I
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0

Local Card 1. First additional card for LOCAL keyword option. See *DEFINE_BOX for a description of the LOCAL option.

Card 2	1	2	3	4	5	6	7	8
Variable	XX	YX	ZX	XV	YV	ZV		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Local Card 2. Second additional card for LOCAL keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	CX	CY	CZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
XMN	Minimum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
XMX	Maximum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMN	Minimum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMX	Maximum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMN	Minimum z-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMX	Maximum z-coordinate. Define in the local coordinate system if the option LOCAL is active.
IFLAG	Flag for protecting elements inside or outside of box: EQ.0: elements inside the box cannot be coarsened EQ.1: elements outside the box cannot be coarsened
XX	X-coordinate on local x-axis. Origin lies at (0,0,0). Define if the LOCAL option is active.
YX	Y-coordinate on local x-axis. Define if the LOCAL option is active.
ZX	Z-coordinate on local x-axis. Define if the LOCAL option is active.
XV	X-coordinate of local xy-vector. Define if the LOCAL option is active.
YV	Y-coordinate of local xy-vector. Define if the LOCAL option is active.
ZV	Z-coordinate of local xy-vector. Define if the LOCAL option is active.
CX	X-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.

VARIABLE	DESCRIPTION
CY	*Y*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CZ	*Z*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.

Remarks:

1. **Multiple Boxes.** Many boxes may be defined. If an element is protected by any box then it may not be coarsened.

*DEFINE_BOX_DRAWBEAD

Purpose: Define a specific box-shaped or tube-shaped volume around a draw bead. This option is useful for draw bead contact. If box-shaped, the volume will contain the draw bead nodes and elements between the bead and the outer edge of the blank. If tubular, the tube is centered around the draw bead. All elements within the tubular volume are included in the contact definition.

Card	1	2	3	4	5	6	7	8
Variable	BOXID	PID	SID	IDIR	STYPE	RADIUS	CID	
Type	I	I	I	I	I	F	I	
Default	0	0	0	3	4	0.0	0	

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
PID	Part ID of blank.
SID	Set ID that defines the nodal points that lie along the draw bead. If a node set is defined, the nodes in the set must be consecutive along the draw bead. If a part or part set is defined, the set must consist of beam or truss elements. Within the part set, no ordering of the elements is assumed, but the number of nodes must equal the number of beam elements plus 1.
IDIR	Direction of tooling movement. The movement is in the global coordinate direction unless the tubular box option is active and CID is nonzero. In this latter case, the movement is in the local coordinate direction. EQ.1: Tooling moves in the x-direction. EQ.2: Tooling moves in the y-direction. EQ.3: Tooling moves in the z-direction (default).
STYPE	Set type: EQ.2: Part set ID EQ.3: Part ID

VARIABLE	DESCRIPTION
	EQ.4: Node set ID
RADIUS	The radius of the tube, which is centered around the draw bead. Elements of part ID, PID, that lie within the tube will be included in the contact. If the radius is not defined, a rectangular box is used instead. This option is recommended for curved draw beads and for draw beads that are not aligned with the global axes.
CID	Optional coordinate system ID. This option is only available for the tubular draw bead.

*DEFINE_BOX_NODES_ADAPTIVE

Purpose: Define mesh fission and fusion at the beginning of an adaptive step according to the path of a moving tool and a set of parameters. This method applies only to shell *h*-adaptivity and is sometimes referred to as "tube" adaptivity due to its adaptive boundary shape. (This tube is more accurately described as a torus.) It can help reduce the computational time for incremental forming or roller hemming simulations while maintaining accuracy in the area of interest. This method is a major improvement over the keyword *DEFINE_BOX_ADAPTIVE (see Remarks and Figure 17-5). Related keywords include *DEFINE_CURVE_BOX_ADAPTIVITY and *CONTROL_ADAPTIVE_CURVE.

Card 1	1	2	3	4	5	6	7	8
Variable	BOXID	NODE	LCX	LCY	LCZ	ITYPE	RADIUS	NPIECE
Type	I	I	I	I	I	I	F	I
Default	none	none	none	none	none	none	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	PID	LEVEL						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
NODE	A reference node ID from which the tube will form in front of it, following the tool path described by LCX, LCY, and LCZ.
LCX, LCY, LCZ	Load curve IDs (see *DEFINE_CURVE) that define the path of the tool in the global *X*, *Y*, and *Z* directions, respectively.
ITYPE	Type of curves: LCX, LCY, and LCZ. Currently, only time as a function of displacement load curves are supported. EQ.2: LCX, LCY, and LCZ are defined as time as a function of

VARIABLE	DESCRIPTION
	displacement.
RADIUS	Radius of the tube that defines the fission/fusion boundary
NPIECE	Number of segments used to approximate the tool path in one adaptive step. Note that the tool's path is divided into several linear segments for approximation.
PID	The deformable part or part set ID on which the tube adaptivity is to be applied (see *PART). GT.0: Part ID LT.0: |PID| is a part set ID. A part set ID can be useful for simulating the forming of tailor-welded blanks.
LEVEL	Desired mesh refinement level. Level set to a value of 1, 2, 3, … allows a maximum of 1, 4, 16, … elements to be created for each original element in the "tube region".

Remarks:

Metal forming simulations for applications such as incremental forming and roller hemming are computationally intensive, primarily due to long tool paths and localized deformation. Several methods have been developed to increase the efficiency of these applications.

The keyword *DEFINE_BOX_ADAPTIVE was one of those methods. While it offered significant CPU time reduction, the adaptive fission box does not stay in front of the tool when the tool path turns around 180 degrees (see Figure 17-5). Additionally, the fusion box, which is supposed to remain behind the tool to coarsen the mesh, does not always stay behind the tool. This drawback is addressed by this keyword.

This keyword creates a tube with a user-specified radius (RADIUS) that encloses the tool path ahead and moves along with a reference node (NODE) located on the tool. The mesh is refined in the area in front of the tube (look-forward adaptivity). The tube moves with the tool. The formed mesh behind the tube is fused according to a user-specified criterion, thus saving CPU time (see Figure 17-6).

From a parameter study of this keyword [Zhu et. al.]. the following was determined:

- In comparison to a convergence study without this keyword, the simulation time was reduced by 46%.
- The optimum tube radius is 10 times the final element size.

- The optimum ADPFREQ (see *CONTROL_ADAPTIVITY) should be about 1.6% of the total simulation time. Note the ADPFREQ indirectly controls how far the tube extends in front of the tool.

Example:

To activate this feature, in addition to using this keyword, *CONTROL_ADAPTIVE also needs to be included in the input deck. See Figure 17-6 and the partial input below.

```
*CONTROL_ADAPTIVE
$  ADPFREQ    ADPTOL    ADPTYP    MAXLVL    TBIRTH    TDEATH     LCADP    IOFLAG
      0.08       1.0         2         2       0.0      13.6         0         1
$  ADPSIZE    ADPASS    IREFLG    ADPENE    ADPTH     MEMORY     ORIENT   MAXEL
                    1         0       0.0      0.0
$  IADPN90    IADPGH    NCFREQ    IADPCL    ADPCTL    CBIRTH     CDEATH   LCLVL
        -1         0         4         0       0.5       0.0       13.6
*DEFINE_BOX_NODES_ADAPTIVE
$       ID      NODE       LCX       LCY       LCZ     ITYPE     RADIUS    NPIECE
         2    114116         1         2         3         2       10.0         8
$      PID     LEVEL
         7         2
```

Reference:

Zhu, X.H., Fan, H.F., Zhang, L., and Xiao, Y.Z., Tube adaptivity for mesh fission/fusion in LS-DYNA, 2017 3rd China LS-DYNA Users' Conference, Shanghai, China.

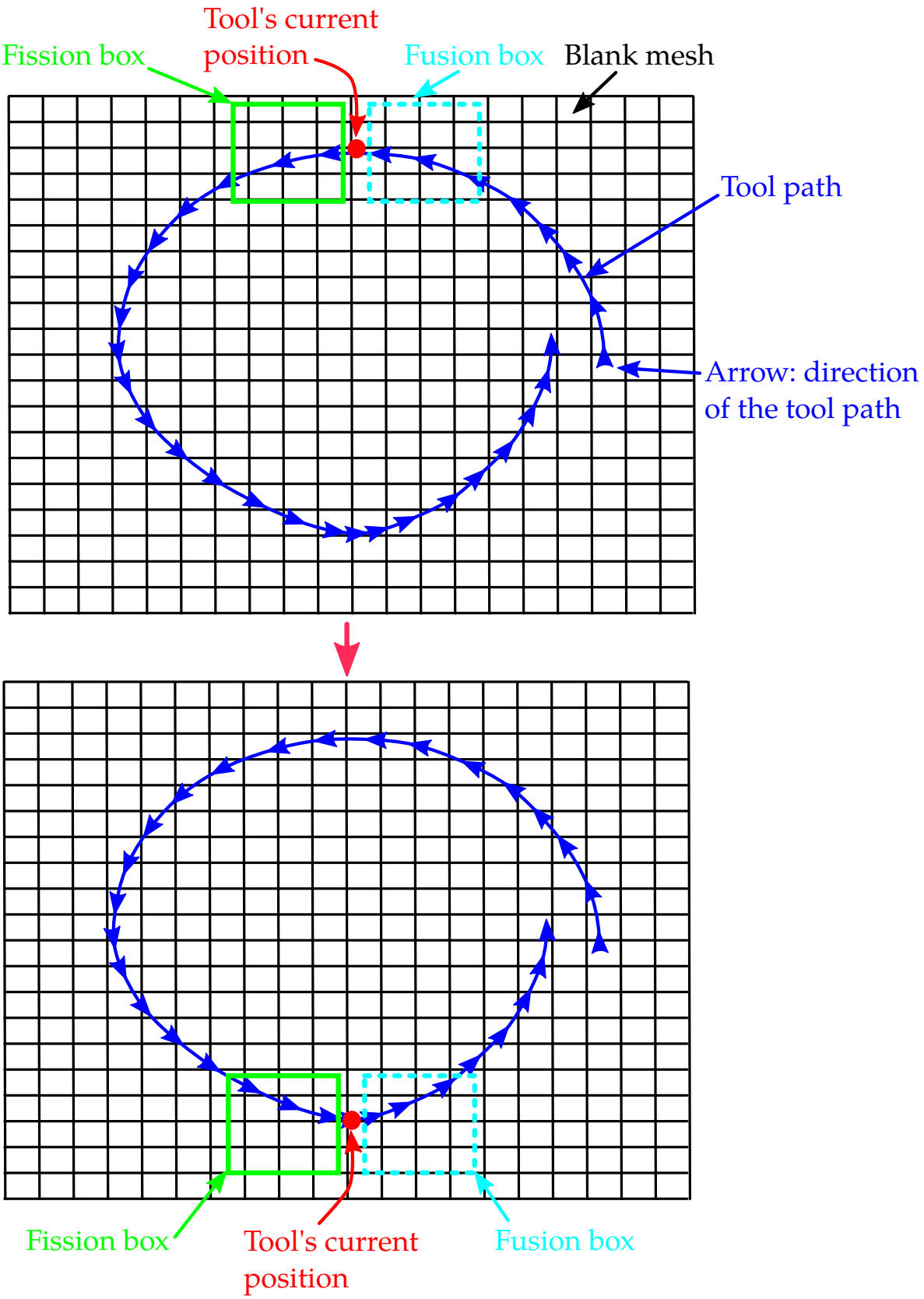


Figure 17-5. Limitation of the existing capability by *DEFINE_BOX_ADAPTIVE.

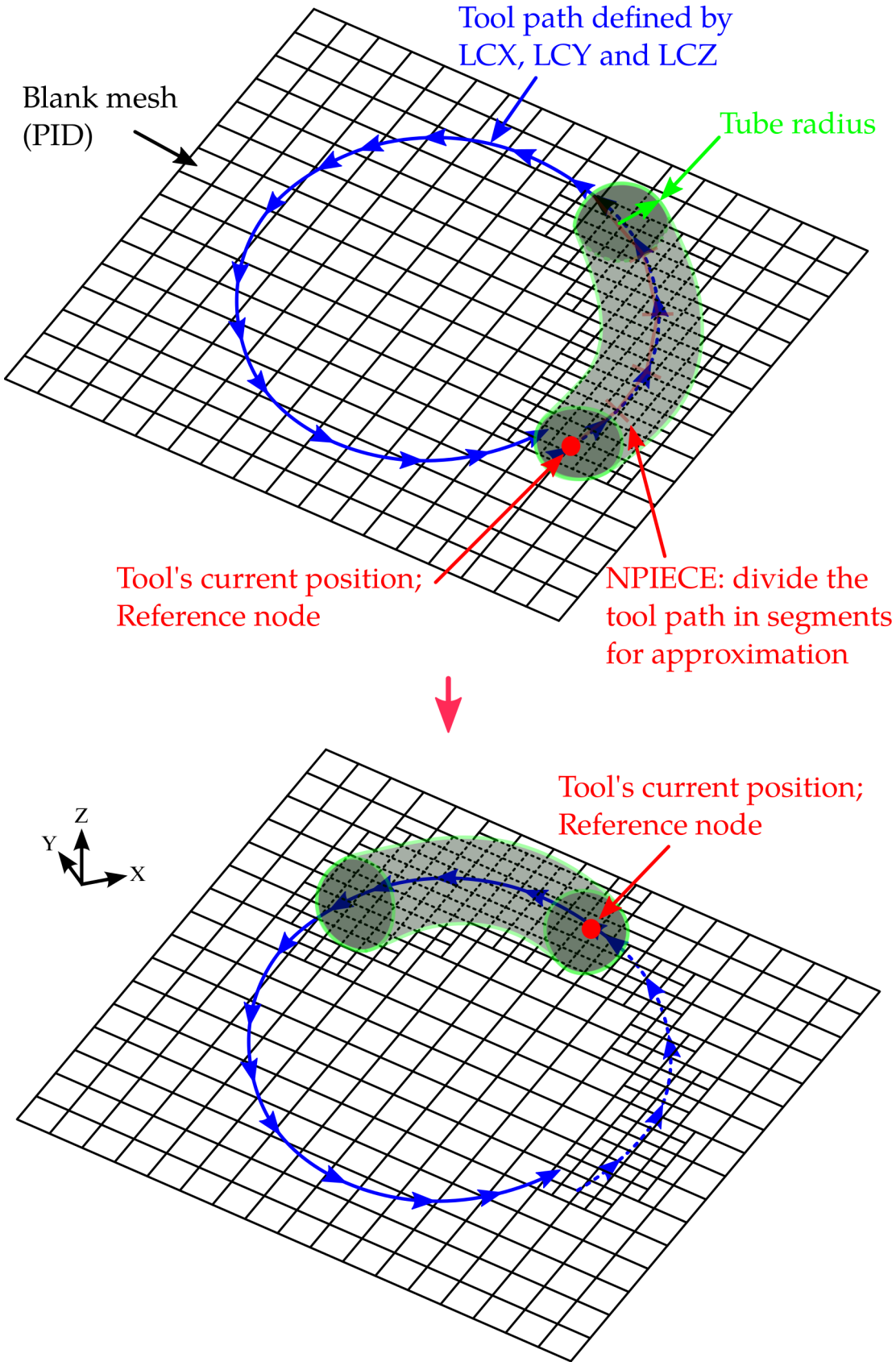


Figure 17-6. "Tube" adaptivity

***DEFINE_BOX_SPH**_{OPTION}_

Available options include:

<BLANK>

LOCAL

Purpose: Define a box-shaped volume for activating and deactivating SPH particles. To define the box, you specify two diagonally opposite corner points. The corner points are assumed to be in the global coordinate system unless the LOCAL keyword option is active. See *DEFINE_BOX for more details about the LOCAL option. When particles are in the box, the SPH particles are active and LS-DYNA computes the SPH approximation (see FORM in *CONTROL_SPH) for the particles. SPH particles are deactivated as they leave the box but reactivate if they re-enter the box. Unlike a box specified with *DEFINE_BOX, the box may move either with a node or in the direction defined by a vector with a motion determined by a load curve.

> **WARNING:** The box specified for deactivating/activating particles by *CONTROL_SPH (see BOXID) must be a *DEFINE_BOX, not a *DEFINE_BOX_SPH. If BOXID references a *DEFINE_BOX_SPH, LS-DYNA will give an error message.

Card 1	1	2	3	4	5	6	7	8
Variable	BOXID	XMN	XMX	YMN	YMX	ZMN	ZMX	VID
Type	I	F	F	F	F	F	F	I
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	LCID	VD	NID	IREACT	IBUFF	ISHOW	PID	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Local Card 1. First additional card for the LOCAL keyword option. See *DEFINE_BOX for a description of the LOCAL option.

Card 3	1	2	3	4	5	6	7	8
Variable	XX	YX	ZX	XV	YV	ZV		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Local Card 2. Second additional card for the LOCAL keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	CX	CY	CZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
BOXID	Box ID. Define unique numbers.
XMN	Minimum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
XMX	Maximum x-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMN	Minimum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
YMX	Maximum y-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMN	Minimum z-coordinate. Define in the local coordinate system if the option LOCAL is active.
ZMX	Maximum z-coordinate. Define in the local coordinate system if the option LOCAL is active.

VARIABLE	DESCRIPTION
VID	Vector ID to describe direction of box motion when VD = 0 or 1; see *DEFINE_VECTOR.
LCID	Load curve ID to describe value of box motion as a function of time; see *DEFINE_CURVE
VD	Velocity/Displacement flag: EQ.0: Velocity, EQ.1: Displacement, EQ.2: Referential node
NID	Referential nodal ID for VD = 2 (SPH box will move with this node).
IREACT	Reactivation flag: EQ.0: Particles outside of the box are permanently deactivated. EQ.1: Deactivated particles get reactivated when they enter the box.
IBUFF	Buffer zone flag: EQ.0: Particles on the edge of the box do not get any special treatment. EQ.1: Particles on the edge of the box are frozen in space and act as neighbors for active particles inside the box. This option is mainly used for fluid simulations to prevent the fluid from spilling out of the activation box.
ISHOW	Create dummy part for visualizing the position of the activation box during post-processing: EQ.0: No part is created. EQ.1: A dummy part is added for visualization.
PID	Part ID used for visualization if ISHOW = 1: EQ.0: A unique part ID is automatically created. GT.0: PID is the part ID for the created part. This should be a unique part ID.
XX	X-coordinate on local x-axis. Origin lies at (0,0,0). Define if the LOCAL option is active.

VARIABLE	DESCRIPTION
YX	*Y*-coordinate on local *x*-axis. Define if the LOCAL option is active.
ZX	*Z*-coordinate on local *x*-axis. Define if the LOCAL option is active.
XV	*X*-coordinate of local *xy*-vector. Define if the LOCAL option is active.
YV	*Y*-coordinate of local *xy*-vector. Define if the LOCAL option is active.
ZV	*Z*-coordinate of local *xy*-vector. Define if the LOCAL option is active.
CX	*X*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CY	*Y*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.
CZ	*Z*-global coordinate of offset vector to origin of local system. Define if the LOCAL option is active.

*DEFINE_CABLE

Purpose: Create a cable consisting of a beam element core surrounded by solid elements. This setup is commonly used for modeling electric cables, where the solid elements represent the insulator and copper core, and the beam elements give additional bending stiffness. This keyword causes the automatic generation of the solid elements from the beam element core. The output file define_cable.k contains the generated elements, nodes, and connectivity for visualization and further pre-processing. Running an input deck containing this keyword with define_cable.k included causes using the solids, nodes, and connectivity from the file instead. Cross-sectional data, such as contact force, stress, and area, are calculated from the solid elements and saved for each corresponding beam element. The automatically output ascii file cablestats contains summary statistics for all the cables present. *DATABASE_CABLE activates outputting element-wise data to binout and/or the ascii file cableout.

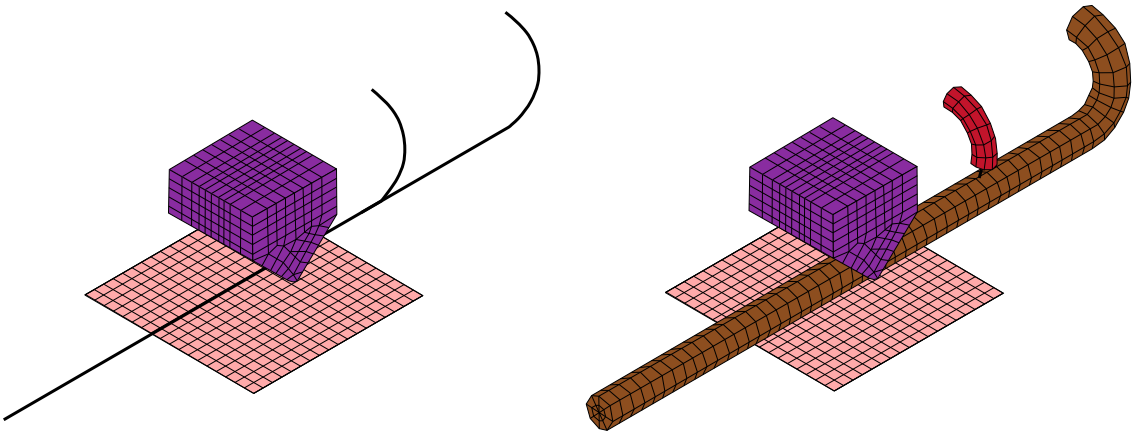


Figure 17-7. Example of initial beam geometry and resulting solid/beam geometry

Card Summary:

Card 1. This card is required.

BPID	NLAYR	NCIRC	NCYCL	NLPID	ELNR	NDNR	TRIM

Card 2. This card is required.

SPID	STHK						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	BPID	NLAYR	NCIRC	NCYCL	SHLPID	ELNR	NDNR	TRIM
Type	I	I	I	I	I	I	I	I
Default	0	0	0	1	0	0	0	0

VARIABLE	DESCRIPTION
BPID	Beam part ID for the cable core. All connected beam elements in the part model a cable. For MPP, all elements in the beam part are put on a single processor.
NLAYR	Number of solid layers in thickness
NCIRC	Number of solid elements on the circumference
NCYCL	Collect data every NCYCL cycle
SHLPID	Optional PID to create shells on the solid faces. This is commonly used for extra null shells that may increase contact stability. This part ID needs to be for an existing part without any elements associated with it. SHLPID on this card cannot be used by any other *DEFINE_CABLE instantiations.
ELNR	Optional starting element number for generated solids. If several cable keywords have an ELNR that causes collisions, ELNR will be automatically adjusted.
NDNR	Optional starting node number for generated nodes. If several cable keywords have an NDNR that causes collisions, NDNR will be automatically adjusted.
TRIM	Set to 1 if adjoint cables need to be trimmed, meaning a cable connected to the interior of another cable will not create new solids such that the connection has overlapping solid elements.

Card 2	1	2	3	4	5	6	7	8
Variable	SLDPID	SLDTHK						
Type	I	F						
Default	0	0.0						

VARIABLE	DESCRIPTION
SLDPID	Solid part ID for the automatically generated solids. This part ID needs to be an existing part without any elements associated with it. SLDPID on this card cannot be used by any other *DEFINE_-CABLE instantiations.
SLDTHK	Thickness for each solid layer

*DEFINE_CONNECTION_PROPERTIES_{OPTION}

Available options include:

<BLANK>

ADD

Purpose: Define failure related parameters for solid element spot weld failure by *MAT_-SPOTWELD_DAIMLERCHRYSLER. For each connection identifier, CON_ID, a separate *DEFINE_CONNECTION_PROPERTIES section must be included. The ADD keyword option allows material specific properties to be added to an existing connection ID. See Remark 2.

Card Summary:

Card 1. This card is required.

CON_ID	PRUL	AREAEQ		DGTYP	MOARFL		

Card 2. This card is omitted if the ADD keyword option is used. See Remark 2.

	DSIGY	DETAN	DDGPR	DRANK	DSN	DSB	DSS

Card 3. This card is omitted if the ADD keyword option is used. See Remark 2.

DEXSN	DEXSB	DEXSS	DLCSN	DLCSB	DLCSS	DGFAD	DSCLMRR

Card 4. This card is included if he ADD keyword option is used. For each shell material with material specific data, define for this CON_ID Cards 4 and 5. Add as many pairs of these cards as necessary. This input is terminated by the next keyword ("*") card. See Remark 2.

MID	SGIY	ETAN	DGPR	RANK	SN	SB	SS

Card 5. This card is included if the ADD keyword option is used. For each shell material with material specific data, define for this CON_ID Cards 4 and 5. Add as many pairs of these cards as necessary. This input is terminated by the next keyword ("*") card. See Remark 2.

EXSN	EXSB	EXSS	LCSN	LCSB	LCSS	GFAD	SCLMRR

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	CON_ID	PRUL	AREAEQ		DGTYP	MOARFL		
Type	F	I	I		I	I		
Default	0	0	0		0	0		

VARIABLE	DESCRIPTION
CON_ID	Connection ID, referenced on *MAT_SPOTWELD_DAIMLER-CHRYSLER. Multiple sets of connection data may be used by assigning different connection IDs.
PRUL	The failure rule number for this connection: EQ.1: Use data of weld partner with lower RANK (default). GE.2: Use *DEFINE_FUNCTION expressions to determine weld data depending on several values of both weld partners. Variables DSIGY, DETAN, DDGPR, DSN, DSB, DSS, DEXSN, DEXSB, DEXSS, and DGFAD must be defined as function IDs, see Remark 5.
AREAEQ	Area equation number for the connection area calculation: EQ.0: (default) $Area_{true} = Area_{modelled}$ EQ.1: Millimeter form; see Remark 4 EQ.-1: Meter form; see Remark 4
DGTYP	Damage type: EQ.0: No damage function is used. EQ.1: Strain based damage EQ.2: Failure function based damage EQ.3 or 4: Fading energy based damage (see Remark 4) EQ.5: Improved version of DGTYP = 4 (see Remark 4)
MOARFL	Modeled area flag: EQ.0: $Area_{modelled}$ goes down with shear (default).

VARIABLE	DESCRIPTION
	EQ.1: $Area_{modelled}$ stays constant.

Card 2	1	2	3	4	5	6	7	8
Variable		DSIGY	DETAN	DDGPR	DRANK	DSN	DSB	DSS
Type		F	F	F	F	F	F	F
Default		none	none	10^{10}	none	none	none	none

VARIABLE	DESCRIPTION
DSIGY	Default yield stress for the spot weld element: GT.0: Constant value LT.0: |DSIGY| references a yield curve or table; see Remark 6.
DETAN	Default tangent modulus for the spot weld element
DDGPR	Default damage parameter for hyperbolic based damage function
DRANK	Default rank value
DSN	Default normal strength
DSB	Default bending strength
DSS	Default shear strength

Card 3	1	2	3	4	5	6	7	8
Variable	DEXSN	DEXSB	DEXSS	DLCSN	DLCSB	DLCSS	DGFAD	DSCLMRR
Type	F	F	F	I	I	I	F	F
Default	1.0	1.0	1.0	0	0	0	none	1.0

VARIABLE	DESCRIPTION
DEXSN	Default exponent on normal stress term
DEXSB	Default exponent on bending stress term
DEXSS	Default exponent on shear stress term
DLCSN	Default curve ID for normal strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed that all strain rate values are given as a natural logarithm of the strain rate.
DLCSB	Default curve ID for bending strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed that all strain rate values are given as a natural logarithm of the strain rate.
DLCSS	Default curve ID for shear strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed that all strain rate values are given as a natural logarithm of the strain rate.
DGFAD	Default fading energy for damage type 3 and type 4
DSCLMRR	Default scaling factor for torsional moment in failure function

Material Data Card 1.

Card 4	1	2	3	4	5	6	7	8
Variable	MID	SGIY	ETAN	DGPR	RANK	SN	SB	SS
Type	F	F	F	F	F	F	F	F
Default	none	none	none	10^{10}	none	none	none	none

VARIABLE	DESCRIPTION
MID	Material ID of the shell material for which properties are defined
SIGY	Yield stress to be used in the spot weld element calculation: GT.0: Constant value LT.0: |SIGY| references a yield curve or table; see Remark 6.

VARIABLE	DESCRIPTION
ETAN	Tangent modulus to be used in the spot weld element calculation
DGPR	Damage parameter for hyperbolic based damage function
RANK	Rank value. See Remark 4.
SN	Normal strength
SB	Bending strength
SS	Shear strength

Material Data Card 2.

Card 5	1	2	3	4	5	6	7	8
Variable	EXSN	EXSB	EXSS	LCSN	LCSB	LCSS	GFAD	SCLMRR
Type	F	F	F	I	I	I	F	F
Default	none	none	none	none	none	none	none	1.0

VARIABLE	DESCRIPTION
EXSN	Exponent on normal stress term
EXSB	Exponent on bending stress term
EXSS	Exponent on shear stress term
LCSN	Curve ID for normal strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed that all strain rate values are given as a natural logarithm of the strain rate.
LCSB	Curve ID for bending strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed that all strain rate values are given as a natural logarithm of the strain rate.
LCSS	Curve ID for shear strength scale factor as a function of strain rate. If the first strain rate value in the curve is negative, it is assumed

VARIABLE	DESCRIPTION
	that all strain rate values are given as a natural logarithm of the strain rate.
GFAD	Fading energy for damage type 3 and 4
SCLMRR	Scaling factor for torsional moment in failure function

Remarks:

1. **Restriction to *MAT_SPOTWELD_DAIMLERCHHRYSLER.** This keyword is used only with *MAT_SPOTWELD_DAIMLERCHRYSLER. The data input is used in a 3 parameter failure model. Each solid spot weld element connects shell elements that may have the same or different materials. The failure model assumes that failure of the spot weld depends on the properties of the welded materials, so this keyword allows shell material specific data to be input for the connection. The default data will be used for any spot weld connected to a shell material that does not have material specific data defined, so it is not necessary to define material specific data for all welded shell materials.

2. **ADD Option.** To simplify data input, the ADD keyword option allows material specific data to be added to an existing *DEFINE_CONNECTION_PROPERTIES table. To use the ADD option, omit Cards 2 and 3, and input only CON_ID on Card 1. Then use Cards 4 and 5 to input material specific data. For each unique CON_ID, control parameters and default values must be input in one set of *DEFINE_CONNECTION_PROPERTIES data. The same CON_ID may be used for any number of sets of material specific data input with the ADD option.

3. **The Three Parameter Failure Function.** The three parameter failure function is

$$f = \left(\frac{\sigma_n}{\sigma_n^F}\right)^{m_n} + \left(\frac{\sigma_b}{\sigma_b^F}\right)^{m_b} + \left(\frac{\tau}{\tau^F}\right)^{m_\tau} - 1 \ ,$$

where the three strength terms are SN, SB, and SS, and the three exponents are EXSN, EXSB, and EXSS. The strengths may be a function of strain rate by using the load curves, LCSN, LCSB, and LCSS. The peak stresses in the numerators are calculated from force resultants and simple beam theory.

$$\sigma_n = \frac{N_{rr}}{A}, \qquad \sigma_b = \frac{\sqrt{M_{rs}^2 + M_{rt}^2}}{Z}, \qquad \tau = \text{SCLMRR} \times \frac{M_{rr}}{2Z} + \frac{\sqrt{N_{rs}^2 + N_{rt}^2}}{A} \ ,$$

where the area is the cross section area of the weld element and Z is given by:

$$Z = \pi \frac{d^3}{32} \ ,$$

where d is the equivalent diameter of the solid spot weld element assuming a circular cross section.

4. **Control Parameters PRUL, AREAQ. And DGTYP.** There are three control parameters that define how the table data will be used for the connection, PRUL, AREAEQ, and DGTYP.

   a) *PRUL.* PRUL determines how the parameters will be used. Because each weld connects two shell surfaces, one weld can have two sets of failure data as well as two values for ETAN and SIGY. For PRUL = 1 (default), a simple rule is implemented and the data with the lower RANK will be used. For PRUL = 2 or 3, function expressions can be used to determine the data based on several input values from both weld partners (see Remark 5 for details).

   b) *AREAEQ.* The second control parameter is AREAEQ which specifies a rule for calculating a true weld cross section area, A_{true} to be used in the failure function in place of the modeled solid element area, A. For AREAEQ = 1, A_{true} is calculated by

$$A_{\text{true}} = \frac{\pi}{4}\left(5\sqrt{t_{\text{min shell}}}\right)^2 \ ,$$

   where $t_{\text{min shell}}$ is the thickness of the welded shell surface that has the smaller thickness. For AREAEQ = −1, A_{true} is calculated by

$$A_{\text{true}} = \frac{\pi}{4}\left(\frac{5}{1000}\sqrt{1000 \times t_{\text{min shell}}}\right)^2 \ ,$$

   The equation for AREAEQ = 1 is valid only for a length unit of millimeters, and AREAEQ = −1 is valid only for a length unit of meters.

   c) *DGTYP.* The third control parameter, DGTYP, chooses from two available damage types. For DGTYP = 0, damage is turned off and the weld fails immediately when $f \geq 0$. For DGTYP > 0, damage is initiated when $f \geq 0$ and complete failure occurs when $\omega \geq 1$. For DGTYP = 1, damage growth is a function of plastic strain:

$$\omega = \frac{\varepsilon^p_{\text{eff}} - \varepsilon^p_{\text{failure}}}{\varepsilon^p_{\text{rupture}} - \varepsilon^p_{\text{failure}}}, \qquad \varepsilon^p_{\text{failure}} \leq \varepsilon^p_{\text{eff}} \leq \varepsilon^p_{\text{rupture}} \ ,$$

   where $\varepsilon^p_{\text{eff}}$ is the effective plastic strain in the weld material. When the value of the failure function first exceeds zero, the plastic strain at failure, $\varepsilon^p_{\text{failure}}$, is set to the current plastic strain, and the rupture strain is offset from the plastic strain at failure by

$$\varepsilon^p_{\text{rupture}} = \varepsilon^p_{\text{failure}} + \text{RS} - \text{EFAIL} \ ,$$

where RS and EFAIL are the rupture strain and plastic strain at failure which are input on the *MAT_SPOTWELD_DAIMLERCHRYSLER card. If failure occurs when the plastic strain is zero, the weld material yield stress is reduced to the current effective stress such that damage can progress.

For DGTYP = 2, damage is a function of the failure function, f:

$$f \geq 0 \Rightarrow \omega = \frac{f}{f_{\text{rupture}}} \ ,$$

where f_{rupture} is the value of the failure function at rupture which is defined by

$$f_{\text{rupture}} = \text{RS} - \text{EFAIL} \ ,$$

and RS and EFAIL are input on the *MAT_SPOTWELD_DAIMLERCHRYSLER card.

Because the DGTYP = 1 damage function is scaled by plastic strain, it will monotonically increase in time. The DGTYP = 2 damage function is forced to be a monotonically increasing function in time by using the maximum of the current value and the maximum previous value. For both DGTYP = 1 and DGTYP = 2, the stress scale factor is then calculated by

$$\widehat{\sigma} = \frac{\text{DGPR} \times (1 - \omega)}{\omega \left(\frac{1}{2} + \sqrt{\frac{1}{4} + \text{DGPR}} \right) + \text{DGPR}} \sigma \ ,$$

This equation becomes nearly linear at the default value of DGPR which is 10^{10}.

For DGTYP = 3, damage is a function of total strain:

$$\omega = \frac{\Delta\varepsilon_n}{\Delta\varepsilon_{\text{fading}}} \ .$$

$\Delta\varepsilon_n$ is the accumulated total strain increment between moment of damage initiation (failure) and current time step t_n

$$\Delta\varepsilon_n = \Delta\varepsilon_{n-1} + \Delta t_n \sqrt{\frac{2}{3} \ \text{tr}(\dot{\boldsymbol{\varepsilon}}_n \dot{\boldsymbol{\varepsilon}}_n^T)} \ , \qquad \Delta\varepsilon|_{t_{\text{failure}}} = 0 \ ,$$

and $\Delta\varepsilon_{\text{fading}}$ is the total strain increment for fading (reduction of stresses to zero)

$$\Delta\varepsilon_{\text{fading}} = \frac{2 \times \text{GFAD}}{\sigma_{\text{failure}}} \ ,$$

where GFAD is the fading energy from input and σ_{failure} is the effective stress at failure. The stress scale factor is then calculated by a linear equation

$$\widehat{\sigma} = (1 - \omega)\sigma \ ,$$

where σ is the Cauchy stress tensor at failure and ω is the actual damage value. Problems can occur, if the loading direction changes after the onset of failure, since during the damage process, the components of the stress tensor are kept constant and hence represent the stress state at failure.

DGTYP = 4 should be used when describing the damage behavior of the spotweld in a more realistic way. For DGTYP = 4, damage is a function of the internal work done by the spotweld after failure,

$$\widehat{\sigma} = (1 - \omega)\sigma^{\mathrm{ep}}, \quad \omega = \frac{G_{\mathrm{used}}}{2 \times \mathrm{GFAD}}, \quad G_{\mathrm{used}} = G_{\mathrm{used}}^{n-1} + \det(\mathbf{F})\, \sigma_{ij}^{\mathrm{ep}} \Delta\varepsilon_{ij} \ .$$

In the above, $\mathbf{F}$ is the deformation gradient. σ^{ep} is a scaled Cauchy stress tensor based on the undamaged Cauchy stress tensor σ^{wd} and scaled in such a way that the same internal work is done in the current time step as in the time step before (equipotential):

$$\sigma^{\mathrm{ep}} = \alpha\sigma^{\mathrm{wd}}, \quad \alpha = \frac{\sigma_{ij}^{n-1,\mathrm{ep}} \Delta\varepsilon_{ij}}{\sigma_{ij}^{\mathrm{wd}} \Delta\varepsilon_{ij}} \ .$$

DGTYP = 4 has two serious disadvantages that may result in unstable behavior after failure:

i) If the spot weld is unloaded during damage, the undamaged Cauchy stress tensor tends to zero and therefore α to infinity.

ii) The factor α is a very instable variable because it depends directly on the total strain increment.

DGTYP = 5 was developed to overcome these disadvantages. In contrast to DGTYP = 4, the undamaged Cauchy stress tensor is only calculated with an elastoplastic material model (not scaled by α). This limitation of the stress tensor makes it more stable during damage. At failure the yield stress is set to the current von Mises stress, and the plastic modulus is set to zero. Calculating the current damage value is the same as for DGTYP = 4. Note that, to achieve realistic behavior for the axial loading situation, you should use the UNIAXIAL keyword option of *MAT_SPOTWELD_DAIMLER. Additionally, with this material model, the spotweld can be deleted based on the current plastic strain with the input field RS.

5. **Failure Rule from *DEFINE_FUNCTION.** The failure rule number PRUL = 2 or 3, is available starting with Release R7. To use this new option, 11 fields must be defined as function IDs: DSIGY, DETAN, DDGPR, DSN, DSB, DSS, DEXSN, DEXSB, DEXSS, DGFAD, and DSCLMRR.

These functions depend on:

(t1, t2) = thicknesses of both weld partners
(sy1, sy2) = initial yield stresses at plastic strain
(sm1, sm2) = maximum engineering yield stresses
r = strain rate
a = spot weld area
fn = normal term in failure function
fb = bending term in failure function
fs = shear term in failure function
(ym1, ym2) = Young's moduli of both weld partners

For DSIGY = 100, such a function could look like:

```
*DEFINE_FUNCTION
             100
func(t1,t2,sy1,sy2,sm1,sm2,r,a,fn,fb,fs,ym1,ym2)=0.5*(sy1+sy2)
```

All the listed arguments in their correct order must be included in the argument list. For PRUL = 2, the thinner part is the first weld partner. For PRUL = 3, the bottom part (nodes 1-2-3-4) is the first weld partner. Since material parameters must be identified from both weld partners during initialization, this feature is only available for a subset of material models at the moment, namely material types 3, 24, 36, 81, 120, 123, 124, 133, 187, 224, 243, 251, and 258. This new option eliminates the need for the ADD option.

6. **Yield Curve or Table for SIGY.** When using this option, a simplified plasticity algorithm is used, assuming a linear behavior within one time increment. Thus, no iterative return mapping has to be performed.

*DEFINE_CONSTRUCTION_STAGES

Purpose: Define times and durations of construction stages.

Card	1	2	3	4	5	6	7	8
Variable	ISTAGE	ATS	ATE	ATR	RTS	RTE	blank	IDYNAIN
Type	I	F	F	F	F	F		I
Default	none	0.0	0.0	none	ATS	ATE		0

VARIABLE	DESCRIPTION
ISTAGE	Stage ID
ATS	Analysis time at start of stage
ATE	Analysis time at end of stage
ATR	Analysis time duration of ramp
RTS	Real time at start of stage
RTE	Real time at end of stage
IDYNAIN	Flag to control output of dynain file at the end of the stage (see Remark 5): EQ.0: default to setting of IDYNAIN on *CONTROL_STAGED_CONSTRUCTION EQ.1: do not write dynain file.

Remarks:

1. **Related Keywords.** See also *CONTROL_STAGED_CONSTRUCTION and *DEFINE_STAGED_CONSTRUCTION_PART.

2. **Stage Start and End Times.** The first stage should start at time zero. There must be no gaps between stages, that is, ATS for each stage must be the same as ATE for the previous stage.

3. **Ramp Time.** The ramp time allows gravity loading and part removal to be applied gradually during the first time period ATR of the construction stage.

4. **Analysis and Real Time.** The analysis always runs in "analysis time" – typically measured in seconds. The "real time" is used only as a number to appear on output plots and graphs and is completely arbitrary. See ITIME on *CONTROL_STAGED_CONSTRUCTION.

5. **IDYNAIN.** If IDYNAIN is zero or blank, it will default to the same setting as IDYNAIN on *CONTROL_STAGED_CONSTRUCTION. The default of *CONTROL_STAGED_CONSTRUCTION is to write a dynain file at the end of each stage. This can be suppressed for individual stages by setting IDYNAIN to 1 on this keyword.

*DEFINE_CONTACT_EXCLUSION

Purpose: Exclude tied nodes from being treated in specific contact interfaces. This keyword is currently only available in the MPP version.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	TITLE						
Type	I	A70						

ID Card 1. This card sets the contact interface IDs of up to 7 tied interfaces.

Card 2	1	2	3	4	5	6	7	8
Variable	TARGET	C1	C2	C3	C4	C5	C6	C7
Type	I	I	I	I	I	I	I	I

Optional ID Cards. More tied interfaces. Include as many cards as necessary. This input ends at the next keyword (*) card.

Card 3	1	2	3	4	5	6	7	8
Variable	C8	C9	C10	C11	C12	C13	C14	C15
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
EID	Exclusion ID
TITLE	Exclusion Title
TARGET	Contact interface from which tied nodes are to be excluded. This must be the ID of a SINGLE_SURFACE, NODE_TO_SURFACE, or SURFACE_TO_SURFACE contact with SOFT ≠ 2.
C*i*	The IDs of TIED contacts: 7 on the first card and 8 per additional card for as many cards as necessary. Any node which is a tracked node in one of these interfaces, and is in fact tied, will not be processed (as a tracked node) in the TARGET interface.

Remarks:

1. **Contact Forces on Excluded Nodes.** If a node is excluded from the Target by this mechanism, contact forces may still be applied to the node due to any tracked or reference nodes impacting the contact segments of which it is a part (no contact SEGMENTS are deleted, only contact NODES).

2. **SURFACE_TO_SURFACE and SINGLE_SURFACE Targets.** If you intend to exclude tied tracked nodes completely from other contacts in the model, this keyword can be used to do so for NODES_TO_SURFACE and ONE_WAY_SURFACE_TO_SURFACE contacts only. For (TWO_WAY) SURFACE_TO_SURFACE and SINGLE_SURFACE contacts, such nodes should be excluded from the contact definitions if so desired.

*DEFINE_CONTACT_VOLUME

Purpose: Define a rectangular, a cylindrical, or a spherical volume in a local coordinate system. The volume can be referenced by *SET_NODE_GENERAL for the purpose of defining a node set consisting of nodes inside the volume, or by *CONTACT_... for the purpose of defining nodes or segments on the SURFA side or the SURFB side of the contact (see SABOXID and SBBOXID on Card 1 of *CONTACT_...).

Card Summary:

Card 1. This card is required.

CVID	CID	TYPE	XC	YC	ZC		

Card 2a. This card is included if and only TYPE = 0.

XMN	XMX	YMN	YMX	ZMN	ZMX		

Card 2b. This card is included if and only if TYPE = 1.

LENGTH	RINNER	ROUTER	D_ANGC				

Card 2c. This card is included if and only if TYPE = 2.

RINNER	ROUTER	D_ANGS					

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	CVID	CID	TYPE	XC	YC	ZC		
Type	I	I	I	F	F	F		
Default	0	0	0	0.	0.	0.		

VARIABLE	DESCRIPTION
CVID	Contact volume ID
CID	Coordinate system ID. Required for rectangular and cylindrical volumes.

VARIABLE	DESCRIPTION
TYPE	Volume type: EQ.0: Rectangle EQ.1: Cylinder EQ.2: Sphere
XC	x-coordinate which defines the origin of coordinate system or the center of the sphere (TYPE= 3) in the global coordinate system.
YC	y-coordinate which defines the origin of coordinate system or the center of the sphere (TYPE = 3) in the global coordinate system.
ZC	z-coordinate which defines the origin of coordinate system or the center of the sphere (TYPE = 3) in the global coordinate system.

Rectangular Prism. Use when TYPE = 0.

Card 2a	1	2	3	4	5	6	7	8
Variable	XMN	XMX	YMN	YMX	ZMN	ZMX		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
XMN	Minimum x-coordinate in local coordinate system
XMX	Maximum x-coordinate in local coordinate system
YMN	Minimum y-coordinate in local coordinate system
YMX	Maximum y-coordinate in local coordinate system
ZMN	Minimum z-coordinate in local coordinate system
ZMX	Maximum z-coordinate in local coordinate system

Cylinder. Use when TYPE = 1.

Card 2b	1	2	3	4	5	6	7	8
Variable	LENGTH	RINNER	ROUTER	D_ANGC				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
LENGTH	Length of cylinder originating at (XC,YC,ZC) and revolving around the local *x*-axis.
RINNER	Inner radius of cylinder
ROUTER	Outer radius of cylinder
D_ANGC	If the included angle between the axis of the cylinder and the normal vector to the contact segment is *less* than this angle, the segment is deleted.

Sphere. Use when TYPE = 2.

Card 2c	1	2	3	4	5	6	7	8
Variable	RINNER	ROUTER	D_ANGS					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
RINNER	Inner radius of sphere
ROUTER	Outer radius of sphere
D_ANGS	If the included angle between a line draw from the center of the sphere to the centroid of the segment and the normal vector to the contact segment is *greater* than this angle, the segment is deleted.

*DEFINE_CONTROL_VOLUME

Purpose: Specify an incompressible control volume. This control volume only works with the implicit solver.

Card 1	1	2	3	4	5	6	7	8
Variable	CVID	SSID	RHO	LCSRC				
Type	I	I	F	I				
Default	none	none	0.0	0				

VARIABLE	DESCRIPTION
CVID	Control volume ID
SSID	Segment set ID giving the control volume geometry. See Remark 2.
RHO	Density of fluid contained in the domain (and entering the domain through the source curve). See Remarks 1 and 5.
LCSRC	Load curve specifying the mass flow rate (mass/time) into the domain. Note that RHO must be set to a nonzero number for this to work. See Remarks 1 and 5.

Remarks:

1. **Definition of a control volume.** A *control volume* is a domain in space that contains an incompressible fluid. The volume of the fluid contained within the domain exclusively determines the domain's volume. Thus, the domain's volume can *only* change if fluid flows into or out of the domain. The flow can be between control volumes or between a control volume and the ambient environment. In both cases, fluid flows through *flow areas*. The flow into a domain can also be defined by a source term determined by combining RHO and LCSRC. Figure 17-8 illustrates two control volumes with one flow area between the control volumes and two flow areas to the ambient environment.

2. **Defining the control volume and flow areas.** For the volume calculation of a control volume to be correct, *the domain must be closed*. In other words, the segment set defining the volume must not contain any holes. So, either the segment set SSID in *DEFINE_CONTROL_VOLUME defines a closed volume, *or* this keyword is complemented by *DEFINE_CONTROL_VOLUME_FLOW_AREA

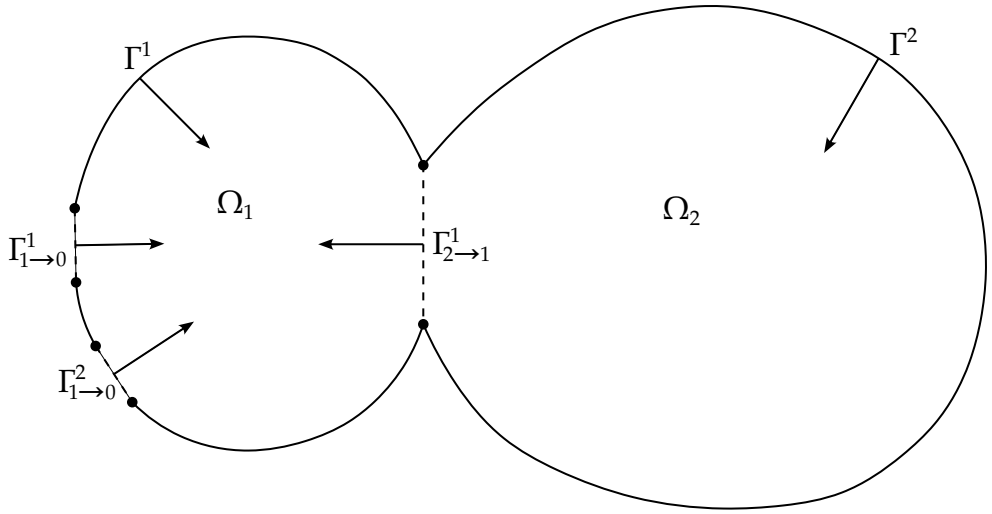


Figure 17-8. Two control volume with one sharing flow areas with the ambient environment. Control volume 1, Ω_1, has boundary Γ_1. It shares two flow areas, $\Gamma^1_{1\to 0}$ and $\Gamma^2_{1\to 0}$, with the ambient environment, and one flow area $\Gamma^1_{2\to 1}$, with control volume 2, Ω_2. Ω_2 has boundary Γ_2 and only shares flow areas with Ω_1.See Remarks 1, 2, and 5.

keywords with *node sets* (FASTYP = 1) defining the perimeters of the holes that the defined segment set SSID doesn't cover. For the latter case, the volume is automatically closed by the segments that are generated by these complementary keyword definitions. If the flow area is defined with a segment set (FASTYP = 2) in *DEFINE_CONTROL_VOLUME_FLOW_AREA, these segments should already be in the definition of SSID. They are *not* added to the set defining the control volume.

The segment set SSID *must* be oriented so that the normal of each segment points towards the interior of the domain as illustrated in Figure 17-8. Furthermore, the segment sets of each flow area must be oriented so that their normal vectors point towards the interior of the *first* domain, CVID1, in the associated *DEFINE_CONTROL_VOLUME_INTERACTION keyword. If the flow area is defined by a node set (FASTYP = 1), the order of the nodes in the set must be clockwise when seen from *outside* the *first* control volume, CVID1. To interface with the ambient environment, leave the second domain, CVID2, blank. Whenever a flow area is defined, the pressure difference between the domains (or domain and ambient) does not result in nodal forces on the segment nodes in the flow area. Thus, the stiffness matrix is nonsymmetric whenever a flow area is defined, so using LCPACK = 3 on *CONTROL_IMPLICIT_SOLVER may improve convergence.

3. **Flow between domains.** *DEFINE_CONTROL_VOLUME_INTERACTION determines the flow between domains or between a domain and the ambient environment. The function defined by LCID specifies the flow rate (volume/time) as a function of the time, pressure drop, and area of the flow area

opening. The curve may be left blank, causing no flow through the opening. The pressure difference used in functions is the one between the first and second domain, meaning $\Delta p = p_{\text{CVID1}} - p_{\text{CVID2}}$. The flow rate is positive into domain CVID2 and negative into domain CVID1.

4. **Output data.** Fluid cavity data, such as the time history of the volume and pressure, is output to the LSDA binout file.

5. **Mathematical formulation.** We define a control volume as Ω_i with boundary Γ_i for $i \in [1, n]$. The volume of a domain, $V_i = V_i(\mathbf{x})$, is obtained by Green's formula, which is a surface integral over Γ_i. Here, $\mathbf{x}$ is the nodal coordinate vector, and it is implicitly understood that the volume calculation depends only on a subset S_i of all the nodes. We also assume that the fluid inside control volume Ω_i has a uniform pressure, p_i, that is assumed given for now. This pressure is relative to the ambient environment, not absolute.

   We define flow areas as $\Gamma^k_{i\to j}$ with $i \in [1, n]$, $j \in [0, n-1]$ where $j < i$, and $k \in \left[1, m_{ij}\right]$. i and j indicate the two domains or the domain and the ambient environment that share a flow area. $j = 0$ is the ambient environment. m_{ij} is the number of flow surfaces between two given domains or between a domain and the ambient environment. The area of a flow surface is $A^k_{i\to j} = A^k_{i\to j}(\mathbf{x})$ and depends only on a subset $S^k_{i\to j}$ of all the nodes. We have $\Gamma^k_{i\to j} \subset \Gamma_i \cap \Gamma_j$ for $j > 0$, meaning that this flow surface is shared between domains Ω_i and Ω_j.

   The flow rate into domain Ω_i is defined as:

$$R_i = R_i^+ - \sum_{j=0}^{i-1} \sum_{k=1}^{m_{ij}} R^k_{i\to j} A^k_{i\to j} + \sum_{j=i+1}^{n} \sum_{k=1}^{m_{ji}} R^k_{j\to i} A^k_{j\to i} \ ,$$

   where R_i^+ is a source term and $R^k_{i\to j}$ is the flow rate density from Ω_i to Ω_j through $\Gamma^k_{i\to j}$. We assume that the individual flow rate densities depend on pressure differences meaning $R^k_{i\to j} = R^k_{i\to j}(p_i - p_j)$. The volume increase of Ω_i can therefore be calculated as

$$\Delta V_i = \int_0^t R_i dt \ .$$

   We now introduce the volume constraint, which in vector notation is written as:

$$\mathbf{c} = \mathbf{V} - \mathbf{V}_0 - \Delta\mathbf{V} = \mathbf{0} \ ,$$

   where $\mathbf{V}_0$ are the volumes at time zero. A standard Lagrangian multiplier approach results in:

$$\mathbf{r}(\mathbf{x}) + \left(\frac{\partial \mathbf{c}}{\partial \mathbf{x}}\right)^T \mathbf{q} = \mathbf{0} \ ,$$

where $\mathbf{q}$ is the vector of Lagrangian multipliers representing the pressure inside the domains. We may now set $\mathbf{p} = \mathbf{q}$ to complete the set of equations in unknowns $\mathbf{x}$ and $\mathbf{p}$.

As discussed in Remark 2, the actual implementation is slightly different in that the nodal force contributions do not involve the pressures exerted on flow areas. So, the last equilibrium equation is changed to

$$\mathbf{r}(\mathbf{x}) + \mathbf{N}^T\mathbf{p} = \mathbf{0},$$

where $\mathbf{N}$ is a vector that accomplishes just that. Deriving the stiffness matrix, while neglecting second-order contributions, we end up with

$$\mathbf{K} = \begin{pmatrix} \dfrac{\partial \mathbf{r}}{\partial \mathbf{x}} & \mathbf{N}^T \\ \dfrac{\partial \mathbf{c}}{\partial \mathbf{x}} & \dfrac{\partial \mathbf{c}}{\partial \mathbf{p}} \end{pmatrix}.$$

It is nonsymmetric in the general case (flow areas exist).

*DEFINE_CONTROL_VOLUME_FLOW_AREA

Purpose: Specify the flow area between two interacting control volumes (see *DEFINE_-CONTROL_VOLUME).

Card 1	1	2	3	4	5	6	7	8
Variable	FAID	FCIID	FASID	FASTYP				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
FAID	Flow area ID
FCIID	Fluid cavity interaction ID referencing the *DEFINE_CONTROL_-VOLUME_INTERACTION for which this area is used
FASID	Set ID giving the flow area
FASTYP	Type of set specifying flow area (see Remark 2): EQ.1: Node set giving the perimeter of the area. The flow area will be automatically meshed. EQ.2: Segment set covering the flow area

Remarks:

1. **Volume calculation of the control volumes.** The flow area segments are added to both control volumes automatically to close them for the volume calculation.

2. **Specifying flow area.** The flow area can be defined with either a segment set or a node set. For the segment set, the nodes forming the segments must be on the boundary of the flow area. Nodes placed inside the flow area will not move as the control volume deforms which may lead to poor accuracy in the flow area calculation. See Remark 4.

   Defining the area with a node set requires ordering the nodes giving the perimeter such that the normal of the area points inward toward the first control volume listed in the interaction and, thus, outward from the second. The

numbering, therefore, proceeds clockwise around a hole in the first control volume when viewed from the outside.

3. **Forces in flow area.** The segments or null shells defining the flow areas on the control volumes do not exert any force.

4. **Calculation of the flow area.** The flow area is calculated as the surface area of the segments defining it, and not as a projected area in the direction of some mean normal direction. For example, if the flow area is meshed about a node in the center for the flow area and the perimeter nodes translate relative to the motionless center node, the resulting mesh will define a cone. The flow area will be the surface area of the cone and not the circular cross-section area.

*DEFINE_CONTROL_VOLUME_INTERACTION

Purpose: Specify the interaction between two control volumes (see *DEFINE_CONTROL_VOLUME).

Card 1	1	2	3	4	5	6	7	8
Variable	FCIID	CVID1	CVID2	LCID	AREA			
Type	I	I	I	I	F			
Default	none	none	none	none	optional			

VARIABLE	DESCRIPTION
FCIID	Fluid cavity interaction ID
CVID1	First control volume ID (see *DEFINE_CONTROL_VOLUME)
CVID2	Second control volume ID (see *DEFINE_CONTROL_VOLUME)
LCID	Load curve ID for a *DEFINE_FUNCTION or user subroutine specifying the pressure gradient-driven interaction between the connected cavities (see Remarks 1 and 2): GT.0: ID for *DEFINE_FUNCTION. The functions's input arguments are time, pressure difference, the area between the control volumes, the current time step size, and a flag for the start of a new time step. Currently, LS-DYNA passes the current time and cavity pressures as input arguments. LT.0: User subroutine `usr_icvflow` in dyn21umat.F determines the pressure gradient.
AREA	Constant area for the case when a flow area is not specified with *DEFINE_CONTROL_VOLUME_FLOW_AREA

Remarks:

1. **Function / user subroutine.** Specifying the load curve as a function with *DEFINE_FUNCTION or as a user subroutine gives the freedom to define various (nonlinear/time-dependent) lumped parameter models, that is, boundary conditions between the cavities.

2. **Area of valve opening.** We assume that the area/size of the valve opening plays no role in the defined interaction for the curve.

*DEFINE_COORDINATE_NODES

Purpose: Define a local coordinate system with three node numbers. The local Cartesian coordinate system is defined in the following steps. If the primary direction is along the x-axis, then the z-axis is computed from the cross product of x and $\bar{y}$, $z = x \times \bar{y}$ (see Figure 17-9); the y-axis is, then, computed as $y = z \times x$. A similar procedure applies if the local axis is along the y or z axes.

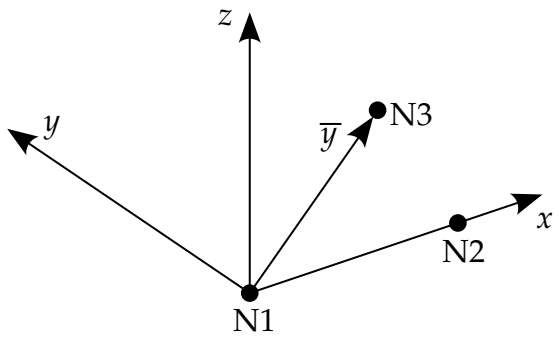


Figure 17-9. Definition of the local coordinate system when DIR = X.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	N1	N2	N3	FLAG	DIR	TYPE	
Type	I	I	I	I	I	A	I	
Default	none	none	none	none	0	X	0	

VARIABLE	DESCRIPTION
CID	Coordinate system ID. A unique number has to be defined.
N1	ID of node located at local origin.
N2	ID of node located along local x-axis if DIR = X, the y-axis if DIR = Y, and along the z-axis if DIR = Z.
N3	ID of a node located in the local xy-plane if DIR = X, the local yz-plane if DIR = Y, and the local zx-plane if DIR = Z.
FLAG	Set to unity, 1, if the local system is to be updated each time step. Generally, this option is *not recommended* when used with nodal SPCs; it can cause excursions in the energy balance because the constraint forces at the node may go through a displacement if the node is partially constrained.

VARIABLE	DESCRIPTION
DIR	Axis defined by the vector pointing from the origin node N1 to node N2. The default direction is the *x*-axis.
TYPE	Coordinate system type: EQ.0: Cartesian EQ.1: Cylindrical EQ.2: Spherical This field is only supported by *BOUNDARY_PRESCRIBED_MOTION, *LOAD_NODE, and *BOUNDARY_SPC to flag that the motion or load is in that coordinate system. The spherical and cylindrical coordinate systems are based on the local Cartesian system found with the input nodes. See Figure 17-10.

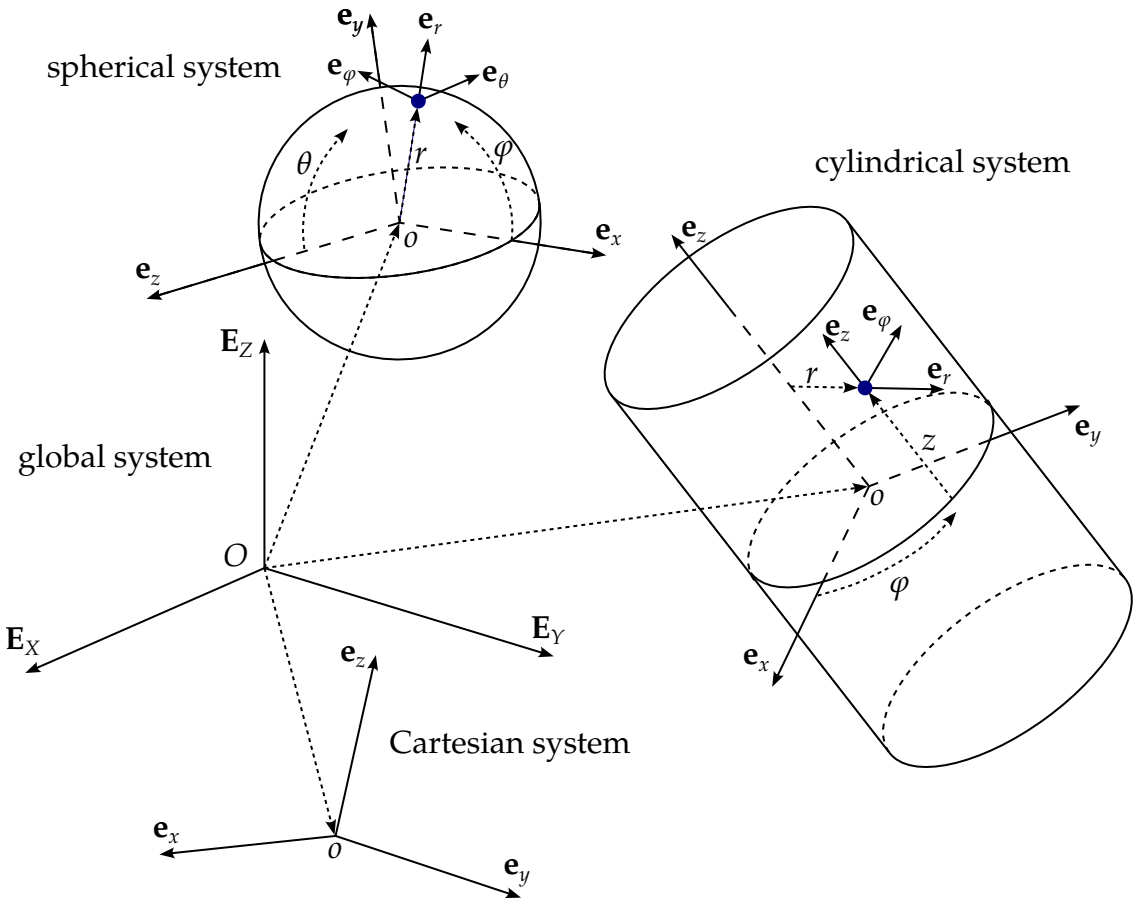


Figure 17-10. Local Cartesian, spherical, and cylindrical coordinate systems. The input data determines the local Cartesian coordinate system, denoted by the orthonormal triplet $\{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$, from which the cylindrical and spherical coordinate systems can be constructed.

Remarks:

The nodes N1, N2, and N3 must be separated by a reasonable distance and not collinear to avoid numerical inaccuracies.

*DEFINE_COORDINATE_SYSTEM_{OPTION}

Available options include:

<BLANK>

IGES

Purpose: Define a local coordinate system.

This card implements the same method as *DEFINE_COORDINATE_NODES, but, instead of reading coordinate positions from nodal IDs, it directly reads the three coordinates from its data cards as Cartesian triples.

Using the IGES option causes the generation of the coordinate system from an IGES file containing three straight curves representing the *x*-, *y*-, and *z*-axes. See Remark 4.

Card Summary:

Card 1. Include this card if the keyword option is unset (<BLANK>).

CID	XO	YO	ZO	XL	YL	ZL	CIDL

Card 2. Include this card if the keyword option is unset (<BLANK>).

XP	YP	ZP	TYPE				

Card 3. Include this card if the IGES keyword option is used.

FILENAME

Data Card Definitions:

Card 1 for <BLANK> Keyword Option.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	XO	YO	ZO	XL	YL	ZL	CIDL
Type	I	F	F	F	F	F	F	I
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
CID	Coordinate system ID. A unique number must be defined.
XO	*X*-coordinate of the origin
YO	*Y*-coordinate of the origin
ZO	*Z*-coordinate of the origin
XL	*X*-coordinate of a point on the local *x*-axis.
YL	*Y*-coordinate of a point on the local *x*-axis.
ZL	*Z*-coordinate of a point on the local *x*-axis.
CIDL	Coordinate system ID applied to the coordinates used to define the current system. The coordinates X0, Y0, Z0, XL, YL, ZL, XP, YP, and ZP are defined with respect to the coordinate system CIDL.

Card 2 for <BLANK> Keyword Option.

Card 2	1	2	3	4	5	6	7	8
Variable	XP	YP	ZP	TYPE				
Type	F	F	F	I				
Default	0.0	0.0	0.0	0				

VARIABLE	DESCRIPTION
XP	*X*-coordinate of a point in the local *xy*-plane.
YP	*Y*-coordinate of a point in the local *xy*-plane.
ZP	*Z*-coordinate of a point in the local *xy*-plane.
TYPE	Coordinate system type: EQ.0: Cartesian EQ.1: Cylindrical EQ.2: Spherical This field is only supported by *BOUNDARY_PRESCRIBED_MOTION, *LOAD_NODE, and *BOUNDARY_SPC to flag that the

VARIABLE	DESCRIPTION
	motion or load is in that coordinate system. The spherical and cylindrical coordinate systems are based on the local Cartesian system found with the points input on this keyword. See Figure 17-10.

Card 1 for IGES Keyword Option.

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of the IGES file containing three curves (see Remark 4 below).

Remarks:

1. **Avoiding numerical inaccuracies.** The coordinates of the points must be separated by a reasonable distance and cannot be collinear to avoid numerical inaccuracies.

2. **Chains of coordinate transformations.** Care must be taken to avoid chains of coordinate transformations because there is no guarantee that they will be executed in the correct order.

3. **LS-PrePost.** A coordinate system can be created using the dialog box located at *Model* (main window) → *CreEnt* → *Define* (see the left pane) → *Coordinate*. This will activate a *Define Coordinate* dialog in the right pane. Select the *Cre* radio button at the top of the right pane and set the *type* dropdown to **SYSTEM*. The next set of radio buttons (below the title input box) sets the method used to define the coordinate system. See Figure 17-11.

   a) The *N+Xaxis* method generates a coordinate system from based on:

      i) a user specified origin,

      ii) one of the three global axes (this is a *severe* restriction), and

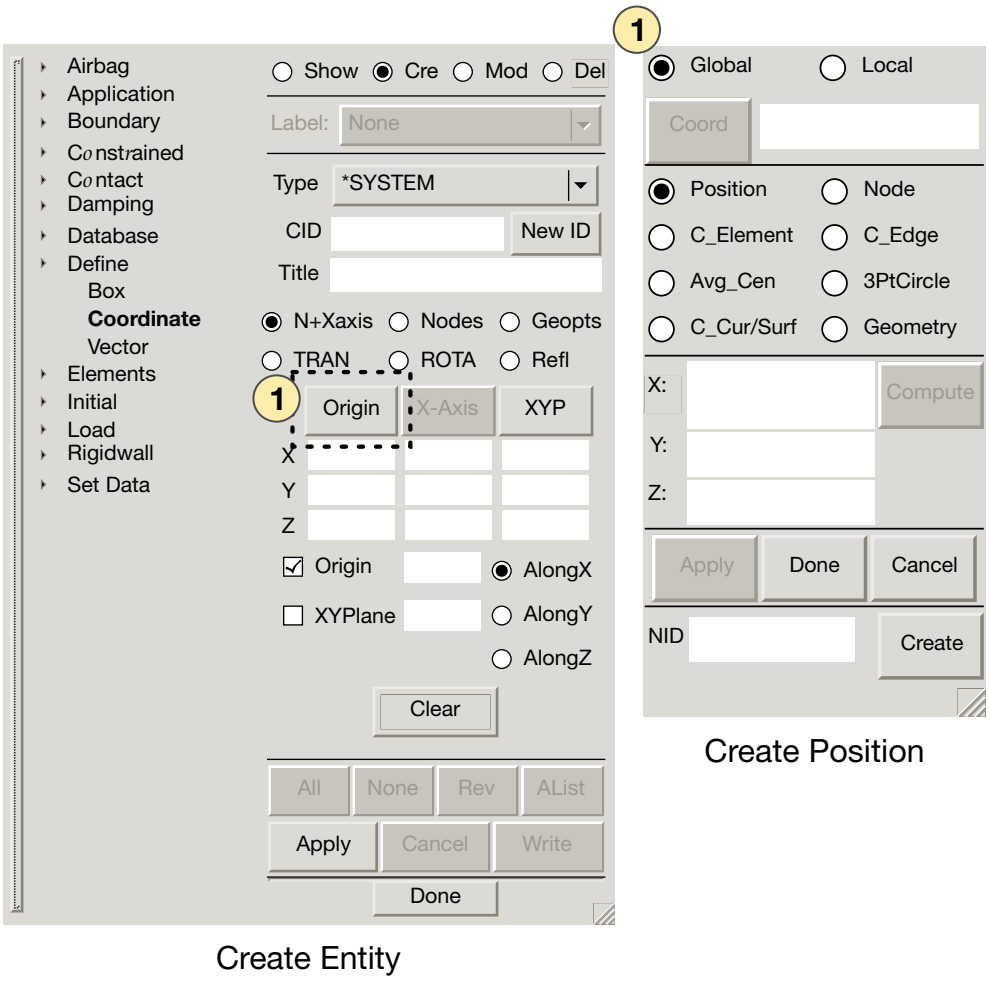


Figure 17-11. *LS-PrePost4.0* Dialog for defining a coordinate system.

iii) a 3rd point.

The 3rd point, together with the specified global axis defines the new system's *xy*-plane. The remaining axes are derived using orthogonality and right-handedness. This method requires the user to pick two points which involves the *Create Position* dialog box, as shown in the left frame of Figure 17-11.

NOTE: After defining each point in the *Create Position* dialog, it is *very important* to use the *done* button. The *Create Entity* dialog stays up and remains interactive while the *Create Position* dialog is also up and interactive. This can be confusing. Returning to the *Create Entity* dialog without choosing *done* is a common mistake.

b) The *node* method generates a coordinate system from three points:

i) The first point specifies the origin.

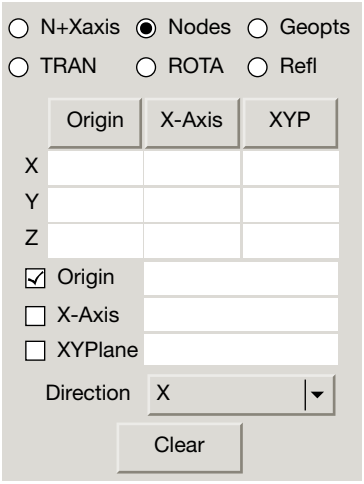


Figure 17-12. Subset of *Create Entity* dialog for both *Nodes* and *Geopts* methods.

ii) The first and second points together specify the *x*-axis.

iii) The three points together specify the *xy*-plane of the new coordinate system. The *y*- and *z*-axis are derived from orthogonality and right-handedness.

c) The *Geopts* option generates the new coordinate system from a global axis and two points. With this method the new system's *z*-axis is set from the *Direction* drop-down. This new system's *xy*-plane is, then, orthogonal to the chosen direction. The remaining two points serve to define the origin and the *x*-axis (by projecting the second point). This option is useful for metal forming application, since, often times, only the *z*-axis is important while the while the *x*- and *y*-axes are not.

4. **IGES.** When option, IGES, is used, three curves in the IGES format will be used to define a local coordinate system. IGES curve entity types 126, 110 and 106 are currently supported. Among the three curves, the longest length will be made as local *Z*-axis, the mid-length will be *Y*-axis and the shortest length *X*-axis. Suggested *X*-, *Y*- and *Z*-axis lengths are 100 mm, 200 mm and 300 mm, respectively.

   All three curves must have one identical point and will be used for the origin of the new local coordinate system. The coordinate system ID for the local system will be based on the IGES file name. The IGES file name must start with a number, followed by an underscore "_", or by a dot. The number preceding the file name will be used as the new local coordinate system ID, which can then be referenced in *MAT_20 cards, for example.

   After the LS-DYNA run, three beam elements of a new PID will be created in place of the three curves representing the local *X*-, *Y*-, and *Z*-axes in the d3plot file for viewing in LS-PrePost. See Figure 17-13.

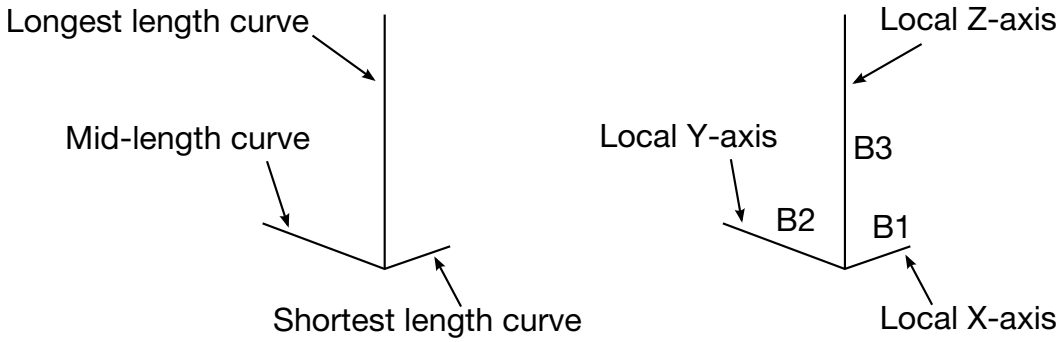


Figure 17-13. Input curves (left). The generated local coordinate system is written to the d3plot file as a part consisting of three beams (right).

The following partial input contains an example in which the keyword is used to create a local coordinate system (CID = 25) from IGES input. The IGES file named, 25_iges, contains three intersecting curves in one of the three supported IGES entity types. The example demonstrates using the IGES coordinate system (ID = 25) to specify the local coordinate system for a rigid body (PID = 2, MID = 2). The keyword, *BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL, then uses this local coordinate system to assign velocities from load curves 3 and 5 for the rigid body motion in the local *x*-direction.

```
*KEYWORD
*DEFINE_COORDINATE_SYSTEM_IGES_TITLE
Flanging OP25
25_iges
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PART
punch
         2         2         2
*MAT_RIGID
$      MID        RO         E        PR         N    COUPLE         M     ALIAS
         2 7.830E-09 2.070E+05      0.28
$      CMO      CON1      CON2
        -1        25    011111
$LCO or A1        A2        A3        V1        V2        V3
25
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*BOUNDARY_PRESCRIBED_MOTION_RIGID_LOCAL
$   typeID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
         2         1         0         3      -1.0         0   0.00241       0.0
         2         1         0         5      -1.0         0 0.0115243   0.00241
```

The keyword can be repeated for each new coordinate system if multiple coordinate systems are needed.

*DEFINE_COORDINATE_VECTOR

Purpose: Define a local coordinate system with two vectors, see Figure 17-14. The vector cross product, $\mathbf{z} = \mathbf{x} \times \mathbf{v}_{xy}$ where $\mathbf{v}_{xy}$ is a vector in the xy-plane, determines the z-axis. The y-axis is then given by $\mathbf{y} = \mathbf{z} \times \mathbf{x}$. If this coordinate system is assigned to a nodal point, then at each time step during the calculation, the coordinate system is incrementally rotated using the angular velocity of the nodal point to which it is assigned.

Card	1	2	3	4	5	6	7	8
Variable	CID	XX	YX	ZX	XV	YV	ZV	NID
Type	I	F	F	F	F	F	F	I
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	0.

VARIABLE	DESCRIPTION
CID	Coordinate system ID. A unique number has to be defined.
XX	X-coordinate on local x-axis. Origin lies at $(0,0,0)$.
YX	Y-coordinate on local x-axis
ZX	Z-coordinate on local x-axis
XV	X-coordinate of local xy-vector
YV	Y-coordinate of local xy-vector
ZV	Z-coordinate of local xy-vector
NID	Optional nodal point ID. The coordinate system rotates with the rotation of this node. If the node is not defined, the coordinate system is stationary. See Remark 2.

Remarks:

1. **Numerical Inaccuracies.** These vectors should be separated by a reasonable included angle to avoid numerical inaccuracies.

2. **Rotation.** Ideally, this nodal point should be attached to a rigid body or a structural part where the nodal point angular velocities are meaningful. It should be

noted that angular velocities of nodes may not be meaningful if the nodal point is attached only to solid elements and even to shell elements where the drilling degree of freedom may be singular, which is likely in flat geometries.

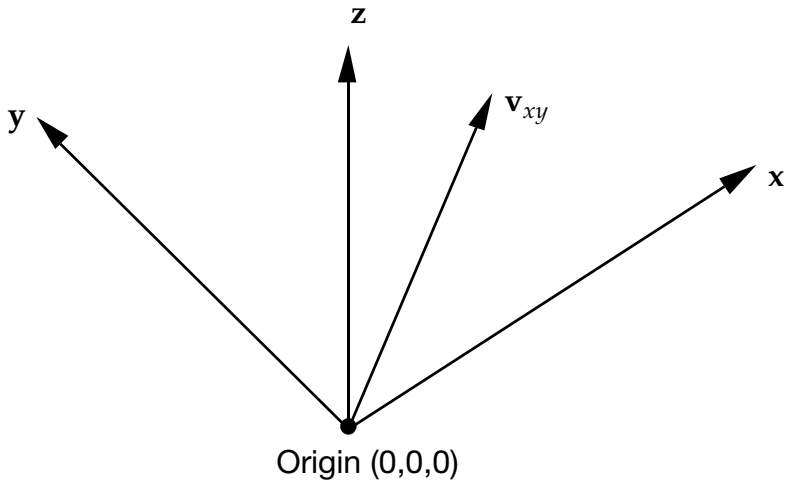


Figure 17-14. Definition of the coordinate system with two vectors.

*DEFINE_CPG_ADAPTIVE

Purpose: Set adaptive options for CPG particle size.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	DISTSW	HMIN	HMAX					
Type	I	F	F					
Default	0	none	none					

VARIABLE	DESCRIPTION
ID	Unique ID for this adaptivity definition
DISTSW	Distance to the wall switch: EQ.0: Inactive EQ.1: Active. The particle size will dynamically vary from HMIN at the wall to HMAX at a distance away from the wall. See Remark 1.
HMIN	Particle size at the wall
HMAX	Particle size away from the wall

Remarks:

1. **Transition between HMIN and HMAX.** The transition distance between HMIN (particle size at the wall) and HMAX (particle size far away from the wall) is automatically calculated by the solver.

*DEFINE_CPG_GAS_PROPERTIES

Purpose: Define extended gas thermodynamic properties for the CPG solver.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	XMM	CP0	CP1	CP2	CP3	CP4	
Type	I	F	F	F	F	F	F	
Default	none	none	0.	0.	0.	0.	0.	

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	MU							
Type	F							
Default	0.							

This card is optional. If using this card, Card 2 must also be included, but it can be a blank line.

Card 3	1	2	3	4	5	6	7	8
Variable	TC							
Type	F							
Default	0.							

VARIABLE	DESCRIPTION
ID	Unique ID for this card
XMM	Molar mass

VARIABLE	DESCRIPTION
CP0, …, CP4	Coefficients of temperature-dependent specific heat at constant pressure $$C_p(T) = C_{p0} + C_{p1}T + C_{p2}T^2 + C_{p3}T^3 + C_{p4}T^4$$
MU	Dynamic viscosity associated with the gas. See Remark 1.
TC	Thermal conductivity of the gas. See Remark 2.

Remarks:

1. **Mixture viscosity.** We use Carr's law to estimate the gas-mixture viscosity composition:

$$\mu_{gm} = \frac{\sum_{i=1}^{N} y_i \mu_i \sqrt{M_i}}{\sum_{i=1}^{N} y_i \sqrt{M_i}}$$

with y_i the mole fraction of the i^{th} gas mixture component, μ_i the viscosity of the i^{th} component, and M_i the molar mass of the i^{th} component.

To obtain a constant viscosity in the entire domain, define the same value for each gas.

Defining viscosity is mandatory for the calculation of a friction force at the wall. See PFRIC in *AIRBAG_CPG.

2. **Mixture thermal conductivity.** The gas thermal conductivity is estimated using a simple mass-fraction-weighted average. In airbag applications, thermal diffusive effects are expected to be small.

3. **Nonmonotonic temperature-dependent specific heat**. When the coefficients of the temperature-dependent specific heat at constant pressure define a nonmonotonic function, the curve is trimmed such that it becomes the following:

$$C_p(T) = \begin{cases} C_{p0} + C_{p1}T_{\min} + C_{p2}T_{\min}^2 + C_{p3}T_{\min}^3 + C_{p4}T_{\min}^4 & \text{for} \quad 0 < T < T_{\min} \\ C_{p0} + C_{p1}T + C_{p2}T^2 + C_{p3}T^3 + C_{p4}T^4 & \text{for} \quad T_{\min} \le T \le T_{\max} \\ C_{p0} + C_{p1}T_{\max} + C_{p2}T_{\max}^2 + C_{p3}T_{\max}^3 + C_{p4}T_{\max}^4 & \text{for} \quad T_{\max} < T \le \infty \end{cases}$$

Here, $T_{\min}$ and $T_{\max}$ centered around the ambient temperature are automatically identified.

*DEFINE_CPG_REGION

Purpose: Define a geometrical region in the CPG domain for applying properties such as initial conditions (see *INITIAL_CPG).

Card 1	1	2	3	4	5	6	7	8
Variable	DRID	DRTYP						
Type	I	I						
Default	none	none						

Card 2	1	2	3	4	5	6	7	8
Variable	P1X	P1Y	P1Z	P2X	P2Y	P2Z	RAD	
Type	F	F	F	F	F	F	F	
Default	0.	0.	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
DRID	Unique ID for this region
DRTYP	Region geometry type: EQ.1: Domain divided by a section plane that acts as a cut-off. The plane's normal vector points outwards from the region. EQ.2: Box EQ.3: Cylinder EQ.4: Sphere
P1X/P1Y/P1Z/ P2X/P2Y/P2Z	Points with meaning that depend on DRTYP: DRTYP.EQ.1: Plane normal and origin point DRTYP.EQ.2: Minimum and maximum coordinates of the box

VARIABLE	DESCRIPTION
	DRTYP.EQ.3: Points defining the center axis
	DRTYP.EQ.4: Origin of the sphere (second point not used)
RAD	Cylinder or sphere radius if DRTYP = 3 or 4, respectively

*DEFINE_CPG_VENT

Purpose: Define extended CPG vent options. See *AIRBAG_CPG.

Card 1	1	2	3	4	5	6	7	8
Variable	DVID	LCT	LCP		PPOP		C23UP	TOPEN
Type	I	I	I		F		F	F
Default	none	0	0		0.0		1.0	optional

VARIABLE	DESCRIPTION
DVID	Unique ID for this card. This is referenced by DVID on *AIRBAG_CPG.
LCT	Optional load curve to impose temperature as a function of time in cases where backflow is detected. Otherwise, TATM in *AIRBAG_CPG is used. See Remark 6 of *AIRBAG_CPG.
LCP	Optional load curve to impose pressure as a function of time. Otherwise, PATM of *AIRBAG_CPG is used. See Remark 6 of *AIRBAG_CPG.
PPOP	Pressure difference between the interior and ambient pressure (PATM) for opening the vent holes. Once the vents are open, they stay open. EQ.0.0: The vent holes are open at the beginning of the simulation unless TOPEN is set to a nonzero value.
C23UP	Scale factor of C23 while switching from a CPG to uniform pressure calculation
TOPEN	Optional time for opening the vent holes. If PPOP is also defined (or nonzero), both conditions need to be met for the vent to open.

*DEFINE_CPM_BAG_INTERACTION

Purpose: To model energy flow from a one airbag (the source) to another airbag (the sink). The source must be an active particle airbag and the sink a control volume (CV) airbag converted from a particle bag.

To track the flow of energy, LS-DYNA automatically determines which vent parts are common to both airbags. At each time step the energy that is vented through the common vents is subtracted from the source and added to the sink. In turn, the sink bag's pressure provides the downstream pressure value for the source bag's venting equation. While this model accounts for energy flow from source to sink, it ignores flow from sink to source.

If CHAMBER is used for the sink CV bag, see Remark 1.

Card 1	1	2	3	4	5	6	7	8
Variable	BAGID1	BAGID2	NSPEC					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
BAGID1	Airbag ID of source CPM particle bag
BAGID2	Airbag ID of sink CV bag switched from CPM bag
NSPEC	The location of the 1st gas component from the CPM bag to be filled in the CV bag. See Remark 2.

Remarks:

1. **Chambers.** Due to the complexity of the bookkeeping, the sink bag may have several chambers, but only one of the chambers may interact with the source bag. LS-DYNA finds this chamber automatically by looking at the commonly shared parts.

2. **NSPEC.** If NSPEC equals zero, the mass and energy vented out from the CPM bag will be directly fed into the CV bag. In this case the two bags are assumed to have the same gas species.

If the CPM bag and the CV bag contain different gas species initially, the gas species of the CPM bag must be listed after the inflator gas species of the CV bag (see NGAS on *AIRBAG_PARTICLE). The order of the gas components should be the same. NSPEC gives the first location of the CPM gas in the CV gas species list.

*DEFINE_CPM_CHAMBER

Purpose: Define airbag chambers for air particle initialization or chamber interaction.

> **NOTE:** The part set that specifies the airbag, SID1, for *AIRBAG_PARTICLE must include the parts that define the chambers, meaning SID1 for this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	NCHM						
Type	I	I						
Default	none	0						

Chamber Definition Card Sets:

Add NCHM chamber definition card sets. Each chamber definition card set consists of a Chamber Definition Card followed by NINTER Interaction Cards.

Chamber Definition Card.

Card 2	1	2	3	4	5	6	7	8
Variable	SID1	SID2	NINTER	CHM_ID				
Type	I	I	I	I				
Default	none	0	0	0				

Interaction Cards. Add NINTER of these. If NINTER = 0, skip this card.

Card 3	1	2	3	4	5	6	7	8
Variable	SID3	ITYPE3	TOCHM					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
ID	Unique ID for this card
NCHM	Number of chambers defined in this card
SID1	Part set defining all parts that constitute the chamber volume. See Remark 1.
SID2	Part set defining the parts whose shell normal vectors need to be flipped, such as separation walls between chambers. See Remark 1.
NINTER	Number of vent hole definition for chamber interaction.
CHM_ID	Chamber ID (see Remark 2)
SID3	Set defining interaction between chambers
ITYPE3	Set type: EQ.0: Part EQ.1: Part set
TOCHM	Chamber ID of the connected chamber

Remarks:

1. **Normal vectors.** Each chamber's volume is calculated based on the part normals pointed inwards. So SID1 would normally have parts with their shell normals pointing inwards. But in some cases, parts may be shared by more than one chamber. In this case, the shell orientation of certain part(s) may need to be flipped for the other chambers in question. In such cases, SID2 can be used to flip the shell normals for specific parts. An example of this is given below and illustrated in Figure 17-15.

```
*SET_PART_LIST
```

```
$#      sid
          1
$#     pid1       pid2       pid3       pid4
          1          2          3          4
*SET_PART_LIST
$#      sid
         20
$#     pid1       pid2
          1          2
*DEFINE_CPM_CHAMBER
$#       id       nchm
      1234          2
$#     sid1       sid2     ninter     chm_id
         20          0          1        998
$#     sid3     itype3      tochm
          2          0        997
$#     sid1       sid2     ninter     chm_id
          1         20          1        997
$#     sid3     itype3      tochm
          2          0        998
```

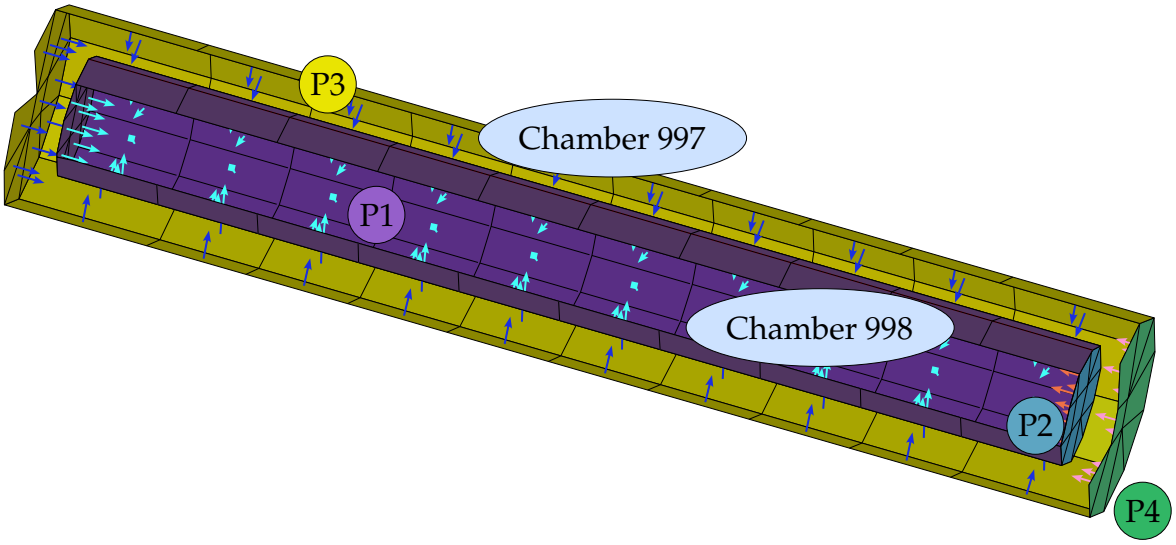


Figure 17-15. Illustration of the example in Remark 1

2. **Particle collisions.** Particles with different chamber ID will not interact in particle to particle collision. This feature will allow program to distinguish particles separated by a thin wall.

3. **Output files.** All chambers data are output to lsda binout database. The utility "l2a" can convert it into abstat_chamber ASCII file and process with LS-PrePost under abstat format.

*DEFINE_CPM_GAS_PROPERTIES

Purpose: Define extended gas thermodynamic properties for CPM airbags.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	XMM	CP0	CP1	CP2	CP3	CP4	
Type	I	F	F	F	F	F	F	
Default	none	none	0.	0.	0.	0.	0.	

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	MUT0	MUT1	MUT2	MUT3	MUT4	CHM_ID	VINI	LCMCF
Type	F	F	F	F	F	I	F	I
Default	0.	0.	0.	0.	0.	0	0.	0

This card is optional. If using this card, Card 2 must also be included, but it can be a blank line.

Card 3	1	2	3	4	5	6	7	8
Variable	PID	PCUT	MCUT	TTIME	ISMTH			
Type	I	F	F	F	I			
Default	0	1e+20	-1e+20	0.	20			

VARIABLE	DESCRIPTION
ID	Unique ID for this card
XMM	Molar mass

VARIABLE	DESCRIPTION
CP0, …, CP4	Coefficients of temperature-dependent specific heat with constant pressure $$C_p(T) = C_{p0} + C_{p1}T + C_{p2}T^2 + C_{p3}T^3 + C_{p4}T^4$$
MUT0, …, MUT4	Coefficients of temperature-dependent Joule-Thomson effect $$\mu_t(T) = \mu_{t0} + \mu_{t1}T + \mu_{t2}T^2 + \mu_{t2}T^3 + \mu_{t2}T^4$$
CHM_ID	Chamber ID (see Remark 1)
VINI	Initial volume for user-defined inflator (see Remark 1): EQ.0.0: User-defined inflator disabled GT.0.0: Initial volume LT.0.0: Calculate volume based on chamber geometry
LCMCF	Optional load curve ID referring to a *DEFINE_CURVE_FUNCTION. If specified, this load curve gives the total mass flow rate of the gas. Each LCM*i* on Card 13 of *AIRBAG_PARTICLE is then not used to give the mass flow rate for component *i*. However, LCM*i* must still be provided because it is used to determine the mass of gas component *i* in the system at time *t* by integrating the curve from 0 to *t*. LCMCF is not supported when using the MOLEFRACTION keyword option with *AIRBAG_PARTICLE.
PID	Part ID to get part pressure
PCUT	Cutoff pressure. If the part pressure found for PID is greater than PCUT, the mass flow rate of this gas is restricted. The pressure compared to PCUT is determined with moving average over the number of cycles specified with ISMTH.
MCUT	Cutoff mass flow rate. If the total mass flow rate, excluding this gas, is declining and below MCUT, the mass flow rate of this gas is restricted.
TTIME	Taper down duration. After satisfying one of the above cutoff conditions (MCUT or PCUT), a linear factor from 1.0 to 0.0 is applied to this mass flow rate. If TTIME equals zero, the flow from this gas is stopped after the cutoff.
ISMTH	Number of cycles to perform the moving average over when smoothing the part pressure to determine if it has exceeded PCUT.

VARIABLE	DESCRIPTION
	This smoothing is performed because the part pressure is noisy. The default is the average over the most recent 20 cycles.

Remarks:

1. **User-defined routines.** If you define CHM_ID and VINI, the user_inflator routine uses this gas property. We provide this routine in the user-defined subroutine Fortran files of the general usermat package. LS-DYNA gives the current chamber volume, pressure, temperature, and time step to the subroutine and expects a return value of change of chamber, burned gas temperature, and mass flow rate to feedback for releasing particles. The abstat_chamber section of binout contains all state data for this chamber.

Example:

```
*AIRBAG_PARTICLE
$====1====$====2====$====3====$====4====$====5====$====6====$====7====$====8====
      1010         1      1011         1         0       0.0       0.0         1
    100000         0         1     300.0   1.0e-04         1
         1         1         1
        61         0       1.0         0         0         1       0.0
   1.0E-04     300.0     -9900
       651       653     -9910
   3000001       1.0
$================================================
*DEFINE_CPM_GAS_PROPERTIES
$====1====$====2====$====3====$====4====$====5====$====6====$====7====$====8====
      9900 2.897E-02 2.671E+01 7.466E-03-1.323E-06
      9910   4.0E-03     20.79
   -610.63   -0.0926
```

*DEFINE_CPM_NPDATA

Purpose: To define extended NPDATA parameters for CPM airbags (see *AIRBAG_PARTICLE).

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HCONV	PFRIC	SDFBLK	KP	INIP	CP	PSFDCF
Type	I	F	F	F	F	I	F	F
Default	none	none	none	1.0	none	none	none	1.0

This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	HCINT	CDEXT						
Type	F	F						
Default	optional	0.0						

VARIABLE	DESCRIPTION
ID	Unique ID for this card (referred to by HCONV on *AIRBAG_PARTICLE)
HCONV	Convective heat transfer coefficient used to calculate heat loss from the airbag's external surface to the ambient. See *AIRBAG_HYBRID developments (Resp. P.O. Marklund). LT.0.0: |HCONV| is a load curve ID defining the heat convection coefficient as a function of time. *DEFINE_CURVE_FUNCTION is supported for specifying this curve.
PFRIC	Friction factor, F_r, if -1.0 < PFRIC ≤ 1.0. Defaults to FRIC from Card 1 on *AIRBAG_PARTICLE if undefined. Otherwise, LE.-1.0: |PFRIC| is the curve ID that defines F_r as a function of the part pressure. GT.1.0: PFRIC is the *DEFINE_FUNCTION ID that defines F_r.

VARIABLE	DESCRIPTION
SDFBLK	Scaling down factor for blockage factor (default = 1.0, no scaling down). The valid factor will be (0.0,1.0]. If 0.0, it is set to 1.0.
KP	Thermal conductivity of the part
INIP	Place initial air particles on the surface: EQ.0: Yes (default) EQ.1: No This feature excludes surfaces from initial particle placement. This option helps prevent particles from being trapped between adjacent fabric layers.
CP	Specific heat
PSFDCF	Additional scale factor for force decay constant. See Remark 1.
HCINT	Convective heat transfer coefficient between CPM particles and the part of interest. This feature is intended for analyses that couple the CPM and thermal solvers.
CDEXT	Drag coefficient of the external air for the part or part set. The CD_-EXT field in *AIRBAG_PARTICLE must be nonzero. GT.0.0: Drag coefficient LT.0.0: Curve ID for the time-dependent drag coefficient

Remarks:

1. **Scale factor for force decay constant.** The particle impact force is gradually applied to an airbag segment by a particular smoothing function with the following form:

$$F_{\text{apply}} = \left[1 - \exp\left(\frac{-\mathrm{d}t}{\tau}\right)\right](F_{\text{current}} + F_{\text{stored}}) \ ,$$

where τ is the force decay constant. PSFDCF scales τ:

$$\tau = \tau \times \text{PSFDCF} \ .$$

*DEFINE_CPM_SWITCH_REGION

Purpose: Define spherical or cylindrical regions of interest from a set of segments where the particle-to-particle collision will be omitted over specified intervals.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	BAGID	SSID	NFREQ	OFF	R	HEIGHT	
Type	I	I	I	I	F	F	F	
Default	none	none	none	1.0	none	none	0.0	

VARIABLE	DESCRIPTION
ID	Unique ID for this option
BAGID	*AIRBAG_PARTICLE ID to apply this option
SSID	Segment set ID
NFREQ	Number of cycles to skip the particle-to-particle collision: GT.0: Number of cycles LT.0: Time-dependent load curve with ID |NFREQ|
OFF	Offset distance of the region's CG from the segment center along the normal direction
R	Radius of the region
HEIGHT	Height of the cylinder for cylindrical domains: EQ.0.0: Spherical domain GT.0.0: Height of the cylindrical domain

*DEFINE_CPM_VENT

Purpose: Define extended vent hole options.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	C23	LCTC23	LCPC23	ENH_V	PPOP	C23UP	IOPT
Type	I	F	I	I	I	F	F	I
Default	none	1.0	none	none	↓	none	none	none

Card 2	1	2	3	4	5	6	7	8
Variable	JT	IDS1	IDS2	IOPT1	PID1	PID2	VANG	LCRED
Type	I	I	I	I	I	I	F	I
Default	0	none	none	none	none	none	0.	none

This card is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	LCAC23	PSETPV	SFPV	LPATM	IBLKOFF
Type	I	I	I	I	I	F	I	I
Default	0	0	0	0	0	0.0	0	0

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	JTND							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ID	Unique ID for this card
C23	Vent hole coefficient. This is the Wang-Nefske leakage parameter.
LCTC23	Load curve defining the vent hole coefficient as a function of time
LCPC23	Load curve or function defining the vent hole coefficient as a function of pressure: GT.0: Load curve ID for *DEFINE_CURVE_FUNCTION or *DEFINE_CURVE LT.0: *DEFINE_FUNCTION ID
ENH_V	Enhance venting option. The default is 0; however, if the Joule-Thomson effect is enabled, this field will be set to 1 automatically. EQ.0: Disable EQ.1: Enable
PPOP	Pressure difference between interior and ambient pressure to open the vent hole. Once the vent is open, it will stay open.
C23UP	Scale factor of C23 while switching from CPM to uniform pressure calculation.
IOPT	Directional venting: EQ.1: In shell normal EQ.2: Against shell normal One-way venting: EQ.10: In shell normal

VARIABLE	DESCRIPTION
	EQ.20: Against shell normal
	Special vent option:
	EQ.100: Enable compression seal vent. Vent area is adjusted according to the formula below (see Remark 1). $$A_{\text{vent}} = \max(A_{\text{current}} - A_0, 0)$$
	EQ.200: Enable push-out vent. Particle remains active while going through this external vent within the range of 2 times of its characteristic length, l_{vent}. $$l_{\text{vent}} = \sqrt{A_{\text{vent}}}$$
JT	Include the Joule-Thomson effect. When the Joule-Thomson effect is enabled, ENH_V is automatically set to 1 (enable).
	EQ.0: Disable
	EQ.1: Use part pressure
	EQ.2: Use chamber pressure
IDS1	JT's up stream condition part ID/chamber ID
IDS2	JT's downstream condition part ID/chamber ID
IOPT1	Upstream chamber ID for one-way vent hole. This will help the code to determine the probability function.
PID1, PID2	PID1 and PID2 indicate parts for determining the local part pressures that can be used to evaluate the vent probability function. Depending on if a chamber is defined, how the local part pressures are evaluated changes (see *DEFINE_CPM_CHAMBER). PID1 and PID2 are optional if a chamber is defined, otherwise they are required input.
	When a chamber is defined, specifying PID1 and PID2 causes the vent probability function to be evaluated from the difference of local part pressures between PID1 and PID2. Otherwise, the calculation involves the chamber pressure. This option is usually used for vents near a long sleeve which causes unrealistic venting using chamber pressure alone.
	When a chamber is not defined, the vent probability function is evaluated from the difference of local part pressures between PID1 and PID2, using the location of the part centers to help determine

VARIABLE	DESCRIPTION
	vent direction. If the part is an external part, the part pressure will be used. If the part is an internal part, the pressure on the shell's positive normal side will be used. If the vent is an external vent, PID1 should be the same as PID2 to avoid input error.
VANG	Cone angle in degrees. Particle going through this vent will be re-directed based on this angle. This option is only valid for an internal vent. GT.0.0: Cone angle (maximum 270°) EQ.0.0: Disabled (default) EQ.-1.0: Direction follows the vent normal (see Remark 2) EQ.-2.0: Direction follows local coordinates system defined by three nodes (see Remark 2)
LCRED	Time dependent probability curve to control CPM particle through the internal vent with VANG option (see Remark 3).
NID1-NID3	Three nodes define a moving coordinate system for the direction of flow through the vent when VANG = -2. (see Remark 4)
LCAC23	Load curve defining vent hole coefficient as a function of current vent area
PSETPV	|PSETPV| is a part set ID for internal airbag parts that interact with the push-out vent (IOPT = 200). The sign determines where the ambient pressure is applied: GT.0: Ambient pressure is applied to elements in these parts that are at least a distance of SFPV × CL away from the vent. LT.0: Ambient pressure is applied to elements in these parts that are within a distance of SFPV × CL of the vent. CL is defined as the $\sqrt{\text{element area}}$.
SFPV	Scale factor for element's characteristic length
LPATM	Load curve for ambient pressure of the external vent. This option only works for the CPM mode.
IBLKOFF	Flag for turning off blockage treatment. This flag only applies when IBLOCK is nonzero on *AIRBAG_PARTICLE. EQ.0: Use IBLOCK from *AIRBAG_PARTICLE (default). EQ.1: Turn off blockage treatment.

VARIABLE	DESCRIPTION
JTND	Node / node set for applying vent reaction force: GT.0: Node ID LT.0: Node Set ID. The average force is evenly applied among the nodes in the node set.

Remarks:

1. **Compression seal vent model.** In order to evaluate bag state variables correctly, the CPM domain needs to be a closed surface for the volume to be well-defined. If the model contains a flap vent which is free to open and close, this option will correctly maintain the bag's integrity.

2. **VANG < 0.** When VANG < 0, all the particles passing through vent are in a collinear stream following the venting direction with no spread, effectively a zero degree angle. The particles disperse after the vent due to particle collisions.

3. **LCRED.** The application of the VANG option upon a particle passing through the vent at a given time depends on the time dependent probability function, LCRED, if defined. This option can be used to strengthen or weaken the directional vent stream.

4. **NID1-NID3.** If NID1, NID2, and NID3 are given, vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, given by NID2 − NID1 and NID3 − NID1, respectively, will form the local XY-plane. The jet direction is determined by the vector $\mathbf{v}_3$ which is given as the cross product of $\mathbf{v}_1$ with $\mathbf{v}_2$, that is, $\mathbf{v}_1 \times \mathbf{v}_2$. If NID3 is zero, the jet direction is determined by the vector $\mathbf{v}_3$ which is given by NID1 − NID2.

Example:

```
*AIRBAG_PARTICLE
$====1====$====2====$====3====$====4====$====5====$====6====$====7====$====8====
      1010         1      1011         1         0       0.0       0.0         1
    100000         0         1     300.0   1.0e-04         1
         1         1         1
        61         0     -9910
   1.0E-04     300.0  2.897E-2  2.671E+1  7.466E-3 -1.323E-6
      1000      1001    4.0E-3     20.79
   3000001       1.0
$=================================================
*DEFINE_CPM_VENT
$====1====$====2====$====3====$====4====$====5====$====6====$====7====$====8====
      9910       1.0         0         0         1       0.0
         1        51         2
```

*DEFINE_CURVE_{OPTION}

Purpose: Define a curve [for example, load (ordinate value) as a function of time (abscissa value)], often loosely referred to as a load curve. The ordinate may represent something other than a load, however, as in the case of curves for constitutive models.

For most constitutive models, *DEFINE_CURVE curves are rediscretized internally with equal intervals along the abscissa for fast evaluation. Rediscretization is *not* used when evaluating loading conditions such as pressures, concentrated forces, or displacement boundary conditions (see Remark 1 for more details).

The curve rediscretization algorithm was enhanced for the 2005 release of version 970. In some instances, the new load-curve routines changed the final results enough to disrupt benchmarks. For validated models, such as those involving barriers and occupants, that require numerical consistency, there are keyword options for reverting to older algorithms.

Available options include:

<OPTION>

3858

5434a

which correspond to the first releases of version 970 and the 2005 release, respectively.

Since input errors and wrong results are sometimes related to load curve usage, a "*Load curve usage*" table is printed in the d3hsp file after all the input is read. This table should be checked to ensure that each curve ID is referenced by the option for which the curve is intended.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	SIDR	SFA	SFO	OFFA	OFFO	DATTYP	LCINT
Type	A	I	F	F	F	F	I	I
Default	none	0	1.	1.	0.	0.	0	0

Point Cards. Put one pair of points per card (2E20.0). Input is terminated at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	A1		01					
Type	E20.0		E20.0					
Default	0.0		0.0					

VARIABLE	DESCRIPTION
LCID	Load curve identification. A unique number or label not containing "." must be specified. When defined by a number, tables (see *DEFINE_TABLE) and load curves may not share common IDs. If LCID is non-numeric, the LS-DYNA execution line should include "plabel = y" for proper (but unfortunately slower) input processing.
SIDR	Flag controlling use of curve during dynamic relaxation. SIDR set to 1 or 2 will activate a dynamic relaxation phase unless IDRFLG = -999 in *CONTROL_DYNAMIC_RELAXATION. EQ.0: Load curve used in the normal analysis phase only or for other applications, EQ.1: Load curve used in the dynamic relaxation phase but not the normal analysis phase, EQ.2: Load curve applies to both the dynamic relaxation phase and normal analysis phase.
SFA	Scale factor for abscissa value. This is useful for simple modifications. EQ.0.0: Default set to 1.0.
SFO	Scale factor for ordinate value (function). This is useful for simple modifications. EQ.0.0: Default set to 1.0.
OFFA	Offset for abscissa values, see Remark 3.
OFFO	Offset for ordinate values (function), see Remark 3.

VARIABLE	DESCRIPTION
DATTYP	Data type. This affects how offsets are applied (see Remark 4). EQ.-100: For defining the proxy, α, from experiments for the chemical shrinkage coefficient as a function of temperature (see *MAT_ADD_CHEM_SHRINKAGE for details) EQ.-2: For fabric stress as a function of strain curves (*MAT_FABRIC) as described below. EQ.0: General case for time-dependent curves, force as a function displacement curves, and stress-strain curves EQ.1: For general (x,y) data curves whose abscissa values do not increase monotonically EQ.6: For general (r,s) data (coordinates in a 2D parametric space) whose values do not increase monotonically. Use for definition of trimming polygons for trimmed NURBS (*ELEMENT_SHELL_NURBS_PATCH, NL > 0).
LCINT	The number of discretization points to use for this curve. If a model contains multiple curves, then the maximum value of all LCINTs is used to determine the memory requirement of each individual curve. Extremely large values should therefore be avoided if many curves are used. EQ.0: Use value of LCINT from *CONTROL_SOLUTION.
A1, A2, …	Abscissa values. See Remark 3.
O1, O2, …	Ordinate (function) values. See Remark 3.

Remarks:

1. ***Warning* concerning rediscretization.** For constitutive models, unless noted otherwise in the material description, LS-DYNA internally rediscretizes the curve with uniform spacing to bypass searching during evaluations. The major drawback of this algorithm is that any detail in the curve on a scale finer than the uniform rediscretization grid will be smoothed-out and lost. *It is, therefore, important to avoid placing a single point off at some value approaching infinity.* The lone point at infinity will cause the resolution of the uniform grid to be coarse relative to the other points, causing the rediscretized curve to be, possibly, featureless.

Therefore, when defining curves for constitutive models, points should generally be spaced as uniformly as possible. Also, since the constitutive model curves are extrapolated, it is important to ensure that extrapolation does not lead to physically meaningless values, such as a negative flow stress. Conversely, extrapolation can be exploited to control the results of evaluations at points far from the input data.

2. The number of points in each rediscretized curve is controlled by the parameter LCINT in *CONTROL_SOLUTION. By changing LCINT to a value greater than the default of 100, the rediscretized curves may better resemble the input curves. The data points of the rediscretized curves are written to messag and d3hsp if the parameter IPCURV is set to 1 in *CONTROL_OUTPUT.

3. **Scaling.** The load curve values are scaled after the offsets are applied, that is:

$$\text{Abscissa value} = \text{SFA} \times (\text{Defined value} + \text{OFFA})$$
$$\text{Ordinate value} = \text{SFO} \times (\text{Defined value} + \text{OFFO})$$

4. **DATTYP.** The DATTYP field controls how the curve is processed during the calculation.

   a) For DATTYP = 0, positive offsets may be used when the abscissa represents time, since two additional points are generated automatically at time zero and at time 0.999 × OFFA with the function values set to zero.

   b) If DATTYP = 1, then the offsets do not create these additional points. Negative offsets for the abscissa simply shift the abscissa values without creating additional points.

   c) For material *MAT_FABRIC with FORM = 4, 14, -14, or 24, set DATTYP = -2 to define stress as a function of strain curves using engineering stress and strain instead of 2nd Piola-Kirchhoff stress and Green strain. We recommend defining a sufficiently large number of points to ensure that the internal nonlinear conversion works well.

   d) For adding chemical shrinkage effects with *MAT_ADD_CHEM_SHRINKAGE, setting DATTYP = -100 signals that the load curve defines the proxy for the chemical shrinkage coefficient as a function of temperature. This proxy is then internally converted to the chemical shrinkage coefficient.

5. **Context-dependent extrapolation.** Load curves are not extrapolated for applied loads, such as pressures, concentrated forces, and displacement boundary conditions. Function values are set to zero if the time, etc., goes off scale. Therefore, extreme care must be observed when defining load curves. In the constitutive models, extrapolation is employed when the values on the abscissa exceed the scale.

6. **Restart.** The curve offsets and scale factors are ignored during restarts if the curve is redefined. See *CHANGE_CURVE_DEFINITION in the restart section.

*DEFINE_CURVE_BOX_ADAPTIVITY

Purpose: To define a polygon adaptive box for sheet metal forming. This keyword is applicable to shell elements. This keyword is used with *CONTROL_ADAPTIVE. This keyword is similar to *DEFINE_BOX_ADAPTIVE. It is only available for SMP.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	PID	LEVEL	DIST1				
Type	I	I	I	F				
Default	none	none	none	none				

Point Cards. Include as many as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	X		Y		Z			
Type	F		F		F			
Default	none		none		none			

VARIABLE	DESCRIPTION
ID	Curve ID; must be unique. The curve must be closed: its first and last point *must* coincide. See Examples.
PID	Sheet blank Part ID, as in *PART.
LEVEL	Adaptive refinement levels, similar to the field MAXLVL in *CONTROL_ADAPTIVE. See Remark 1.
DIST1	Depth in the Z-direction that the curve defined with Card 2 will be extruded. Currently this variable must be input as a negative value. The box depth in the Z-direction will be extended in the $-Z$-direction by $Z_{min} - \|DIST1\|$ and in $+Z$-direction by $Z_{max} + \|DIST1\|$, where Z_{min} and Z_{max} are the minimum and maximum Z coordinates in all the (X, Y, Z) data points input with Card 2. See Remark

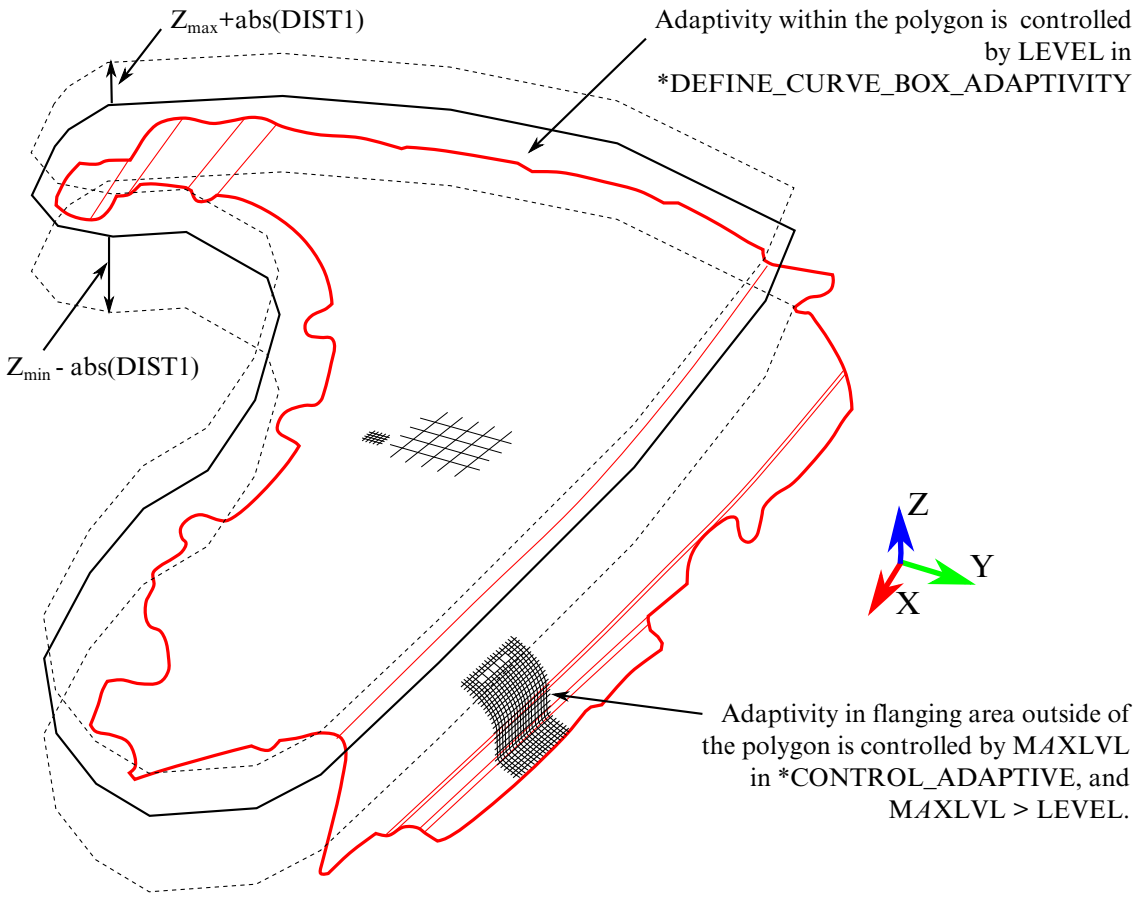


Figure 17-16. Defining an adaptive polygon box

VARIABLE	DESCRIPTION
	3 and Figure 17-16.
X	*X* coordinate of a point on the curve. See Remark 3.
Y	*Y* coordinate of a point on the curve. See Remark 3.
Z	*Z* coordinate of a point on the curve. See Remark 3.

Remarks:

1. **Mesh Refinement Definition.** Within the polygon, the variable LEVEL has priority over MAXLVL in *CONTROL_ADAPTIVE, but is limited by the minimum element size controlled by ADPSIZE. A larger LEVEL than MAXLVL value will enable more mesh refinement within the polygon, up to the size defined by ADPSIZE, than outside of the box. However, mesh refinement when LEVEL > MAXLVL is not recommended. This keyword (and *DEFINE_BOX_ADAPTIVE) is intended for the MAXLVL > LEVEL case in which the polygon box excludes the local areas of interest. For this case, refinement inside the local

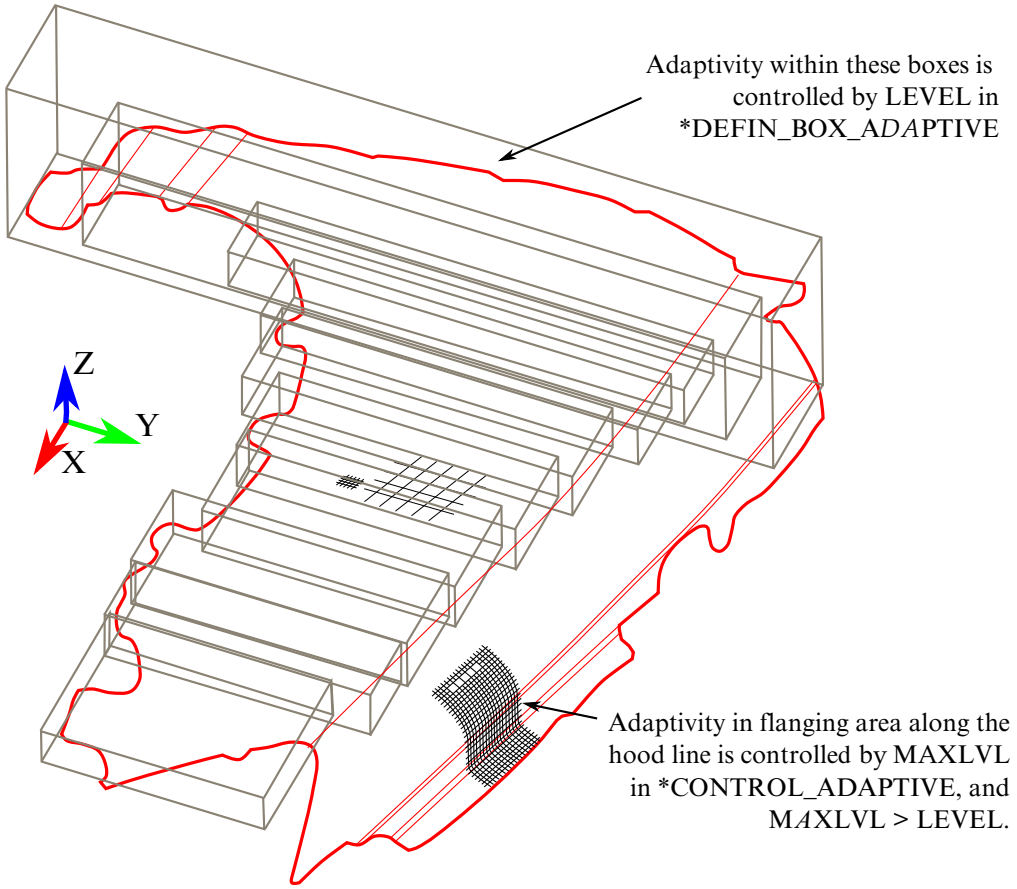


Figure 17-17. Defining adaptive boxes

areas will be controlled by MAXLVL while outside of the area will be controlled by LEVEL as shown in Figure 17-16.

2. **Advantages.** With this keyword only one box needs to be defined to specify the selected region while *DEFINE_ADAPTIVE_BOX requires multiple boxes to be defined for the same region as shown in Figure 17-17.

3. **Curve Location.** The 3D curve (closed polygon) defined by (X, Y, Z) data pairs should be near the sheet blank after the blank is auto-positioned in the beginning of a simulation. Similar to *DEFINE_BOX_ADAPTIVE, only the elements on the sheet blank initially within the polygon will be considered for use with this keyword. Local coordinate system is not supported at the moment.

4. **IGES Format.** The 3D curve can be converted from IGES format to the format required here following the procedure outlined in keyword *INTERFACE_BLANKSIZE.

Examples:

A partial keyword example is provided below, where inside the polygon mesh has no

refinement (LEVEL = 1), while outside of the box, the mesh is refined 5 levels (MAXLVL = 5). The final minimum element size is defined as 0.4. It is noted that the first point and last point of the polygon are the same, closing the polygon box.

```
*CONTROL_ADAPTIVE
$  ADPFREQ    ADPTOL    ADPTYP    MAXLVL    TBIRTH    TDEATH     LCADP    IOFLAG
   &adpfq1       5.0         2         5       0.0 1.000E+20                 1
$  ADPSIZE    ADPASS    IREFLG    ADPENE     ADPTH    MEMORY    ORIENT     MAXEL
       0.4         1         0   &lookfd       0.0                 0         0
$  IADPE90              NCFREQ    IADPCL    ADPCTL    CBIRTH    CDEATH     LCLVL
        -1
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE_BOX_ADAPTIVITY
$       ID       PID     LEVEL     DIST1
        99         1         1     -25.0
$(3E20.0):
          -59.573399           -6.698870          -40.224651
          -90.728516           24.456253          -40.224651
...
          -23.213169           18.088070          -10.954337
           14.353654           16.130911          -10.954337
          -31.070744           -5.785467          -40.487387
          -59.573399           -6.698870          -40.224651
```

DEFINE_CURVE_COMPENSATION_CONSTRAINT_*OPTION

Purpose: Allow for the definition of a localized die face region for springback compensation of stamping tools. This keyword is only available for double precision.

Options available include:

BEGIN

END

> **NOTE:** *DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN and *DEFINE_CURVE_COMPENSATION_CONSTRAINT_END are not valid in the context of a general keyword input deck. Instead, they may only be used inside of an *INCLUDE_COMPENSATION_CURVE include file.

The required option, which must be either BEGIN or END, distinguishes between two different closed curves. When taken together, these curves identify a portion of the die for applying springback compensation, and a transition region for which compensation smoothly tapers off.

Card 1	1	2	3	4	5	6	7	8
Variable	CRVID	INOUT	TYPE					
Type	I	I	I					
Default	none	0	none					

Point Cards. Include as many cards as needed. This input ends at the next keyword ("*") card. Only the projection of this curve onto the *xy*-plane is used.

Card 2	1	2	3	4	5	6	7	8
Variable	X		Y		Z			
Type	F		F		F			
Default	0.0		0.0		0.0			

VARIABLE	DESCRIPTION
CRVID	Curve ID; must be unique. The curve must be closed: its first and last point *must* coincide.
INOUT	Flag to indicate local area to be compensated: EQ.0: For this option, the compensated region of the die consists of all points for which the projection onto the *xy*-plane is exterior to the projection of the BEGIN curve. The projection of the END curve is assumed exterior to the BEGIN curve. The transition region, then, consists of all die points for which the projection is between the BEGIN and END curves. All other points on the die are uncompensated. EQ.1: For this option, the compensated region of the die consists of all points for which the projection onto the *xy*-plane is interior to the projection of the BEGIN curve. The projection of the END curve is assumed exterior to the BEGIN curve. The transition region, then, consists of all die points for which the projection is between the BEGIN and END curves. All other points on the die are uncompensated. See Figure 17-18.
TYPE	Type code - must be "0".
X	*x*-coordinate of a point on the curve.
Y	*y*-coordinate of a point on the curve.
Z	*z*-coordinate of a point on the curve.

Motivation:

Sometimes springback occurs in a localized region of the die face. Since other parts of the die face are better left undisturbed, a localized compensation makes the most sense to bring the part shape back to the design intent. A typical example will be the front portion along the grill and headlamp or the rear portion along the windshield of a trimmed hood inner panel. A decklid (or trunk lid) inner also exhibits similar needs. Once the localized areas are identified, iterative compensation scheme may be employed within this localized region to bring the springback panel back to design shape.

Modeling Details:

Referring to Figure 17-18, the keywords *DEFINE_CURVE_COMPENSATION_CON-

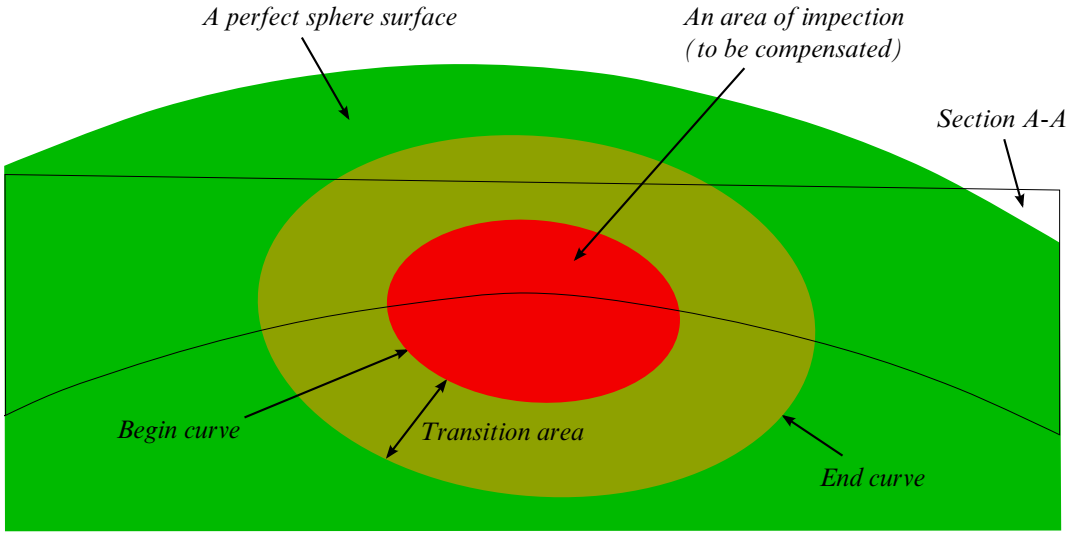


Figure 17-18. Local area compensation.

STRAINT_BEGIN and *DEFINE_CURVE_COMPENSATION_CONSTRAINT_END must be used together in a file, which in turn will be included using *INCLUDE_COMPENSATION_CURVE. The keyword BEGIN precedes the keyword END; each is defined by discrete points. In addition, each curve must form a closed loop. The area formed between the two curves is a transition area and will be affected in the compensated tooling. Multiple disconnected curves can be joined together and output in the .xyz format required here using *Curve* → *Merge* → *Multiple Method* in LS-PrePost4.0 and later.

The curve can be a 3D piecewise linear curve with coordinates in x, y and z. However, z-coordinates are ignored, meaning the tooling to be compensated must be positioned so the draw direction is in the global z-direction; otherwise, an error will occur. In addition, it is assumed that both "blank before springback" and "blank after springback" will be smaller than rigid tools in dimension. Also, the rigid tool meshes should be discretized fine enough to provide sufficient degrees of freedom for the compensation.

Example – Single Region:

A complete input deck is provided below for a local compensation simulation. The keyword files state1.k and state2.k consist of model (nodes and elements) information for the blank before and after springback, respectively. If the blank is adaptively refined, the adaptive constraints must be included in the keyword files. The keyword file tools.k consists of the stamping tools (with PID 1, 2, 3, and 4) positioned in the home position. The keyword file curvesxy.xyz includes this keyword with both variations, defining the two closed-loop curves used to define a localized area.

```
*KEYWORD
*TITLE
LS-Dyna971 Compensation Job
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*INTERFACE_COMPENSATION_NEW
$    METHOD        SL        SF      ELREF      PSIDm      UNDCT     ANGLE    NOLINEAR
```

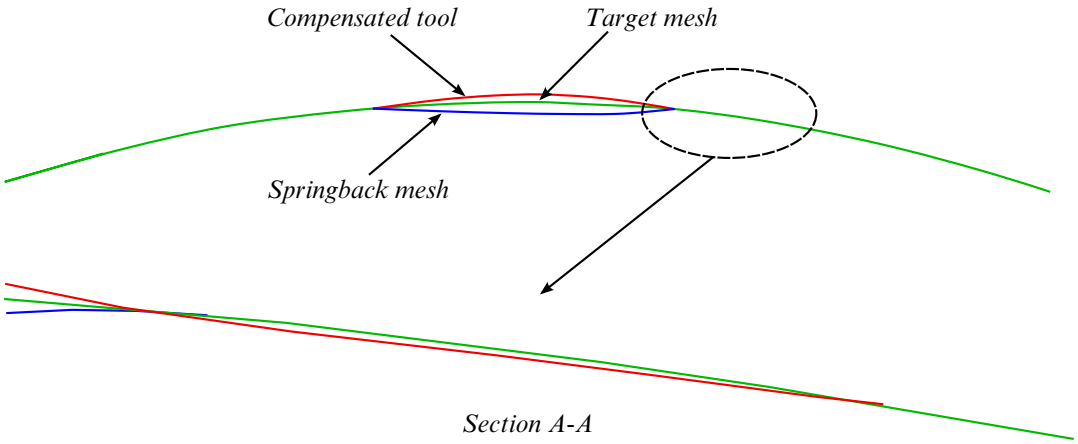


Figure 17-19. Local compensation details.

```
         6    10.000     0.700         0         1         0         0         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
state1.k
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
state2.k
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
state1.k
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
state1.k
*INCLUDE_COMPENSATION_CURRENT_TOOLS
tools.k
*INCLUDE_COMPENSATION_CURVE
curvesxy.xyz
*SET_PART_LIST
         1
1,2,3,4
*END
```

A portion of the file curvesxy.xyz is shown below,

```
*KEYWORD
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN
$       CID     IN/OUT       TYPE
          1          1          0
    -1.86925e+02     1.83338e+03    -1.55520e+01
    -1.83545e+02     1.83003e+03    -1.55469e+01
    -1.80162e+02     1.82668e+03    -1.55428e+01
    -1.91811e+02     1.83884e+03    -1.56014e+01
    -1.90187e+02     1.83701e+03    -1.55852e+01
    -1.88560e+02     1.83519e+03    -1.55688e+01
    -1.86925e+02     1.83338e+03    -1.55520e+01
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_END
$       CID     IN/OUT       TYPE
          2          1          0
    -4.07730e+02     1.61371e+03    -8.04858e+01
    -3.84480e+02     1.59890e+03    -7.99169e+01
    -3.61193e+02     1.58423e+03    -7.93471e+01
    -3.37832e+02     1.56984e+03    -7.87756e+01
    -4.49289e+02     1.67556e+03    -8.04582e+01
    -4.35672e+02     1.65473e+03    -8.05162e+01
    -4.21764e+02     1.63396e+03    -8.05530e+01
    -4.07730e+02     1.61371e+03    -8.04858e+01
*END
```

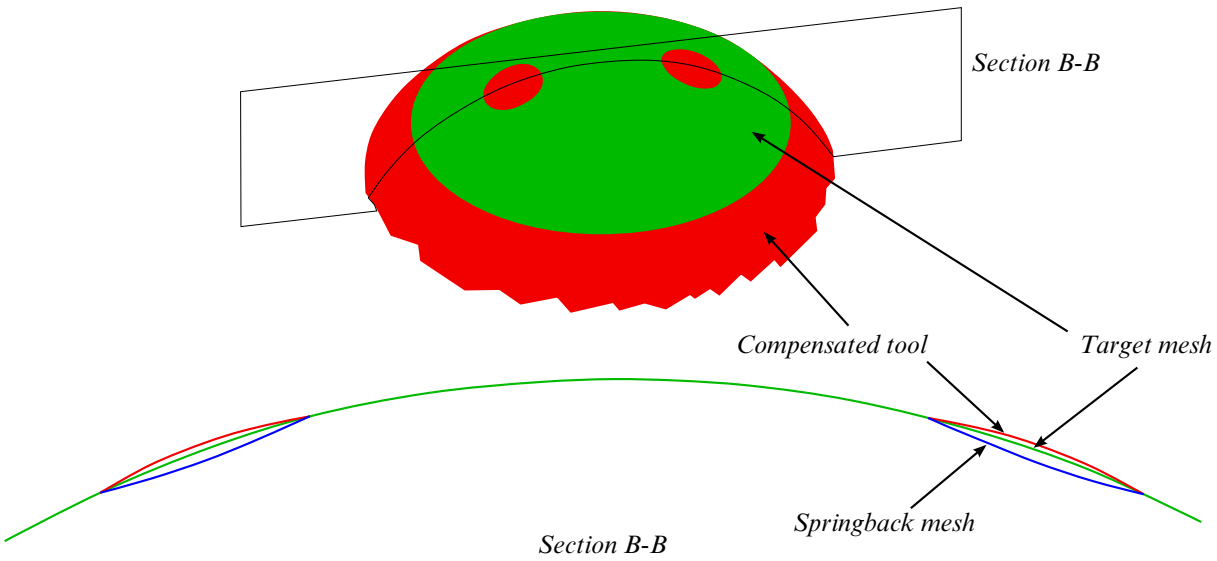


Figure 17-20. Multi-region local compensation.

It is noted the first point and last point are exactly the same, forming a closed loop. In Figure 17-18, local area compensation is to be performed in the center portion of a rigid sphere. Based on springback and target meshes, the compensated tool mesh is obtained, and smooth transition areas are achieved (see Figure 17-19). Here the compensation scale factor of 0.7 is used.

Example – Multiple Regions:

Multi-region localized compensation is also possible by defining multiple pairs of the BEGIN and END versions of this keyword, each forming a localized region. For example, for localized compensation of two regions, the file curvesxy.xyz will read as follows,

```
*KEYWORD
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN
$       CID     IN/OUT       TYPE
          1          1          0
     3.67967e+02     1.63423e+03    -6.98532e+01
     3.60669e+02     1.62992e+03    -6.92921e+01
     3.53586e+02     1.62525e+03    -6.88777e+01
           ⋮               ⋮              ⋮
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_END
$       CID     IN/OUT       TYPE
          2          1          0
     4.12534e+02     1.75537e+03    -5.83975e+01
     3.98853e+02     1.75264e+03    -5.58860e+01
     3.85292e+02     1.74921e+03    -5.35915e+01
           ⋮               ⋮              ⋮
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN
$       CID     IN/OUT       TYPE
          3          1          0
    -4.37478e+02     2.67393e+03    -1.70421e+02
    -4.45605e+02     2.67209e+03    -1.71724e+02
    -4.53649e+02     2.66985e+03    -1.72894e+02
           ⋮               ⋮              ⋮
*DEFINE_CURVE_COMPENSATION_CONSTRAINT_END
$       CID     IN/OUT       TYPE
```

```
         4         1         0
-4.49426e+02     2.79057e+03    -2.18740e+02
-4.63394e+02     2.78749e+03    -2.20955e+02
-4.77223e+02     2.78370e+03    -2.22938e+02
      ⋮               ⋮              ⋮
*END
```

Figure 17-20 (top) shows an example of two localized areas of the sphere to be compensated. The compensation results are shown in Figure 17-20 (bottom). Again, a compensation scale factor of 0.7 was used, and smooth transition areas are achieved.

*DEFINE_CURVE_DRAWBEAD

Purpose: This keyword simplifies the definition of a draw bead, which previously required the use of many keywords.

> **NOTE:** This option has been deprecated in favor of *DEFINE_MULTI_DRAWBEADS_IGES.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	TCTYPE	VID	PID	BLKID	PERCT	LCID	
Type	I	I	I	I	I	F	I	
Default	none	none	none	none	none	0.0	none	

Point Cards. For TCTYPE = 1 define points on the curve. Input is terminated at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	CX	CY	CZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

IGES Card. For TCTYPE = 2 set an IGES file.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
CID	Draw bead curve ID; must be unique.
TCTYPE	Flag to indicate input curve data format: EQ.1: XYZ data, EQ.2: IGES format data.
VID	Vector ID, as defined by *DEFINE_VECTOR. This vector is used to project the supplied curves to the rigid tool, defined by the PID below.
PID	Part ID of a rigid tool to which the curves are projected and attached.
BLKID	Part ID of the blank.
PERCT	Draw bead lock percentage or draw bead force. GT.0: Percentage of the full lock force for the bead defined. This is the ratio of desired restraining force over the full lock force. The value should be between 0.0 and 100.0. LT.0: Absolute value is the draw bead force.
LCID	Load curve ID defining material hardening curve of the sheet blank, BLKID.
CX, CY, CZ	Points on the curve.
FILENAME	IGES file name.

Remarks:

1. This feature implements the following input algorithm for drawbeads:

   a) It reads a draw bead curve in either XYZ or IGES format

   b) projects the curve to the rigid tool specified

   c) creates extra node set and attaches it to the rigid tool.

   d) With supplied material hardening curve (LCID), full lock force is calculated.

   There is no need to define *CONTACT_DRAWBEAD and *CONSTRAINED_-RIGID_BODIES since they are treated internally within the code.

2. The "curve" menu in LS-PrePost can be used to break or join multiple disconnected curves, and output in either 'XYZ' or IGES format.

3. The following partial keyword example defines a draw bead curve ID 98 (IGES file "bead1.iges") to restrain blank part ID 63. Full lock force is calculated from the strain hardening curve ID 400. The draw bead is projected along vector ID 991, and is attached to a rigid tool of part ID 3.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
-8
*KEYWORD
*DEFINE_VECTOR
991,0.0,0.0,0.0,0.0,0.0,10.0
*DEFINE_CURVE_DRAWBEAD
$       CID    TCTYPE       VID       PID     BLKID      PERCT      LCID
         98         2       991         3        63     52.442       400
bead1.iges
*MAT_037
$       MID        R0         E        PR      SIGY      ETAN         R
HCLID
          1  7.89E-09  2.00E+05       0.3     240.0                 1.6
400
*DEFINE_CURVE
400
0.0,240.0
0.02,250.0
...
1.0, 350.0
*END
```

Revision information:

This feature is available starting in LS-DYNA R5 Revision 62464.

*DEFINE_CURVE_DUPLICATE

Purpose: Define a curve by optionally scaling and offsetting the abscissa and ordinates of another curve defined by the *DEFINE_CURVE keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	RLCID	SFA	SFO	OFFA	OFFO		
Type	I	I	F	F	F	F		
Default	none	none	1.	1.	0.	0.		

VARIABLE	DESCRIPTION
LCID	Load curve ID. Tables (see *DEFINE_TABLE) and load curve ID's must be unique.
RLCID	Reference load curve ID.
SFA	Scale factor for abscissa value of curve ID, RLCID. This value scales the SFA value defined for RLCID. EQ.0.0: default set to 1.0.
SFO	Scale factor for ordinate value (function) of curve ID, RLCID. This value scales the SFO value defined for RLCID. EQ.0.0: default set to 1.0.
OFFA	Offset for abscissa values. This value is added to the OFFA value defined for RLCID.
OFFO	Offset for ordinate values (function). This value is added to the OFFO value defined for RLCID.

*DEFINE_CURVE_ENTITY

Purpose: Define a curve of straight line segments and circular arcs that defines an axisymmetric surface. This curve can only be used with the keyword *CONTACT_ENTITY for the load curve entity GEOTYP = 11.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	SFA	SFO	SFR	OFFA	OFFO	OFFR	
Type	I	F	F	F	F	F	F	
Default	none	1.	1.	1.	0.	0.	0.	

Point Cards. Put one point per card. Include as many cards as needed. This input terminates at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	A*i*		O*i*		R*i*		IFLAG	
Type	F		F		F		I	
Default	0.0		0.0		optional		↓	

VARIABLE	DESCRIPTION
LCID	Load curve ID. Tables (see *DEFINE_TABLE) and load curves may not share common IDs. LS-DYNA allows load curve IDs and table IDs to be used interchangeably. A unique number has to be defined.
SFA	Scale factor for axis value. This is useful for simple modifications. EQ.0.0: default set to 1.0.
SFO	Scale factor for radius values. This is useful for simple modifications. EQ.0.0: default set to 1.0.

VARIABLE	DESCRIPTION
SFR	Scale factor for circular radius. This is useful for simple modifications. EQ.0.0: default set to 1.0.
OFFA	Offset for axis values. See Remark 1.
OFFO	Offset for radius values. See Remark 1.
OFFR	Offset for circular radius. See Remark 1.
A*i*	Z-axis coordinates along the axis of rotation.
O*i*	Radial coordinates from the axis of rotation
R*i*	Radius of arc between points (A_i, O_i) and (A_{i+1}, O_{i+1}). If zero, a straight line segment is assumed.
IFLAG	Defined if $\|Ri\| > 0.0$. Set to 1.0 if the center of the arc is inside the axisymmetric surface and to -1.0 if the center is outside the axisymmetric surface.

Remarks:

1. **Scaling.** The load curve values are scaled after the offsets are applied, that is:

$$\text{Axis value} = \text{SFA} \times (\text{Defined value} + \text{OFFA})$$
$$\text{Radius value} = \text{SFO} \times (\text{Defined value} + \text{OFFO})$$
$$\text{Circular value} = \text{SFR} \times (\text{Defined value} + \text{OFFR})$$

*DEFINE_CURVE_FEEDBACK

Purpose: Define information that is used as the solution evolves to scale the ordinate values of the specified load curve ID. This keyword is usually used in connection with sheet metal forming calculations.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	PID	BOXID	FLDID				
Type	I	I	I	I				
Default	none	none	0	none				

Card 2	1	2	3	4	5	6	7	8
Variable	FSL	TSL	SFF	SFT	BIAS			
Type	F	F	F	F	F			
Default	none	none	1.0	1.0	0.0			

VARIABLE	DESCRIPTION
LCID	ID number for load curve to be scaled.
PID	Active part ID for load curve control
BOXID	Box ID. Elements of specified part ID contained in box are checked. EQ.0: all elements of the active part are checked.
FLDID	Load curve ID which defines the flow limit diagram as shown in Figure 17-21.
FSL	If the ratio, $r = \varepsilon_{\text{major}_{\text{workpiece}}}/\varepsilon_{\text{major}_{\text{fld}}}$, exceeds FSL, then the scale factor for flow, SFF, is active.
TSL	Thickness strain limit. If the thickness strain limit is exceeded, then the scale factor for thickening, SFT, is active.

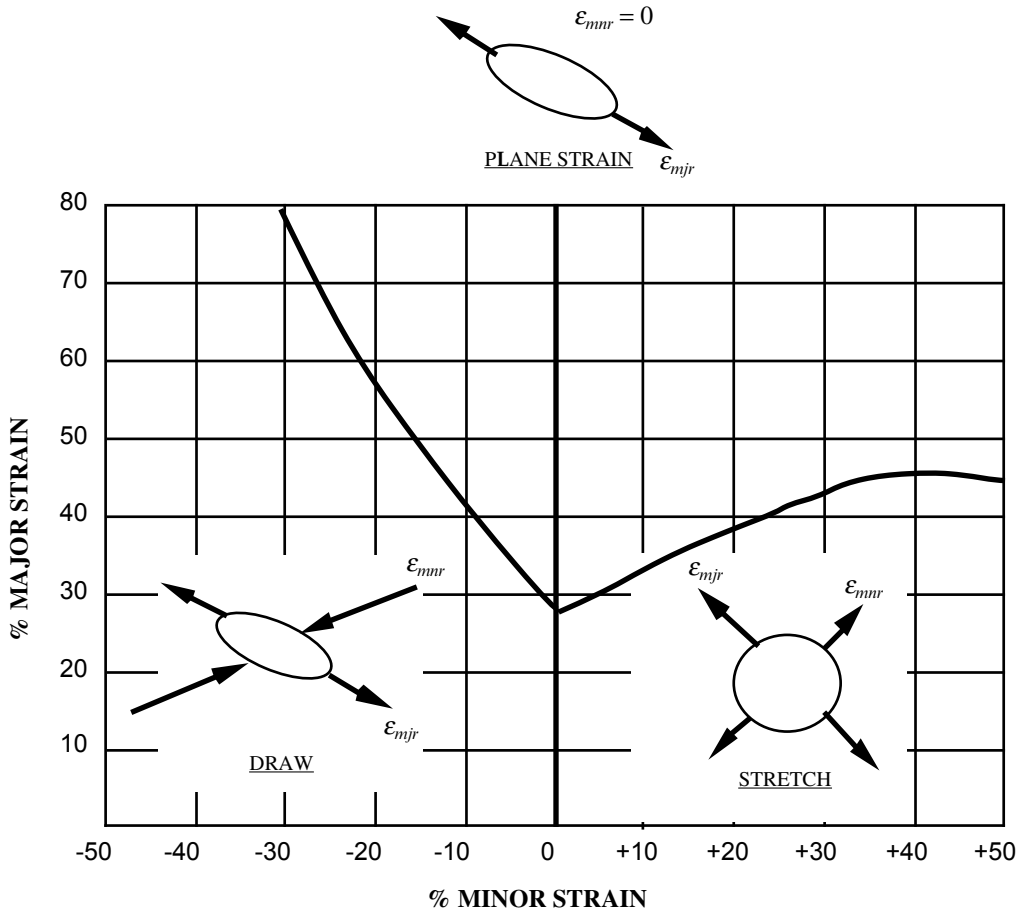


Figure 17-21. Flow limit diagram.

VARIABLE	DESCRIPTION
SFF	Scale factor for the flow limit diagram.
SFT	Scale factor for thickening.
BIAS	Bias for combined flow and thickening. Bias must be between -1 and 1.

Remarks:

This feature scales the ordinate values of a load curve according to a computed scale factor, S_f, that depends on both the major strain, r, and the through thickness, t. At each time step the load curve is scaled by S_f according to,

$$S^{n+1}_{\text{scaled load curve}} = S_f(r,t) \times S^{n}_{\text{load curve}},$$

where the superscript denotes the time step. The scale factor depends on r, which is a strain measure defined as,

$$r = \frac{\varepsilon_{\text{major}_{\text{workpiece}}}}{\varepsilon_{\text{major}_{\text{fld}}}}.$$

The scale factor, then, is given by,

$$S_f = \begin{cases} 1 & r < \text{FSL}, \quad t < \text{TSL} \\ \text{SFF} & r > \text{FSL}, \quad t < \text{TSL} \\ \text{SFT} & r < \text{FSL}, \quad t > \text{TSL} \\ \frac{1}{2}(1 - \text{BIAS}) \times \text{SFF} + \frac{1}{2}(1 + \text{BIAS}) \times \text{SFT} & r > \text{FSL}, \quad t > \text{TSL} \end{cases}$$

Usually SFF is slightly less than unity and SFT is slightly greater than unity so that $S_{\text{load curve}}$ changes insignificantly from time step to time step.

*DEFINE_CURVE_FLC

Purpose: This keyword allows for defining the Forming Limit Diagram (FLD) using sheet metal thickness t and strain hardening value n. This keyword is applicable to shell elements only.

This feature is available in LS-DYNA Revision 61435 and later releases.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	TH	TN					
Type	I	F	F					
Default	none	0.0	0.0					

VARIABLE	DESCRIPTION
LCID	Load curve ID.
TH	Sheet metal thickness.
TN	Strain hardening value of the sheet metal, as in power law (Swift).

Remarks:

1. **Related Keywords.** This keyword is used in conjunction with keyword *MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC_NLP_FAILURE and is for shell elements only. For detailed formula of calculating the FLD based on sheet metal thickness and n-value, please refer to the following paper: *Ming F. Shi, Shawn Gelisse, "Issues on the AHSS Forming Limit Determination", IDDRG 2006*.
2. **Material and Geometric Restrictions.** It is noted that this FLD calculation method is limited to sheet metal steels with thickness equal to or less than 2.5 mm, and it is not suitable for aluminum sheets. For aluminum sheets, *DEFINE_CURVE can be used to input the FLC for the field ICFLD in *MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC_NLP_FAILURE.

Example:

In a validation example shown in Figure 17-22, a single shell element is stretched in three typical strain paths (linear): uniaxial, plane strain and equi-biaxial. Strain limits for each

path are recovered when the history variable (Formability Index limit in *MAT_037) reaches 1.0, shown in Figure 17-23. The top most point (strain limit) of each strain path coincides with the FLC curve calculated according to the paper, indicating the FLC defined by this keyword is working correctly. As shown in a partial keyword file below, the FLC is defined using a thickness value of 1.5 and n-value of 0.159. The LCID of 891 is used in keyword *MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC_NLP_FAILURE.

```
*MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC_NLP_FAILURE
$      MID        RO         E        PR      SIGY      ETAN         R     HLCID
         1 7.830E-09 2.070E+05      0.28       0.0       0.0    -0.864       200
$      IDY        EA       COE     ICFLD
                                      891
*DEFINE_CURVE_FLC
$ LCID, TH, TN
891,1.5,0.159
$ DP600 NUMISHEET'05 Xmbr, Power law fitted
*DEFINE_CURVE
200
0.000,395.000
0.001,425.200
0.003,440.300
...
```

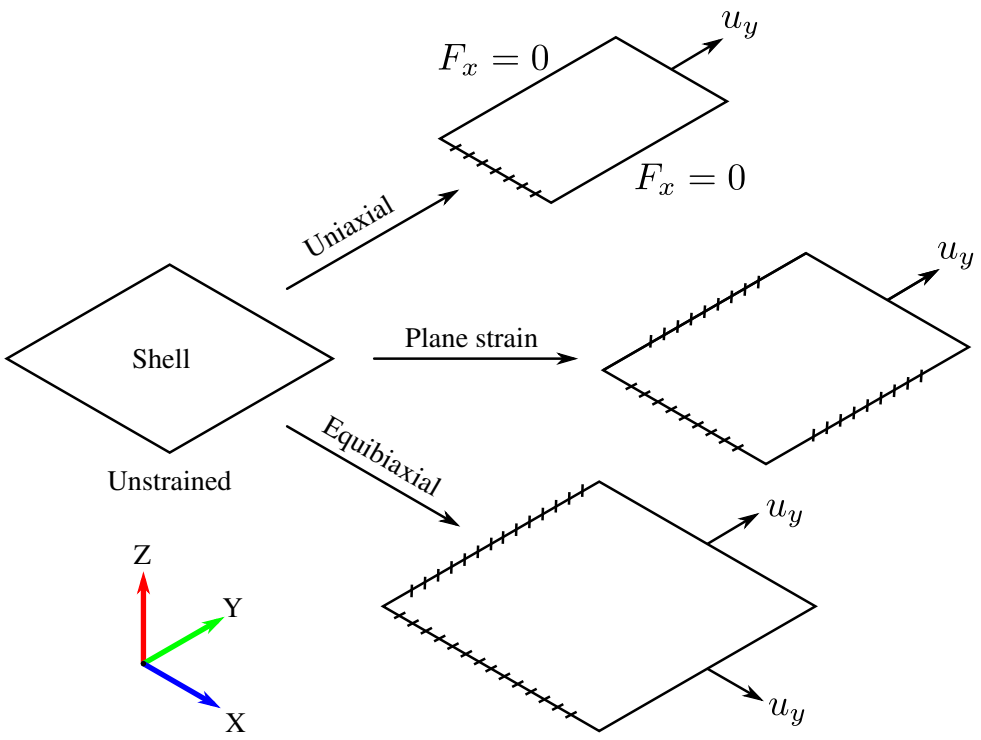


Figure 17-22. A single shell strained in three different strain paths

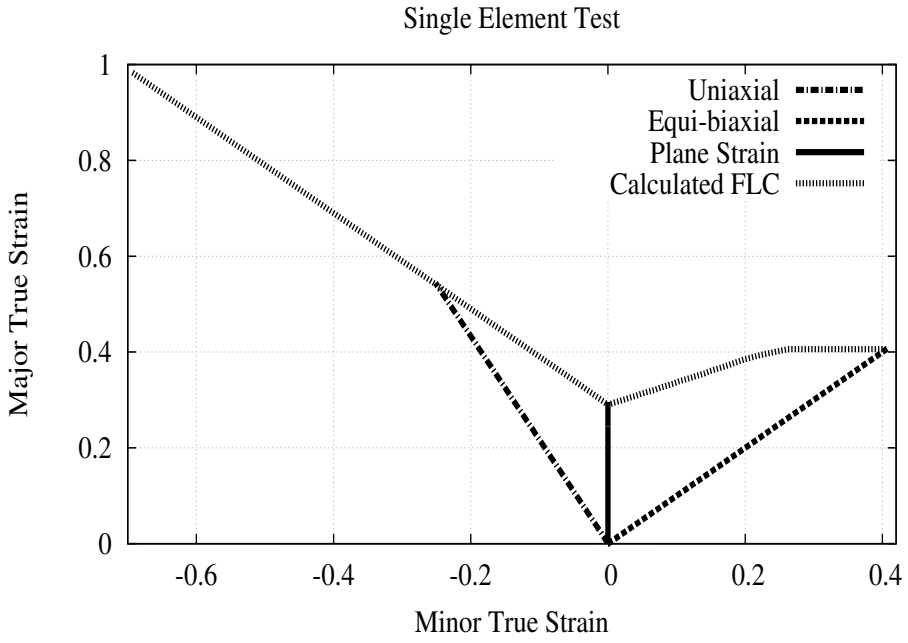


Figure 17-23. Validation of the FLC defined by this keyword

*DEFINE_CURVE_FLD_FROM_TRIAXIAL_LIMIT

Purpose: Create a forming limit diagram (FLD) curve from a given stress triaxial limit curve, which can be referred to by material models utilizing a FLD curve, such as *MAT_-037_NLP_FAILURE or *MAT_260B. The conversion assumes plane stress and Von-Mises yield criterion. The converted FLD curve can be found in the ".o" file (a scratch file from batch queue run). A related keyword is *DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_-FLD.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							
Default	none							

Point Cards. Put one pair of points per card (2E20.0). Input is terminated at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	A1		O1					
Type	E20.0		E20.0					
Default	0.0		0.0					

VARIABLE	DESCRIPTION
LCID	The FLD curve ID to be created.
A1, A2, …	Abscissas represent stress triaxialities, typically ranging from -1/3 to 2/3.
O1, O2, …	Ordinates represent equivalent plastic strains to fracture.

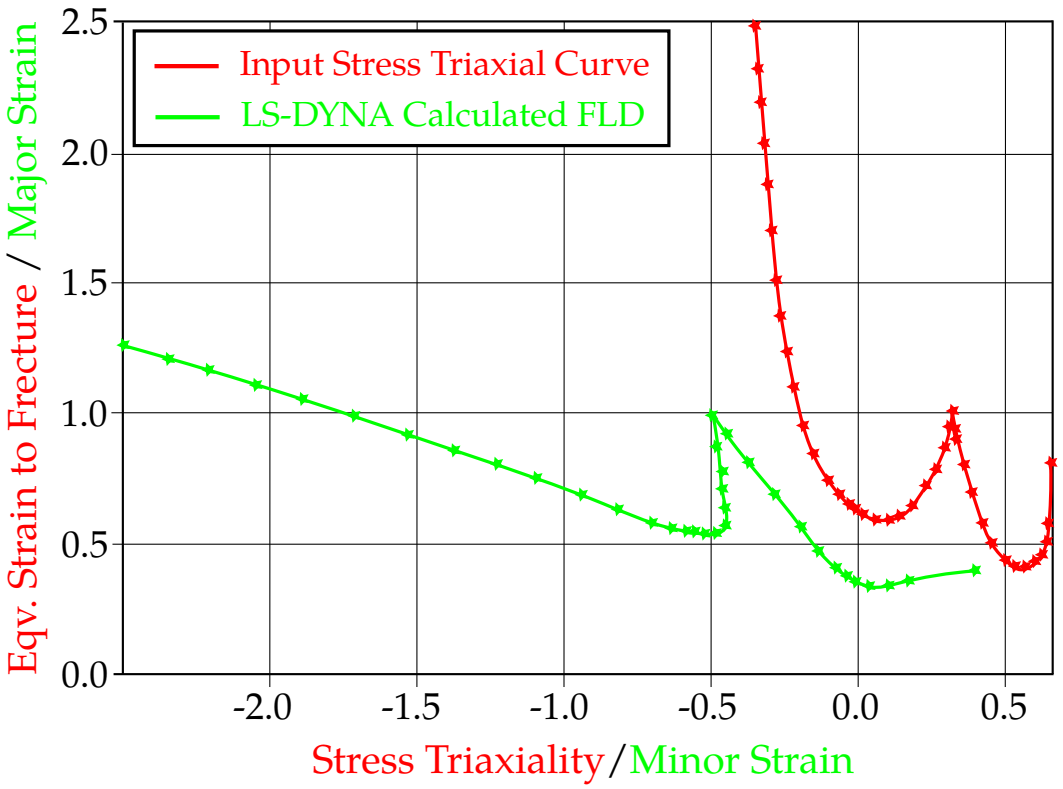


Figure 17-24. LS-DYNA calculated FLD from input stress triaxial curve [1].

Example:

The keyword input below includes the FLD limit curve obtained from Li et al. [1]. Figure 17-24 shows the calculated FLD curve from a uniaxial tension model on a single element using LS-DYNA, which matches the FLD curve from the paper.

```
*DEFINE_CURVE_FLD_FROM_TRIAXIAL_LIMIT
909
-.3284545, 2.485632
-.3193636, 2.327586
-.3102727, 2.198276
-.3011818, 2.04023
-.2875454, 1.882184
-.2739091, 1.70977
-.2602727, 1.522989
-.2466363, 1.37931
-.2239091, 1.235632
-.2011818, 1.106322
-.1648182, .9626437
-.133, .8477012
-8.754542E-02, .7471265
-4.663633E-02, .6896552
-1.481815E-02, .6465518
3.36367E-03, .632184
3.972731E-02, .6178162
8.518185E-02, .6034483
.1397273, .6034483
.1897273, .6465518
.2351819, .7183908
```

```
.2715455, .7758621
.3033637, .862069
.3306364, .9770116
.3488182, .9195403
.3715455, .8189656
.3988182, .7040231
.4351819, .5890805
.4715455, .5028736
.517, .4310345
.5533637, .4166667
.5760909, .4166667
.617, .4310345
.6397273, .4885058
.6533636, .5603449
.6670001, .8045977
*end
```

Revision Information:

This feature is available starting from Revision 116631.

References:

[1] Li, Yaning et al, "Prediction of shear-induced fracture in sheet metal forming," Journals of Material Processing Technology, Volume 210, issue 14, (2010).

*DEFINE_CURVE_FUNCTION

Purpose: Define a curve [for example, load (ordinate value) versus time (abscissa value)] where the ordinate is given by a function expression. The function can reference other curve definitions, kinematical quantities, forces, interpolating polynomials, intrinsic functions, and combinations thereof. Please note that many functions require the definition of a local coordinate system (see Remark 1 below). To output the curve to an ASCII database, see *DATABASE_CURVOUT. This command is not for defining curves for material models. Note that arguments appearing in square brackets "[ ]" are optional.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	SIDR						
Type	I	I						
Default	none	0						

Function Cards. Insert as many cards as needed. These cards are combined to form a single line of input. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	FUNCTION							
Type	A80							
Remarks	1							

VARIABLE	DESCRIPTION
LCID	Load curve ID. Tables (see *DEFINE_TABLE) and load curves may not share common ID's. LS-DYNA allows load curve ID's and table ID's to be used interchangeably. A unique number has to be defined.
SIDR	Stress initialization by dynamic relaxation: EQ.0: Load curve used in transient analysis only or for other applications, EQ.1: Load curve used in stress initialization but not transient analysis,

VARIABLE	DESCRIPTION
	EQ.2: Load curve applies to both initialization and transient analysis.
FUNCTION	Arithmetic expression involving a combination of the following possibilities.

Constants and Variables:

FUNCTION	DESCRIPTION
TIME	Current simulation time
TIMESTEP	Current simulation time step
PI	Proportionality constant relating the circumference of a circle to its diameter
DTOR	Degrees to radians conversion factor ($\pi/180$)
RTOD	Radians to degrees conversion factor ($180/\pi$)
NCYCLE	Current integration cycle
IDRFLG	Carries a value of 1 (unity) during dynamic relaxation and 0 (zero) in the transient phase

Intrinsic Functions:

FUNCTION	DESCRIPTION
ABS(a)	Absolute value of a
AINT(a)	Nearest integer whose magnitude is not larger than a
ANINT(a)	Nearest whole number to a
MOD(a_1, a_2)	Remainder when integer a_1 is divided by integer a_2
SIGN(a_1, a_2)	Transfer sign of a_2 to magnitude of a_1
MAX(a_1, a_2)	Maximum of a_1 and a_2
MIN(a_1, a_2)	Minimum of a_1 and a_2
SQRT(a)	Square root of a

FUNCTION	DESCRIPTION
EXP(a)	e raised to the power of a
LOG(a)	Natural logarithm of a
LOG10(a)	Log base 10 of a
SIN(a)	Sine of a
COS(a)	Cosine of a
TAN(a)	Tangent of a
ASIN(a)	Arc sine of a
ACOS(a)	Arc cosine of a
ATAN(a)	Arc tangent of a
ATAN2(a_1, a_2)	Arc tangent of a_1/a_2
SINH(a)	Hyperbolic sine of a
COSH(a)	Hyperbolic cosine of a
TANH(a)	Hyperbolic tangent of a

Load Curves:

FUNCTION	DESCRIPTION
LCn	Ordinate value of curve n defined elsewhere (see *DEFINE_CURVE)
DELAY(LC$n, t_{\text{delay}}, y_{\text{def}}$)	Delays curve n, defined by *DEFINE_CURVE_FUNCTION, *DEFINE_FUNCTION or DEFINE_CURVE, by a constant t_{delay} when simulation time ≥ t_{delay}, and sets the delayed curve value to a constant y_{def} when time < t_{delay}, that is, $$f_{\text{delay}}(t) = \begin{cases} f(t - t_{\text{delay}}) & t \geq t_{\text{delay}} \\ y_{\text{def}} & t < t_{\text{delay}} \end{cases}$$ For a nonlinear curve, a t_{delay} equal to more than 5,000 time steps might compromise the accuracy and must be used with caution. When t_{delay} is a negative integer, delay time is input in terms

FUNCTION	DESCRIPTION
	of time step; $\lvert t_{delay} \rvert$ is the number of delay time steps. In this case, $\lvert t_{delay} \rvert$ is limited to a maximum of 100. For example, t_{delay} = -2 delays the curve by 2 time steps.

Coordinate Functions:

FUNCTION	DESCRIPTION
CX(n)	Value of x-coordinate for node n.
CY(n)	Value of y-coordinate for node n.
CZ(n)	Value of z-coordinate for node n.

Displacement Functions:

FUNCTION	DESCRIPTION
DM(n_1[,n_2])	Magnitude of translational displacement of node n_1 relative to node n_2. Node n_2 is optional and if omitted, the displacement is computed relative to ground.
DMRB(n)	Magnitude of translational displacement of rigid body having a part ID of n
DX(n_1[,n_2,n_3])	x-translational displacement of node n_1 relative to node n_2 expressed in the local coordinate system of node n_3. In other words, at any time t, the function returns the component of relative displacement that lies in the x-direction of the local coordinate system at time t. If node n_2 is omitted, it defaults to ground. If node n_3 is not specified, the displacement is reported in the global coordinate system.
DY(n_1[,n_2,n_3])	y-translational displacement of node n_1 relative to node n_2 expressed in the local coordinate system of node n_3. In other words, at any time t, the function returns the component of relative displacement that lies in the y-direction of the local coordinate system at time t. If node n_2 is omitted it defaults to ground. If node n_3 is not specified, the displacement is reported in the global coordinate system.
DZ(n_1[,n_2,n_3])	z-translational displacement of node n_1 relative to node n_2 expressed in the local coordinate system of node n_3. In other

FUNCTION	DESCRIPTION
	words, at any time t, the function returns the component of relative displacement that lies in the z-direction of the local coordinate system at time t. If node n_2 is omitted, it defaults to ground. If node n_3 is not specified, the displacement is reported in the global coordinate system.
DXRB(n)	x-translational displacement of rigid body having a part ID of n
DYRB(n)	y-translational displacement of rigid body having a part ID of n
DZRB(n)	z-translational displacement of rigid body having a part ID of n
AX(n_1[,n_2])	Rotational displacement of node n_1 about the local x-axis of node n_2. If n_2 is not specified, then it defaults to ground. In computing this value it is assumed that the rotation about the other two axes (y-, z-axes) of node n_2 is zero. See Remark 1.
AX2(n_1[,n_2])	Rotational displacement of node n_1 about the local x-axis of node n_2. If n_2 is not specified, then it defaults to the local x -axis of node n_1. This is recommended over AX, since it has fewer limitations. See Remark 1.
AY(n_1[,n_2])	Rotational displacement of node n_1 about the local y-axis of node n_2 . If n_2 is not specified, then it defaults to ground. In computing this value it is assumed that the rotation about the other two axes (x-, z-axes) of node n_2 is zero. See Remark 1.
AY2(n_1[,n_2])	Rotational displacement of node n_1 about the local y-axis of node n_2. If n_2 is not specified, then it defaults to the local y -axis of node n_1. This is recommended over AY, since it has fewer limitations. See Remark 1.
AZ(n_1[,n_2])	Rotational displacement of node n_1 about the local z-axis of node n_2. If n_2 is not specified, then it defaults to ground. In computing this value it is assumed that the rotation about the other two axes (x-, y-axes) of node n_2 is zero. See Remark 1.
AZ2(n_1[,n_2])	Rotational displacement of node n_1 about the local z-axis of node n_2. If n_2 is not specified, then it defaults to the local z-axis of node n_1. This is recommended over AZ, since it has fewer limitations. See Remark 1.

FUNCTION	DESCRIPTION
PSI(n_1[,n_2])	First angle in the body2:313 Euler rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.
THETA(n_1[,n_2])	Second angle in the body2:313 Euler rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.
PHI(n_1[,n_2])	Third angle in the body2:313 Euler rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.
YAW(n_1[,n_2])	First angle in the body3:321 yaw-pitch-roll rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.
PITCH(n_1[,n_2])	Second angle in the body3:321 yaw-pitch-roll rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.
ROLL(n_1[,n_2])	Third angle in the body3:321 yaw-pitch-roll rotation sequence which orients node n_1 in the frame of node n_2. If n_2 is omitted, it defaults to ground. See Remark 1.

Velocity Functions:

FUNCTION	DESCRIPTION
VM(n_1[,n_2])	Magnitude of translational velocity of node n_1 relative to node n_2. Node n_2 is optional and if omitted the velocity is computed relative to ground.
VR(n_1[,n_2])	Relative radial translational velocity of node n_1 relative to node. If node n_2 is omitted, it defaults to ground.
VX(n_1[,n_2,n_3])	x-component of the difference between the translational velocity vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used.
VY(n_1[,n_2,n_3])	y-component of the difference between the translational velocity vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted it defaults to ground. Node n_3 is

FUNCTION	DESCRIPTION
	optional and if not specified the global coordinate system is used.
VZ(n_1[,n_2,n_3])	z-component of the difference between the translational velocity vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used.
WM(n_1[,n_2])	Magnitude of angular velocity of node n_1 relative to node n_2. Node n_2 is optional and if omitted the angular velocity is computed relative to ground.
WX(n_1[,n_2,n_3])	x-component of the difference between the angular velocity vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.
WY(n_1[,n_2,n_3])	y-component of the difference between the angular velocity vectors of node n_1 and node $n_2$in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.
WZ(n_1[,n_2,n_3])	z-component of the difference between the angular velocity vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node $n_2$is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.

Acceleration Functions:

FUNCTION	DESCRIPTION
ACCM(n_1[,n_2])	Magnitude of translational acceleration of node n_1 relative to node n_2. Node n_2 is optional and if omitted the acceleration is computed relative to ground. See Remark 1.
ACCX(n_1[,n_2,n_3])	x-component of the difference between the translational acceleration vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.

FUNCTION	DESCRIPTION
ACCY(n_1[,n_2,n_3])	y-component of the difference between the translational acceleration vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node $n_3$is optional and if not specified the global coordinate system is used. See Remark 1.
ACCZ(n_1[,n_2,n_3])	z-component of the difference between the translational acceleration vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.
WDTM(n_1[,n_2])	Magnitude of angular acceleration of node n_1 relative to node n_2. Node n_2 is optional and if omitted the angular acceleration is computed relative to ground. See Remark 1.
WDTX(n_1[,n_2,n_3])	x-component of the difference between the angular acceleration vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.
WDTY(n_1[,n_2,n_3])	y-component of the difference between the angular acceleration vectors of node $n_1$and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.
WDTZ(n_1[,n_2,n_3])	z-component of the difference between the angular acceleration vectors of node n_1 and node n_2 in the local coordinate system of node n_3. If node n_2 is omitted, it defaults to ground. Node n_3 is optional and if not specified the global coordinate system is used. See Remark 1.

Generic Force Functions:

FUNCTION	DESCRIPTION
FM(n_1[,n_2])	Magnitude of the SPC force acting on node n_1 minus the force acting on node n_2. Node n_2 is optional; if omitted the force acting only on n_1 is returned. See Remark 1.
FX(n_1[,n_2,n_3])	x-component of SPC force acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the force acting

FUNCTION	DESCRIPTION
	on n_2 is subtracted from the force acting on n_1. See Remark 1.
FY(n_1[,n_2,n_3])	y-component of SPC force acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the force acting on n_2 is subtracted from the force acting on n_1. See Remark 1.
FZ(n_1[,n_2,n_3])	z-component of SPC force acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the force acting on n_2 is subtracted from the force acting on n_1. See Remark 1.
TM(n_1[,n_2])	Magnitude of SPC torque acting on node n_1 minus the torque acting node n_2. Node n_2 is optional; if omitted the torque acting only on n_1 is returned. See Remark 1.
TX(n_1[,n_2,n_3])	x-component of the SPC torque acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the torque acting on n_2 is subtracted from the torque acting on n_1. See Remark 1.
TY(n_1[,n_2,n_3])	y-component of the SPC torque acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the torque acting on n_2 is subtracted from the torque acting on n_1. See Remark 1.
TZ(n_1[,n_2,n_3])	z-component of the SPC torque acting on node n_1 as computed in the optional local system of node n_3. If n_2 is specified, the torque acting on n_2 is subtracted from the torque acting on n_1. See Remark 1.

Sensor Functions:

FUNCTION	DESCRIPTION
SENSOR(n)	Returns a value of 1.0 if *SENSOR_CONTROL of control ID n has a status of "on". If the sensor has a status of "off", then the returned value is equal to the value of the TYPEID field on *SENSOR_CONTROL when the TYPE field is set to "function", otherwise SENSOR(n) returns zero.
SENSORD(n,*idflt*)	Returns the current value of *SENSOR_DEFINE sensor having ID n. Idflt is the optional filter ID defined using *DEFINE_FILTER.

Contact Force Functions:

FUNCTION	DESCRIPTION
RCFORC(*id*, *iba*, *comp*, *local*)	Returns the component *comp* (see description below) of contact interface *id* (see *CONTACT_..._ID) as calculated in the local coordinate system *local* (see *DEFINE_COORDINATE_...). If *local* equals zero, then forces are reported in the global coordinate system. Forces are reported for the SURFA side when *iba* = 1 or SURFB side when *iba* = 2. The admissible values of *comp* and their corresponding force component are as follows: comp.EQ.1: *x* force component comp.EQ.2: *y* force component comp.EQ.3: *z* force component comp.EQ.4: Resultant force

Element Specific Functions:

FUNCTION	DESCRIPTION
BEAM(*id*, *jflag*, *comp*, *rm*)	Returns the force component *comp* (see description below) of beam *id* as calculated in the local coordinate system *rm*. Forces are reported in the global coordinate system if *rm* is zero. If *rm* equals -1 the beam's *r*, *s*, and *t* force/moment is returned. If *jflag* is set to zero, then the force/torque acting on n_1 end of the beam is returned; otherwise if *jflag* is set to 1, the force/torque on the n_2 end of the beam is returned. See *ELEMENT_BEAM for the nodal connectivity rule defining n_1 and n_2. Admissible values of *comp* are 1-8 and correspond to the following components: comp.EQ.1: Force magnitude comp.EQ.2: *x* force (axial *r*-force, rm = -1) comp.EQ.3: *y* force (*s*-shear force, rm = -1) comp.EQ.4: *z* force (*t*-shear force, rm = -1) comp.EQ.5: Torque magnitude

FUNCTION	DESCRIPTION
	comp.EQ.6: *x* torque (torsion, rm = -1) comp.EQ.7: *y* torque (*s*-moment, rm = -1) comp.EQ.8: *z* torque (*t*-moment, rm = -1)
ELHIST(*eid*, *etype*, *comp*, *ipt*, *local*)	Returns the elemental quantity *comp* (see description below) of element e*id* as calculated in the local coordinate system *local*. Quantities are reported in the global coordinate system if *local* is zero. The parameter *ipt* specifies whether the quantity is for a particular integration point or maximum, minimum, or averaging is applied across the integration points. The following element classes, specified with *etype*, are supported: etype.EQ.0: Solid etype.EQ.2: Thin shell Following are admissible values of *comp* and the corresponding elemental quantity: comp.EQ.1: *x* stress comp.EQ.2: *y* stress comp.EQ.3: *z* stress comp.EQ.4: *xy* stress comp.EQ.5: *yz* stress comp.EQ.6: *zx* stress comp.EQ.7: Effective plastic strain comp.EQ.8: Hydrostatic pressure comp.EQ.9: Effective stress comp.EQ.11: *x* strain comp.EQ.12: *y* strain comp.EQ.13: *z* strain comp.EQ.14: *xy* strain comp.EQ.15: *yz* strain comp.EQ.16: *zx* strain Integration point options, specified with *ipt*, are:

FUNCTION	DESCRIPTION
	ipt.GE.1: Quantity is reported for integration point number ipt ipt.EQ.-1: Maximum of all integration points (default) ipt.EQ.-2: Average of all integration points ipt.EQ.-3: Minimum of all integration points ipt.EQ.-4: Lower surface integration point ipt.EQ.-5: Upper surface integration point ipt.EQ.-6: Middle surface integration point The local coordinate option *local* currently defaults to the global coordinate system for solid elements and other coordinate system options are unavailable. In the case of thin shell elements the quantity is reported only in the element local coordinate system. local.EQ.1: Global coordinate system (solid elements) local.EQ.2: Element coordinate system (thin shell elements)
JOINT(*id*, *jflag*, *comp*, *rm*)	Returns the force component *comp* (see description below) due to rigid body joint *id* as calculated in the local coordinate system *rm*. If *jflag* is set to zero, then the force/torque acting on n_1 end of the joint is returned. The force/torque on the n_2 end of the joint is returned if jflag is set to 1. See *CONSTRAINED_JOINT for the rule defining n_1 and n_2. Admissible values of *comp* are 1-8 and correspond to the following components: comp.EQ.1: Force magnitude comp.EQ.2: x force comp.EQ.3: y force comp.EQ.4: z force comp.EQ.5: Torque magnitude comp.EQ.6: x torque

FUNCTION	DESCRIPTION
	comp.EQ.7: y torque
	comp.EQ.8: z torque

Nodal Specific Functions:

FUNCTION	DESCRIPTION
TEMP(n)	Returns the temperature of node n
DIST(n)	Returns distance traveled by node n
EPOT(n)	Returns the electric potential of node n of a piezoelectric material
ECHG(n)	Returns the reactive electric charge of node n of a piezoelectric material, conjugate to prescribed nodal electric potential.
ECHGBC(n)	Returns the summation of the reactive electric charges of all nodes associated with a prescribed electric potential boundary condition (see *BOUNDARY_PZEPOT) with an ID = n.

General Functions:

FUNCTION	DESCRIPTION
CHEBY($x, x_0, a_0, \ldots, a_{30}$)	Evaluates a Chebyshev polynomial at the user specified value x. The parameters $x_0, a_0, a_1, \ldots, a_{30}$ are used to define the constants for the polynomial defined by: $$C(x) = \sum a_j T_j(x - x_0)$$ where the functions T_j is defined recursively as $$T_j(x - x_0) = 2(x - x_0)T_{j-1}(x - x_0) - T_{j-2}(x - x_0)$$ where $$T_0(x - x_0) = 1$$ $$T_1(x - x_0) = x - x_0$$
FORCOS($x, x_0, \omega[, a_0, \ldots, a_{30}]$)	Evaluates a Fourier cosine series at the user specified value x. The parameters $x_0, a_0, a_1, \ldots, a_{30}$ are used to define the constants for the series defined by: $$F(x) = \sum a_j T_j(x - x_0)$$ where

FUNCTION	DESCRIPTION
	$T_j(x - x_0) = \cos[j\omega(x - x_0)]$
FORSIN$(x, x_0, \omega[, a_0, \ldots, a_{30}])$	Evaluates a Fourier sine series at the user specified value x. The parameters $x_0, a_0, a_1, \ldots, a_{30}$ are used to define the constants for the series defined by: $F(x) = \sum a_j T_j(x - x_0)$ where $T_j(x - x_0) = \sin[j\omega(x - x_0)]$
IF(a_1, a_2, a_3, a_4)	Arithmetic if conditional where a_i can be a constant or any legal expression described in *DEFINE_CURVE_-FUNCTION. For example, a_1='CX(100)' sets the first argument to be the x-coordinate of node 100. $\text{IF} = \begin{cases} \text{the value of } a_2 & \text{if the value of } a_1 < 0 \\ \text{the value of } a_3 & \text{if the value of } a_1 = 0 \\ \text{the value of } a_4 & \text{if the value of } a_1 > 0 \end{cases}$
MORLET(*freq, mag)*	Evaluates Morlet wavelet function with central frequency of *freq* and peak value of *mag*: $M(t) = \text{mag} \times e^{-0.5 \times g^2(t)} \times \cos(5 \times f(t) - 20)$ where $f(t) = 1.25 \times \text{freq} \times t$ and $g(t) = f(t) - 4.$
PIDCTL(*meas, ref, kp, ki, kd, tf, ei0, sint, umin, umax*)	Evaluates the control signal of a PID controller $u(t) = \text{kp} \times e(t) + \text{ki} \times \int_0^t e(\tau)d\tau + \text{kd} \times \frac{\text{d}e(t)}{\text{d}t}$ where $e(t)$ is the control error defined as the difference between the reference value *ref* and the measured value, *meas* $e(t) = \text{ref} - \text{meas}$ The control parameters are proportional gain *kp*, integral gain *ki*, derivative gain *kd* and low-pass filter *tf* for the derivative calculation $\frac{\text{d}e(t_n)}{\text{d}t} = \frac{\text{d}e(t_{n-1})}{\text{d}t}\frac{tf}{\Delta t + tf} + \frac{\Delta t}{\Delta t + tf} \times \frac{e(t_n) - e(t_{n-1})}{\Delta t}$

FUNCTION	DESCRIPTION
	ei0 is the initial integral value at time = 0. *sint* is the sampling interval. *umin* and *umax*, the lower and upper limits of a control signal, can be used to represent the saturation limits of an actuator. When the signal is not within the limits, it is clipped to the saturation limit; that is, integration is skipped to avoid integrator wind-up. Input parameter can be a constant or any legal expression described in *DEFINE_CURVE_FUNCTION. For example, *meas = 'CX(100)'* measures the *x*-coordinate of node 100, *ref* = 'LC200' uses curve 200 as the reference value.
POLY$(x, x_0, a_0, \dots, a_{30})$	Evaluates a standard polynomial at the user-specified value x. The parameters x_0, a_0, a_1, …, a_{30} are used to define the constants for the polynomial defined by: $$P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \cdots + a_n(x - x_0)^n$$
RICKER(*freq*, *mag*)	Evaluates Ricker wavelet function with central frequency of *freq* and peak value of *mag*: $$R(t) = \text{mag} \times (1 - 2 \times g(t))/e^{g(t)} \ ,$$ where $$f(t) = \text{PI} \times \text{freq} \times (t - 1/\text{freq})$$ and $$g(t) = f^2(t)$$
SHF$(x, x_0, a, \omega[, \phi, b])$	Evaluates a Fourier sine series at the user specified value x. The parameters x_0, a_0, a_1, …, a_{30} are used to define the constants for the series defined by: $$\text{SHF} = a\sin[\omega(x - x_0) - \phi] + b$$
STEP(x, x_0, h_0, x_1, h_1)	Approximates the Heaviside function with a cubic polynomial using the equation: $$\text{STEP} = \begin{cases} h_0 & \text{if } x \le x_0 \\ h_0 + (h_1 - h_0)\left[\dfrac{(x - x_0)}{(x_1 - x_0)}\right]^2 \left\{3 - 2\left[\dfrac{(x - x_0)}{(x_1 - x_0)}\right]\right\} & \text{if } x_0 < x < x_1 \\ h_1 & \text{if } x \ge x_1 \end{cases}$$

Electromagnetic Solver (EM) Functions:

FUNCTION	DESCRIPTION
EM_ELHIST(*iele*, *ifield*, *idir*)	Returns the elemental quantity of element *iele* in the global reference frame.
EM_NDHIST(*inode*, *ifield*, *idir*)	Returns the nodal quantity of node *inode* in the global reference frame.
EM_PAHIST(*ipart*, *ifield*, *idir*)	Returns the value integrated over the whole part given by *ipart*. *ifield* can be 7, 8 and 11 only. Admissible values of *ifield* are 1 - 10 and correspond to the following variables: EQ.1: Scalar potential EQ.2: Vector potential EQ.3: Electric field EQ.4: **B** field EQ.5: **H** field EQ.6: Current density EQ.7: Lorentz force EQ.8: Joule heating EQ.9: Conductivity EQ.10: Relative permeability EQ.11: Magnetic energy (in the conductor only) Admissible values of *idir* are 1-4 and correspond to the following components: EQ.1: *x*-component EQ.2: *y*-component EQ.3: *z*-component EQ.4: Norm
EM_ISOHIST(*iiso*, *ifield*, *idir*)	Returns the global values that are calculated for each isopotential. Variable *iiso* is the isopotential ID. Admissible values of *ifield* are 1 - 3 and correspond to the following variables: EQ.1: Scalar potential EQ.2: Current

FUNCTION	DESCRIPTION
	EQ.3: Resistance Admissible value of *idir* is 1.
EM_ISOCHIST(*iisoch*, *ifield*, *idir*)	Returns the global values that are calculated for each isopotential connection. The variable *iisoch* is the isopotential connection ID. Admissible values of *ifield* are 1 - 6 and correspond to the following variables: EQ.1: Scalar potential EQ.2: Current EQ.3: Resistance EQ.4: Charge EQ.5: Power EQ.6: Energy Admissible value of *idir* is 1.
EM_BPHIST(*ibp*, *ifield*, *idir*)	Returns the global values that are calculated for each prescribed boundary connection. The variable *ibp* is the prescribed boundary ID (see *EM_-BOUNDARY_PRESCRIBED). Admissible values of *ifield* are 1 - 3 and correspond to the following variables: EQ.1: Scalar potential EQ.2: Current EQ.3: Resistance The only admissable value of *idir* is 1.

ICFD Solver (ICFD) Functions:

FUNCTION	DESCRIPTION
ICFD_NDHIST(*inode*, *ifield*, *idir*)	Returns the nodal quantity of node *inode* in the global reference frame. Admissible values of *ifield* are 1-8 and correspond to the following variables: EQ.1: Node position EQ.2: Velocity field

FUNCTION	DESCRIPTION
	EQ.3: Time-averaged velocity EQ.4: Density EQ.5: Dynamic viscosity EQ.6: Pressure EQ.7: Turbulent dynamic viscosity EQ.8: Level set Admissible values of *idir* are 1-4 and correspond to the following components: EQ.1: *x*-component EQ.2: *y*-component EQ.3: *z*-component EQ.4: Norm
ICFD_PAHIST(*ipart*, *ifield*, *idir*)	Returns the value integrated over the whole ICFD surface part given by *ipart*. Admissible values of *ifield* are 1-11 and correspond to the following variables: EQ.1: Pressure force EQ.2: Shear force EQ.3: Total force EQ.4: Moment due to pressure EQ.5: Moment due to shear EQ.6: Total moment EQ.7: Flow rate EQ.8: Time-averaged flow rate EQ.9: Pressure EQ.10: Time averaged Pressure EQ.11: Area Admissible values of *idir* are 1-4 and correspond to the following components: EQ.1: *x*-component EQ.2: *y*-component

FUNCTION	DESCRIPTION
	EQ.3: z-component EQ.4: Norm
ICFD_TPD(*itpd*, *ifield*)	Returns the quantity given by an ICFD TPD ID *itpd* defined in *ICFD_DATABASE_TPD. The only admissible value of *ifield* is 1.

CPG Solver Functions:

FUNCTION	DESCRIPTION
CPG_PAHIST(*ipart*, *ifield*, *ibag*)	Returns the value integrated over the whole surface part given by *ipart* for a given CPG airbag ID *ibag*. Admissible values of *ifield* correspond to the following variables: EQ.2: Static pressure EQ.6: Surface area EQ.7: Static temperature EQ.11: Mass that leaked or entered EQ.12: Volume flow rate EQ.13: Mass flow rate EQ.14: Heat convection EQ.15: Heat convection rate EQ.18: Unblocked area EQ.19: Total temperature

Remarks:

1. **Local coordinate systems required for rotational motion.** A local coordinate system *must* be attached to nodes if they are referenced by functions involving rotational motion, for example, angular displacement or angular velocity. The local coordinate system is attached to the node using *DEFINE_COORDINATE_NODES, and FLAG = 1 is a requirement. Furthermore, the three nodes that comprise the coordinate system must lie on the same body. Similarly, a local coordinate system must also be attached to node n_3 if n_3 is referenced in functions: DX, DY, DZ, VX, VY, VZ, WX, WY, WZ, ACCX, ACCY, ACCZ, WDTX, WDTY, WDTZ, FX, FY, FZ, TX, TY, or TZ.

2. **Default is radians.** Unless otherwise noted, units of radians are always used for the arguments and output of functions involving angular measures.

3. **Reserved variable names.** See Appendix U for a list of variable names reserved internally by LS-DYNA.

The following examples serve only as an illustration of syntax.

Example 1:

Define a curve 10 whose ordinate is,

$$f(x) = \frac{1}{2}(\text{ordinate of load curve 9}) \times (\text{magnitude of translation velocity at node 22})^3.$$

```
*DEFINE_CURVE_FUNCTION
10
0.5*lc9*vm(22)**3
```

Example 2:

Define a curve 101 whose ordinate is,

$$f(x) = -2(\text{z translational displacement of node 38}) \times \sin(20\pi t).$$

```
*DEFINE_CURVE_FUNCTION
101
-2.*dz(38)*sin(2.*pi*10.*time)
```

Example 3:

Define a curve 202 whose ordinate is,

$$f(x) = \begin{cases} \cos(4\pi t) \; if \; t \leq 5. \\ 0. \; if \; t > 5. \end{cases}$$

```
*DEFINE_CURVE_FUNCTION
202
If(TIME-5.,COS(4.*PI*TIME),COS(4.*PI*TIME),0.)
```

*DEFINE_CURVE_SMOOTH

Purpose: Define a smoothly varying curve using few parameters. This shape is useful for velocity control of tools in metal forming applications.

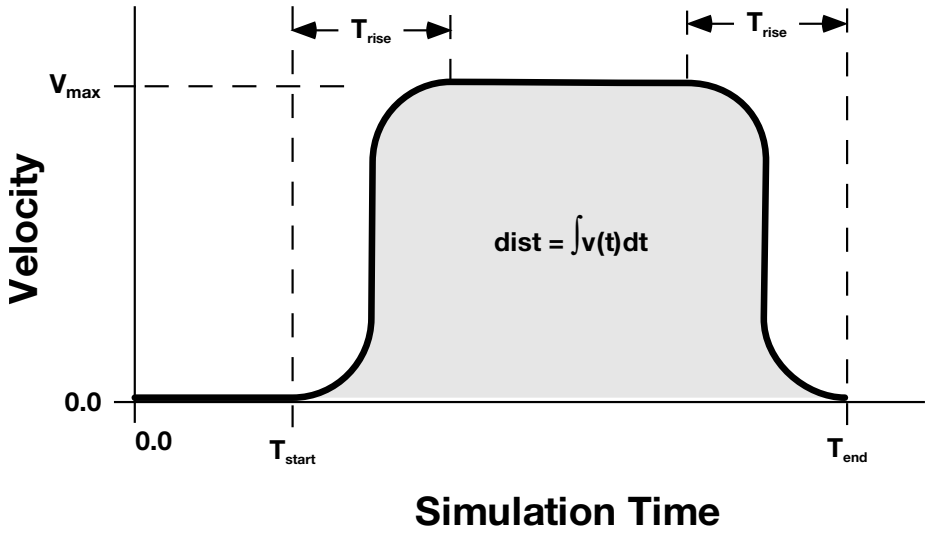


Figure 17-25. Smooth curve created automatically using *DEFINE_CURVE_-SMOOTH. This shape is commonly used to control velocity of tools in metal forming applications as shown in this figure but can also be used for other applications in place of any standard load curve.

Card	1	2	3	4	5	6	7	8
Variable	LCID	SIDR	DIST	TSTART	TEND	TRISE	VMAX	
Type	I	I	F	F	F	F	F	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
LCID	Load curve ID, must be unique
SIDR	Stress initialization by dynamic relaxation: EQ.0: Load curve is used in transient analysis only or for other applications, EQ.1: Load curve is used in stress initialization but not transient analysis.

VARIABLE	DESCRIPTION
	EQ.2: Load curve applies to both initialization and transient analysis.
DIST	Total distance tool will travel (area under curve).
TSTART	Time curve starts to rise
TEND	Time curve returns to zero. If TEND is nonzero, VMAX will be computed automatically to satisfy required travel distance DIST. Input either TEND or VMAX.
TRISE	Rise time
VMAX	Maximum velocity (maximum value of curve). If VMAX is non-zero, TEND will be computed automatically to satisfy required travel distance DIST. Input either TEND or VMAX.

*DEFINE_CURVE_STRESS

Purpose: This keyword defines a material hardening curve based on a few commonly used material hardening laws. The hardening curve can also be a weighted combination of some of the laws. The load curve ID for this curve can be referenced in the load curve ID field used by a specific material model. This feature is applicable to all material models with a hardening curve that can be defined by a load curve using *DEFINE_CURVE.

Curve Definition Card(s). Only one card is needed unless ITYPE = 11. When ITYPE = 11, a second card with the form of CARD 1 is needed with the same LCID as the first card. See Remark 2.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	ITYPE	P1	P2	P3	P4	P5	P6
Type	I	I	F	F	F	F	F	F
Default	none	none	none	1	none	none	none	none

VARIABLE	DESCRIPTION
LCID	Load curve ID for the stress-strain curve to be generated.
ITYPE	Type of hardening law:

EQ.1: Swift power law:

$$\sigma = K\left(e_0 + \epsilon_p\right)^n ,$$

where σ is true effective stress, e_0 is the residual plastic strain at the initial yield point, K is a strength coefficient, ϵ_p is true effective plastic strain, and n is the work hardening coefficient.

EQ.2: Voce law with form:

$$\sigma = \sigma_0 + R_{\text{sat}}\left(1.0 - e^{-\zeta\epsilon_p}\right) ,$$

where σ_0 is the initial yield stress, R_{sat} is the stress differential between σ_0 and the saturated stress, and ζ is a strain coefficient.

EQ.3: Voce law with form:

$$\sigma = A - Be^{-C\epsilon_p} ,$$

where A, B, and C are material constants.

VARIABLE **DESCRIPTION**

EQ.4: Hockett-Sherby law:

$$\sigma = A - Be^{-C\epsilon_p^H},$$

where A, B, C, and H are material constants.

EQ.5: Stoughton-Yoon hardening law:

$$\sigma = A - Be^{-C\epsilon_p^m} + D\epsilon_p,$$

where A, B, C, m and D are material constants, such that $0 < m < 1.0$ and $D \geq 0.0$. According to Stoughton-Yoon, "with the exception of metals exhibiting Yield Point Elongation (YPE) effects, this function can represent the stress-strain response for both mild and AHSS steel and aluminum, from the initial yield point, throughout the small strain range, up to the highest strains realized in bulge tests". Note that if $D = 0.0$, this law reduces to the Hockett-Sherby law (ITYPE = 4). Also note that if $m = 1.0$ and $D = 0.0$, this law reduces to the ITYPE = 3 Voce law.

EQ.11: A weighted combination of ITYPE = 1 and any of the other ITYPEs. See Remark 2.

P1, P2, P3
P4, P5, P6

ITYPE	P1	P2	P3	P4	P5	P6
1	K	n	e_0			
2	σ_0	R_{sat}	ζ	w_2		
3	A	B	C	w_2		
4	A	B	C	H	w_2	
5	A	B	C	M	D	w_2
11	K	n	e_0	w_1		

Where w_1 and w_2 are the weighting factors used for ITYPE = 11; see Remark 2.

Remarks:

1. **Default for P2.** For ITYPE = 1, the default value (also the minimum) for P2 is 0.000001.

2. **Combining Hardening Laws.** With ITYPE = 11, the Swift power law (ITYPE = 1) can be combined with another law through a weighted sum. To do this two cards are needed with the same LCID. The first card has ITYPE = 11 in place of ITYPE = 1. This card also requires a weight, w_1, in the P4 field which is the weight applied to the Swift power law in the sum. The second card has the ITYPE of the law that is being summed to the Swift power law. This card also requires a weight, w_2, in the P*i* field specified for that ITYPE. The following example generates a hardening curve (LCID 90903) that is the sum of 0.5 (w_1) times the Swift power law and 0.8 (w_2) times the Hockett-Sherby law (ITYPE = 4). This curve then has the equation

$$\sigma = 0.5\left(350.0\left(0.01 + \epsilon_p\right)^{0.22}\right) + 0.8\left(162.2 - 72.2e^{-4.34\epsilon_p^{1.2}}\right).$$

```
*MAT_037
$      MID        RO         E        PR      SIGY      ETAN         R     HLCID
         1 7.900E-09 2.070E+05      0.30                        -1.45     90903
*DEFINE_CURVE_STRESS
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$ ITYPE = 1: power law.  P1 = K, P2 = n, P3 = e0, P4 = 0.5.
$ ITYPE = 4: Hockett-Sherby law.  P1 = A, P2 = B, P3 = C, P4 = H, P5 = 0.8.
$                       VOCE: sigma = A-B*exp(-C*eps**H)
$     LCID      TYPE        P1        P2        P3        P4        P5
     90903        11     350.0      0.22      0.01       0.5
     90903         4     162.2      72.2      4.34       1.2       0.8
```

Revision Information:

This feature is available starting from Revision 113640. ITYPE = 5 (Stoughton-Yoon) is available starting from Revision 114803.

*DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_FLD

Purpose: Create a stress triaxial limit curve from a given forming limit diagram (FLD) curve, which can be referred to by material models utilizing a stress triaxial limit curve, such as *MAT_ADD_EROSION or *MAT_260B, or by keywords, such as *CONTROL_-FORMING_ONESTEP. The conversion assumes plane stress and Von-Mises yield criterion. The converted stress triaxial curve can be found in the ".o" file (a scratch file from batch queue run). A related keyword is *DEFINE_CURVE_FLD_FROM_TRIAXIAL_-LIMIT.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							
Default	none							

Point Cards. Put one pair of points per card (2E20.0). Input is terminated at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	A1		O1					
Type	E20.0		E20.0					
Default	0.0		0.0					

VARIABLE	DESCRIPTION
LCID	The stress triaxial limit curve ID to be created.
A1, A2, …	Abscissas represent minor true strains of a FLD curve.
O1, O2, …	Ordinates represent major true strains of a FLD curve.

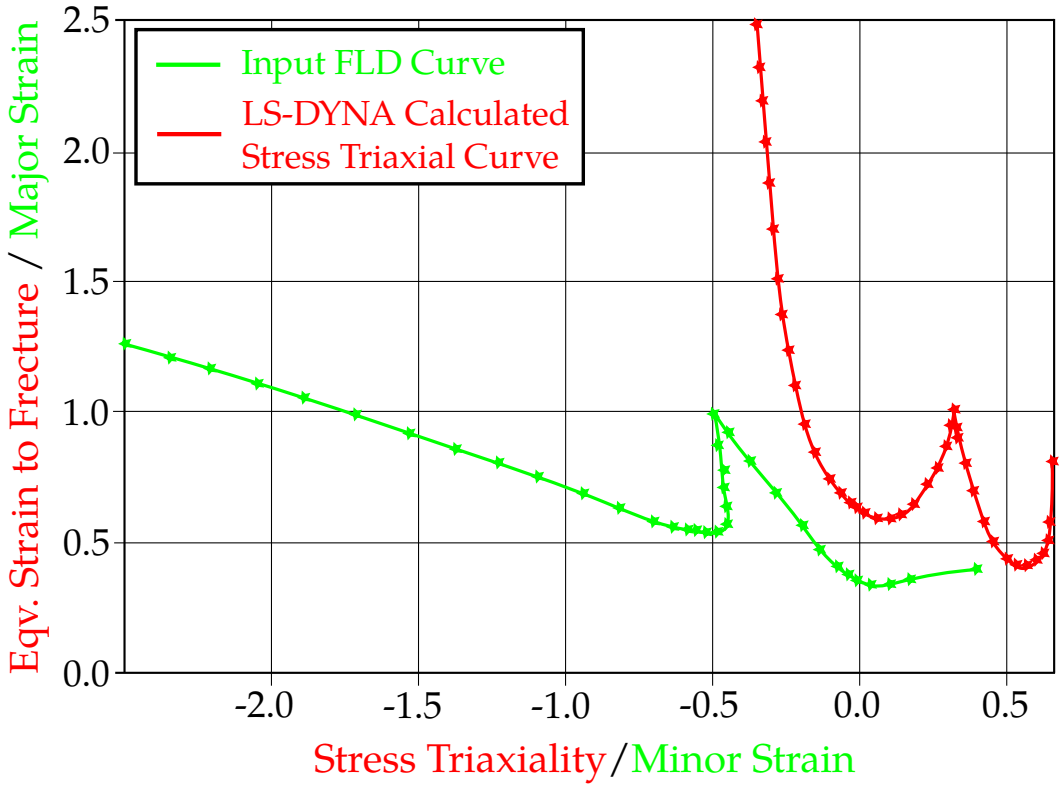


Figure 17-26. LS-DYNA calculated stress triaxial curve from FLD curve [1].

Example:

The keyword input below includes the FLD limit curve obtained from Li et al. [1]. Using a uniaxial tension model on a single element, Figure 17-26 shows the calculated stress triaxial curve from LS-DYNA, which matches the triaxial curve from the paper.

```
*DEFINE_CURVE_TRIAXIAL_LIMIT_FROM_FLD
909
-2.485543, 1.260929
-2.326915, 1.211872
-2.196562, 1.173440
-2.037161, 1.115458
-1.876514, 1.064689
-1.701195, 0.9987057
-1.511570, 0.9169926
-1.364909, 0.8546170
-1.215432, 0.8004163
-1.080362, 0.7465187
-0.9267878, 0.6888055
-0.8039308, 0.6348159
-0.6904781, 0.5923799
-0.6199200, 0.5716710
-0.5669760, 0.5526081
-0.5458882, 0.5490728
-0.5156967, 0.5524927
-0.4797515, 0.5568793
-0.4477362, 0.5742416
-0.4447934, 0.6287722
-0.4554417, 0.7088588
```

```
-0.4556653, 0.7716601
-0.4686948, 0.8609383
-0.4924086, 0.9770012
-0.4380930, 0.9192049
-0.3606666, 0.8170972
-0.2779830, 0.6991526
-0.1942268, 0.5787448
-0.1300228, 0.4857036
-6.8558961E-02, 0.4028141
-2.8324760E-02, 0.3741716
-1.5549607E-03, 0.3616189
5.8079619E-02, 0.3408428
0.1153265, 0.3534369
0.1781329, 0.3710328
0.4022586, 0.4023391
*end
```

Revision Information:

This feature is available starting from Revision 116631.

References:

[1] Li, Yaning et al, "Prediction of shear-induced fracture in sheet metal forming," Journals of Material Processing Technology, Volume 210, issue 14, (2010).

***DEFINE_CURVE_TRIM**_{OPTION}_

Available options include:

<BLANK>

3D

2D

Purpose: Define curves and controls for sheet blank trimming in sheet metal forming. This keyword can also be used to define mesh adaptivity along a curve within a band prior to the start of a simulation, using the variable TCTOL in this keyword and *CONTROL_ADAPTIVE_CURVE.

> **NOTE:** The option NEW was replaced by 2D starting in Revision 115518.

The three keyword options for *DEFINE_CURVE_TRIM cause different trimming algorithms to be used. The <BLANK> (or no option) case is the oldest method. It is used for 2D trimming. However, it is not recommended because it is slow due to each node on the blank being checked to determine if it is inside the curve or not. Because this method is not recommended, the remarks section will only discuss the 2D and 3D keyword options. When the keyword option 3D is used, the trimming is processed based on the element normal, meaning that the trimming curve is projected to the nearest element using the element normal. The option 2D is used to trim in a fixed direction specified by a vector, that is, the curve is projected onto the model using the same vector. This option is also a 2D trimming. Currently this keyword applies to:

- 2D and 3D adaptive trimming of adaptive shell elements,
- 2D and 3D adaptive trimming of non-adaptive solids,
- 2D and 3D adaptive trimming of adaptive-meshed sandwiched parts (limit to a core of one layer of solid elements with outer layers of shell elements; see "IFSAND" under *CONTROL_ADAPTIVE),
- 2D and 3D adaptive trimming of adaptive-meshed sandwiched parts (a core of multiple layers of solid elements with outer layers of shell elements),
- 2D and 3D adaptive trimming (in conjunction with *CONTROL_FORMING_TRIMMING_SOLID_REFINEMENT) of adaptive-meshed sandwiched parts (a core of multiple layers of solid elements with outer layers of shell elements), and,
- 2D trimming of thick shell elements (TSHELL).

This keyword is not applicable to axisymmetric solids or 2D plane strain/stress elements. Related keywords include *ELEMENT_TRIM, *CONTROL_FORMING_TRIMMING, *CONTROL_FORMING_TRIMMING_SOLID_REFINEMENT, *CONTROL_ADAPTIVE_CURVE, *INCLUDE_TRIM, and *INCLUDE. Another closely related keyword is

*CONTROL_FORMING_TRIM_MERGE, which automatically closes an open-ended trim curve within a user-specified tolerance.

> **NOTE:** Trimming of shell and solid elements in LS-PrePost is supported starting with LS-PrePost 4.0, under *Application* → *MetalForming* → *Easy Setup*.

Card Summary:

Card 1a. Include this card if the keyword option is not used.

TCID	TCTYPE			TCTOL			

Card 1b. Include this card for the 2D keyword option.

TCID	TCTYPE	TFLG	TDIR	TCTOL	DEPTH	NSEED1	NSEED2

Card 1c. Include this card for the 3D keyword option.

TCID	TCTYPE		TDIR	TCTOL	TOLN	NSEED1	NSEED2

Card 2a. Include as many of this card as needed if TCTYPE = 1. The next keyword ("*") card terminates this input.

CX	CY	CZ		

Card 2b. Include this card if TCTYPE = 2.

FILENAME

Data Card Definitions:

No Keyword Option Card. Include this card if not using a keyword option.

Card 1a	1	2	3	4	5	6	7	8
Variable	TCID	TCTYPE			TCTOL			
Type	I	I			F			
Default	none	1			0.25			

VARIABLE	DESCRIPTION
TCID	ID number for trim curve
TCTYPE	Trim curve type: EQ.1: Curve data in XYZ format, obtained following procedures outlined in Figures under *INTERFACE_BLANKSIZE. In addition, only this format is allowed in *INTERFACE_-COMPENSATION_3D. EQ.2: IGES trim curve
TCTOL	Tolerance limiting size of small elements created during trimming (see Figure 17-28). LT.0: "Simple" trimming, producing jagged edge mesh When used together with *CONTROL_ADAPTIVE_CURVE, it is a distance from the curve out (both sides). Within this distance the blank mesh will be refined, as stated in remarks below.

2D Keyword Option Card. Including this card if using the 2D keyword option.

Card 1b	1	2	3	4	5	6	7	8
Variable	TCID	TCTYPE	TFLG	TDIR	TCTOL	DEPTH	NSEED1	NSEED2
Type	I	I	I	I	F	F	I	I
Default	none	1	none	0	0.25	0.0	none	none

VARIABLE	DESCRIPTION
TCID	ID number for trim curve
TCTYPE	Trim curve type: EQ.1: Curve data in XYZ format, obtained following procedures outlined in Figures under *INTERFACE_BLANKSIZE. In addition, only this format is allowed in *INTERFACE_-COMPENSATION_3D. EQ.2: IGES trim curve
TFLG	Element removal option:

VARIABLE	DESCRIPTION
	EQ.-1: Remove material outside curve. EQ.1: Remove material inside curve.
TDIR	ID of a vector (*DEFINE_VECTOR) giving the trim direction (see Figure 17-27). EQ.0: Default vector (0.0,0.0,1.0) is used. The curve is defined in global *XY*-plane and projected onto the mesh in the global *Z*-direction to define the trim line.
TCTOL	Tolerance limiting size of small elements created during trimming (see Figure 17-28). LT.0: "Simple" trimming, producing jagged edge mesh When used together with *CONTROL_ADAPTIVE_CURVE, it is a distance from the curve out (both sides). Within this distance the blank mesh will be refined, as stated in remarks below.
DEPTH	The trimming depth is DEPTH – 1. If the distance between the element and the curve is larger than this value, then it will not be cut. This feature prevents trimming through to the opposite side of the part.
NSEED1/ NSEED2	A node ID on the blank in the area that remains after trimming: LT.0: |NSEED*i*| is a node which may not necessarily be on the blank. See Remark 6.

3D Keyword Option Card. Include this card for the 3D keyword option.

Card 1c	1	2	3	4	5	6	7	8
Variable	TCID	TCTYPE	TFLG	TDIR	TCTOL	TOLN	NSEED1	NSEED2
Type	I	I	I	I	F	F	I	I
Default	none	1	0	0	0.25	2.0	none	none

VARIABLE	DESCRIPTION
TCID	ID number for trim curve

VARIABLE	DESCRIPTION
TCTYPE	Trim curve type: EQ.1: Curve data in XYZ format, obtained following procedures outlined in Figures under *INTERFACE_BLANKSIZE. In addition, only this format is allowed in *INTERFACE_-COMPENSATION_3D. EQ.2: IGES trim curve
TFLG	Side of the surface for which trimming will start for sandwich parts: EQ.1: Top surface (default) EQ.2: Bottom surface
TDIR	Indicate whether the trim curve is near the top or bottom surface of the solids or laminates; see 3D (normal) trimming of solid elements. EQ.0: The trim curve is located near the bottom surface (default). EQ.1: The trim curve is located near the top surface.
TCTOL	Tolerance limiting size of small elements created during trimming (see Figure 17-28). LT.0: "Simple" trimming, producing jagged edge mesh When used together with *CONTROL_ADAPTIVE_CURVE, it is a distance from the curve out (both sides). Within this distance the blank mesh will be refined, as stated in remarks below.
TOLN	Maximum gap between the trimming curve and the mesh. If the gap is bigger than this value, this section in the curve will not be used.
NSEED1/ NSEED2	A node ID on the blank in the area that remains after trimming: LT.0: |NSEED*i*| is a node which may not necessarily be on the blank. See Remark 6.

Point Cards. Additional cards for TCTYPE = 1. Put one point per card (2E20.0). Input is terminated at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	CX		CY		CZ			
Type	F		F		F			
Default	0.0		0.0		0.0			

VARIABLE	DESCRIPTION
CX, CY, CZ	*X*, *Y*, *Z*-coordinates of trim curve.

IGES File Card. Additional card for TCTYPE = 2.

Card 2b	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	Name of IGES database containing trim curve(s).

Trimming Capability Summary:

This keyword and its options deal with trimming of the following scenarios:

	2D (along one direction)	3D (element normal)	2D & 3D Double Trim	Adaptive mesh
Shell	Yes	Yes	Yes	Yes
Solids	Yes	Yes	Yes	N/A
Sandwiched Parts (Laminates)	Yes	Yes	Yes	Yes
TSHELL	Yes	N/A	N/A	N/A

Remarks:

1. **IGES Trim Curve.** Only IGES entities 110 and 106 are supported when using TCTYPE = 2. The eZ-Setup for trimming function ensures correct IGES files are written for a trimming simulation.

2. **Trim Curve Requirements.** Enclosed trimming curves (same start and end points) are required for all options. Also, for each enclosed trimming curve, only one curve segment is acceptable for the option 3D; while several curve segments are acceptable with the option 2D. Curves can be manipulated through the *Merge* and *break* features in LS-PrePost4.0, found under *Curve/Merge* (always select *piecewise* under *Merge*) and *break*.

   For 3D trimming, trim curves need to be sufficiently close to the part. Trim curves can be processed using curve projection onto the mesh in LS-PrePost. This feature is accessible under *GeoTol/Project/Closest Proj/Project to Element/By Part*. Using a double precision LS-DYNA executable may also help in this situation.

   For the option 2D, Revision 68643 and later releases enable trimming of a part where trim lines go beyond the part boundary. This is illustrated in Figure 17-33.

3. **Order of Trimming.** This keyword in combination with *ELEMENT_TRIM trims the requested parts before a job starts (pre-trimming). This combination can also handle an adaptive mesh. If the keyword *ELEMENT_TRIM is not included, the parts are trimmed after the job is terminated (post-trimming).

4. **TCTOL.** The trimming tolerance TCTOL limits the size of the smallest element created during trimming. A value of 0.0 places no limit on element size. A value of 0.5 restricts new elements to be at least half of the size of the parent element. A value of 1.0 allows no new elements to be generated, only repositioning of existing nodes to lie on the trim curve. A negative tolerance value activates "simple" trimming, where entire elements are removed, leaving a jagged edge.

5. **Mesh Refinement along the Trim Curve for Shell Elements.** When large elements are along the trim curve, the blank mesh can be pre-adapted along the trim curve before trimming by adding the keyword *CONTROL_ADAPTIVE_CURVE for a better quality trim edge. Care should be taken when using this keyword since an excessive number of elements can be created when setting the fields. Note that this keyword can only be used for shell elements.

   The variable TCTOL can be used to control the mesh refinement along a curve when used together with *CONTROL_ADAPTIVE_CURVE. In this scenario, it is the distance of the entire refinement width. The mesh will be refined in the beginning of the simulation. This method offers greater control on the number of elements to be generated during mesh refinement. A detailed description and

example are provided under the manual section under *CONTROL_ADAPTIVE_CURVE.

Sometimes it is helpful to conduct a check of the trimmed mesh along the edge in the same trimming input deck using the keyword *CONTROL_CHECK_SHELL. This is especially useful for the next continued process simulation. For detailed usage, see that manual page.

6. **Seed Nodes.** A seed node is used to define which side of the drawn panel will be kept after trimming. With the frequent application of adaptive re-meshing, the seed node for trimming is often unknown until the draw forming is complete. When NSEED*i* is negative, an extra node unrelated to the blank and tools can be used as the seed node, enabling simulating a trimming process independent of the previous process simulation results. The extra node can be defined using keyword *NODE. If the seed node is too far away from the blank, it will be projected onto the blank and the new position will be used as the seed node. Typically, this node can be selected from the stationary tool in its home position. Selecting a seed node is quite easy in *Trimming* process of LS-PrePost4.0 eZSetup for metal forming application.

   A partial keyword input example for the trimming of a double-attached *NUMISHEET2002 fender outer* with the option 2D is listed below, where a 2D trimming is performed with IGES file doubletrim.iges in the global *Z*-axis, with two nodes of negative ID 43356 and 18764 assigned to the variables NSEED1 and NSEED2, respectively. The two seed nodes are defined off the stationary lower post and do not necessarily need to be a part of the post as shown in Figure 17-29. The drawn panels in the wire frame are shown in Figure 17-30, along with the thickness/thinning contour (Figure 17-31). In Figure 17-32, the drawn panels are trimmed and separated.

```
*KEYWORD
*CONTROL_TERMINATION
0.000
*CONTROL_SHELL
......
*CONTROL_OUTPUT
......
*DATABASE_BINARY_D3PLOT
......
*DATABASE_EXTENT_BINARY
......
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*SET_PART_LIST
......
*PART
Blank
......
*SECTION_SHELL
......
*MAT_3-PARAMETER_BARLAT
......
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*INCLUDE_TRIM
```

```
drawn.dynain
*ELEMENT_TRIM
         1
*DEFINE_CURVE_TRIM_2D
$#     TCID    TCTYPE      TFLG      TDIR     TCTOL     DEPTH    NSEED1    NSEED2
         1         2                    0     0.250         1    -43356    -18764
doubletrim.iges
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*NODE
18764,-184.565,84.755,78.392
43356,-1038.41,119.154,78.375
*INTERFACE_SPRINGBACK_LSDYNA
......
*END
```

Alternatively, if the fields NSEED*i* are not defined, the seeds can be defined using *DEFINE_TRIM_SEED_POINT_COORDINATES. A partial keyword input is provided below for trimming of the same double-attached fender outer.

```
*INCLUDE_TRIM
drawn.dynain
*ELEMENT_TRIM
         1
*DEFINE_CURVE_TRIM_2D
$#     TCID    TCTYPE      TFLG      TDIR     TCTOL     DEPTH    NSEED1    NSEED2
         1         2                    0     0.250         1
doubletrim.iges
*DEFINE_TRIM_SEED_POINT_COORDINATES
$    NSEED        X1        Y1        Z1        X2        Y2        Z2
         2  -184.565    84.755    78.392  -1038.41   119.154    78.375
```

7. **Nodal Temperature.** Starting in Revision 81105, trimming of blank with nodal temperature (related to hot and warm stamping) is available. At the end of the trimming, the file new_temp_ic.inc will be output along with the usual dynain file. Furthermore, the keyword *CONTROL_CHECK_SHELL can be used together when trimming with nodal temperature.

8. ***INCLUDE_TRIM.** We recommend using the keyword *INCLUDE_TRIM at all times, either for trimming or for mesh refinement purposes, except in the case where the to-be-trimmed sheet blank has no stress and strain information (no *INITIAL_STRESS_SHELL, and *INITIAL_STRAIN_SHELL cards present in the sheet blank keyword or dynain file). In that case the keyword *INCLUDE must be used. For solid element and thick shell element trimming, the keyword *INCLUDE_TRIM *must* be used. A check box to indicate that the blank is free of stress and strain information is provided in the Trimming process in the eZ-Setup for users to set up a trimming input deck under the circumstance.

2D and 3D trimming of solid elements and laminates:

In all trimming of solids and laminates, only a dynain file is written out (no d3plot files will be output).

2D trimming of solid elements

As of Revision 92088, 2D (option 2D) trimming in any direction (defined by a vector) of solid elements is available. An illustration of the 2D trim is shown in Figure 17-34. A partial keyword example is provided below, where trim curves in trimcurves2d.iges are being used to perform a solid element trimming along a vector defined along the global Z-axis.

```
*KEYWORD
*INCLUDE_TRIM
incoming.dynain
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PARAMETER_EXPRESSION
...
*CONTROL_TERMINATION
$   ENDTIM
0.0
*CONTROL_OUTPUT
...
*DATABASE_XXX
...
*PART
 Solid Blank
$#     pid     secid       mid
  &blk1pid  &blk1sec  &blk1mid
*SECTION_Solid
&blk1sec,&elform
*MAT_PIECEWISE_LINEAR_PLASTICITY
...
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$   Trim cards
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*CONTROL_FORMING_TRIMMING
$     PSID
   &blksid
*DEFINE_TRIM_SEED_POINT_COORDINATES
$ NSEED,X1,Y1,Z1,X2,Y2,Z2
1,&seedx,&seedy,&seedz
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_CURVE_TRIM_2D
$#    tcid    tctype      tflg      tdir     tctol     depth    nseed1    nseed2
         2         2         0         1   0.10000  1.000000         0         0
$# filename
trimcurves2d.iges
*DEFINE_VECTOR
$#     vid        xt        yt        zt        xh        yh        zh       cid
         1     0.000     0.000     0.000     0.000     0.000  1.000000         0
*INTERFACE_SPRINGBACK_LSDYNA
$     PSID
&blksid,&nshv
*END
```

Currently, 2D trimming of solids in some cases may be approximate. The trimming will trim the top and bottom sides of the elements, not crossing over to the other sides. This can be seen, for instance, in trimming involving a radius.

3D (normal) trimming of solid elements

3D trimming (option 3D) of solid elements is available. The trim curve should be created based on the solid element normal. The field TDIR indicates whether the curve is closer to the top (TDIR = 1) or bottom (TDIR = 0) surface of the solid; see Figure 17-36. The projection of trim curves onto either the top or bottom surface of the blank is important to ensure a smooth, successful trim.

To change the example in the 2D case to the 3D case, the option 2D is changed to 3D, and trimcurves3d.iges is used. The field TDIR is set to "1" since the trim curve is on the positive side of the element normal (Figure 17-36). Since 3D trimming is along the element normal directions, *DEFINE_VECTOR is no longer needed.

```
*DEFINE_CURVE_TRIM_3D
$#     tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
         2         2         0         1   0.10000  1.000000         0         0
$# filename
trimcurves3d.iges
```

2D and 3D trimming of non-adaptive-meshed sandwiched parts (laminates)

The laminates can have multiple layers of solid elements, sandwiched by a top and a bottom layer of shell elements. Note that the nodes of the shell elements must share the nodes with the solid elements at the top and bottom layers. See Figure 17-35 as an example. The input deck is similar to those used for trimming of solid elements, except the variable ITYP under *CONTROL_FORMING_TRIMMING should be set to "1" to activate the trimming of laminates in both 2D and 3D conditions:

```
*CONTROL_FORMING_TRIMMING
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$     PSID                ITYP
   &blksid                   1
```

Again, in the case of 3D trimming, the projection of trim curves onto either the top or bottom surface of the blank is important to ensure a smooth and successful trim.

2D and 3D trimming of adaptive-meshed sandwiched parts - one core layer of solids

2D and 3D trimming of adaptive-meshed laminates with one core layer of solid elements sandwiched by shell elements are available. The shell elements must share the same nodes as the solid elements.

*Note elements are refined automatically along the trim curves until no dependent nodes (see *CONSTRAINED_ADAPTIVITY) would be cut by the trim curves. In addition, the keyword *CONTROL_ADAPTIVE_CURVE must not be used since it is only applied to shell elements and would cause error termination otherwise.* This trimming, unlike for shell elements, requires no additional adaptivity-related keyword inputs for mesh refinement. An example of the trimming on the 2005 NUMISHEET Cross Member is shown in Figure 17-37.

2D and 3D trimming of adaptive-meshed sandwiched parts - multiple layers of solids:

This capability is available when used with the keyword *CONTROL_FORMING_TRIMMING_SOLID_REFINEMENT. This keyword allows for the elements along the trim curves to be refined. Please refer to the corresponding manual pages.

2D trimming of thick shell elements (TSHELL):

2D trimming of thick shells is supported. Note that, by definition, thick shells have only one layer of elements. Note also *INCLUDE_TRIM (not *INCLUDE) must be used to include the dynain file to be trimmed. The input deck for 2D trimming of thick shells is similar to that used for solid-element trimming.

2D and 3D trimming of double-attached solids and laminates:

These features are available starting in Revision 110140. Both seed point coordinates can be specified in *DEFINE_TRIM_SEED_POINT_COORDINATES to define a seed coordinate for each part, as shown below:

```
*DEFINE_TRIM_SEED_POINT_COORDINATES
$    NSEED         X1         Y1         Z1         X2         Y2         Z2
         2   -184.565     84.755     78.392   -1038.41    119.154     78.375
```

In Figure 17-38, a 2D double-trimming example on a sandwiched part is shown using the 2005 NUMISHEET cross member. Two trim curves and two seed nodes are defined for each to be trimmed portion.

Note again, *INCLUDE_TRIM (not *INCLUDE) must be used to include the dynain file to be trimmed.

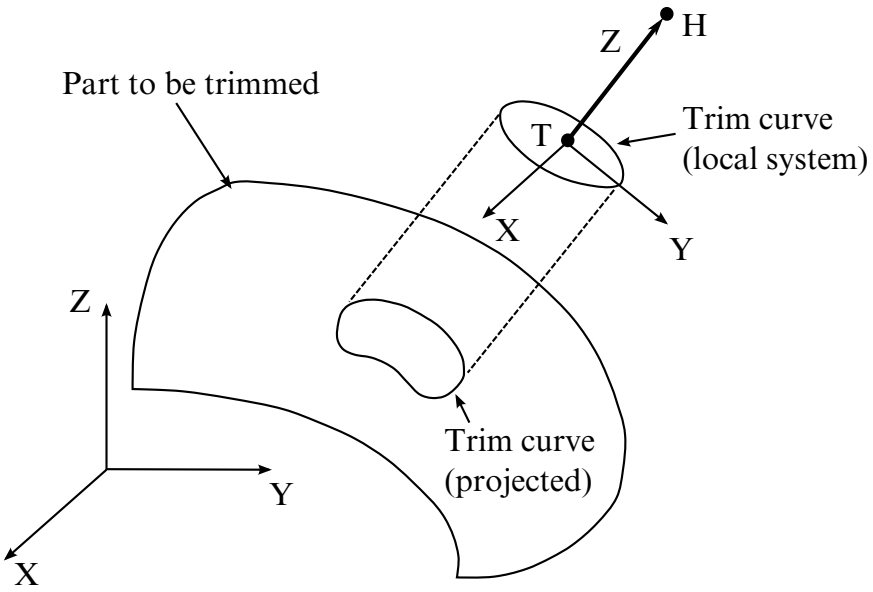


Figure 17-27. Trimming Orientation Vector. The tail (**T**) and head (**H**) points define a local coordinate system (x, y, z). The global coordinate system is named (X, Y, Z). The local x-direction is constructed in the Xz-plane. If X and z nearly coincide ($|X \cdot z| > 0.95$), then the local x-direction is instead constructed in the Yz-plane. Trim curve data is input in the xy-plane and projected in the z-direction onto the deformed mesh to obtain the trim line.

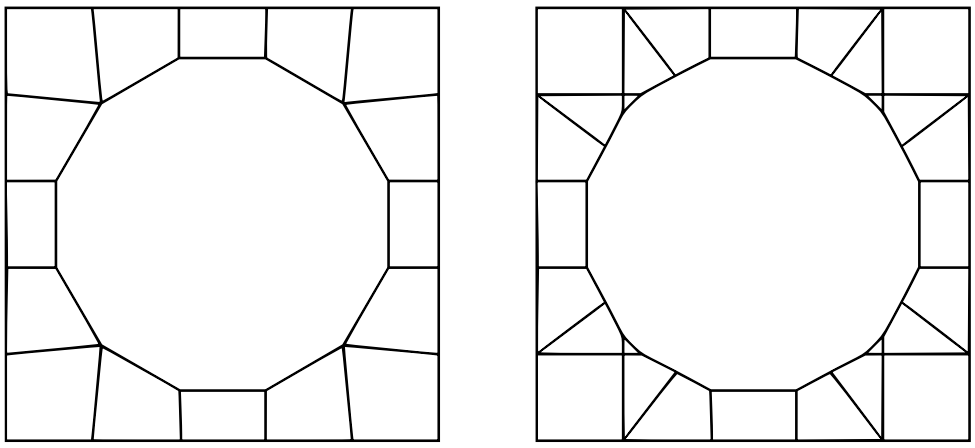


Figure 17-28. Trimming Tolerance. The tolerance limits the size of the small elements generated during trimming. The default tolerance (left) produces large elements. Using a tolerance of 0.01 (right) allows smaller elements and more detail in the trim line.

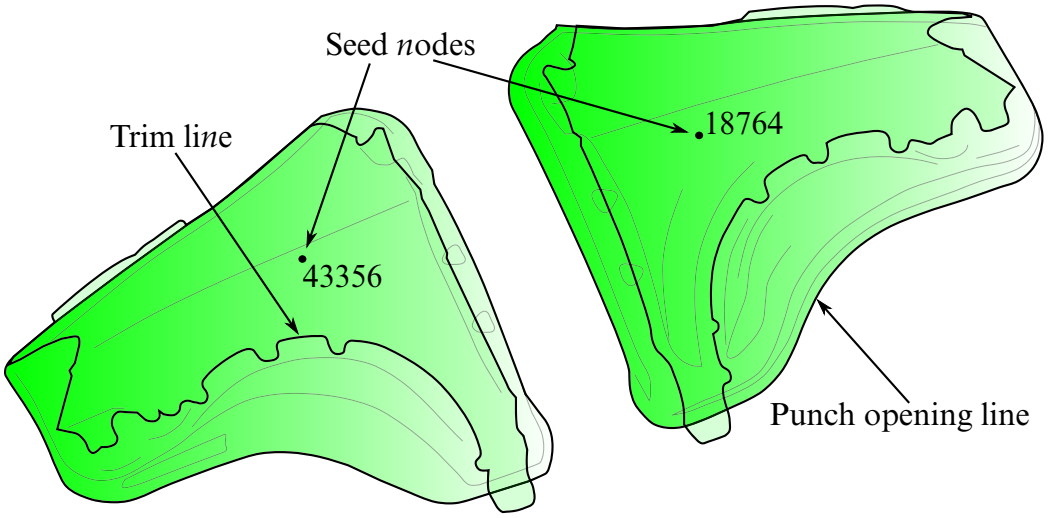


Figure 17-29. Trimming of a double-attached part (NUMISHEET2002 Fender Outer).

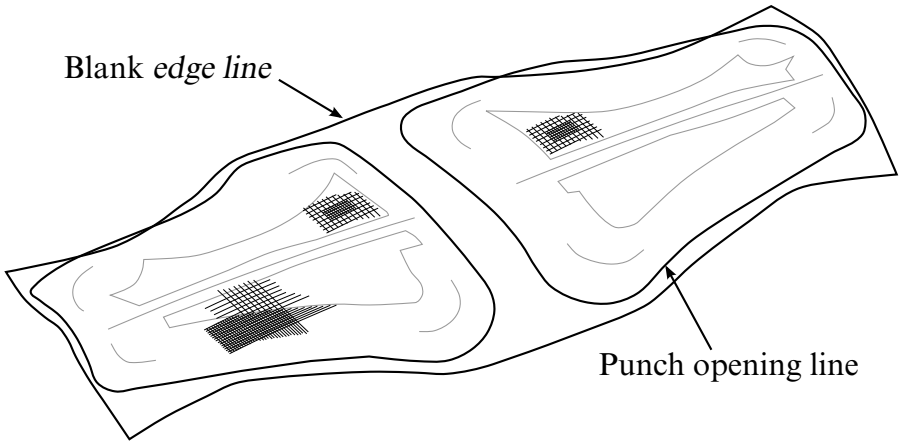


Figure 17-30. The fender outer (draw complete) in wireframe mode.

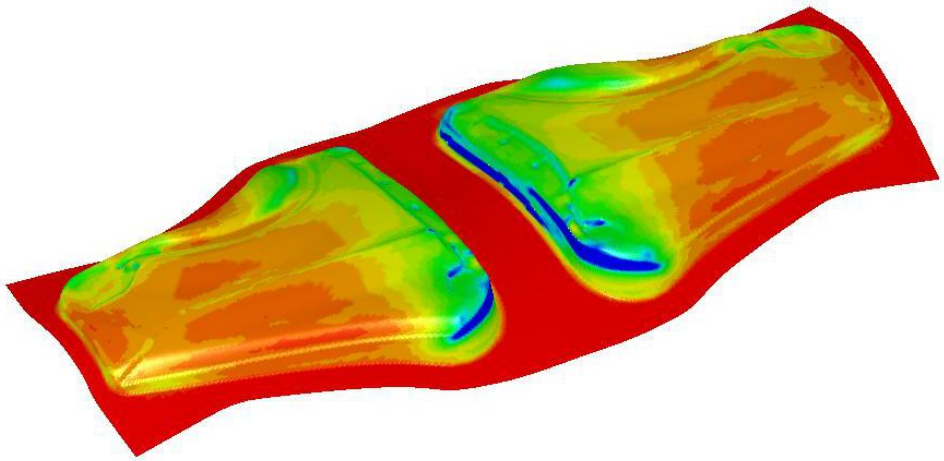


Figure 17-31. The fender outer - thickness/thinning plot on the drawn panel.

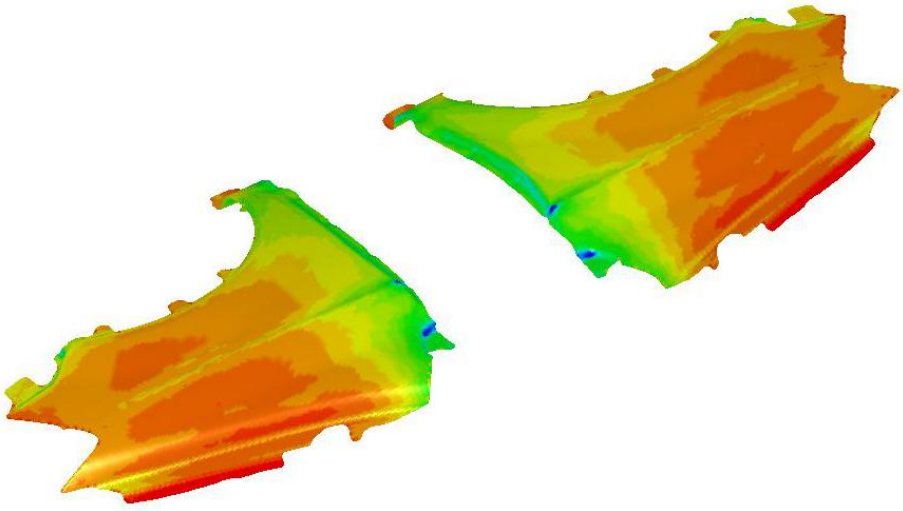


Figure 17-32. The fender outer trim complete using the NSEED1/NSEED2 feature.

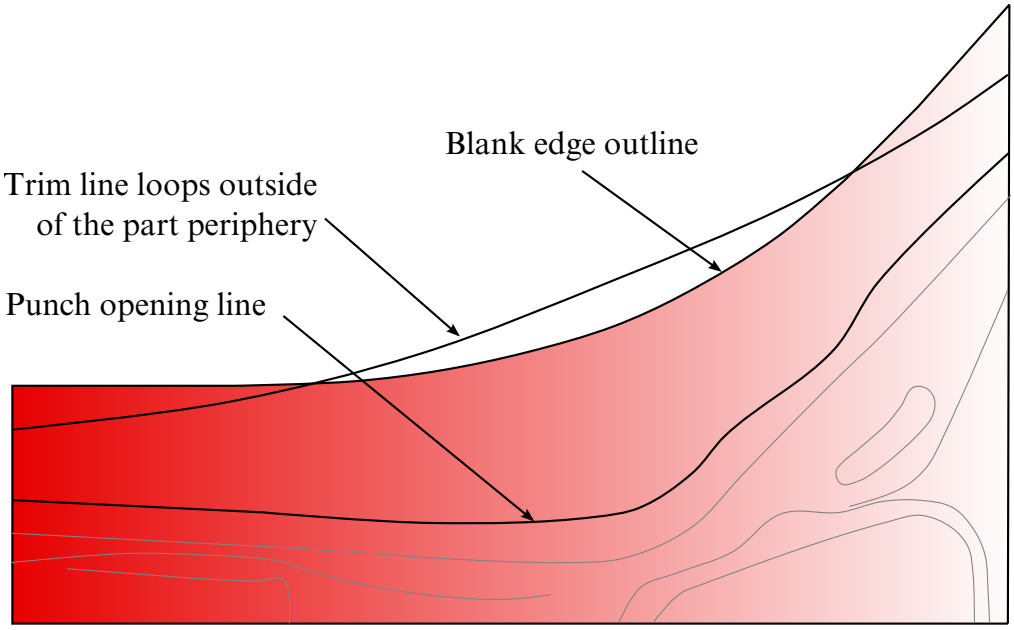


Figure 17-33. Revision 68643 deals with trim curves going beyond part boundary.

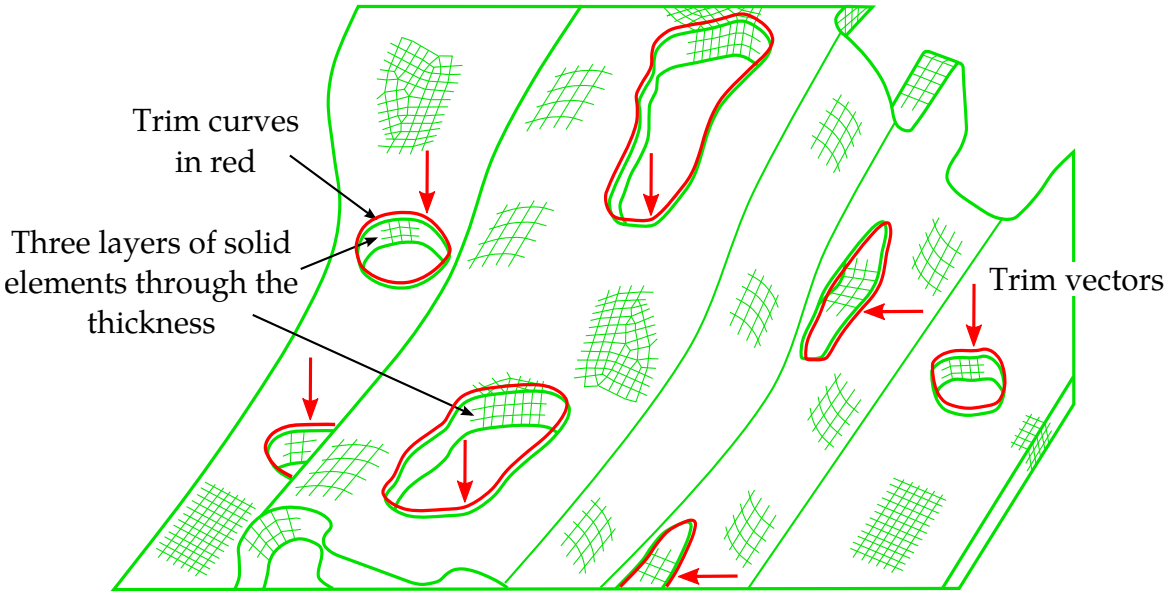


Figure 17-34. 2D trimming of solids using *DEFINE_CURVE_TRIM_2D

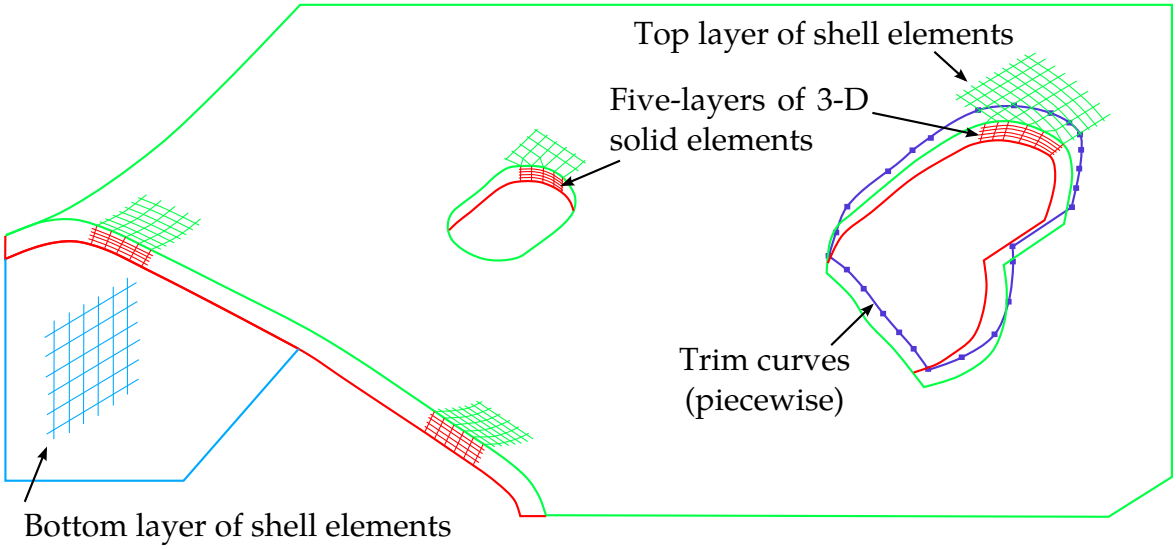
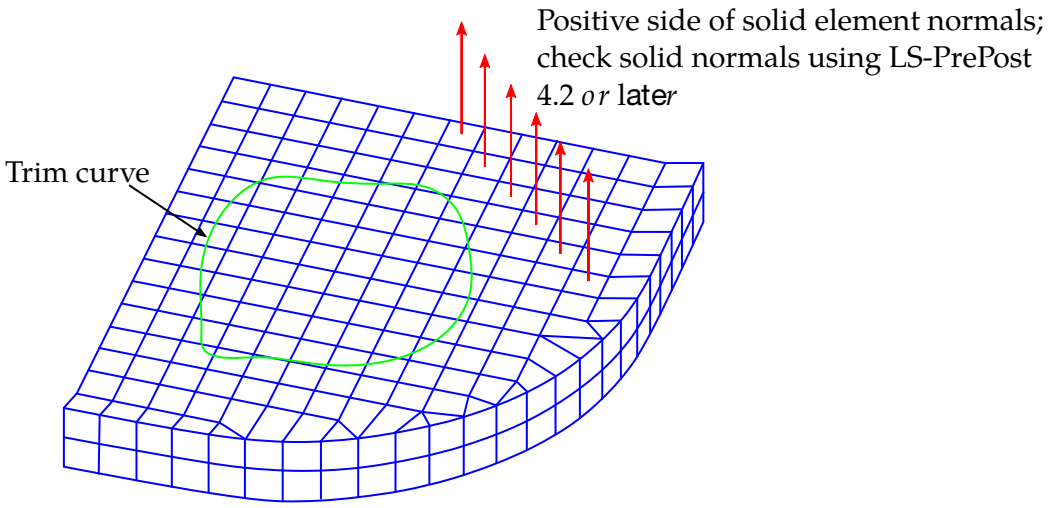


Figure 17-35. 3D trimming of laminates (a core of multiple-layers of solid elements with top and bottom layers of shell elements) using *DEFINE_CURVE_-TRIM_3D. Note that shell elements must share the same nodes with the solid elements at the top and bottom layer.



All solid element normals must be consistent. If the trim curve is close to the positive normal side, set TDIR = 1; otherwise set TDIR = 0. Respacing the curve with more points and projecting the respaced curve to the top or bottom solid surface may help the trimming.

Figure 17-36. Define trim curve for 3D trimming of solid and laminates.

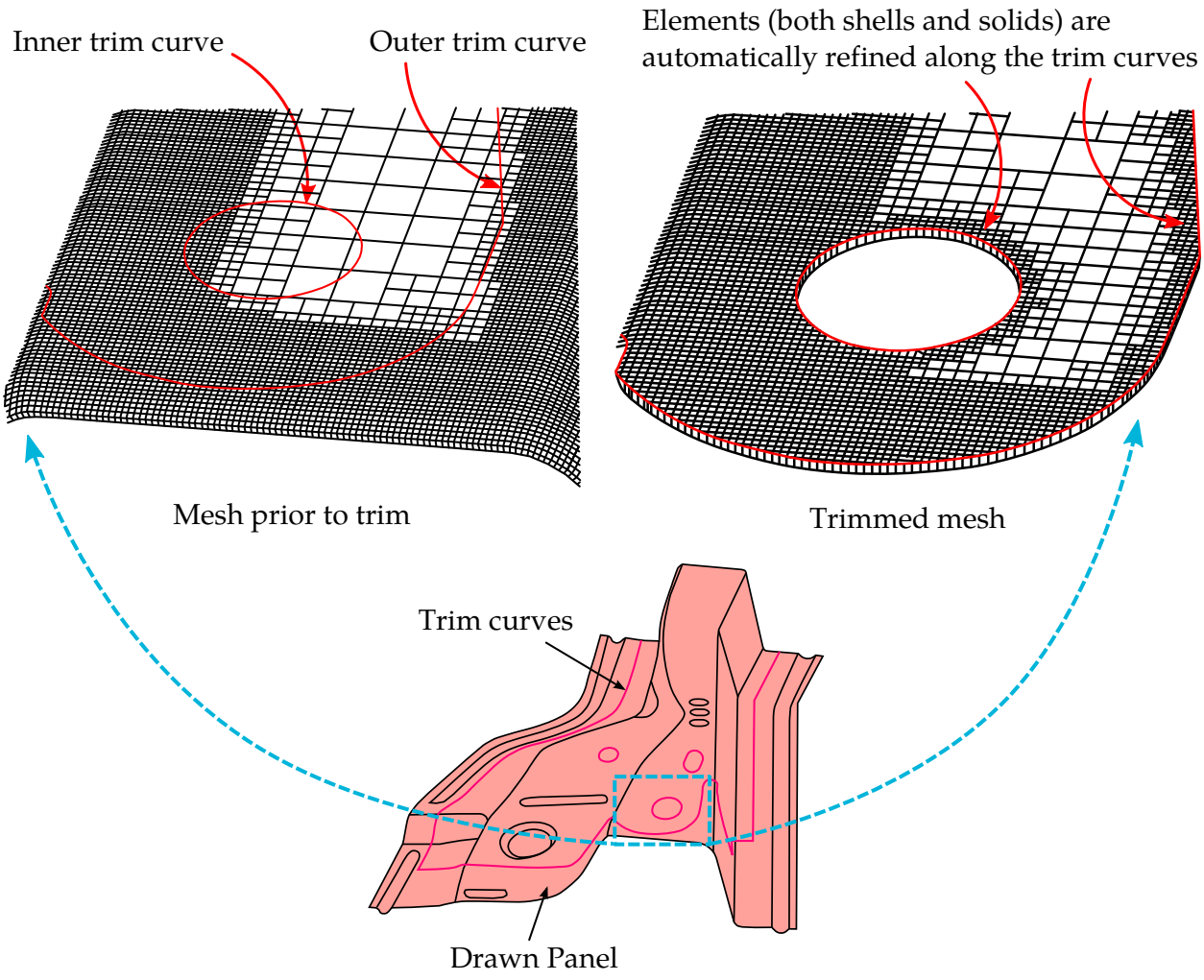


Figure 17-37. Trimming of an adaptive mesh on a sandwiched part. Meshes are automatically refined along the trim curves. Note that only one layer of solid element as a core is allowed for adaptive-meshed sandwich parts without CONTROL_FORMING_TRIMMING_SOLID_REFINEMENT. Also, the top and bottom layer of shells must share the same nodes as the solid elements.

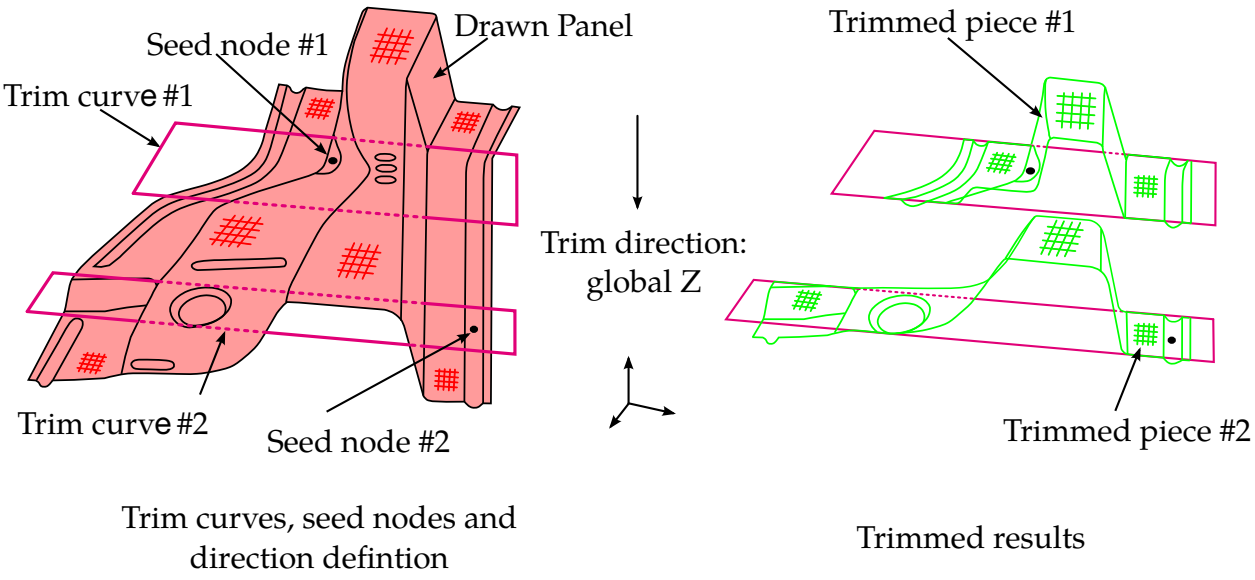


Figure 17-38. An example of 2D trimming of sandwiched part (laminates) from
2005 NUMISHEET benchmark - cross member.

*DEFINE_DE_ACTIVE_REGION

Purpose: To define an interested region for Discrete Elements (DE) for high efficiency collision pair searching. Any DE leaving this domain will not be considered in the future DE searching and will also be disabled in the contact algorithm.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	XM/R	YM	ZM	TBIRTH	TDEATH	NFREQ
Type	I	I	F	F	F	F	F	I
Default	none	0	0.	0.	0.	0.	10^{20}	1

VARIABLE	DESCRIPTION
ID	Set ID/Box ID/Node ID
TYPE	Type for ID: EQ.0: Part set ID EQ.1: Box ID EQ.2: Node ID (moving sphere) EQ.3: Node ID (box-shaped region that moves with this node)
XM, YM, ZM	For TYPE = 0 or 1, factor for region's margin on each direction based on region length. The static coordinates limits are determined either by the part set or box. To provide a buffer zone, these factors, XM, YM, and ZM, can be used. The margin in each direction is calculated in the following way: Let X_{max} and X_{min} be the limits in the x-direction given by the set or box. Then, $$\Delta X = X_{max} - X_{min} \ .$$ The margin is, then, computed from the input as, $$X_{margin} = \mathrm{XM} \times \Delta X$$ Then the corresponding limits for the active region are, $$X'_{max} = X_{max} + X_{margin}$$ $$X'_{min} = X_{min} - X_{margin}$$

VARIABLE	DESCRIPTION
	For TYPE = 3, the region is a box that moves with the node. The node is also the region's center. The location of the region is updated every NFREQ cycles. XM, YM, and ZM give the distance in each direction that the region extends from the node. For instance, let X_{node} be the position of the node in the x-direction. Let X_{max} and X_{min} be minimum and maximum x values defining the region. X_{max} and X_{min} are then calculated as: $$X_{max} = X_{node} + XM$$ $$X_{min} = X_{node} - XM$$
R	Radius of the region which is centered at the Node ID for TYPE = 2
TBIRTH, TDEATH	Birth and death times for the active region when Node ID is used (TYPE = 2)
NFREQ	Number of cycles between updates of the region's location for TYPE = 3 (default = 1)

*DEFINE_DE_BOND

Purpose: Define a bond model for bonds between discrete element spheres (DES). Note that *DEFINE_DE_BOND_OVERRIDE may override the bond properties between DES.

Card Summary:

Card 1. This card is required.

SID	STYPE	BDFORM	IDIST	MAXCN	BINARY		

Card 2. This card is required.

PBN	PBS	PBNS	PBSS	SFA	ALPHA	BENDSF	MAXGAP

Card 3a. Include this card when BDFORM = 3.

ARSP	BRSP						

Card 3b. Include this blank card when IDIST ≠ 0 and BDFORM ≠ 3.

Card 4a. Include this card for IDIST = -1 or 1.

PBN_SD	PBS_SD	PBNS_SD	PBSS_SD				

Card 4b. Include this card for IDIST = -2 or 2.

MPBN	MPBS	MPBNS	MPBSS				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	BDFORM	IDIST	MAXCN	BINARY		
Type	I	I	I	I	I	I		
Default	none	0	1	0	0	0		

VARIABLE	DESCRIPTION
SID	Node set, part set, or part ID for which bond properties apply

VARIABLE	DESCRIPTION
STYPE	SID type: EQ.0: DES node set EQ.2: DES part set EQ.3: DES part
BDFORM	Bond formulation: EQ.1: Linear bond formulation (default) EQ.3: Bond strength that increases linearly with deformation rate. See Remark 3.
IDIST	Distribution of bond properties (see Remarks 4 and 5): EQ.-2: Weibull distribution (non-deterministic) EQ.-1: Gaussian distribution (non-deterministic) EQ.0: Single property (default) EQ.1: Gaussian distribution (deterministic) EQ.2: Weibull distribution (deterministic)
MAXCN	Maximum coordination number for outputting the DES particles that belong to a bond. If the coordination number of a DES particle that belongs to a bond is less than or equal to this value, the DES particle is output at the initial time state. EQ.0: Do not print out any bonded DES particles at the initial time state.
BINARY	Output file type for printing the bonded DES particle if the coordination number is less than or equal to MAXCN: EQ.0: Do not write the data to a file. EQ.1: Write an ASCII file. EQ.2: Write the data to binary database binout. EQ.3: Write the data to an ASCII file and to the binary database binout.

Bond Formulation Card.

Card 2	1	2	3	4	5	6	7	8
Variable	PBN	PBS	PBNS	PBSS	SFA	ALPHA	BENDSF	MAXGAP
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	1.0	0.0	1.0	10^{-4}

VARIABLE	DESCRIPTION
PBN	Parallel-bond modulus [Pa]. See Remarks 1 and 2. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MPBN ≠ 0, this value is the scale parameter.
PBS	Parallel-bond stiffness ratio, shear stiffness/normal stiffness. See Remark 2. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MPBS ≠ 0, this value is the scale parameter.
PBNS	Parallel-bond maximum normal stress. A zero value defines an infinite maximum normal stress. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MPBNS ≠ 0, this value is the scale parameter.
PBSS	Parallel-bond maximum shear stress. A zero value defines an infinite maximum shear stress. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MPBSS ≠ 0, this value is the scale parameter.
SFA	Bond radius multiplier
ALPHA	Numerical damping, 0.0 ≤ ALPHA ≤ 1.0
BENDSF	Influence of bending/twisting on the bond failure criteria (see Remark 6). EQ.-1.0: No bending/twisting is considered in the bond failure criteria. EQ.0.0: Defaults to 1.0. GT.0.0: Scale factor for the bending/twisting component in the bond failure criteria.
MAXGAP	Maximum gap between two bonded spheres:

VARIABLE	DESCRIPTION
	GT.0.0: Defines the ratio of the smaller radius of two bonded spheres as the maximum gap, that is, MAX-GAP × $\min(r_1, r_2)$.
	LT.0.0: Absolute value is used as the maximum gap.

Rate Sensitivity Parameters Card (BDFORM = 3). Include this card when BDFORM = 3.

Card 3a	1	2	3	4	5	6	7	8
Variable	ARSP	BRSP						
Type	F	F						
Default	0.0	0.0						

VARIABLE	DESCRIPTION
ARSP	Deformation rate sensitivity parameter for parallel-bond maximum normal stress. A zero value ignores deformation rate effects on bond strength.
BRSP	Deformation rate sensitivity parameter for parallel-bond maximum shear stress. A zero value ignores deformation rate effects on bond strength. t

Include this blank card when BDFORM ≠ 3 and IDIST ≠ 0.

Card 3b	1	2	3	4	5	6	7	8
Variable								
Type								
Default								

Standard Deviation for Gaussian Distribution Card. Include this card IDIST = -1 or 1.

Card 4a	1	2	3	4	5	6	7	8
Variable	PBN_SD	PBS_SD	PBNS_SD	PBSS_SD				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
PBN_SD	Standard deviation for parallel-bond modulus. PBN on Card 2 is the mean. EQ.0.0: Uniform distribution
PBS_SD	Standard deviation for parallel-bond stiffness ratio, shear stiffness/normal stiffness. PBS on Card 2 is the mean. EQ.0.0: Uniform distribution
PBNS_SD	Standard deviation for parallel-bond maximum normal stress. PBNS on Card 2 is the mean. EQ.0.0: Uniform distribution
PBSS_SD	Standard deviation for parallel-bond maximum shear stress. PBSS on Card 2 is the mean. EQ.0.0: Uniform distribution

Shape Parameters for Weibull Distribution Card. Include this card if IDIST = -2 or 2.

Card 4b	1	2	3	4	5	6	7	8
Variable	MPBN	MPBS	MPBNS	MPBSS				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
MPBN	Shape parameter (heterogeneity index) for parallel-bond modulus. PBN on Card 2 is the associated scale parameter. EQ.0.0: No distribution. PBN is the parallel-bond modulus.
MPBS	Shape parameter (heterogeneity index) for parallel-bond stiffness ratio, shear stiffness/normal stiffness. PBS on Card 2 is the associated scale parameter. EQ.0.0: No distribution. PBS is the parallel-bond stiffness ratio.
MPBNS	Shape parameter (heterogeneity index) for parallel-bond maximum normal stress. PBNS on Card 2 is the associated scale parameter.. EQ.0.0: No distribution. PBNS is the parallel-bond maximum normal stress.
MPBSS	Shape parameter (heterogeneity index) for parallel-bond maximum shear stress. PBSS on Card 2 is the associated scale parameter. EQ.0.0: No distribution. PBSS is the parallel-bond maximum shear stress.

Remarks:

1. **Normal force.** The normal force between two bonded discrete elements with radii r_1 and r_2 is calculated as

$$\Delta f_{\mathrm{n}} = \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_n \ ,$$

where

$$A = \pi r_{\mathrm{eff}}^2$$
$$r_{\mathrm{eff}} = \min(r_1, r_2) \times \mathrm{SFA}$$

2. **Shear force.** The shear force is calculated as

$$\Delta f_{\mathrm{s}} = \mathrm{PBS} \times \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_s \ .$$

3. **Linearly increasing bond strength with increasing loading rate.** With BDFORM = 3, bond strength increases linearly with an increased loading rate. The following equations give the maximum normal and shear stresses:

$$\sigma_n(v_{\text{rel}}) = \text{PBNS} + \text{ARSP} \times \frac{v_{\text{rel}}}{r_1 + r_2}$$
$$\tau(v_{\text{rel}}) = \text{PBSS} + \text{BRSP} \times \frac{v_{\text{rel}}}{r_1 + r_2}$$

Here PBNS and PBSS correspond to the maximum bond strength in quasistatic loading, ARSP and BRSP are the rate sensitivity parameters, and v_{rel} is the relative velocity between two bonded particles with radii r_1 and r_2.

4. **Distributions of bond properties.** IDIST determines the distribution of bond properties for the DES in SID. It specifically affects the parallel-bond modulus, stiffness ratio, maximum normal stress, and maximum shear stress. IDIST = 0 means all the DES in SID have the same bond properties.

   The distribution of bond properties for IDIST = -1 and 1 follows a Gaussian distribution:

$$f(x|\text{prop_mean}, \text{prop_sd}) = \frac{1}{\text{prop_sd}\sqrt{2\pi}} e^{\frac{(x-\text{prop_mean})^2}{2\times\text{prop_sd}^2}},$$

   where prop_mean is the mean value of the property, prop_sd is the standard deviation, and x is the statistical distribution of the property. For this distribution, you input the following pairs to specify the mean and standard deviation for the four bond properties: (PBN, PBN_SD), (PBS, PBS_SD), (PBNS, PBNS_SD), and (PBSS, PBSS_SD).

   The distribution of bond properties for IDIST = -2 and 2 follows a Weibull distribution:

$$f(n|\text{prop_sf}, \text{mprop}) = \frac{\text{mprop}}{\text{prop_sf}} \left(\frac{n}{\text{prop_sf}}\right)^{\text{mprop}-1} e^{-\left(\frac{n}{\text{prop_sf}}\right)^{\text{mprop}}},$$

   where prop_sf is the scale factor for the property, mprop is the shape factor for the property, and n is the statistical distribution of the property. For this distribution, you input the following pairs to specify the scale factor and shape factor for the four bond properties: (PBN, MPBN), (PBS, MPBS), (PBNS, MPBNS), and (PBSS, MPBSS).

5. **Deterministic versus non-deterministic distributions.** We have implemented both deterministic and non-deterministic versions for the distributions of bond properties (IDIST ≠ 0). By deterministic, we mean that each run uses the same seed and has the same sequence of numbers that fit the distribution. Thus, multiple runs with the same input deck yield the same results. In contrast, the non-deterministic algorithms obtain a random seed each time. Therefore, each run with the same input deck may have a different solution as the sequence of numbers changes each time.

6. **BENDSF.** BENDSF enables removing or reducing the influence of twisting/bending on the bond failure criteria. The following equations determine bond failure:

$$\bar{\sigma}_{\max} = \frac{-\bar{F}_n}{A} + \text{BENDSF} \times \frac{|\bar{M}_s| r_{\text{eff}}}{I} \geq \text{PBNS}$$
$$\bar{\tau}_{\max} = \frac{-\bar{F}_s}{A} + \text{BENDSF} \times \frac{|\bar{M}_n| r_{\text{eff}}}{J} \geq \text{PBSS}$$

Here, the $\bar{}$ means average. I and J are the moment of inertia and polar moment of inertia. A and r_{eff} are as described in Remark 1.

*DEFINE_DE_BOND_OVERRIDE

Purpose: Override the bond values specified with *DEFINE_DE_BOND for bonds between certain discrete element spheres. This feature allows you to define heterogeneous bonds.

Unless *DEFINE_DE_BOND_OVERRIDE is required for defining heterogeneous bonds, we recommend not using it. The algorithms invoked by including *DEFINE_DE_BOND_OVERRIDE require more data passing during the calculation. Thus, it is more computationally expensive.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE		IRAND				
Type	I	I		I				
Default	none	0		0				

Card 2	1	2	3	4	5	6	7	8
Variable	PBN	PBS	PBN_S	PBS_S	SFA	ALPHA	BENDSF	
Type	F	F	F	F	F	F	F	
Default	none	none	none	none	1.0	0.0	1.0	

Additional card if IRAND = 1

Card 3	1	2	3	4	5	6	7	8
Variable	PBNMAX	PBSMAX	PBN_SMX	PBS_SMX				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
SID	Node set, part set, or part for which the bond properties will be overridden
STYPE	SID type: EQ.0: DES node set EQ.2: DES part set EQ.3: DES part
IRAND	Random distribution of bond properties: EQ.0: Single property (default) EQ.1: Uniform distribution
PBN	Parallel-bond modulus [Pa]. For IRAND = 1, this is the minimum value in the distribution. See Remarks 1 and 2.
PBS	Parallel-bond stiffness ratio. shear stiffness/normal stiffness. For IRAND = 1, PBS is the minimum value of the parallel-bond stiffness ratio in the distribution. See Remark 2.
PBN_S	Parallel-bond maximum normal stress. A zero value defines an infinite maximum normal stress. For IRAND = 1, PBN_S is the minimum value used for the parallel-bond maximum normal stress in the distribution.
PBS_S	Parallel-bond maximum shear stress. A zero value defines an infinite maximum shear stress.
SFA	Bond radius multiplier
BENDSF	Influence of bending/twisting on the bond failure criteria (see Remark 4). EQ.-1.0: No bending/twisting is considered in the bond failure criteria. EQ.0.0: Defaults to 1.0. GT.0.0: Scale factor for the bending/twisting component in the bond failure criteria.
ALPHA	Numerical damping
PBNMAX	Maximum value of PBN. Used for random distribution.

VARIABLE	DESCRIPTION
PBSMAX	Maximum value of PBS. Used for random distribution.
PBN_SMAX	Maximum value of PBN_S. Used for random distribution.
PBS_SMAX	Maximum value of PBS_S. Used for random distribution.

Remarks:

1. **Normal force.** The normal force between two bonded discrete elements with radii r_1 and r_2 is calculated as

$$\Delta f_{\mathrm{n}} = \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_n \ ,$$

where

$$A = \pi r_{\mathrm{eff}}^2$$
$$r_{\mathrm{eff}} = \min(r_1, r_2) \times \mathrm{SFA}$$

For IRAND = 1, PBN is selected from the uniform distribution.

2. **Shear force.** The shear force is calculated as

$$\Delta f_{\mathrm{s}} = \mathrm{PBS} \times \frac{\mathrm{PBN}}{(r_1 + r_2)} \times \mathrm{A} \times \Delta u_s \ .$$

For IRAND = 1, PBN and PBS are selected from their uniform distributions.

3. **Maximum gap between bonded spheres.** The maximum gap between bonded spheres is inherited from MAXGAP on *DEFINE_DE_BOND and thus not specified with this keyword.

4. **BENDSF.** BENDSF enables removing or reducing the influence of twisting/bending on the bond failure criteria. The following equations determine bond failure:

$$\bar{\sigma}_{\max} = \frac{-\bar{F}_n}{A} + \mathrm{BENDSF} \times \frac{|\bar{M}_s| r_{\mathrm{eff}}}{I} \geq \mathrm{PBNS}$$
$$\bar{\tau}_{\max} = \frac{-\bar{F}_s}{A} + \mathrm{BENDSF} \times \frac{|\bar{M}_n| r_{\mathrm{eff}}}{J} \geq \mathrm{PBSS}$$

Here, the $\bar{}$ means average. I and J are the moment of inertia and polar moment of inertia. A and r_{eff} are as described in Remark 1.

*DEFINE_DE_BY_PART

Purpose: Define control parameters for discrete element spheres by part ID. This card overrides the values set in *CONTROL_DISCRETE_ELEMENT.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NDAMP	TDAMP	FRICS	FRICR	NORMK	SHEARK	
Type	I	F	F	F	F	F	F	
Default	none	0.	0.	0.	0.	0.01	2/7	

Card 2	1	2	3	4	5	6	7	8
Variable	GAMMA	VOL	ANG	HT_DE	INT_TMP	SPH_CP	TE_COEF	HT_CND
Type	F	F	F	I	F	F	F	F
Default	0.	0.	0.	0	0.	0.	0.	0.

Card 3 is optional.

Card 3	1	2	3	4	5	6	7	8
Variable	LNORM	LSHEAR		FRICD	DC			
Type	I	I		F	F			
Default	0.	0.		FRICS	0.			

VARIABLE	DESCRIPTION
PID	Part ID of DES nodes
NDAMP	Normal damping coefficient
TDAMP	Tangential damping coefficient

VARIABLE	DESCRIPTION
FRICS	Static coefficient of friction (see Remarks): EQ.0.0: 3 DOF NE.0.0: 6 DOF (consider rotational DOF)
FRICR	Rolling friction coefficient
NORMK	Optional scale factor of normal spring constant (Default = 0.01)
SHEARK	Optional ratio between SHEARK/NORMK (Default = 2/7)
GAMMA	Liquid surface tension
VOL	Volume fraction
ANG	Contact angle
HT_DE	Heat transfer for discrete spheres flag: EQ.0: Heat transfer between DES is not considered. EQ.1: Heat transfer between DES is considered.
INT_TMP	Initial temperature of the DES in K or °C
SPH_CP	Specific heat capacity of DES in J/(kg K)
TE_COEF	Thermal expansion coefficient of DES in K^{-1} or $°C^{-1}$
HT_CND	Thermal conductivity between DES in W/(m K)
LNORM	Load curve ID of a curve that defines normal stiffness as a function of the norm penetration ratio
LSHEAR	Load curve ID of a curve that defines shear stiffness as a function of the norm penetration ratio
FRICD	Dynamic coefficient of friction. By default, FRICD = FRICS. See Remarks.
DC	Exponential decay coefficient. See Remarks.

Remarks:

The friction coefficient is assumed to depend on the relative velocity v_{rel} of the two DEM in contact

$$\mu_c = \text{FRICD} + (\text{FRICS} - \text{FRICD})e^{-\text{DC}\times|v_{\text{rel}}|}.$$

*DEFINE_DE_COHESIVE

Purpose: Define a cohesive force mode for discrete element spheres. This keyword overrides the values set in *CONTROL_DISCRETE_ELEMENT.

Card 1	1	2						
Variable	SID	STYPE						
Type	I	I						
Default	0	0						

Card 2	1	2	3	4				
Variable	GAMMA	VOL	ANG	GAP				
Type	F	F	F	F				
Default	0.	0.	0.	0.				

VARIABLE	DESCRIPTION
SID	Node set ID, part set ID or part ID defining DES with cohesive force
STYPE	SID type: EQ.0: node set EQ.1: part set EQ.2: part
GAMMA	Liquid surface tension
VOL	Volume fraction
ANG	Contact angle
GAP	Spatial limit for the existence of liquid bridge between particles. A liquid bridge will exist when the distance between two particles is less or equal to $\min(\text{GAP}, d_{\text{rup}})$ where d_{rup} is the rupture distance

VARIABLE	DESCRIPTION
	of the bridge automatically calculated by LS-DYNA. EQ.0.0: maximum radius of DES NE.0.0: value of spatial limit

*DEFINE_DE_FLOW_DRAG

Purpose: Apply drag force to discrete element spheres (DES).

Card Summary:

Card 1. This card is required.

CD	RHO	MU	VX	VY	VZ	TBIRTH	TDEATH

Card 2. This card is required.

VS	DFLAG	SFN	SFS				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	CD	RHO	MU	VX	VY	VZ	TBIRTH	TDEATH
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10^{20}

VARIABLE	DESCRIPTION
CD	Drag coefficient, C_d EQ.-1: C_d determined based on Morrison 2013. See Remark 2. EQ.-2: C_d determined based on Allen 2018. See Remark 2. Card 1.1 must be included.
RHO	Flow density, ρ
MU	Dynamic viscosity
VX, VY, VZ	Flow velocity
TBIRTH	Birth time
TDEATH	Death time

Card 2	1	2	3	4	5	6	7	8
Variable	VS	DFLAG	SFN	SFS				
Type	F	I	F	F				
Default	0.0	1	1.0	1.0				

VARIABLE	DESCRIPTION
VS	Sound speed, V_s for CD = -2. See Remark 2.
DFLAG	Influence of neighbors on drag treatment: EQ.1: Consider only shadowing effect (default) EQ.2: See Remark 3. EQ.3: See Remark 3.
SFN, SFS	Scale factors for $C_{d1_\text{eff}}$and C_{d2_eff} respectively when DFLAG = 2.

Remarks:

1. **Drag Force.** The drag force is calculated as:

$$f_{\text{drag}} = \frac{1}{2} C_d \rho v^2 \frac{\pi D^2}{4} \quad .$$

Here C_d is the drag coefficient and D is the diameter of the DES.

2. **Drag Coefficient.** For CD = -1 and -2, the drag coefficient is found using predefined correlations.

If CD = -1, the drag coefficient is determined by the following correlation (Morrison 2013):

$$C_d = \frac{24}{\text{Re}} + \frac{2.6\left(\frac{\text{Re}}{5.0}\right)}{1+\left(\frac{\text{Re}}{5.0}\right)^{1.52}} + \frac{0.411\left(\frac{\text{Re}}{2.63\times 10^5}\right)^{-7.94}}{1+\left(\frac{\text{Re}}{2.63\times 10^5}\right)^{-8.00}} + \frac{0.25\frac{\text{Re}}{10^6}}{1+\frac{\text{Re}}{10^6}} \quad .$$

Re is the Reynolds number which for a sphere is defined as:

$$\text{Re} = \frac{\rho v D}{\mu} \quad .$$

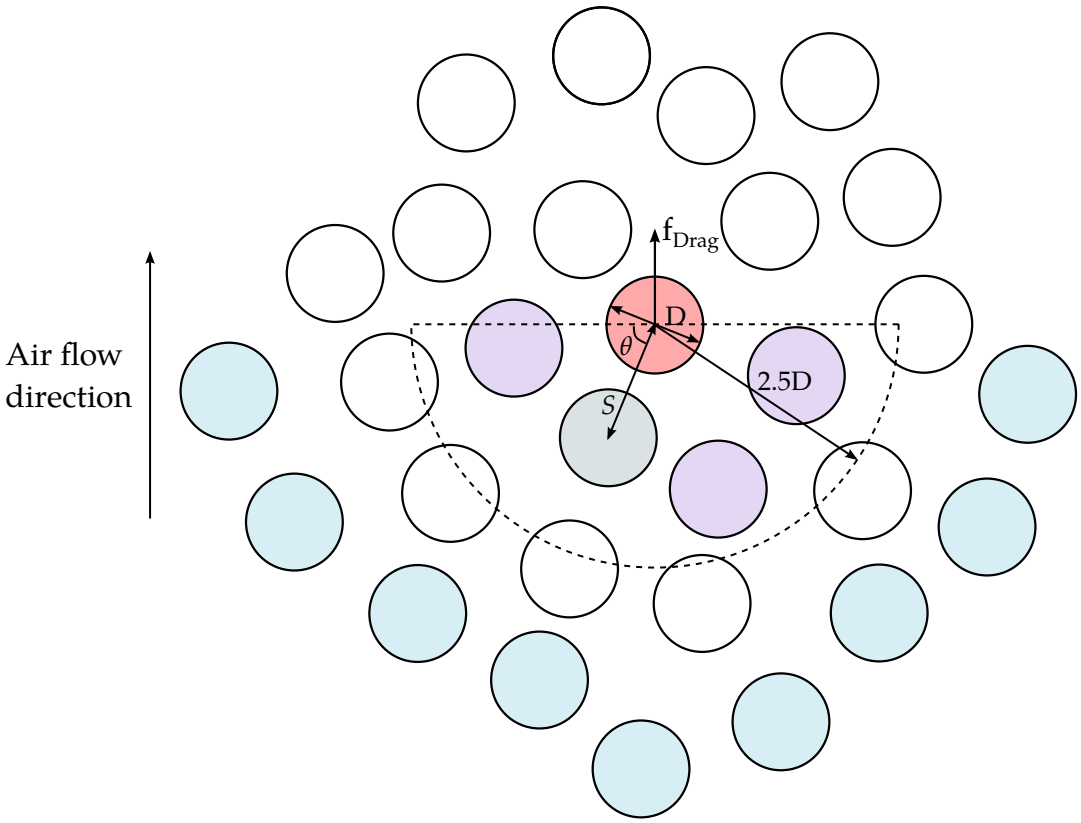


Figure 17-39. Schematic of how neighboring particles affect the drag treatment for DFLAG = 2 and 3. In this schematic the particles in blue are the leading particles that move at supersonic speed. We have developed two different two different methods for how the coefficient of drag is influenced by neighboring particles for particles that are not leading particles. In both methods, LS-DYNA checks for particles in the hemisphere in front of a given particle. In this schematic, the given particle is in red. The neighboring particles in the hemisphere are purple and grey. For DFLAG = 2, the grey particle is the closest neighbor of the red particle.

where ρ is the fluid density, v is the flow speed, D is the particle diameter, and μ is the dynamic viscosity of the fluid. C_d as obtained from this correlation is valid between $0.1 < \mathrm{Re} < 10^6$.

If CD = -2, the drag coefficient is determined by the following correlation (Allen 2018):

$$C_d = \begin{cases} 0.995 & \text{for} \quad M \geq 2.0 \\ 0.920 + 0.0375M & \text{for} \quad 1.2 \leq M < 2.0 \\ -0.163 + 0.9400M & \text{for} \quad 0.7 \leq M < 1.2 \\ 0.418 + 0.1100M & \text{for} \quad 0.2 \leq M < 0.7 \\ 0.440 & \text{for} \quad M < 0.2 \end{cases}$$

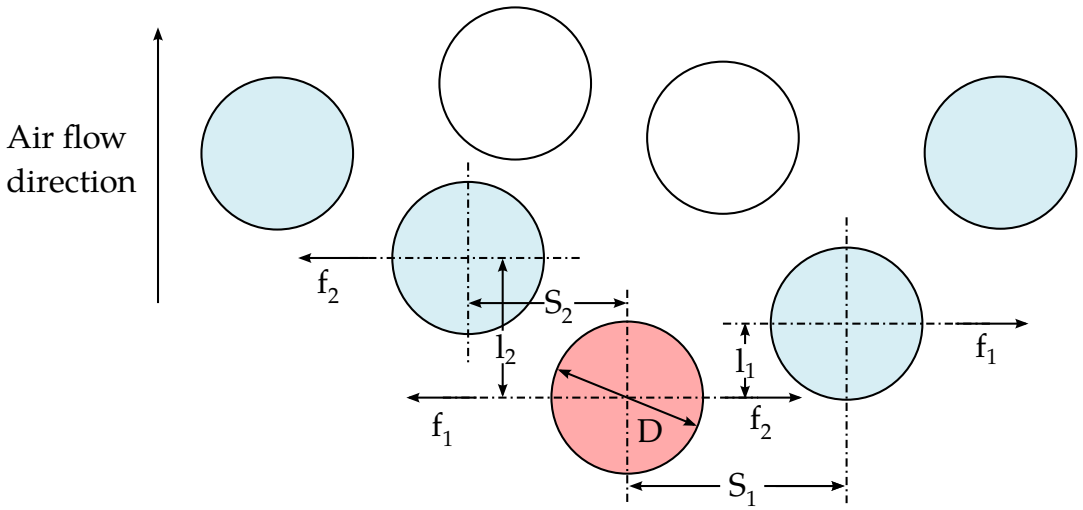


Figure 17-40. Illustration of perpendicular drag force in the lead particles for DFLAG = 2 and 3. In this schematic, the red particle interacts with both of its neighbors.

Here, M is the Mach number defined as:

$$M = \frac{V}{V_s},$$

where V is the pellet velocity and V_s is the sound speed.

3. **Influence of Neighbors on Drag Treatment.** The influence of neighboring particles on the drag treatment of a DES depends on the setting of DFLAG. By default, LS-DYNA only considers the shadowing effect (DFLAG = 1). We have implemented two other treatments.

   For DFLAG = 2 and 3, consider a group of particles as shown in Figure 17-39. The particles in blue are the leading particles in the flow. These particles travel with supersonic speed. They create a shock wave in front of them. Because of the shock wave, the leading particles may experience drag side forces perpendicular to the traveling direction from their neighboring leading particles (see Figure 17-40). For the particles in the layers behind the leading particles, their drag coefficient may be influenced by the particles in front of them. DFLAG = 2 and 3 are intended to model these two phenomena.

   For both DFLAG = 2 and 3, we model the perpendicular side force with the same equation. These side forces are only from leading particle interactions and thus only apply to leading particles. The side drag coefficient is given by:

$$\frac{C_{\text{dperp_eff}}}{C_d} = \left(7.20e^{-2.644\frac{S}{D}}\right)(1-\frac{l}{D}), for\ l \le D\ and\ S < 2.5D$$

C_d in this equation is the normal drag coefficient, and D is the diameter of the particle. l and S are as indicated in Figure 17-40, where l_1 and l_2 are examples of l, and S_1 and S_2 are examples of S.

For the drag force in the traveling direction, the coefficient of drag for the leading particles will be CD as specified in Card 1. The leading particles have the largest drag coefficient. The coefficient of drag for all the other particles are modified based on the value of DFLAG. For this modification, LS-DYNA looks at the hemisphere with radius $2.5D$ in front of a given particle in the flow as shown in Figure 17-39.

a) *DFLAG* = *2*. The coefficient of drag is first determined with the following equation:

$$\frac{C_{d\text{trav_eff}}}{C_d} = \left(1.0 + 4.644e^{-2.225\frac{S}{D}}\right)\cos\theta + \left(1 - e^{-0.5\frac{S}{D}}\right)\sin\theta\,, \text{ for } S < 2.5D$$

S is the distance from the center of the particle to its closest neighbor in the hemisphere. In Figure 17-39, the particle of interest is red and its closest neighbor is grey. θ is defined as shown in the figure. C_d is the coefficient given in CD. This coefficient is then scaled down by considering the number of particles, NHP, in the hemisphere to get the traveling direction coefficient of drag:

$$C_{\text{dtrav}} = \max(0, 1.0 - 0.02 \times \text{NHP})C_{\text{dtrav_eff}}$$

Note that if the particle has no neighboring particles, then the coefficient of drag is that given by Card 1.

b) *DFLAG* = *3*. The coefficient of drag is determined with the following equation (Numerical Investigation of Drag Forces on Particle Clouds in Non-Newtonian Flow, Efe Kinaci, Johannes Kepler University Linz, September 2015):

$$\frac{C_{\text{dtrav_eff}}}{C_d} = \alpha^{1-\chi}, \text{ for } S < 2.5D$$

Here,

$$C_d = \left(0.63 + \frac{4.8}{\text{Re}_i^{0.5}}\right)^2$$

$$\chi = 3.7 - 0.65e^{\frac{(1.5-\log_{10}\text{Re}_i)^2}{2}}$$

$$\text{Re}_i = \frac{\alpha\rho v D}{\mu}$$

In the above equations, α is the void fraction, ρ is the fluid density, v is the flow velocity, D is the diameter of the particle, and μ is dynamic viscosity of the fluid.

If there is no neighboring particle within 2.5*D*, the drag coefficient is based on the logic of CD = -1.

*DEFINE_DE_HBOND

Purpose: To define a heterogeneous bond model for discrete element sphere (DES). (This command, along with *INTERFACE_DE_HBOND, are no longer being developed.)

Card Summary:

Card 1. This card is required

SID	STYPE	HBDFM	IDIM				

Card 1.1. This card is included if and only if HBDFM = 2.

PBK_SF	PBS_SF	FRGK	FRGS	BONDR	ALPHA	DMG	FRMDL

Card 2. This card is optional.

PRECRK	CKTYPE		ITFID				

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	HBDFM	IDIM				
Type	I	I	I	I				
Default	none	0	1	3				

VARIABLE	DESCRIPTION
SID	DES nodes
STYPE	Set type for SID: EQ.0: DES node set EQ.1: DES node EQ.2: DES part set EQ.3: DES part

VARIABLE	DESCRIPTION
HBDFM	Bond formulation: EQ.1: (Reserved) EQ.2: Nonlinear heterogeneous bond formulation for fracture analysis based on the general material models defined in the material cards. DES elements with different material models can be defined within one bond.
IDIM	Space dimension for DES bonds: EQ.2: For 2D plane strain problems EQ.3: For 3D problems

Nonlinear Heterogeneous Bond Card. Include this card if and only if HBDFM = 2.

Card 1.1	1	2	3	4	5	6	7	8
Variable	PBK_SF	PBS_SF	FRGK	FRGS	BONDR	ALPHA	DMG	FRMDL
Type	F	F	F	F	F	F	F	I
Default	1.0	1.0	none	none	none	0.0	1.0	1

VARIABLE	DESCRIPTION
PBK_SF	Scale factor for volumetric stiffness of the bond.
PBS_SF	Scale factor for shear stiffness of the bond.
FRGK	Critical fracture energy release rate for volumetric deformation due to the hydrostatic pressure. EQ.0: A zero value specifies an infinite energy release rate for unbreakable bonds. LT.0: A negative value defines the energy release rate under volumetric compression (i.e. positive pressure) and FRGS defined below is used under volumetric expansion (i.e. negative pressure).
FRGS	Critical fracture energy release rate for shear deformation. EQ.0: A zero value specifies an infinite energy release rate

VARIABLE	DESCRIPTION
	for unbreakable bonds. FRGK.LT.0: See description for FRGK.
BONDR	Influence radius of the DES nodes.
ALPHA	Numerical damping
DMG	Continuous parameter for damage model. EQ.1.0: The bond breaks if the fracture energy in the bond reaches the critical value. Microdamage is not calculated. GT.0.5.and.LT.1.0: Microdamage effects being once the fracture energy reaches DMG × FMG[K,S]. Upon the onset of microdamage, the computed damage ratio will increase (monotonically) as the fracture energy grows. Bond weakening from microdamage is modeled by reducing the bond stiffness in proportion to the damage ratio.
FRMDL	Fracture model: EQ.1: Fracture energy of shear deformation is calculated based on deviatoric stresses. EQ.2: Fracture energy of shear deformation is calculated based on deviatoric stresses, *excluding the axial component (along the bond).* EQ.3,4: Same as 1&2, respectively, but FRGK and FRGS are read as the total failure energy density and will be converted to the corresponding critical fracture energy release rate. The total failure energy density is calculated as the total area under uniaxial tension stress-strain curve. EQ.5,6: Same as 3&4, respectively, as FRGK and FRGS are read as the total failure energy density but will not be converted. Instead, the failure energy within the bond will be calculated. Models 1&2 are more suitable for brittle materials, and Models 5&6 are easier for ductile materials. Models 3&4 can be used for moderately ductile fracture accordingly. This is the default fracture model applied to all DES parts, even if they have different material models. More fracture models can be

VARIABLE	DESCRIPTION
	defined for different materials by specifying an interface ID (ITFID) in the optional card.

Pre-Crack Card. This card is optional.

Card 2	1	2	3	4	5	6	7	8
Variable	PRECRK	CKTYPE		ITFID				
Type	I	I		I				
Default	none	0		0				

VARIABLE	DESCRIPTION
PRECRK	Shell set, define 3D surfaces of the pre-crack
CKTYPE	Type of PRECRK: EQ.0: Part set EQ.1: Part
ITFID	ID of the interface *INTERFACE_DE_HBOND, which defines different failure models for the heterogeneous bonds within each part and between two parts, respectively.

*DEFINE_DE_INJECT_BONDED_{OPTION}

Available options include:

<BLANK>

ELLIPSE

Purpose: This keyword injects bonded discrete element spheres (DES) from a specified region at a flow rate given by a user defined curve. Like *DEFINE_DE_INJECTION, when the option is blank, the region from which the DES emanate is assumed rectangular; the ELLIPSE option indicates that the region is elliptical. The first two cards are very similar to *DEFINE_DE_INJECTION. In contrast to *DEFINE_DE_INJECTION, this keyword contains additional cards to inject the bonded patterns. The bond properties of Card 3 are the same as the properties of Card 2 in *DEFINE_DE_BOND. In conjunction with the keyword *DEFINE_DE_INJECT_-SHAPE, this keyword randomly selects different shape patterns and injects them at the mass flow rate (RMASS) defined in Card 2.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	SID	XC	YC	ZC	XL	YL	CID
Type	I	I	F	F	F	F	F	I
Default	none	none	0.0	0.0	0.0	0.0	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	RMASS	VX	VY	VZ	TBEG	TEND		
Type	F	F	F	F	F	F		
Default	none	0.0	0.0	0.0	0.0	10^{20}		

Card 3	1	2	3	4	5	6	7	8
Variable	PBN	PBS	PBN_S	PBS_S	SFA	ALPHA	MAXGAP	
Type	F	F	F	F	F	F	F	
Default	none	none	0.0	0.0	1.0	0.0	10^{-4}	

Card 4	1							
Variable	NSHAPE							
Type	I							
Default	0							

Bonded Shape Patterns. Additional cards for NSHAPE ≠ 0. Define NSHAPE patterns, requiring ceil(NSHAPE/8) cards.

Card 5	1	2	3	4	5	6	7	8
Variable	ISHAPE	ISHAPE	ISHAPE	ISHAPE	ISHAPE	ISHAPE	ISHAPE	ISHAPE
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
PID	Part ID of new generated DES nodes
SID	Node set ID. Nodes and DES properties are generated automatically during input phase based on the user input and assigned to this SID.
XC, YC, ZC	x, y, z coordinates of the center of injection plane

VARIABLE	DESCRIPTION
XL	For rectangular planes XL specifies the planar length along the x-axis in the coordinate system specified by CID. For elliptical planes XL specifies the length of the major axis.
YL	For rectangular planes YL specifies the planar length along the y-axis in the coordinate system specified by CID. For elliptical planes YL specifies the length of the minor axis.
CID	Optional local coordinate system ID, see *DEFINE_COORDINATE_SYSTEM
RMASS	Mass flow rate GE.0.0: Constant mass flow rate LT.0.0: |RMASS| is a curve ID defining the mass flow rate as a function of time.
VX, VY, VZ	Vector components defining the initial velocity of injected DES specified relative the coordinate system defined by CID.
TBEG	Birth time; time for injection to begin
TEND	Death time; time for injection to end
PBN	Parallel-bond modulus [Pa].
PBS	Parallel-bond stiffness ratio. Shear stiffness/normal stiffness.
PBN_S	Parallel-bond maximum normal stress. A zero value defines an infinite maximum normal stress.
PBS_S	Parallel-bond maximum shear stress. A zero value defines an infinite maximum shear stress.
SFA	Bond radius multiplier.
ALPHA	Numerical damping.
MAXGAP	Maximum gap between two bonded spheres GT.0.0: When MAXGAP is positive, the maximum allowed gap is determined on a bond-by-bond basis as a function of the radii of the two involved spheres. The maximum gap is determined by multiplying the minimum of the two radii by the value of MAXGAP.

VARIABLE	DESCRIPTION
	LT.0.0: Absolute value is used as the maximum gap.
NSHAPE	Number of shape patterns.
ISHAPE	The pattern ID defined in *DEFINE_DE_INJECT_SHAPE. Only the first NSHAPE number of IDs will be used.

*DEFINE_DE_INJECT_SHAPE

Purpose: Define the bonded shape patterns. When used with *DEFINE_DE_INJECT_-BONDED, these shape patterns will be injected with equal probability. Each pattern contains several discrete elements, and the pattern can be in any direction.

There are two options to specify a desired bonded injection pattern:

1. Give the position and radius of each DE in the pattern (IAUTO = 0). The relative positions and radii of the DEs should be carefully defined to generate bonds between the neighboring DEs; see the field MAXGAP in *DEFINE_DE_INJECTED_BONDED.

2. Select the shape from predefined pattern types (IAUTO = 1). This method requires the length of the desired shape in all directions and a particle radius. All the particles will have the same radius. The patterns are restricted to lines, prisms with equilateral triangle faces, and cuboids.

Card Summary:

Card Sets: Include as many sets of Cards 1 and 2a/b as desired. Each set defines one shape pattern. This input terminates at the next keyword ("*") card.

Card 1. This card is required.

ID	NDE	IAUTO	ITYPE				

Card 2a. If IAUTO = 0, include NDE of this card, one for each DE in the pattern.

X	Y	Z	R				

Card 2b. Include this card if IAUTO = 1.

LX	LY	LZ	R				

Data Card Definitions:

Card 1	1	2	3	4				
Variable	ID	NDE	IAUTO	ITYPE				
Type	I	I	I	I				
Default	none	none	0	0				

VARIABLE	DESCRIPTION
ID	ID of the shape pattern
NDE	Number of DEs in this pattern
IAUTO	Flag for how to specify the bonded shape patterns: EQ.0: Give each particle's relative position and radius EQ.1: Use predefined pattern types
ITYPE	Bond particles patterns when IAUTO = 1: EQ.1: Line EQ.2: Cuboid EQ.3: Prism with equilateral triangle faces

Card 2 for IAUTO = 0. Include one card for each DE in the pattern (NDE total cards for each Card 1).

Card 2a	1	2	3	4				
Variable	X	Y	Z	R				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
X, Y, Z	Coordinates of the DEs in this pattern
R	Radii of the DEs in this pattern

Card 2 for IAUTO = 1. Include this card to define the length of the bonded pattern as well as the radius for each DE in the pattern.

Card 2b	1	2	3	4				
Variable	LX	LY	LZ	R				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
LX, LY, LZ	Length of bonded pattern in *X*, *Y*, and *Z*-directions
R	Radius of each DE in this pattern. All DEs in the pattern have the same radius.

*DEFINE_DE_INJECTION_{OPTION}

Available options include:

<BLANK>

ELLIPSE

Purpose: This keyword injects discrete element spheres (DES) from specified a region at a flow rate given by a user defined curve. When the option is blank, the region from which the DES emanate is assumed rectangular. The ELLIPSE keyword option indicates that the region is to be elliptical.

Card Summary:

Card 1. This card is required.

PID	SID	XC	YC	ZC	XL	YL	CID

Card 2. This card is required.

RMASS	RMIN	RMAX	VX	VY	VZ	TBEG	TEND

Card 3. This card is optional.

IFUNC	NID	IMULTI	LCVX	LCVY	LCVZ	IRAND	

Card 3.1. Include this card when IMULTI > 1.

R1	P1	R2	P2	R3	P3	R4	P4

Card 1	1	2	3	4	5	6	7	8
Variable	PID	SID	XC	YC	ZC	XL	YL	CID
Type	I	I	F	F	F	F	F	I
Default	none	none	0.0	0.0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
PID	Part ID of generated DES nodes

VARIABLE	DESCRIPTION
SID	Node set ID. Nodes and DES properties are generated automatically during input phase based on the user input and assigned to this SID.
XC, YC, ZC	x, y, z coordinate of the center of injection plane
XL	For rectangular planes XL specifies the planar length along the x-axis in the coordinate system specified by CID. For elliptical planes XL specifies the length of the major axis.
YL	For rectangular planes YL specifies the planar length along the y-axis in the coordinate system specified by CID. For elliptical planes YL specifies the length of the minor axis.
CID	Optional local coordinate system ID; see *DEFINE_COORDINATE_SYSTEM or *DEFINE_COORDINATE_NOTES

Card 2	1	2	3	4	5	6	7	8
Variable	RMASS	RMIN	RMAX	VX	VY	VZ	TBEG	TEND
Type	F	F	F	F	F	F	F	F
Default	none	none	RMIN	0.0	0.0	0.0	0.0	10^{20}

VARIABLE	DESCRIPTION
RMASS	Mass flow rate (see Remark 2) LT.0: Curve ID
RMIN	Minimum DES radius (ignored if IMULTI > 1)
RMAX	Maximum DES radius (ignored if IMULTI > 1)
VX, VY, VZ	Vector components defining the initial velocity of injected DES specified relative the coordinate system defined by CID.
TBEG	Birth time
TEND	Death time

Optional card 3.

Card 3	1	2	3	4	5	6	7	8
Variable	IFUNC	NID	IMULTI	LCVX	LCVY	LCVZ	IRAND	
Type	I	I	I	I	I	I	I	
Default	0	0	1	0	0	0	0	

VARIABLE	DESCRIPTION
IFUNC	Distribution of particle radii (ignored if IMULTI > 1): EQ.0: Uniform distribution (default) EQ.1: Gaussian distribution (see Remark 1)
NID	An optional node ID. If defined, the center of injection plane follows the motion of this node.
IMULTI	Flag for giving a specified mass distribution of injected particles with given radii: EQ.1: Inject the particles with distribution IFUNC using the radii specified with RMIN and RMAX (default). GT.1: Inject particles with IMULTI different radii, R*i*, with each different size having a specified mass distribution, P*i*, given in Card 3.1. IMULTI cannot be greater than 4.
LCVX	Load curve defining initial injection velocity in the *x*-direction
LCVY	Load curve defining initial injection velocity in the *y*-direction
LCVZ	Load curve defining initial injection velocity in the *z*-direction
IRAND	Enable a more randomized injection patten with IRAND = 1.

Included when IMULTI > 1.

Card 3.1	1	2	3	4	5	6	7	8
Variable	R1	P1	R2	P2	R3	P3	R4	P4
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
R*i*	Injected particle radius. IMULTI radii may be specified.
P*i*	Mass percentage of injected particle with radius R*i*

Remarks:

1. **Gaussian Distribution of Particle Radii.** The distribution of particle radii for IFUNC = 1 follows a Gaussian distribution:

$$(r|r_0,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(r-r_0)^2}{2\sigma^2}} ,$$

where the mean radius is given by

$$r_0 = \frac{1}{2}(r_{\max} + r_{\min})$$

and the standard deviation is

$$\sigma = \frac{1}{2}(r_{\max} - r_{\min}) .$$

2. **Maximum Estimated Mass Flow Rate.** The maximum estimated mass flowrate injected through plane (XL,YL) is:

$$Q_{\max} = 0.9 \times \pi \times \rho \times \mathrm{XL} \times \mathrm{YL} \times \frac{1}{6}\mathrm{V}_{\max}$$

where

$$V_{\max} = \sqrt{\mathrm{VX}^2 + \mathrm{VY}^2 + \mathrm{VZ}^2}$$

If RMASS > $Q_{\max}$, LS-DYNA issues the warning message: "input mass flowrate may exceed the maximum estimated rate."

3. **Restrictions for Rebalancing.** Dynamic rebalancing does not work with injections. Thus, NCRB in *CONTROL_DISCRETE_ELEMENT should be set to zero

when using this feature. Instead, you can use *CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION for rebalancing purposes.

*DEFINE_DE_INTERNAL_SKIP

Purpose: DES defined in PID will ignore internal particle to particle interaction force.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	TYPE						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
PID	Part set ID or part ID. TYPE below indicates the ID type.
TYPE	PID type: EQ.0: Part set EQ.1: Part

*DEFINE_DE_MASSFLOW_PLANE

Purpose: Measure DES mass flow rate across a defined plane. See also the accompanying keyword *DATABASE_DEMASSFLOW which controls the output frequency.

Card 1	1	2	3	4	5	6	7	8
Variable	PRTCLSID	SURFSID	PTYPE	STYPE				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
PRTCLSID	Node set ID, node ID, part set ID or part ID specifying DES to be measured. PTYPE below indicates the ID type specified by PRTCLSID.
SURFSID	Part set ID or part ID defining the surface. STYPE below indicates the ID type specified by SURFSID.
PTYPE	PRTCLSID type: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
STYPE	SURFSID type: EQ.0: Part set EQ.1: Part

*DEFINE_DE_MESH_BEAM

Purpose: Generate and place discrete element sphere (DES) elements along the axis of beam elements. By default, the generated DES is treated as a shadow (inactive) element on the beam element. It only transfers forces between the ICFD element and beam element to make ICFD coupling easier.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	TYPE	NQUAD	DESPID	DESXID	NSID	RSF	IACTIVE
Type	I	I	I	I	I	I	F	I
Default	none	none	1	Rem 1	Rem 1	0	1.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	MASS	INERTIA	RADIUS					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
SID	Part or part set ID for the region of the mesh upon which the DES elements will be placed
TYPE	SID type: EQ.0: Part set ID EQ.1: Part ID
NQUAD	Number of equally spaced DES elements created along the axis of beam element (maximum NQUAD = 4)
DESPID	Part ID for generated DES elements
DESXID	Section ID for generated DES elements

VARIABLE	DESCRIPTION
NSID	Create a node set with ID NSID (see *SET_NODE) for the nodes generated by this keyword. By default, no node set is created.
RSF	Scale factor of DES radius
IACTIVE	Activate DES: EQ.0: DES is inactive and used as a shadow (default). EQ.1: DES is active.
MASS	DES Mass: GT.0: DES mass EQ.-1: The DES particle radius (r) is $0.5 \times$ Beam Length/NQUAD, the DES mass (m) is $4\pi\rho r^3/3$, and the moment of inertia is $2mr^2/5$. Input fields INERTIA and RADIUS are ignored.
INERTIA	Mass moment of inertia (ignored if MASS = -1)
RADIUS	Particle radius (ignored if MASS = -1)

Remarks:

1. **DESPID and DESXID.** The part ID and/or section ID will be generated by this card if they are not provided in the input.

*DEFINE_DE_MESH_SURFACE

Purpose: Generate and place discrete element sphere (DES) elements on the surface of shell elements. By default, the generated DES is treated as a shadow (inactive) element on the surface. It only transfers forces between the ICFD element and shell element to make ICFD coupling easier.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	TYPE	NQUAD	DESPID	DESXID	NSID	RSF	IACTIVE
Type	I	I	I	I	I	I	F	I
Default	none	none	1	Rem 1	Rem 1	0	1.0	0

VARIABLE	DESCRIPTION
SID	Part or part set ID for the region of the mesh upon which the DES elements will be placed
TYPE	SID type: EQ.0: Part set ID EQ.1: Part ID
NQUAD	Number of equally spaced DES elements created on a shell element in each local shell direction. For instance, NQUAD × NQUAD DES elements will be created on the surface a quad shell. (Maximum NQUAD = 4)
DESPID	Part ID for generated DES elements
DESXID	Section ID for generated DES elements
NSID	If defined, this card creates a node set with ID NSID (see *SET_-NODE) for the nodes generated by this card.
RSF	Scale factor for determining the DES radius: GE.0.0: Scale factor on the internally determined radius based on shell thickness. LT.0.0: DES radius is $0.5 \times \lvert RSF \rvert \times (\text{maximum diagonal length})/NQUAD$

VARIABLE	DESCRIPTION
	for rectangular segments and $0.5 \times \lvert RSF \rvert \times (\text{maximum side length})/NQUAD$ for triangular segments.
IACTIVE	Activate DES: EQ.0: DES is inactive and used as a shadow (default). EQ.1: DES is active.

Remarks:

1. **DESPID and DESXID.** Part ID and/or section ID will be generated by this card if they are not provided in the input.

*DEFINE_DE_MESH_VOLUME

Purpose: Generate and place discrete element sphere (DES) elements inside the volume of solid elements. By default, the generated DES is treated as a shadow (inactive) element in the volume. It only transfers forces between the ICFD element and solid element to make ICFD coupling easier.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	TYPE	NQUAD	DESPID	DESXID	NSID	RSF	IACTIVE
Type	I	I	I	I	I	I	F	I
Default	none	none	1	Rem 1	Rem 1	0	1.0	0

VARIABLE	DESCRIPTION
SID	Part or part set ID for the region of the mesh upon which the DES elements will be placed
TYPE	SID type: EQ.0: Part set ID EQ.1: Part ID
NQUAD	Number of equally spaced DES elements created on a shell element in each local shell direction. For instance, NQUAD × NQUAD DES elements will be created on the surface a quad shell. (Maximum NQUAD = 4)
DESPID	Part ID for generated DES elements
DESXID	Section ID for generated DES elements
NSID	If defined, this card creates a node set with ID NSID (see *SET_-NODE) for the nodes generated by this card.
RSF	Scale factor for determining the DES radius: GE.0.0: Scale factor on the internally determined radius based on shell thickness
IACTIVE	Activate DES: EQ.0: DES is inactive and used as a shadow (default).

VARIABLE	DESCRIPTION
	EQ.1: DES is active.

Remarks:

1. **DESPID and DESXID.** The part ID and/or section ID for the generated DES particles will be created by this keyword if they are not provided in the input.

*DEFINE_DE_PARTS_INTERACTION

Purpose: Provide contact parameters for the interaction between discrete element spheres that are in different parts or in the same part. This card overrides the values set in *CONTROL_DISCRETE_ELEMENT and *DEFINE_DE_BY_PART.

Card 1	1	2	3	4	5	6	7	8
Variable	PID1	PID2	NDAMP	TDAMP	FRICS	FRICR	NORMK	SHEARK
Type	I	I	F	F	F	F	F	F
Default	none	none	0.	0.	0.	0.	0.01	2/7

VARIABLE	DESCRIPTION
PID1	Part ID of DES nodes (see Remark 1)
PID2	Part ID of DES nodes
NDAMP	Normal damping coefficient
TDAMP	Tangential damping coefficient
FRICS	Static coefficient of friction (see Remark 2): EQ.0.0: 3 DOF NE.0.0: 6 DOF (consider rotational DOF)
FRICR	Rolling friction coefficient
NORMK	Optional scale factor of normal spring constant (Default = 0.01)
SHEARK	Optional ratio between SHEARK/NORMK (Default = 2/7)

Remarks:

1. **Interaction within a single part.** PID1 and PID2 can be the same part for providing contact parameters to specify interaction within a single part.

2. **Friction.** The friction coefficient is assumed to depend on the relative velocity v_{rel} of the two DEM in contact:

$$\mu_c = \text{FRICD} + (\text{FRICS} - \text{FRICD})e^{-\text{DC}\times|v_{\text{rel}}|}.$$

*DEFINE_DE_PATTERN_OUTPUT

Purpose: Output when DES cross specified planes and what their coordinates are when they cross the planes. This feature is for visualizing the pattern of shotgun pellets at various locations during post-processing. The data is output to dem_pattern. With this keyword, you specify the line to which the planes are orthogonal and where the plane is along the line from an origin point.

Card Summary:

Card 1. This card is required.

PID	PTYPE	XO	YO	ZO	XH	YH	ZH

Card 2. This card is required.

NSET							

Card 2.1. Include NSET/8 of this card.

DIST1	DIST2	DIST3	DIST4	DIST5	DIST6	DIST7	DIST8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PTYPE	XO	YO	ZO	XH	YH	ZH
Type	I	I	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PID	Part ID/part set ID of the DES
PTYPE	Type for PID: EQ.0: Part EQ.1: Part set
XO, YO, ZO	Coordinates of the origin from which the distance to the planes is measured along the vector (XH,YH,ZH) – (XO,YO,ZO)

VARIABLE	DESCRIPTION
XH, YH, ZH	Head of the direction to which the planes are orthogonal

Card 2	1	2	3	4	5	6	7	8
Variable	NSET							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
NSET	Number of planes for which patterns will be recorded

Include NSET/8 of this card to specify the distance of each plane from the origin

Card 2.1	1	2	3	4	5	6	7	8
Variable	DIST1	DIST2	DIST3	DIST4	DIST5	DIST6	DIST7	DIST8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
DIST1, . .	Distance of each plane from (XO, YO,ZO)

*DEFINE_DE_TEMP

Purpose: Change the radius of discrete element sphere (DES) particles due to a change in temperature. Note that *DEFINE_DE_TEMP does not transfer heat from particle to particle.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	LCID	SFT	ALPHA	INI_TEMP		
Type	I	I	I	F	F	F		
Default	none	0	0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
SID	Node set, part set, or part ID for which the temperature effect applies
STYPE	SID type: EQ.0: DES node set EQ.2: DES part set EQ.3: DES part
LCID	Load curve ID for the temperature as a function of time during the dynamic process
SFT	Scale factor on the temperature given with LCID
ALPHA	Coefficient of thermal expansion (1/°C or 1/°F), α
INI_TEMP	Initial temperature of the DES particles (°C or °F)

Remarks:

1. **Change in radius.** This feature changes the radius of a particle due to a change in temperature. The change in radius is given by:

$$\Delta r = r \times \alpha \times \Delta T$$

Here, Δr is the change in radius of the particle, α is the coefficient of thermal expansion, r is the radius of the particle, and ΔT is the change in temperature.

*DEFINE_DE_TO_BEAM_COUPLING

Purpose: To define a coupling interface between discrete element spheres (DES) and beams.

Card 1	1	2	3	4	5	6	7	8
Variable	DESID	BEAMID	DESTYP	BEAMTYP				
Type	I	I	I	I				
Default	0	0	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	FRICS	FRICD	DAMP	BSORT				
Type	F	F	F	I				
Default	0	0	0	100				

VARIABLE	DESCRIPTION
DESID	Node set ID, node ID, part set ID or part ID specifying the DES in the coupling. DESTYP below indicates the ID type.
BEAMID	Part set ID or part ID specifying the beams in the coupling. BEAMTYP below indicates the ID type.
DESTYP	Type for DESID: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
BEAMTYP	Type for BEAMID: EQ.0: Part set EQ.1: Part

VARIABLE	DESCRIPTION
FRICS	Friction coefficient
FRICD	Rolling friction coefficient
DAMP	Damping coefficient
BSORT	Number of cycles between bucket sorts

*DEFINE_DE_TO_CPG_COUPLING

Purpose: Define a coupling interface between discrete element spheres (DES) and a CPG airbag (*AIRBAG_CPG_ID).

Card 1	1	2	3	4	5	6	7	8
Variable	DESID	BAGID	DESTYP	DRAG	HTC	ICTYP		
Type	I	I	I	F	F	I		
Default	0	0	0	0.0	0.0	0		

VARIABLE	DESCRIPTION
DESID	Part set ID or part ID specifying the DES in the coupling. DESTYP below indicates the ID type.
BAGID	ID of the CPG airbag in the coupling (see BAGID on *AIRBAG_CPG)
DESTYP	Type for DESID: EQ.0: Part set EQ.1: Part
DRAG	Drag coefficient (C_D) used for the force exchange between DES and CPG
HTC	Heat transfer coefficient used for the thermal coupling between DES and CPG
ICTYP	Flag to indicate the coupling direction of the solvers (see Remark 1): EQ.0: Two-way coupling between CPG and DES. The coupling force and heat transfer rate between CPG and DES are passed to both solvers. EQ.1: One-way coupling. The coupling force and heat transfer rate are only passed to CPG. EQ.2: One-way coupling. The coupling force and heat transfer rate are only passed to DES.

Remarks:

1. **Coupling terms.** The force to be exchanged between CPG and DES is calculated as:

$$\mathbf{F}_{\text{cpg-des}} = C_{\text{D}}\left(\frac{1}{2}\rho_{\text{cpg}}\|\mathbf{V}_{\text{cpg}} - \mathbf{V}_{\text{des}}\|(\mathbf{V}_{\text{cpg}} - \mathbf{V}_{\text{des}})\text{Area}_{\text{des (cross-sectional area)}}\right)$$

Typically, $C_{\text{D}} = 1$ for spherical particles. The area in the force calculation is the cross-sectional area of the DES. For a sphere, this area is πr^2, where r is the radius of the sphere. The heat transfer that will be exchanged between CPG and DES is calculated as:

$$\dot{Q}_{\text{cpg-des}} = \text{HTC}(T_{\text{cpg}} - T_{\text{des}})\text{Area}_{\text{des (total surface area)}}$$

In the above equation, the area is the total surface area of the DES. For a sphere, this area is $4\pi r^2$.

*DEFINE_DE_TO_CPM_COUPLING

Purpose: Define a coupling interface between discrete element spheres (DES) and a CPM airbag (*AIRBAG_PARTICLE_ID).

Card 1	1	2	3	4	5	6	7	8
Variable	DESID	BAGID	DESTYP					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
DESID	Part set ID or part ID specifying the DES in the coupling. DESTYP below indicates the ID type.
BAGID	*AIRBAG_PARTICLE ID specifying the CPM airbag in the coupling
DESTYP	Type for DESID: EQ.0: Part set EQ.1: Part

*DEFINE_DE_TO_SURFACE_COUPLING_{OPTION}

Purpose: To define a non-tied coupling interface between discrete element spheres (DES) and a surface defined by a shell or solid part(s). This coupling is not currently implemented for thick shell parts.

Available options include:

<BLANK>

TRANSDUCER

The TRANSDUCER keyword option creates a force transducer to measure coupling interface forces. With this option, LS-DYNA outputs the translational forces from the active surface coupling interfaces to dem_rcforc. The *only* fields read when using this option are SURFID, SURFTYP, and CID_RCF. See Remark 5.

Card Summary:

Card 1. This card is required.

DESID	SURFID	DESTYP	SURFTYP	ISOFT			SBOX

Card 2. This card is required.

FRICS	FRICD	DAMP	BSORT	LCVX	LCVY	LCVZ	WEARC

Card 3. Include this card if WEARC < 0.

W1	W2	W3	W4	W5	W6	W7	W8

Card 4. This card is optional.

SFP	SFT			IDEACT	CID_RCF	BT	DT

Card 5. This card is optional.

HT_TRSF	TH_CND	SURF_HT					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	DESID	SURFID	DESTYP	SURFTYP	ISOFT			SBOX
Type	I	I	I	I	I			I
Default	none	none	0	0	0			0

VARIABLE	DESCRIPTION
DESID	Node set ID, node ID, part set ID, or part ID specifying DES in the coupling. DESTYP below indicates the ID type.
SURFID	Part set ID or part ID specifying the surface. SURFTYP below indicates the ID type.
DESTYP	DESID type: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
SURFTYP	SURFID type: EQ.0: Part set EQ.1: Part EQ.2: Segment set (TRANSDUCER keyword option only; see Remark 5)
ISOFT	Contact stiffness evaluation: EQ.0: Default is based on DES properties only. $K_a = K_{\text{des}}$ EQ.1: Geometric mean is based on both DES and contact segment's properties. $K_b = \sqrt{K_{des} \times \text{SFP} \times \text{bulk modulus of the surface}}$ EQ.2: Soft option is based on the mass of the system and the time

VARIABLE	DESCRIPTION
	step size. $K_c = \max(K_b, \frac{\text{SFP} \times \max(M_{\text{des}}, M_{n_1}, M_{n_2}, M_{n_3}, M_{n_4})}{dt^2})$
SBOX	BOX ID. Exclude segments belonging to SURFID that are outside of the box during initialization.

Card 2	1	2	3	4	5	6	7	8
Variable	FRICS	FRICD	DAMP	BSORT	LCVX	LCVY	LCVZ	WEARC
Type	F	F	F	I	I	I	I	F
Default	0.0	0.0	0.0	100	0	0	0	0.

VARIABLE	DESCRIPTION
FRICS	Friction coefficient
FRICD	Rolling friction coefficient
DAMP	Damping coefficient (unitless). If a discrete element sphere impacts a rigid surface with a velocity v_{impact}, the rebound velocity is $v_{\text{rebound}} = (1 - \text{DAMP})v_{\text{impact}}$
BSORT	Number of cycles between bucket sorts; the default value is 100. For blast simulations with very high DEM particle velocity, we suggest setting BSORT = 20 or smaller. LT.0: |BSORT| is the minimum number of cycles between bucket sorts. This value can be increased during runtime by tracking the velocity of potential coupling pairs. This feature only works with MPP currently.
LVCX	Load curve defining surface velocity in the x-direction
LVCY	Load curve defining surface velocity in the y-direction
LVCZ	Load curve defining surface velocity in the z-direction
WEARC	WEARC is the wear coefficient:

VARIABLE	DESCRIPTION
	GT.0: Archard's Wear Law; see Remark 1.
	EQ.-1: Finnie Wear Law; an additional card is required
	LE.-100: User-defined wear model; an additional card is required, see Remark 6.

User-Defined Wear Parameter Cards. Additional card for WEARC < 0.

Card 3	1	2	3	4	5	6	7	8
Variable	W1	W2	W3	W4	W5	W6	W7	W8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
W1-W8	For WEARC = -1, W1 is the yield stress of the target material. For WEARC ≤ -100, W*i* are user-defined wear parameters.

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	SFP	SFT			IDEACT	CID_RCF	BT	DT
Type	F	F			I	I	F	F
Default	1.0	1.0			0	0	0.0	10^{20}

VARIABLE	DESCRIPTION
SFP	Scale factor on contact stiffness. By default, SFP = 1.0. The contact stiffness is calculated as

$$K_n = \text{SFP} \times \begin{cases} K \times r \times \text{NormK} & \text{if NormK} > 0 \\ |\text{NormK}| & \text{if NormK} < 0 \end{cases}$$

where K is the bulk modulus, r is the discrete element radius, and NormK is the scale factor of the spring constant defined in *CON-

VARIABLE	DESCRIPTION
	TROL_DISCRETE_ELEMENT.
SFT	Scale factor for surface thickness (scales true thickness). True thickness is the element thickness of the shell elements. This option applies only to contact with shell elements.
IDEACT	DES particles will be automatically deactivated after contacting the surface when IDEACT = 1.
CID_RCF	Coordinate system ID. Force resultants are output to the demforc file in a local system.
BT	Birth time
DT	Death time

This card is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	HT_TRSF	TH_CND	SURF_HT					
Type	I	F	F					
Default	0	0.	0.					

VARIABLE	DESCRIPTION
HT_TRSF	Flag to enable heat transfer between DES and the surface: EQ.0: No heat transfer between DES and the surface EQ.1: Consider heat transfer between DES and the surface
TH_CND	Thermal conductivity between DES and the surface in W/(m K)
SURF_HT	Constant surface temperature in K or °C

Remarks:

1. **Archard's wear law.** If WEARC > 0, then the wear on the shell surface is calculated using Archard's wear law

$$\dot{h} = \frac{\text{WEARC} \times f_n \times v_t}{A}$$

where,

h = wear depth
f_n = normal contact force from DE
v_t = tangential sliding velocity of the DE on shell
A = area of contact segment

The wear depth is output to the interface force file.

2. **Finnie's wear law.** If WEARC = -1, Finnie's wear law determines the wear on the shell surface. The model of Finnie relates the rate of wear to the rate of kinetic energy of particle impact on a surface as:

$$Q = \begin{cases} \frac{mv^2}{8p}\left(\sin 2\alpha - 3\sin^2\alpha\right) & \text{if } \tan\alpha < \frac{1}{3} \\ \frac{mv^2}{24p}\cos^2\alpha & \text{if } \tan\alpha > \frac{1}{3} \end{cases}$$

where Q is the volume of the material removed from the surface, m is particle mass, α is the impact angle, and p is the yield stress of the target material input in field W1. The wear depth is output to the interface force file.

3. **Interface force file.** *DATABASE_BINARY_DEMFOR controls the output interval of the coupling forces to the DEM interface force file. This interface force file is activated by the command line option "dem=", for example,

lsdyna i=**inputfilename.k** … dem=**interfaceforce_filename**

The DEM interface force file can be read into LS-PrePost to plot coupling pressure and forces on the surface segments.

4. **Coupling force output.** *DATABASE_RCFORC controls the output interval of the coupling forces to the ASCII demrcf file. This output file is analogous to the rcforc file for *CONTACT.

5. **TRANSDUCER.** The TRANSDUCER keyword option causes LS-DYNA to collect and record the translational forces from a section of the surface side of an active surface coupling interface. The collected forces are from surface nodes instead of surface segments. Therefore, the total force may be a little different from the parent interface. With this option, *SURFID, SURFTYP, and CID_RCF are the only input not ignored*.

6. **User-defined wear model.** When WEARC≤ −100, subroutine `userdewear` in file dyn21cnt.f of the usermat library defines the wear model; see comments within the subroutine for instructions. LS-DYNA passes parameters W1-W8 to

the subroutine. Setting $-108 \leq$WEARC< -100 activates passing information between time steps with history variables. A maximum of eight history variables can be defined. The $\min(|\text{WEARC}| - 100, 8)$ gives the number of available history variables. The interface force file contains these history variables for post-processing.

*DEFINE_DE_TO_SURFACE_TIED

Purpose: To define a tied-with-failure coupling interface between discrete element spheres (DES) and a surface defined by shell part(s) or solid part(s). This coupling is currently not implemented for tshell part(s).

Card 1	1	2	3	4	5	6	7	8
Variable	DESID	SURFID	DESTYP	SURFTYP				
Type	I	I	I	I				
Default	0	0	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	NFLF	SFLF	NEN	MES	LCID	NSORT	MAXGAP	
Type	F	F	F	F	I	I	F	
Default	none	none	2.	2.	0	100	0.0	

VARIABLE	DESCRIPTION
DESID	Node set ID, node ID, part set ID or part ID specifying DES in the tied interface. DESTYP below indicates the ID type.
SURFID	Part set ID or part ID specifying the surface. SURFTYP below indicates the ID type.
DESTYP	DESID type: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
SURFTYP	SURFID type: EQ.0: Part set

VARIABLE	DESCRIPTION
	EQ.1: Part
NFLF	Normal failure force. Only tensile failure, that is, tensile normal forces, will be considered in the failure criterion
SFLF	Shear failure force
NEN	Exponent for normal force
MES	Exponent for shear force. Failure criterion: $\left(\frac{\lvert f_n \rvert}{\text{NFLF}}\right)^{\text{NEN}} + \left(\frac{\lvert f_s \rvert}{\text{SFLF}}\right)^{\text{MES}} \geq 1.$ Failure is assumed if the left side is larger than 1. f_n and f_s are the normal and shear interface force.
LCID	Optional curve ID defining a scale factor as a function of time. The scale factor is applied to NFLF and SFLF, making the failure forces time-dependent.
NSORT	Number of cycles between bucket sorts
MAXGAP	Determines maximum gap between DES and the surface: GT.0.0: The maximum gap is determined by scaling the DES radius by MAXGAP, that is, MAXGAP × r_{DES} LT.0.0: |MAXGAP| is used as the maximum gap.

Remarks:

Both NFLF and SFLF must be defined. If failure in only tension or shear is required, then set the other failure force to a large value (10^{10}).

DEFINE_DEATH_TIMES_*OPTION

Available options include:

NODES

SET

RIGID

Purpose: To dynamically define the death times for *BOUNDARY_PRESCRIBED_MOTION based on the locations of nodes and rigid bodies. Once a node or rigid body moves past a plane or a geometric entity, the death time is set to the current time. The input in this section continues until the next keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	GEO	N1	N2	N3				
Type	I	I	I	I				
Default	none	0	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	X_T	Y_T	Z_T	X_H	Y_H	Z_H	R	FLAG
Type	F	F	F	F	F	F	F	I
Default	Rem 1	Rem 1	Rem 1	Rem 2	Rem 2	Rem 2	none	1

ID Cards. Set the list of nodes and rigid bodies affected by this keyword. This input terminates at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	NSID1	NSID2	NSID3	NSID4	NSID5	NSID6	NSID7	NSID8
Type	I	I	I	I	I	I	I	I

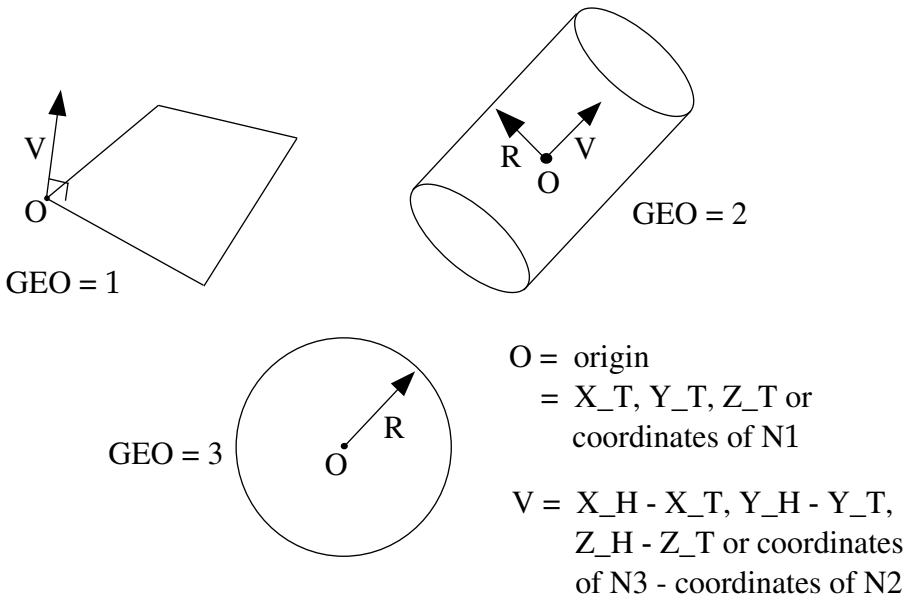


Figure 17-41. Geometry types.

VARIABLE	DESCRIPTION
GEO	Geometric entity type: EQ.1: Plane EQ.2: Infinite cylinder EQ.3: Sphere
N1	Node defining the origin of the geometric entity. See Remark 1.
N2	Node defining the tail of the orientation vector. See Remark 2.
N3	Node defining the head of the orientation vector. See Remark 2.
X_T	*x*-coordinate of the origin of the geometric entity and the tail of the orientation vector. See Remark 1.
Y_T	*y*-coordinate of the origin of the geometric entity and the tail of the orientation vector. See Remark 1.
Z_T	*z*-coordinate of the origin of the geometric entity and the tail of the orientation vector. See Remark 1.
X_H	*x*-coordinate of the head of the orientation vector. See Remark 2.
Y_H	*y*-coordinate of the head of the orientation vector. See Remark 2.

VARIABLE	DESCRIPTION
Z_H	z-coordinate of the head of the orientation vector. See Remark 2.
R	Radius of cylinder or sphere.
FLAG	Flag for where the node or rigid body must be with respect to the geometric entity for the death time to be set to the current time: EQ.1: The node or rigid body is outside of the geometric entity or on the positive side of the plane as defined by the normal direction. EQ.-1: Node or rigid body is inside the geometric entity or on the negative side of the plane.
NSIDi	i^{th} node, node set, or rigid body

Remarks:

1. **Origin.** Either N1 or X_T, Y_T, and Z_T should be specified, but not both.

2. **Orientation Vector.** Either N2 and N3 or X_H, Y_H, and Z_H should be specified, but not both. Specifying N2 and N3 is equivalent to setting the head of the vector equal to the tail of the vector (X_T,Y_T, Z_T) plus the vector from N2 to N3.

*DEFINE_DRIFT_REMOVE

Purpose: Minimize the drift of a structure over a loop by modifying the curves for specifying accelerations and forces.

Card 1	1	2	3	4	5	6	7	8
Variable	OPTION							
Type	I							
Default	none							

Curve Card. Include as many of this card as needed. Input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Specifies how curves are handled. EQ.0: no changes are made to the curves. EQ.1: modify to minimize the drift.
ID[N]	ID of Nth curve to be modified

Remarks:

When data from an accelerometer is integrated to obtain velocities and displacements numerical noise leads to drift. For signals that are necessarily periodic, like data collected as a car drives around a track, the resulting quadrature loses its periodicity. This keyword activates an algorithm that approximately restores periodicity.

Because its chief property is *not* being periodic we approximate noise, a_{noise}, by a function that is not periodic, namely, a parabola (or possibly a straight line)

$$a_{\text{noise}}(t) = a_0 + a_1 t + a_2 t^2 \ .$$

Since a_{noise} cannot be allowed in the context of periodicity we seek to find an $a_{\text{corrected}}$

$$a_{\text{corrected}} = a_{\text{data}} - a_{\text{noise}}$$

that removes as much "noise" as possible. We therefore find the a_i that minimize the norm of the corrected velocity,

$$\int_{t_0}^{t_1} [v_{\text{corrected}}(t)]^2 \text{dt} \ ,$$

where the corrected velocity is the quadrature of the corrected noise,

$$v_{\text{corrected}}(t) = \int_{t_0}^{t} [a_{\text{data}}(\tau) - a_{\text{noise}}(\tau)] \, \text{d}\tau = v_{\text{data}}(t) - \left(a_0 t + \frac{1}{2} a_1 t^2 + \frac{1}{3} a_2 t^3 \right).$$

For each curve listed on the data cards LS-DYNA approximately removes the drift by numerically solving the above minimization problem.

DEFINE_ELEMENT_DEATH_*OPTION

Available options include:

SOLID

SOLID_SET

BEAM

BEAM_SET

SHELL

SHELL_SET

THICK_SHELL

THICK_SHELL_SET

Purpose: Set a time or space condition for deleting an element or element set during a simulation. This keyword is only for deformable elements, not rigid body elements.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID	TIME	BOXID	INOUT	IDGRP	CID	PERCENT	
Type	I	F	I	I	I	I	F	
Default	none	0.0	{∅}	0	0	global	0.0	

VARIABLE	DESCRIPTION
EID/SID	Element ID or element set ID identifying elements that are considered for deletion, either by meeting the BOXID/INOUT criterion or the independent TIME/IDGRP/PERCENT criterion.
TIME	Deletion time for elimination of the element or element set. If BOXID is nonzero, a TIME value of zero is reset to 10^{16}.
BOXID	An optional box ID; see CID below and *DEFINE_BOX. An element is immediately deleted upon meeting the condition of being inside the box (or outside the box, depending on INOUT), without regard to TIME, IDGRP, or PERCENT. For an element set, only the element (or elements) in the set meeting the condition is deleted at

VARIABLE	DESCRIPTION
	each time step. Before R11, the elements in a single IDGRP were all deleted when one element in the group met the condition despite the value of PERCENT. As of R11, all the elements in the group are deleted when PERCENT of the elements in the group meet the condition.
INOUT	Flag pertaining to BOXID: EQ.0: Elements inside the box EQ.1: Elements outside the box
IDGRP	Group ID. Elements sharing the same positive value of IDGRP are considered to be in the same group. All elements in a group will be simultaneously deleted one cycle after a percentage of the elements (specified in PERCENT) fail. There is no requirement that each *DEFINE_ELEMENT_DEATH command have a unique IDGRP. In other words, elements in a single group can come from multiple *DEFINE_ELEMENT_-DEATH commands. Elements in which IDGRP = 0 are not assigned to a group and thus deletion of one element does not cause deletion of the other elements.
CID	Coordinate ID for transforming box BOXID. If CID is not specified, the box is in the global coordinate system. The box rotates and translates with the coordinate system only if the coordinate system is flagged for an update every time step.
PERCENT	Deletion percentage. EQ.0.0: When one element fails, all elements in the group will be deleted (default). GT.0.0: Percentage of elements failed before elements in group IDGRP are deleted.

DEFINE_ELEMENT_EROSION_*OPTION

Available options include:

IGA

SHELL

TSHELL

Purpose: Defines the number of failed integration points per layer that fails a layer and the number of failed layers that triggers element deletion (erosion) during the simulation. This keyword is only for deformable shells and deformable thick shells. This keyword must be used in conjunction with a material model that has a failure option, or else with *MAT_ADD_EROSION. The criteria specified on this card will override other criteria related to the number of failed integration points for element deletion, such as NUMFIP on *MAT_ADD_EROSION or NUMINT on other *MAT cards.

Either the SHELL option or the TSHELL option is required, except for in the case of isogeometric analysis. Isogeometric analysis requires the IGA option and currently only supports isogeometric shell elements for STYP = 3 or 4.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYP	NUMFIP	NIPF				
Type	I	I	F	I				
Default	none	none	-100.	1				

VARIABLE	DESCRIPTION
SID	Element ID, element set ID, part ID, or part set ID; see *PART, *SET_PART or *SET_T/SHELL_*OPTION*.
STYP	ID type of SID: EQ.1: Shell or tshell element ID EQ.2: Shell or tshell element set ID EQ.3: Part ID EQ.4: Part set ID

VARIABLE	DESCRIPTION
NUMFIP	Number of layers which must fail prior to element deletion. LT.0.0: |NUMFIP| is the percentage of layers which must fail prior to element deletion.
NIFP	Number of integration points within one layer that need to fail to indicate a failed layer

*DEFINE_ELEMENT_GENERALIZED_SHELL

Purpose: Define a general 3D shell formulation to be used in combination with *ELEMENT_GENERALIZED_SHELL. The objective of this feature is to allow the rapid prototyping of new shell element formulations by adding them through the keyword input file.

All necessary information, like the values of the shape functions and their derivatives at various locations (at the integration points and at the nodal points), must be defined using this keyword. An example for a 9-noded generalized shell element with 4 integration points in the plane is given in Figure 17-42 to illustrate the procedure. The element formulation ID (called ELFORM) used in this keyword needs to be greater or equal than 1000 and will be referenced through *SECTION_SHELL (see Figure 19-4 in *ELEMENT_GENERALIZED_SHELL).

Card 1	1	2	3	4	5	6	7	8
Variable	ELFORM	NIPP	NMNP	IMASS	FORM			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

Weights and Shape Function Values/Derivatives at Gauss Points:

These cards are read according to the following pseudo code:

```
for i = 1 to NIPP {
        read Card 2(i)
        for k = 1 to NMNP {
                read Card 3(i,k)
        }
} // comment: Read in NIPP × (1 + NMNP) cards
```

Weight Cards. Provide weight for integration point *i*.

(Card 2)$_i$	1	2	3	4	5	6	7	8
Variable	WI							
Type	F							

Integration Point Shape Function Value/Derivatives Cards. Provide the value of the k^{th} shape function and its derivative at the i^{th} integration point.

(Card 3)$_{ik}$	1	2	3	4	5	6	7	8
Variable	NKI		DNKIDR		DNKIDS			
Type	F		F		F			

For FORM = 0 or FORM = 1, Shape Function Derivatives at Nodes:

These cards are read according to the following pseudo code:

```
for l = 1 to NMNP {
        for k = 1 to NMNP {
                read Card 4a(l,k)
        }
} // comment: Read in NMNP × NMNP cards
```

Nodal Shape Function Derivative Cards. The value of the k^{th} shape function's derivative at the l^{th} nodal point.

(Card 4a)$_{lk}$	1	2	3	4	5	6	7	8
Variable	DNKLDR		DNKLDS					
Type	F		F					

For FORM = 2 or FORM = 3, Shape Function 2nd derivative at Gauss Points:

> NOTE: For FORM = 2 and FORM = 3 it is assumed that the shape functions are at least C1 continuous (having a continuous derivative).

The cards for this method are read according to the following pseudo code:

```
for i = 1 to NIPP {
        for k = 1 to NMNP {
                read Card 4b(i,k)
        }
} // comment: Read in NGP × NMNP cards
```

Nodal Shape Function Second Derivative Cards. The value of the k^{th} shape function's second derivative at the i^{th} integration point.

(Card 4b)*ik*	1	2	3	4	5	6	7	8
Variable	D2NKIDR2		D2NKIDRDS		D2NKIDS2			
Type	F		F		F			

VARIABLE	DESCRIPTION
ELFORM	Element Formulation ID which is referenced by *SECTION_SHELL to connect *ELEMENT_GENERALIZED_SHELL with the appropriate shell formulation. The chosen number needs to be greater or equal than 1000.
NIPP	Number of in-plane integration points.
NMNP	Number of nodes for this element formulation.
IMASS	Option for lumping of mass matrix: EQ.0: row sum EQ.1: diagonal weighting.
FORM	Shell formulation to be used EQ.0: shear deformable shell theory with rotational DOFs (shell normal evaluated at the nodes) EQ.1: shear deformable shell theory without rotational DOFs (shell normal evaluated at the nodes) EQ.2: thin shell theory without rotational DOFs (shell normal evaluated at the integration points) EQ.3: thin shell theory with rotational DOFs (shell normal evaluated at the integration points)
WI	Integration weight at integration point *i*.

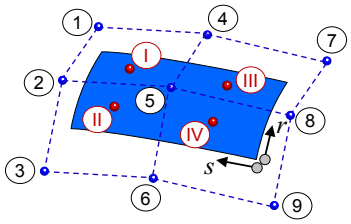
VARIABLE	DESCRIPTION
NKI	Value of the shape function N_k evaluated at integration point i.
DNKIDR	Value of the derivative of the shape function N_k with respect to the local coordinate r at the integration point i $\left(\frac{\partial N_k^i}{\partial r}\right)$.
DNKIDS	Value of the derivative of the shape function N_k with respect to the local coordinate s at the integration point i $\left(\frac{\partial N_k^i}{\partial s}\right)$.
DNKLDR	Value of the derivative of the shape function N_k with respect to the local coordinate r at the nodal point l $\left(\frac{\partial N_k^l}{\partial r}\right)$.
DNKLDS	Value of the derivative of the shape function N_k with respect to the local coordinate s at the nodal point l $\left(\frac{\partial N_k^l}{\partial s}\right)$.
D2NKIDR2	Value of the second derivative of the shape function N_k with respect to the local coordinate r at the integration point i $\left(\frac{\partial^2 N_k^i}{\partial r^2}\right)$.
D2NKIDRDS	Value of the second derivative of the shape function N_k with respect to the local coordinates r and s at the integration point i $\left(\frac{\partial^2 N_k^i}{\partial r \partial s}\right)$.
D2NKIDS2	Value of the second derivative of the shape function N_k with respect to the local coordinate s at the integration point i $\left(\frac{\partial^2 N_k^i}{\partial s^2}\right)$.

Remarks:

1. **Interpolation Shell Elements.** For post-processing and the treatment of contact boundary conditions, the use of interpolation shell elements (see *ELEMENT_-INTERPOLATION_SHELL and *CONSTRAINED_NODE_INTERPOLATION) is necessary.

2. **Input Order.** Data input order for the NMNP nodal points must match the definition of the connectivity of the element in *ELEMENT_GENERALIZED_-SHELL.

Example:




```
*DEFINE_ELEMENT_GENERALIZED_SHELL
$#  elform      nipp      nmnp     imass      form
      1001         4         9         0         1
$# integration point 1 (i=1)
$#                wi
 1.3778659577546E-04
$#               nki              dnkidr              dnkids
 1.7098997698601E-01 3.3723996630918E+00 2.4666694616947E+00
...
$# integration point 2 (i=2)
 2.2045855324077E-04
 5.4296436772101E-02 1.9003752917745E+00 7.8327025592051E+00
...
$# integration point 3 (i=3)
...
$# integration point 4 (i=4)
...
$# node 1 (l=1)
$#            dnkldr              dnklds
 4.8275862102259E+00 3.5310344763662E+01
...
$# node 2 (l=2)
 2.4137931051130E+00 8.8275861909156E+00
...
[...]
$# node 9 (l=9)
...
```

W1
k=1
k=2-9
W2
NMNP Lines
1 (W3)+ NMNP Lines
1 (W4)+ NMNP Lines
k=1
k=2-9
NMNP Lines
NMNP Lines

Block A

Block B

Figure 17-42. Example of a generalized shell formulation with *DEFINE_ELEMENT_GENERALIZED SHELL.

*DEFINE_ELEMENT_GENERALIZED_SOLID

Purpose: Define a general 3D solid formulation to be used in combination with *ELEMENT_GENERALIZED_SOLID. New solid element formulations may be rapidly prototyped in the input deck using this feature.

All necessary information, like the values of the shape functions and their derivatives at all integration points, must be defined using this keyword. The example shown in Figure 17-43 illustrates this procedure by prototyping an 18-noded generalized solid element with 8 integration points. The element formulation ID (called ELFORM) used in this keyword needs to be greater or equal than 1000 and will be referenced through *SECTION_SOLID (see Figure 19-5 in *ELEMENT_GENERALIZED_SOLID).

Card 1	1	2	3	4	5	6	7	8
Variable	ELFORM	NIP	NMNP	IMASS				
Type	I	I	I	I				
Default	none	none	none	none				

Weights and Shape Function Values/Derivative at Gauss Points:

These cards are read according to the following pseudo code:

```
for i = 1 to NIP {
        read Card 2(i)
        for k = 1 to NMNP {
                read Card 3(i,k)
        }
} // comment: Read in NIP × (1 + NMNP) cards
```

Weight Cards. Provide weight for integration point *i*.

(Card 2)*i*	1	2	3	4	5	6	7	8
Variable	WI							
Type	F							

Integration Point Shape Function Value/Derivatives Cards. Provide the value of the k^{th} shape function and its derivative at the i^{th} integration point.

(Card 3)$_{ik}$	1	2	3	4	5	6	7	8
Variable	NKI		DNKIDR		DNKIDS		DNKIDT	
Type	F		F		F		F	

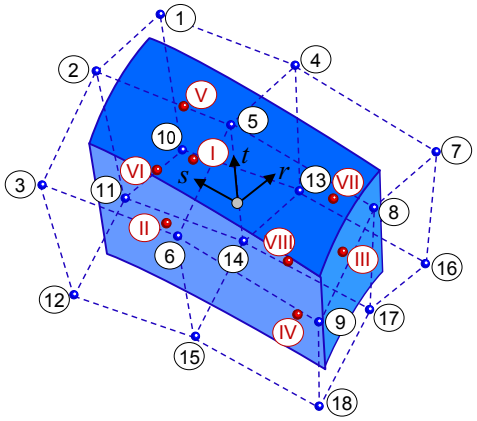
VARIABLE	DESCRIPTION
ELFORM	Element Formulation ID referenced by *SECTION_SOLID to connect *ELEMENT_GENERALIZED_SOLID with the appropriate solid formulation. The chosen number needs to be greater or equal than 1000.
NIP	Number of integration points.
NMNP	Number of nodes for this element formulation.
IMASS	Option for lumping of mass matrix: EQ.0: row sum EQ.1: diagonal weighting.
WI	Integration weight at integration point *i*.
NKI	Value of the shape function N_k evaluated at integration point *i*.
DNKIDR	Value of the derivative of the shape function N_k with respect to the local coordinate *r* at the integration point *i* $\left(\frac{\partial N_k^i}{\partial r}\right)$.
DNKIDS	Value of the derivative of the shape function N_k with respect to the local coordinate *s* at the integration point *i* $\left(\frac{\partial N_k^i}{\partial s}\right)$.
DNKIDT	Value of the derivative of the shape function N_k with respect to the local coordinate *t* at the integration point *i* $\left(\frac{\partial N_k^i}{\partial t}\right)$.

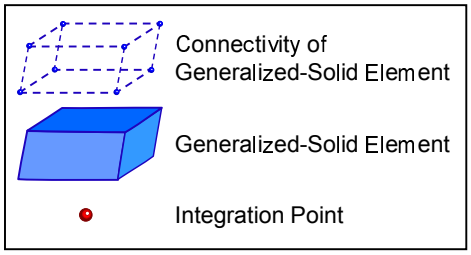
Remarks:

1. **Post-processing.** For post-processing the use of interpolation solid elements (see *ELEMENT_INTERPOLATION_SOLID and *CONSTRAINED_NODE_INTERPOLATION) is necessary.

2. **Input order.** Data input order for the NMNP nodal points must correlate with the definition of the connectivity of the element in *ELEMENT_GENERALIZED_SOLID.

Example:






```
*DEFINE_ELEMENT_GENERALIZED_SOLID
$#   elform       nip      nmnp     imass
        1001         8        18         0
$# integration point 1 (i=1)
$#                  wi
 1.3778659577546E-04
$#                 nki               dnkidr               dnkids               dnkidt
 1.7098997698601E-01 3.3723996630918E+00 2.4666694616947E+00 1.5327451653258E+00
...
$# integration point 2 (i=2)
 2.2045855324077E-04
 5.4296436772101E-02 1.9003752917745E+00 7.8327025592051E+00 3.258715871621E+00
...
[...]
$# integration point 8 (i=8)
 3.8574962585875E-04
 2.6578426581235E-01 1.6258741125438E+00 2.9876495873627E+00 5.403982758392E+00
 ...
```

W1
k=1
k=2,18
W2
NMNP Lines
W8
NMNP Lines
Block A

Figure 17-43. Example of a generalized solid formulation with *DEFINE_ELEMENT_GENERALIZED_SOLID

*DEFINE_FABRIC_ASSEMBLIES

Purpose: Define lists of part sets to properly treat fabric bending between parts.

Define as many cards as needed for the assemblies, using at most 8 part sets per card. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SPID1	SPID2	SPID3	SPID4	SPID5	SPID6	SPID7	SPID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
SPID*n*	Part set ID that comprises an assembly.

Remarks:

The materials *MAT_FABRIC and *MAT_FABRIC_MAP are equipped with an optional coating feature to model the fabric's bending resistance. See the related parameters ECOAT, SCOAT and TCOAT on these material model manual entries.

The default behavior for these coatings, *which this keyword changes*, excludes T-interesections, and, furthermore requires that all fabric elements must have a consistently oriented normal vector. In Figure 17-44, the left connection of fabric elements is permitted by the default functionality while the right one is not. However, with using this keyword the proper bending treatment for the right connectivity can be activated by adding the following input to the deck

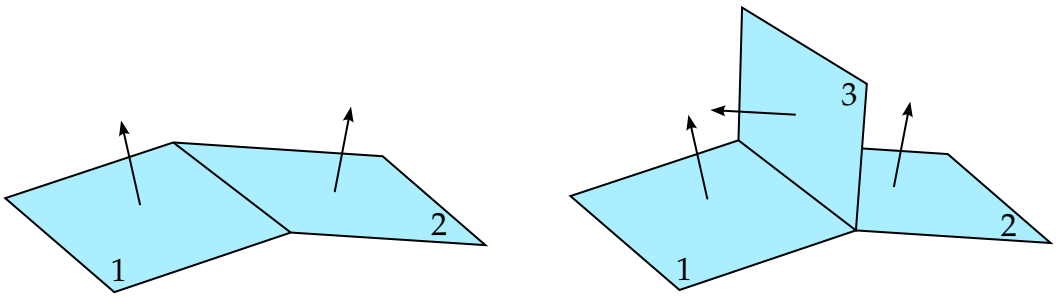


Figure 17-44. Bending in fabric, intended use of *DEFINE_FABRIC_ASSEMBLIES. The numbers indicate the parts the elements belong to.

```
*SET_PART_LIST
1
1,2
*SET_PART_LIST
2
3
*DEFINE_FABRIC_ASSEMBLIES
1,2
```

which decouples part 3 from the other two parts in terms of bending, thus creating a moment free hinge along the edge between part sets 1 and 2. Bending between parts 1 and 2 is unaffected since these are contained in the same fabric assembly.

For several instances of this keyword in an input deck, the list of assemblies is appended. If assemblies are defined and there happens to be fabric parts that do not belong to any of the specified assemblies, then these parts are collected in a separate unlisted assembly. The restriction on consistent normal vectors and on having no T-intersections applies to all elements within an assembly.

*DEFINE_FIBERS

Purpose: Define carbon fibers and their related properties in a matrix for a one-step inverse forming simulation. This keyword works *only* with the keyword *CONTROL_FORMING_ONESTEP. . This keyword requires a double precision executable. We developed this feature with Ford Motor Company.

Fiber ID and Property Card.

Card 1	1	2	3	4	5	6	7	8
Variable	IDF	IDP	NUMF	N1	N2	EFB	SHR	HRGLS
Type	I	I	I	I	I	F	F/I	F
Default	none	none	none	optional	optional	none	none	1.0

Fiber Orientation Card.

Card 2	1	2	3	4	5	6	7	8
Variable	ALPHA1	ALPHA2	ALPHA3					
Type	F	F	F					
Default	none	none	none					

Reference Fiber Orientation Card. Define this card if N1 and N2 in Card 1 are undefined or zero.

Card 2	1	2	3	4	5	6	7	8
Variable	X1	Y1	Z1	X2	Y2	Z2		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
IDF	ID of a fiber set to be defined. It must be unique number; see Figure 17-45.
IDP	Part ID of the matrix material associated with the fiber set; see Figure 17-45.
NUMF	Number of fiber orientations; see Figure 17-45.
N1, N2	Direction from Node 1 (N1) to Node 2 (N2) defines the reference fiber orientation; see Figure 17-45. If not defined or zero, Card 3 is required.
EFB	Effective stiffness of the fiber in its orientation which typically equals Young's Modulus times fiber cross sectional area fraction. Fiber cross sectional area fraction is typically between 0.25 and 0.5.
SHR	Shear stiffness of the fiber: GT.0: shear stiffness, LT.0: |SHR| is the load curve ID defining shear stiffness as a function of shear strain.
HRGLS	Hourglass coefficient for stiffness type hourglass control.
ALPHA1, ALPHA2, ALPHA3	Initial orientation angles of the first, second and third fibers relative to reference fiber orientation defined by N2 − N1, respectively.
X1, Y1, Z1, X2, Y2, Z2	If N1 and N2 are undefined, or zero, Card 3 is required to define the coordinates for N1 and N2. The Z1 and Z2 coordinate values must be defined close to the part. Based on the coordinate inputs, LS-DYNA will find the nearest nodes to define N1 and N2 from the model. See Figure 17-45.

Remarks:

1. **Matrix Material.** *MAT_024 is supported to define the matrix material.

2. **Predicted Flat Blank Orientation.** The method of orienting the unfolded blank with three nodes defined in *CONTROL_FORMING_ONESTEP_AUTO_CONSTRAINT is disabled when *DEFINE_FIBERS is used. Use the dynain (not repositioned.k) file for the predicted flat blank.

3. **Coulomb Friction.** Coulomb friction can be specified with *CONTROL_FORMING_ONESTEP_FRICTION.

4. **Draw Bead Forces.** Use of *CONTROL_FORMING_ONESTEP_DRAWBEAD is not recommended.

5. **Matrix Element Direction.** Element direction of the matrix must be consistent. The directions can be displayed in LS-PrePost4.6, under *EleTol* → *EleEdit* → *Direction* → *Shell* → *Vector* → *Apply*. To align all element directions with an existing element direction, use *EleTol* → *EleEdit* → *Direction* → *Shell* → *Orient* → *Pick seed* → *Apply*.

6. **Output for Visualization.** *ELEMENT_FIBER_INFO and *ELEMENT_BEAM (new part IDs) are automatically created and output in onestepresult file to display the fiber orientations after forming in LS-PrePost; see Figure 17-45.

   Additional history variables are written in the onestepresult file to display various angles in color contours (see Figure 17-45):

   a) History variable #1: the angle between two fibers,

   b) History variable #2: the angle between the first fiber and the element direction,

   c) History variable #3: the angle between the second fiber and the element direction.

   These color contour fringe plots can be overlaid with beam elements to verify the fiber orientations. To request history variable output, set NEIPS in *DATABASE_EXTENT_BINARY to a minimum of 3.

Example:

A partial keyword example below defines a fiber set ID 1 in a matrix material with part ID 121. The first and second fiber orientations are at 45° and 135°, respectively, from the reference fiber direction.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_FIBERS
$       IDF       IDP      NUMF        N1        N2       Efb       SHR     HRGLS
          1       121         2                       120000.0     -1011       1.0
$    ALPHA1    ALPHA2    ALPHA3
       45.0     135.0
$        X1        Y1        Z1        X2        Y2        Z2
     2650.0     800.0     808.0    2940.0     800.0     800.0
*DEFINE_CURVE
1011
0.0,0.0
0.1,1.0
0.5,5.0
```

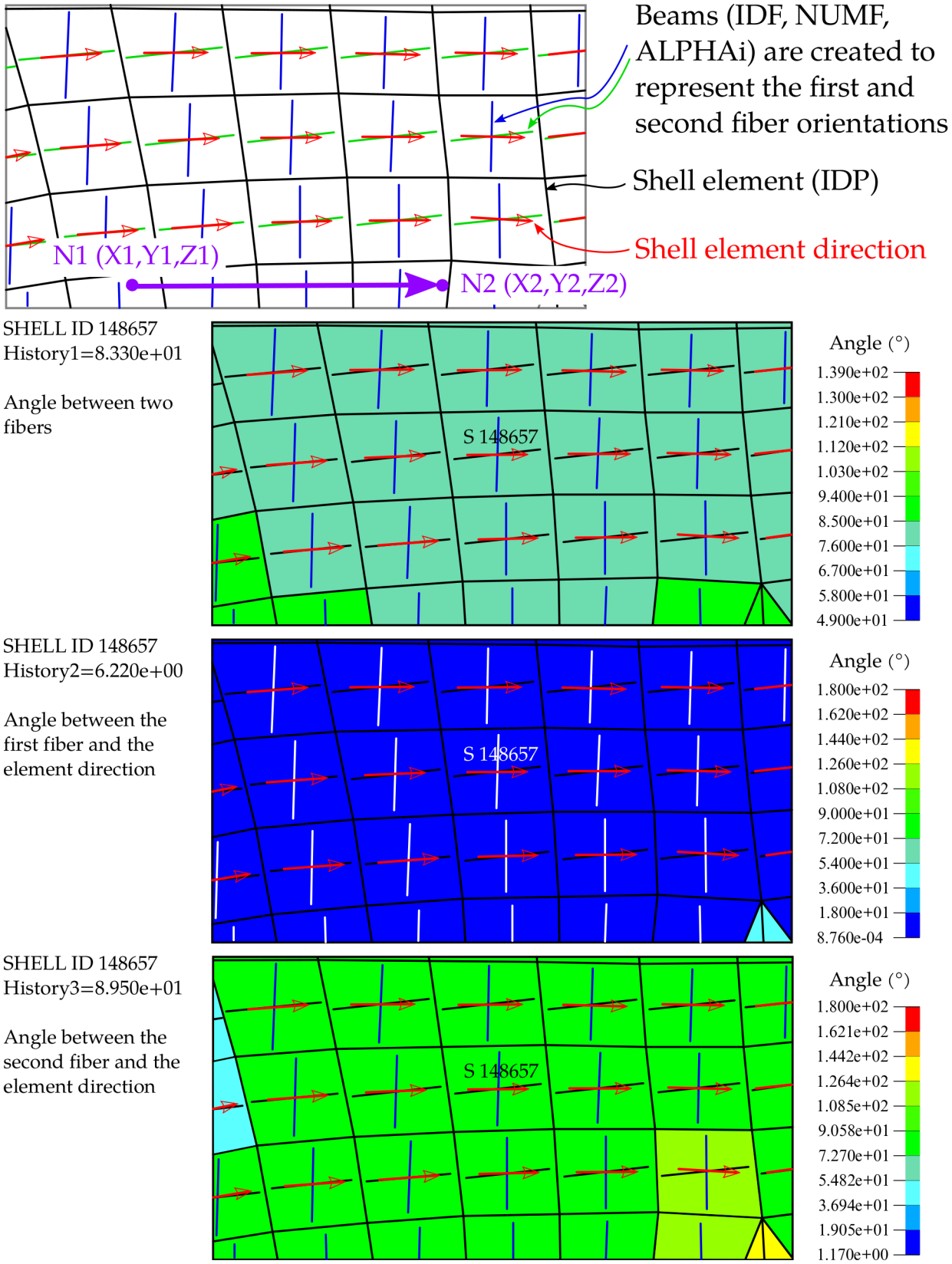


Figure 17-45. *DEFINE_FIBERS inputs and outputs.

*DEFINE_FIELD_{OPTION}

Available options include:

<BLANK>

TITLE

Purpose: Define a field associated with a point cloud; see *DEFINE_POINT_CLOUD. Field data, defined on a point cloud, are automatically interpolated at required locations for the purpose of analysis. The interpolation is performed through radial basis functions. Parameters defined on *DEFINE_FIELD control the interpolation.

Card Summary:

Card Title. This card is included if the TITLE keyword option is used.

TITLE

Card 1. This card is required.

FID	PCID	NV	LMTSRC	SRCRAD	REFLEN		

Card 2. This card is required. Include ceil[NV × number of points/8] cards. This input ends if an empty field is found OR at the next keyword ("*") card.

V1	V2	V3	V4	V5	V6	V7	V8

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

VARIABLE	DESCRIPTION
TITLE	Name or description of the field defined in this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	FID	PCID	NV	LMTSRC	SRCRAD	REFLEN		
Type	I	I	I	I	F	F		
Default	none	none	none	0	0.0	0.0		

VARIABLE	DESCRIPTION
FID	Field ID. A unique ID number must be used.
PCID	ID of the point cloud referenced by this field. See *DEFINE_POINT_CLOUD.
NV	Number of data specified for each point of the point cloud PCID. See Remark 1.
LMTSRC	Flag to restrict the interpolation domain to a subset of data points of the point cloud PCID. See Remark 4. This flag activates a search algorithm that restricts the interpolation domain based on a point-to-element distance criterion. For any element associated with a *PART_..._FIELD that references this field FID: EQ.0: Perform global interpolation by interpolating field data from all data points of point cloud PCID. EQ.1: Restrict the interpolation domain and search for all data points of point cloud PCID whose distance from centroid of the element is ≤ a distance determined by SRCRAD.
SRCRAD	Radius of the interpolation domain. If LMTSRC = 0, this parameter is disregarded. If LMTSRC = 1, this parameter restricts the interpolation domain. Based on the value of SRCRAD, the interpolation domain is restricted to all data points whose distance from the element centroid is: LT.0.0: Less than or equal to |SRCRAD| × REFLEN EQ.0.0: Less than or equal to REFLEN GT.0.0: Less than or equal to SRCRAD
REFLEN	Reference length used for scaling of radial basis functions, and, if required, to restrict the interpolation domain:

VARIABLE	DESCRIPTION
	EQ.0.0: For any element, the solver automatically computes a reference length.
	GT.0.0: User-defined reference length. Any value can be specified.

Field Data Cards. Include ceil[NV × number of points/8] cards. This input ends at the first missing field or the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	V1	V2	V3	V4	V5	V6	V7	V8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
V*i*	Field data specified at points of the point cloud PCID. See Remark 2. NV × number of points defined in PCID should be defined. The data should be input such that the first NV data are for the first point, the second NV data are for the second point, etc.

Remarks:

1. **Field dimension**. The format of *DEFINE_FIELD is general in the sense that no distinction is made between scalar, vector, or tensor data at the keyword level.

2. **Data type.** A field represents a consistent set of data associated with one physical property or parameter.

3. **Precision**. To specify more significant digits for the field data, invoke the "long format" for this keyword, meaning *DEFINE_FIELD_*{OPTION}*+. For the "long format," the format of Card 1 is (4I20,2F20), while the format of Card(s) 2 is (8F20). The precision of the executable may also limit the actual number of significant digits of the input field data used for the analysis. See Long Format Input in the section General Card Format of Getting Started in this manual for more details regarding the number of significant digits and the "long format."

4. **Interpolation domain**. By default, all data points defined in PCID are used to interpolate the field at each integration point. While this approach guarantees that at least one data point is always included in the interpolation domain, the interpolation algorithm may become excessively expensive for larger data sets.

Therefore, we highly recommend setting LMTSRC to 1, REFLEN to 0, and SRCRAD to a negative value with a magnitude greater than 1.0 (|SRCRAD| > 1). If a warning or error message due to data points not being found within the search domain occurs, we recommend increasing the density of the data points. For the most accurate results, we recommend defining at least one data point per element, located close to the element's centroid.

*DEFINE_FILTER

Purpose: Define a general-purpose filter. It is currently used by *SENSOR_SWITCH.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	TYPE	DATA1	DATA2	DATA3	DATA4	DATA5	DATA6	DATA7
Type	A10							

VARIABLE	DESCRIPTION
ID	Identification number
TITLE	Title for this filter
TYPE	One of the three currently defined filter types: DISCRETE, CONTINUOUS, or CHAIN
DATA1-7	Filter-type-specific data, which determines what the filter does.

Filter Types:

FILTER	DESCRIPTION
DISCRETE	The discrete filter operates on a fixed number of values of the input data. The first data field is an A10 character field, which gives the type of operation the filter performs: MIN, MAX, and AVG are the available options. The second data field is an I10 field, giving the number of input values over which the minimum, maximum, or average is computed.
CONTINUOUS	Like the DISCRETE filter, except that it operates over a fixed time interval. The first data field is identical to that for the DISCRETE option. The second data field is an F10 field and provides the duration of the filter. For example, if AVG is given

FILTER	DESCRIPTION
	and the duration is set to 0.1, a running-time-step-weighted average is computed over the last 0.1 time steps of the simulation.
CHAIN	Here, data fields 1-7 are all I10 fields, and give the IDs of a list of other filters (including other CHAIN filters, if desired), each of which will be applied in order. So the raw data is fed to the filter indicated by DATA1. The output of that is fed to the next filter, and so on, with up to 7 filters in the chain. List only as many filters as you need.

*DEFINE_FORMING_ADAPTIVE_PART_INITIAL

Purpose: Refine the mesh of a deformable part set in local regions at time zero. This keyword is intended for refining the mesh of blank parts in small regions for spot welding simulations or where contact occurs between the blank and rigid pins in gravity loading processes. In the case of rigid pins, this keyword enables contact to occur between the pins and the blank because, otherwise, the pins are too small compared to the blank's initial coarse mesh. Thus, contact is missed without this refinement. The local regions for refinement on the blank are determined by projecting the parts of local regions onto the blank.

Card 1	1	2	3	4	5	6	7	8
Variable	PSTID	PSTBID	VX	VY	VZ	SMIN	IMOVE	
Type	I	I	F	F	F	F	I	
Default	none	none	0.0	0.0	0.0	0.0	0	

VARIABLE	DESCRIPTION
PSTID	Part set of local regions that will be projected onto the parts of the blank (PSTBID) to determine where the blank will be refined
PSTBID	Part set of the deformable body (blank) to be refined
VX, VY, VZ	Vector for projecting the parts of PSTID onto the parts of PSTBID
SMIN	Minimum element size after refinement
IMOVE	Flag for treatment of the vector (VX, VY, VZ): EQ.0: Use (VX, VY, VZ) as the projection vector. EQ.1: Ignore (VX, VY, VZ) and use the movement vector defined with *PART_MOVE instead for the projection. This option may be ideal when including this keyword with *INCLUDE_TRANSFORM because the vector changes from the transformation. This option enables determining the vector only once.

*DEFINE_FORMING_BLANKMESH

Purpose: A rectangular sheet metal blank can be defined with square elements and an arbitrary mesh orientation. This blank can, then, be trimmed with *ELEMENT_BLANKING and *DEFINE_CURVE_TRIM_2D into a developed blank with a complex periphery and inner hole cutouts. This keyword is renamed from the previous keyword *CONTROL_FORMING_BLANKMESH.

Card 1	1	2	3	4	5	6	7	8
Variable	IDMSH	ELENG	LENG1	LENG2	ANGLE	NPLANE	CID	
Type	I	F	F	F	F	I	I	
Default	none	0.0	0.0	0.0	0.0	1	0	

Card 2	1	2	3	4	5	6	7	8
Variable	PIDBK	NID	EID	XCENT	YCENT	ZCENT	SHIFT1	SHIFT2
Type	I	I	I	F	F	F	F	F
Default	1	1	1	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
IDMSH	A unique ID for the blank mesh (not the blank PID) to be generated
ELENG	The edge length of the square element to be generated.
LENG1	First edge length of the rectangular blank along the following axis in the coordinate system (CID) defined: NPLANE.EQ.1: X-axis NPLANE.EQ.2: Y-axis NPLANE.EQ.3: Z-axis
LENG2	Second edge length of the rectangular blank along the following axis in the coordinate system (CID) defined: NPLANE.EQ.1: Y-axis

VARIABLE	DESCRIPTION
	NPLANE.EQ.2: *Z*-axis NPLANE.EQ.3: *X*-axis
ANGLE	An angle about the following axis of the CID defined, starting from its respective axis, to rotate the blank (the orientation of the mesh) to be generated. The sign of the rotation angle follows the right-hand rule. NPLANE.EQ.1: about *Z*-axis, starting from the positive *X*-axis NPLANE.EQ.2: about *X*-axis, starting from the positive Y-axis NPLANE.EQ.3: about *Y*-axis, starting from the positive *Z*-axis
NPLANE	A plane in which a flat blank to be generated, in reference to the coordinate system defined (CID): EQ.1: *XY*-plane (default) EQ.2: *YZ*-plane EQ.3: *XZ*-plane
CID	ID of a local coordinate system defined by *DEFINE_COORDINATE_SYSTEM. Default is 0 representing the global coordinate system.
PIDBK	Part ID of the blank, as defined by *PART
NID	Starting node ID of the blank to be generated
EID	Starting element ID of the blank to be generated
XCENT	*X*-coordinate of the center of the blank
YCENT	*Y*-coordinate of the center of the blank
ZCENT	*Z*-coordinate of the center of the blank
SHIFT1	First shift distance of the blank in the following axis in the coordinate system (CID) defined. NPLANE.EQ.1: *X*-axis NPLANE.EQ.2: *Y*-axis NPLANE.EQ.3: *Z*-axis
SHIFT2	Second shift distance of the blank in the following axis in the

VARIABLE	DESCRIPTION
	coordinate system (CID) defined: NPLANE.EQ.1: Y axis NPLANE.EQ.2: Z-axis NPLANE.EQ.3: X-axis

About the Keyword:

A flat and rectangular blank can be defined and meshed, which can be trimmed with IGES curves to a desired blank shape with complex periphery and inner cutouts. The mesh orientation is always parallel to the rectangular blank edge and can be changed by defining the field ANGLE. The blank outlines and inner holes can be defined using IGES curves and included in *DEFINE_CURVE_TRIM_2D (not_3D). This feature is very useful for interactive input deck set up of metal forming simulation.

Example:

A complete keyword example of generating a flat blank with PID 1 is provided blow. Referring to Figure 17-46, the rectangular blank mesh is generated in the global XY-plane with its center at the global origin and a size of 1100.0×1050.0 mm. The square element edge length is 12 mm. The node and element ID numbers start at 8000 and 9000, respectively. An inner cut-out curve is provided by the IGES file innerholes.iges, and the blank outer line is defined with the IGES file outerlines.iges. Both files are included in the keyword *DEFINE_CURVE_TRIM_2D. A seed point is defined indicating the portion that remains after trimming.

```
*KEYWORD
*SET_PART_LIST
1
1
*CONTROL_TERMINATION
0.0
*ELEMENT_BLANKING
$# psid
1
*DEFINE_FORMING_BLANKMESH
$    IDMSH     ELENG     LENG1     LENG2     ANGLE    NPLANE       CID
         3      12.0   1100.00    1050.0
$    PIDBK       NID       EID     XCENT     YCENT     ZCENT    SHIFT1    SHIFT2
         1      8000      9000
*DEFINE_CURVE_TRIM_2D
$#    tcid    tctype      TFLG      TDIR     TCTOL      TOLN
     11111         2                 0  0.250000  1.000000
innerholes.iges
*DEFINE_CURVE_TRIM_2D
$#    tcid    tctype      TFLG      TDIR     TCTOL      TOLN
     11112         2                 0  0.250000  1.000000
outerlines.iges
*DEFINE_TRIM_SEED_POINT_COORDINATES
1,264.0,0.0,-200.0
```

```
*PART
Blank
$#     pid     secid       mid
         1         1         1
*SECTION_SHELL
$#   secid    elform      shrf       nip
         1         2  1.000000         1
$#      t1        t2        t3        t4
  0.500000  0.500000  0.500000  0.500000
*MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC
$      MID        RO         E        PR      SIGY      ETAN         R
         1 7.9000E-9 2.0700E+5  0.300000 179.80000     1.000       1.0
*INTERFACE_SPRINGBACK_LSDYNA
1
*END
```

The blank and mesh orientation can be rotated about a particular axis. Following the right-hand rule, the blank in this case is rotated about *Z*-axis for a positive 30° (ANGLE=30.0) starting from the positive *X*-axis, as shown in Figure 17-47.

Revision information:

This feature is available in LS-DYNA starting in Revision 59165. This keyword name was changed from *CONTROL_FORMING_BLANKMESH to *DEFINE_FORMING_BLANKMESH in Revision 69074. The variable NPLANE is available in Revision 69128. Some important updates are available in Revision 123603.

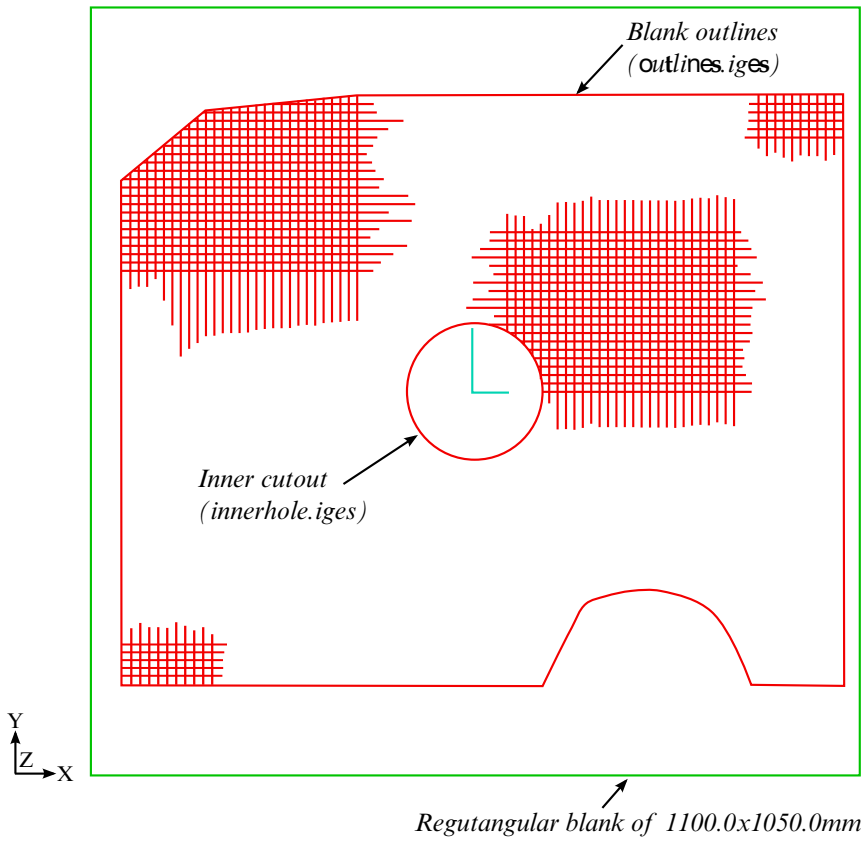


Figure 17-46. A rectangular blank mesh is generated and trimmed into a developed blank shape, with an inner cutout defined by innerhole.iges and outline defined by outlines.iges.

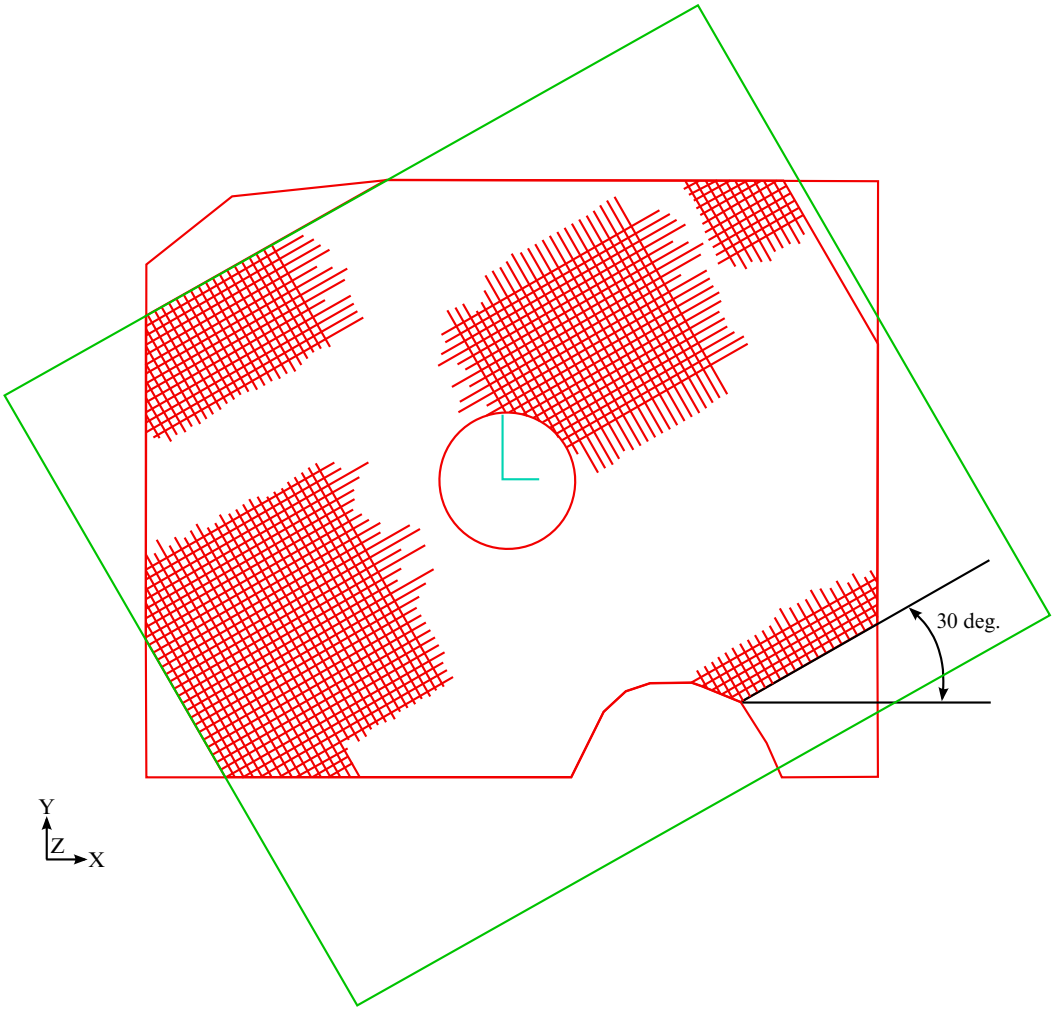


Figure 17-47. A rectangular blank mesh is generated in an angle (ANGLEX = 30) and trimmed into a developed blank shape, with an inner cutout defined by innerhole.iges and outline defined by outlines.iges. Note the blank mesh orientation is always parallel to the rectangular blank boundary.

*DEFINE_FORMING_CLAMP

Purpose: This keyword acts as a macro that replaces the combination of cards required to model a clamping process. It is available for double precision executables with the implicit solver. A related keyword is *DEFINE_FORMING_CONTACT.

Define Clamp Card. Define one card for each clamp set. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	CLP1	CLP2	VID	GAP	AT	DT		
Type	I	I	I	F	F	F		
Default	none	none	none	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
CLP1	Part ID of a moving rigid body clamp, defined by *PART and *MAT_020 (*MAT_RIGID).
CLP2	Part ID of a fixed rigid body clamp, defined by *PART and *MAT_-020. This is sometimes called "net pad".
VID	Define CLP1 moving direction: GT.0: Vector ID from *DEFINE_VECTOR, specifying the moving direction of CLP1 LT.0: Absolute value is a node ID, whose normal vector will be used to define the moving direction of CLP1.
GAP	Final desired distance between CLP1 and CLP2 at the end of clamping.
AT	Begin time for CLP1's move.
DT	Duration of CLP1's move.

About the keyword:

One typical application of this keyword is to estimate springback during the clamping of a formed panel on a checking fixture. Net pads (lower fixed regular or square pads) of a

few millimeters thick are placed according to GD&T (*Geometry Dimensioning and Tolerancing*) requirements on a support platform, typically taking shape of the nominal product. Each net pad (CLP2, see Figure 17-48) has a corresponding moving clamp (CLP1).

The movable clamp (CLP1) is initially open so that the formed panel can be loaded onto the net pads. Four-way and two-way position gaging pins are used to initially locate and load the panel in the fixture before CLP1 is moved to close with the net pad (CLP2). A white light scan is then performed on the panel, and scan data is processed to ascertain the degree of panel conformance to the required nominal shape. Even with the clamps fully closed, some severely distorted panels will significantly deviate from the nominal shape when the residual stresses from the forming process are too great.

Although, unrelated to this keyword another common method to determine the panel springback amount is the *free state* check which involves a white light scan on the panel secured on a platform but with no additional forces (clamps – CLP1s) applied to deform the panel (to the net pads CLP2s, for example). LS-DYNA can model both methods and job setups can be easily done by selecting *Implicit Static Flanging* process in LS-PrePost 4.2's *Metal Forming Application → eZ-Setup*. Furthermore, once springback has been determined, die compensation (*INTERFACE_COMPENSATION_3D) can then be performed to minimize or eliminate the springback; the resulting compensated die shapes can be surfaced, re-machined to produce panels that are within dimensional tolerance.

Since the clamp is, typically, modelled with a much coarser mesh than that on a blank, *CONTACT_FORMING_SURFACE_TO_SURFACE should be used. Rotating-type of clamps are currently not supported.

Application example:

A partial keyword example of using the feature is listed below. Referring to Figure 17-48, the drawn and trimmed blank is positioned between the clamps CLP1 and CLP2. The implicit termination "time" is set at 1.0, with a stepping size of 0.25, for a total of four steps – two steps each for the two CLP1. With the original blank thickness of 1.0 mm, the CLP1s are set to close with the lower CLP2s at "time" of 1.0, leaving a total GAP of 1.02 mm. Note the VIDs are defined as "-46980", indicating that the moving clamps (CLP1) will move in the normal direction defined by Node #46980. The contact definition between the panel and the clamps are defined using *DEFINE_FORMING_CONTACT.

```
*KEYWORD
*INCLUDE
./trimmed.dynain
./nets.k
*CONTROL_TERMINATION
1.0
*CONTROL_IMPLICIT_forming
1
*control_implicit_general
1,0.25
*CONTROL_SHELL
```

```
⋮
*DATABASE_EXTEND_BINARY
⋮
*PART
Blank
$      PID     SECID       MID
         1         1         1
Clamp1
2,2,2
Clamp2
3,2,3
Clamp3
4,2,2
Clamp4
5,2,3
*MAT_TRANSVERSELY_ANISOTROPIC_ELASTIC_PLASTIC
$      MID        RO         E        PR      SIGY      ETAN         R     HLCID
         1 2.700E-09 12.00E+04      0.28       0.0       0.0     0.672         2
*MAT_RIGID
$#     mid        ro         e        pr         n    couple         m     alias
         2 7.8500E-9 2.1000E+5  0.300000
$#     cmo      con1      con2
  1.000000         7         7
$# lco or a1      a2        a3        v1        v2        v3
     0.000     0.000     0.000     0.000     0.000     0.000

*MAT_RIGID
$#     mid        ro         e        pr         n    couple         m     alias
         3 7.8500E-9 2.1000E+5  0.300000
$#     cmo      con1      con2
  1.000000         4         7
$# lco or a1      a2        a3        v1        v2        v3

*SETION_SHELL
1,16,,7
1.0,1.0,1.0,1.0
*LOAD_BODY_Z
      9997       1.0
*DEFINE_CURVE
      9997
              0.0000           9810.0000
              1.0000           9810.0000
*DEFINE_FORMING_CLAMP
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$     CLP1      CLP2       VID       GAP        AT        DT
         3         2    -46980      1.02       0.0       0.5
         5         4    -46980      1.02       0.5       0.5
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DEFINE_FORMING_CONTACT
$      IPS       IPM        FS    ONEWAY
         1         2     0.125         1
         1         3     0.125         1
         1         4     0.125         1
         1         5     0.125         1
*END
```

Note with this keyword, the formed panel needs to be pre-positioned properly with respect to the clamps by users, and auto-position (*CONTROL_FORMING_AUTOPOSITION) cannot be used. Furthermore, prescribed motions and clamp motion curves do not need to be defined for the clamps.

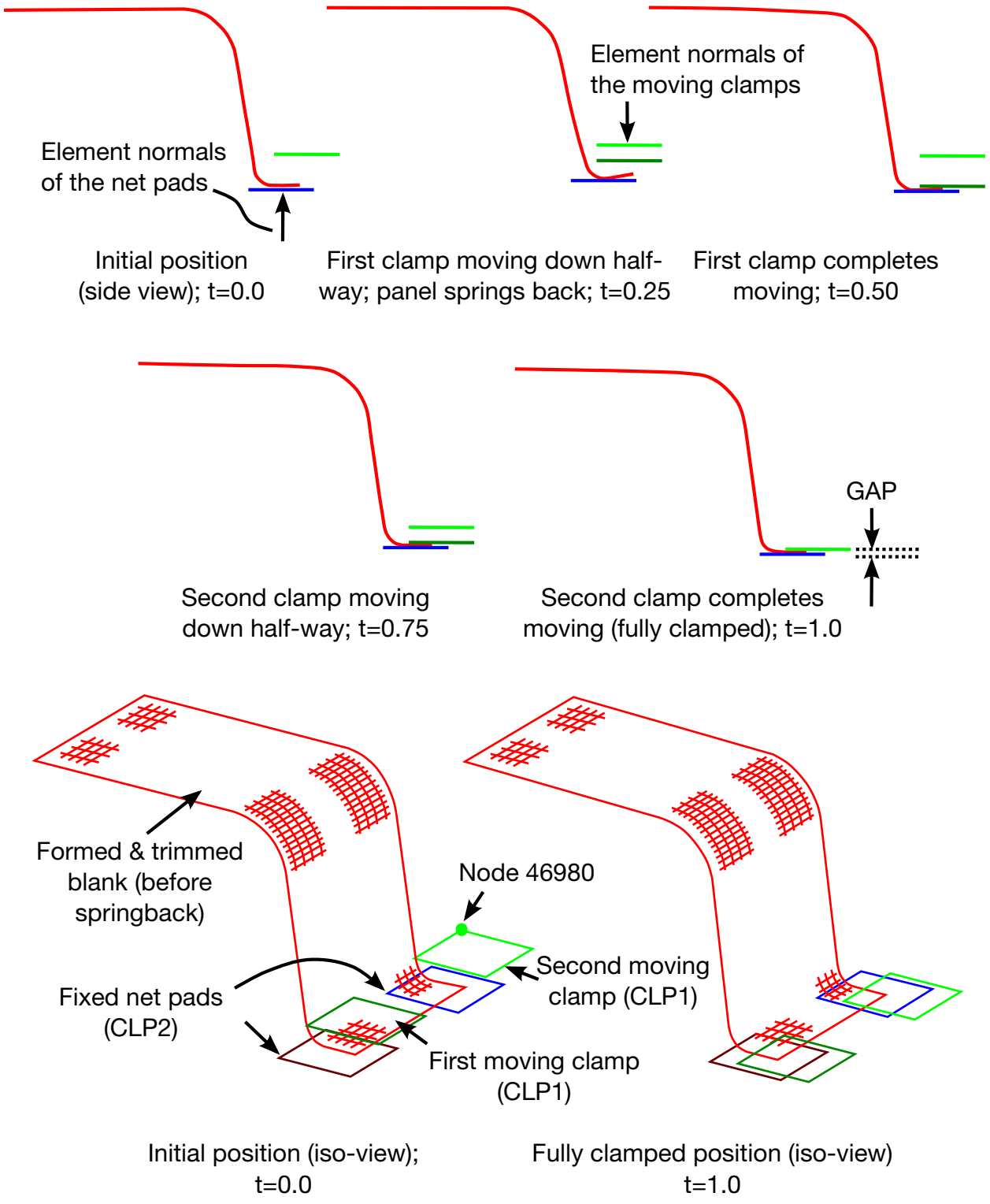


Figure 17-48. Variable definitions and an example of using the negative VID.

*DEFINE_FORMING_ONESTEP_PRIMARY

> **NOTE:** This keyword was previously named *DEFINE_-FORMING_ONESTEP_MASTER. The old name is supported for backward compatibility. *DEFINE_-FORMING_ONESTEP_PRIMARY is the preferred name starting with R13.

Purpose: Define a primary blank to which a second (constrained) blank is connected using *CONSTRAINED_SPOTWELD. Such a configuration is often called "patch-welded blanks." Note this keyword must be used together with *CONTROL_FORMING_ONESTEP. This feature is available in SMP with a double-precision executable.

Card 1	1	2	3	4	5	6	7	8
Variable	SLPID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SLPID	Part ID of the primary blank to which a constrained blank is welded using *CONSTRAINED_SPOTWELD

Example:

A partial keyword example is given below. Two patches, with PIDs 2 and 3, are spot-welded onto the primary blank (PID 1) using multiple *CONSTRAINED_SPOTWELD keywords. The primary blank is specified using *DEFINE_FORMING_ONESTEP_PRIMARY. The spot welds can be generated using LS-PrePost.

```
*KEYWORD
*TITLE
Patch welded blanks - One Step Simulation
*PARAMETER
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
I nip              3
R dt            0.25
R autobd        1.50
R tsclmax       1.00
R tsclmin       0.80
R epsmax        0.20
I nslovr           2
I ilimit           1
R dctol         0.01
```

```
R deltau        0.00
I lsolvr           5
$ Reposition Nodes
I nid1          6344
I nid2          1950
I nid3          4113
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*PART
Part to be unfolded - 3
$#     pid     secid       mid
         1         1         1

         2         1         1

         3         1         1
*SECTION_SHELL
$#   secid    elform      shrf       nip     propt   qr/irid     icomp     setyp
         1        16   0.83333      &nip         1         0         0         1
$#      t1        t2        t3        t4
  1.000000  1.000000  1.000000  1.000000
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*INCLUDE
2patches.k
*INCLUDE
sim_Mat.k
*INCLUDE
EZone_step.k
*CONSTRAINED_SPOTWELD_ID
$#     wid
         1
$#      n1        n2        sn        ss         n         m        tf        ep
      8396      2019     0.000     0.000     0.000     0.0001.0000E+201.0000E+20
*CONSTRAINED_SPOTWELD_ID
         3
      8289      1894     0.000     0.000     0.000     0.0001.0000E+201.0000E+20
*CONSTRAINED_SPOTWELD_ID
        13
      8216      1817     0.000     0.000     0.000     0.0001.0000E+201.0000E+20
*CONSTRAINED_SPOTWELD_ID
        14
      1966      8355     0.000     0.000     0.000     0.0001.0000E+201.0000E+20
*CONTROL_FORMING_ONESTEP_QUAD2
$#  option   TSCLMAX    autobd   TSCLMIN    EPSMAX
         7  &tsclmax   &autobd  &tsclmin   &epsmax

*DEFINE_FORMING_ONESTEP_PRIMARY
$    SLPID
         1
*END
```

*DEFINE_FP_TO_SURFACE_COUPLING

Purpose: Define a tied coupling interface between fluid particles modeled with implicit smoothed particle Galerkin (ISPG) and a surface.

Card 1	1	2	3	4	5	6	7	8
Variable	FP	SURF	FPTYPE	SURFTYPE				
Type	I	I	I	I				
Default	none	none	1	0				

Card 2	1	2	3	4	5	6	7	8
Variable	SBC	SCA				SFP		
Type	I	F				F		
Default	0	0.5				0.1		

VARIABLE	DESCRIPTION
FP	Part/part set ID for the fluid particles
SURF	Segment set ID specifying the surface. Currently the segment set should be generated from the 8-noded hexahedral elements.
FPTYPE	Type for FP: EQ.0: Part set ID EQ.1: Part ID
SURFTYPE	Type for SURF: EQ.0: Segment set ID
SBC	Type of boundary condition. EQ.0: Free-slip boundary EQ.1: Non-slip boundary

VARIABLE	DESCRIPTION
SCA	Static (equilibrium) contact angle in radians
SFP	Stiffness coefficient along the normal direction of the contact interface. SFP should be less than 1.0. If SFP is too small, large penetrations can occur.

*DEFINE_FRICTION

Purpose: Define friction coefficients between parts for use with the contact options:

SINGLE_SURFACE

AIRBAG_SINGLE_SURFACE

AUTOMATIC_GENERAL

AUTOMATIC_SINGLE_SURFACE

AUTOMATIC_SINGLE_SURFACE_MORTAR

AUTOMATIC_NODES_TO_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE

AUTOMATIC_SURFACE_TO_SURFACE_MORTAR

AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE

ERODING_SINGLE_SURFACE

The input ends at the next keyword ("*"). Default friction values are used for any part ID pair that is not defined. Contact interfaces that are defined with SOFT = 2 on optional card A may use this part data for any keyword that is supported by the segment-to-segment contact.

The coefficient tables specified by the following cards are activated when FS he (see t card of *CONTACT) is set to -2.0. This feature overrides the coefficients defined in *PART_-CONTACT (which are turned on only when FS is set to -1.0).

When only *one* friction table is defined, it is used for *all* contacts having FS set to -2.0. Otherwise, for each contact with FS equal to -2.0, the keyword reader assigns a table to each *CONTACT by matching the value of FD from *CONTACT with an ID from Card 1 below. Failure to match FD to an ID causes error termination.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	FS_D	FD_D	DC_D	VC_D	ICNEP		
Type	I	F	F	F	F	I		
Default	none	0.0	0.0	0.0	0.0	0		

Friction *ij* card. Sets the friction coefficients between parts (or part sets) i and j. Add as many of these cards to the deck as necessary. The next keyword ("*") card terminates the friction definition.

Card 2	1	2	3	4	5	6	7	8
Variable	PID*i*	PID*j*	FS*ij*	FD*ij*	DC*ij*	VC*ij*	PTYPE*i*	PTYPE*j*
Type	I	I	F	F	F	F	A	A
Default	none	none	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ID	Identification number.
FS_D	Default value of the static coefficient of friction. The frictional coefficient is assumed to depend on the relative velocity v_{rel} of the surfaces in contact, $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ Default values are used when the part pair is undefined. For Mortar contact, $\mu_c = \text{FS}$, that is, dynamic effects are ignored.
FD_D	Default value of the dynamic coefficient of friction. The frictional coefficient is assumed to depend on the relative velocity v_{rel} of the surfaces in contact, $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ Default values are used when the part pair is undefined. For Mortar contact, $\mu_c = \text{FS}$, that is, dynamic effects are ignored.
DC_D	Default value of the exponential decay coefficient. The frictional coefficient is assumed to depend on the relative velocity v_{rel} of the surfaces in contact,

VARIABLE	DESCRIPTION
	$$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\lvert v_{\text{rel}}\rvert}.$$ Default values are used when the part pair is undefined. For Mortar contact, $\mu_c = \text{FS}$, that is, dynamic effects are ignored.
VC_D	Default value of the coefficient for viscous friction. This is necessary to limit the friction force to a maximum. A limiting force is computed $$F_{\text{lim}} = \text{VC} \times A_{\text{cont}} \ ,$$ where A_{cont} is the area of the segment contacted by the node in contact. The suggested value for VC is to use the yield stress in shear $\text{VC} = \sigma_o/\sqrt{3}$ where σ_o is the yield stress of the contacted material. Default values are used when the part pair is undefined.
ICNEP	Flag to check for non-existing parts, or part sets, (PID*i*, PID*j*) on Card 2: EQ.0: The existence of parts or part sets is checked, and an error occurs when any is missing (default). EQ.1: The existence of parts or part sets is checked, and lines with non-existent parts will be ignored.
PID*i*	Part or part set ID *i*
PID*j*	Part or part set ID *j*
FS*ij*	Static coefficient of friction between parts *i* and *j*
FD*ij*	Dynamic coefficient of friction between parts *i* and *j*
DC*ij*	Exponential decay coefficient between parts *i* and *j*
VC*ij*	Viscous friction between parts *i* and *j*
PTYPE*i*, PTYPE*j*	EQ."PSET": when PTYPE*i* or PTYPE*j* refers to a *SET_PART.

*DEFINE_FRICTION_ORIENTATION

Purpose: Define different coefficients of friction (COF) for specific directions, specified using a vector and angles in degree. In addition, COF can be scaled according to the amount of pressure generated in the contact interface. This keyword works in SMP only with *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE, including the SMOOTH keyword option. In MPP it works with *CONTACT_FORMING_ONE_-WAY_SURFACE_TO_SURFACE_ORTHO_FRICTION, including the SMOOTH keyword option.

This feature is developed jointly with the Ford Motor Company.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCID	LCIDP	V1	V2	V3		
Type	I	I	I	F	F	F		
Default	none	0	0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
PID	Part ID to which directional and pressure-sensitive COF is to be applied. See *PART.
LCID	ID of the load curve defining COF as a function of orientation in degrees
LCIDP	ID of the load curve defining COF scale factor as a function of pressure
V1, V2, V3	Vector components of vector **V** defining zero-degree (rolling) direction

Remarks:

Load curves LCID and LCIDP are not extrapolated beyond what are defined. It is recommended that the definition is specified for the complete range of angle and pressure values expected. One edge of all elements on the sheet metal blank must align initially with the vector defined by V1, V2, and V3.

This feature was initially intended for use with *CONTACT_FORMING_ONE_WAY_-SURFACE_TO SURFACE in SMP. It later was further enhanced to be used in

combination with *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE_ORTHO_FRICTION for MPP applications. With this contact, the fields LCID and LCIDP are overridden by the friction factor input in ORTHO_FRICTION, while the vector **V** defines the first orthogonal direction. It furthermore allows the convenience of defining SURFA and SURFB in *CONTACT as part sets in which case the segment sets necessary for ORTHO_FRICTION are generated automatically with orientation according to vector **V**. A detailed keyword example is shown in Figure 17-55.

Example:

The following is a partial keyword input of using this feature to define directional frictions and pressure-sensitive COF scale factor.

```
*DEFINE_FRICTION_ORIENTATION
$      PID      LCID     LCIDP        V1        V2        V3
         1        15        16       1.0        0.        0.
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ define COF vs.  orientation angle
*DEFINE_CURVE
15
0.0, 0.3
90.0, 0.0
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ define COF scale factor vs.  pressure
*DEFINE_CURVE
16
0.1, 0.3
0.2, 0.3
0.3, 0.3
0.4, 0.2
0.5, 0.5
0.6, 0.4
```

Referring to Figure 17-49, a deformable blank is clamped with 1000 N of force between two rigid plates and is pulled 90 mm in the X-direction using displacement control. Initial and final positions of the blank are shown in Figure 17-50. The normal force is recovered from the RCFORC file, as shown in Figure 17-51, which agrees with what is applied. The frictional force (pulling force) in the X-direction is 89 N, as shown in Figure 17-52. A hand calculation from the input verifies this result:

$$\underbrace{[\text{clamping force}]}_{1000N} \times \underbrace{[X\text{-dir coefficient}]}_{0.3} \times \underbrace{[\text{coefficient scale factor at 0.27 pressure}]}_{0.3} = 90N$$

The interface pressure can be output from an LS-DYNA run when 'S=filename' is included in the command line. The binary output can be viewed from LS-PrePost.

The element directions are automatically aligned with the vector **V**. The left side of Figure 17-53 shows the element directions of the incoming sheet blank. The keyword will re-orient the element directions based on the vector **V** specified, which has the component of (1.0,0.0,0.0) in this case. The re-oriented element directions for the blank are shown on the right side of the figure.

Following the numeric directions provided in Figure 17-54, LS-PrePost4.0 can be used to check the element directions of a sheet blank.

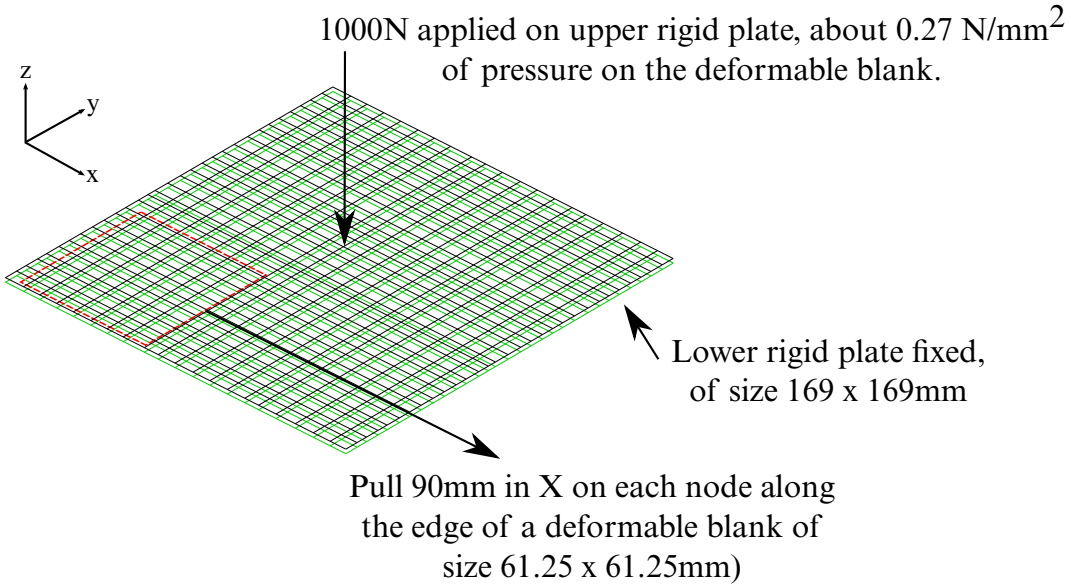


Figure 17-49. Boundary and loading conditions of a small test model.

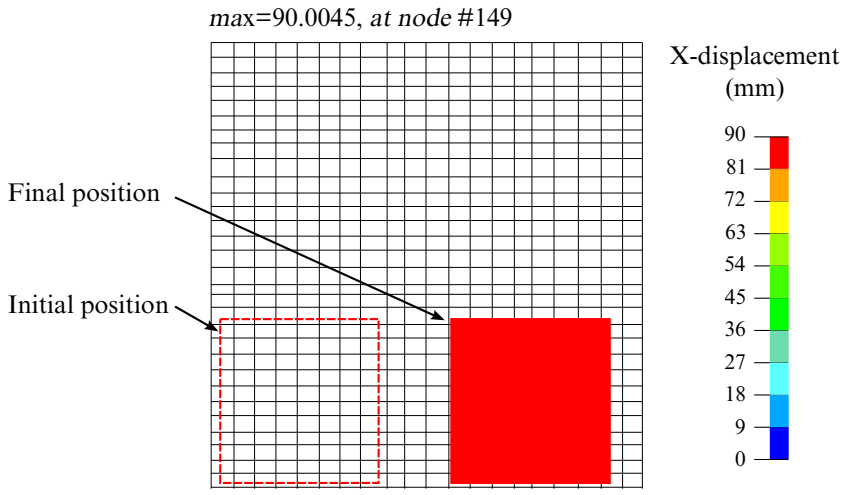


Figure 17-50. Initial and final position of the blank.

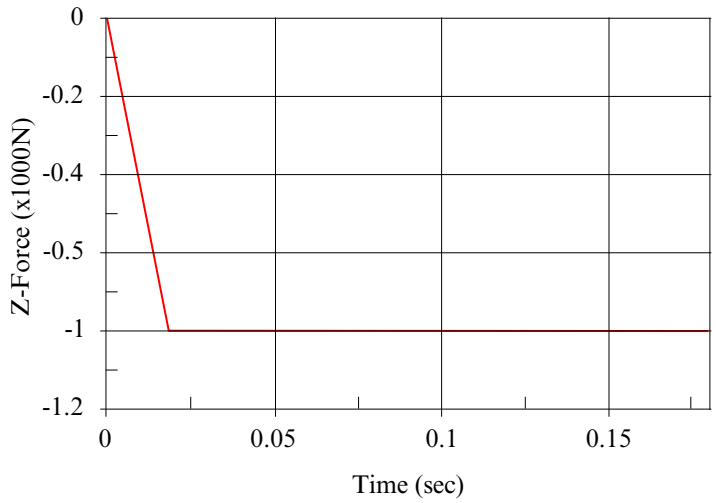


Figure 17-51. Normal force from RCFORC file.

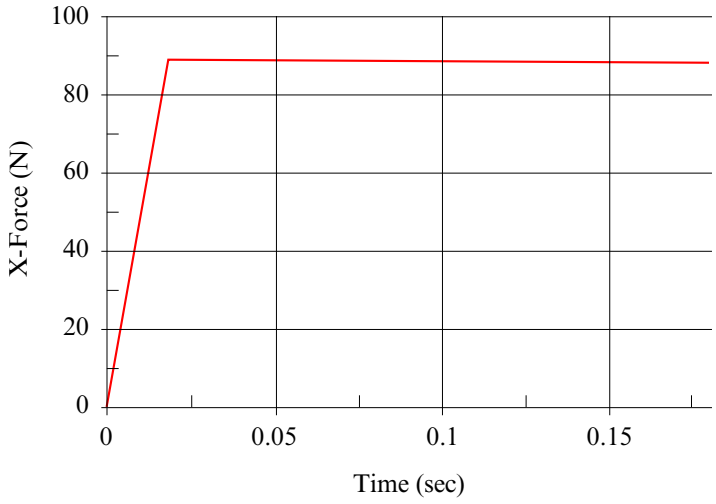


Figure 17-52. Pulling force (frictional force) from RCFORC file.

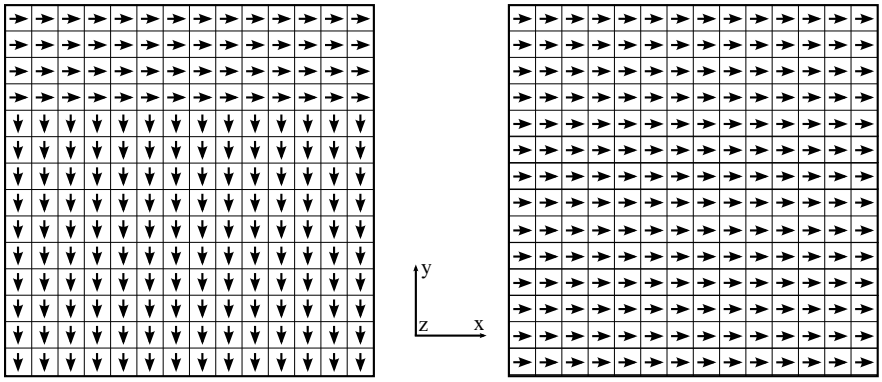


Figure 17-53. Element directions (N1-N2) of an incoming sheet blank (left) and directions after re-orientation.

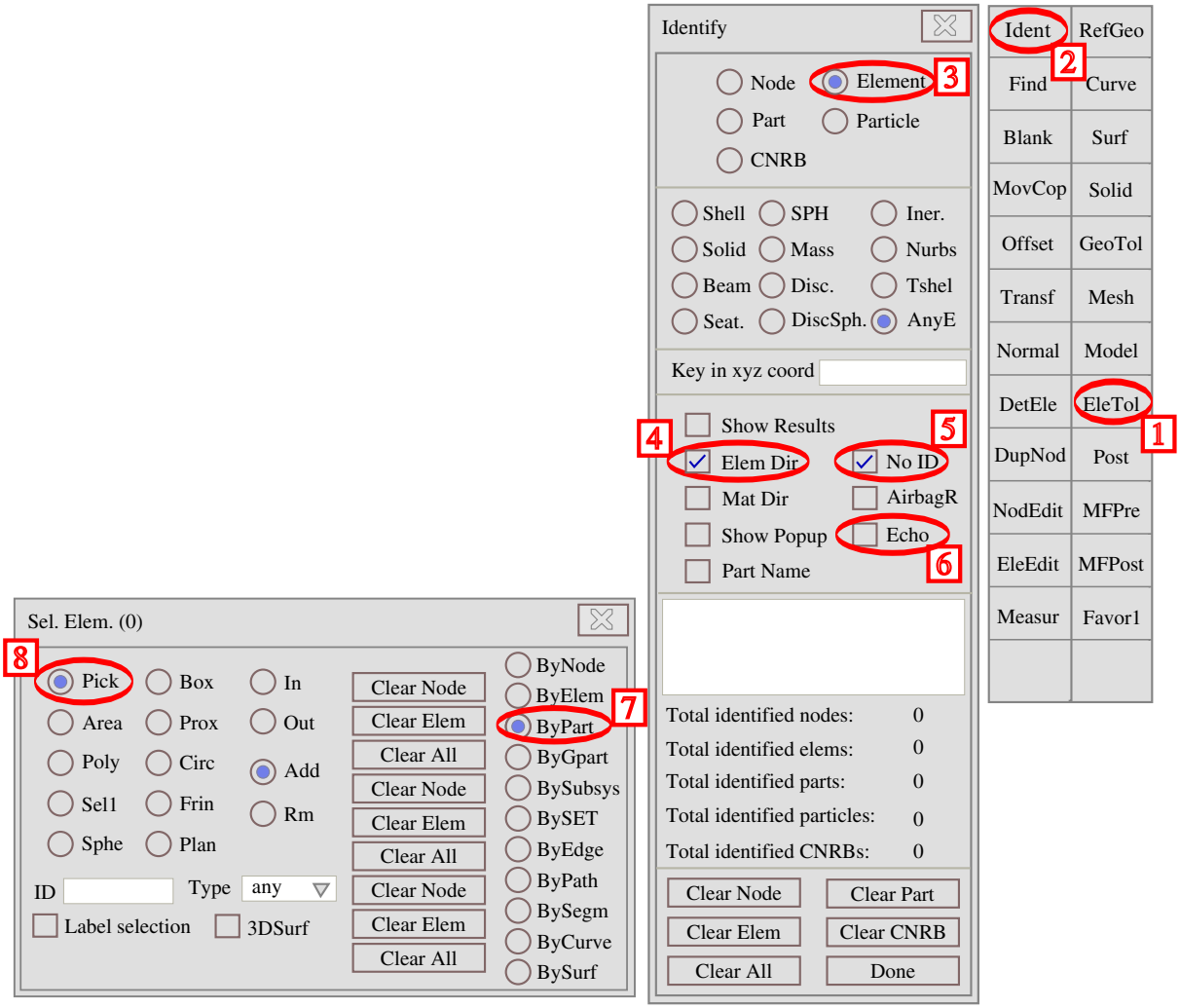


Figure 17-54. Checking element directions (N1-N2) by part using LS-PrePost

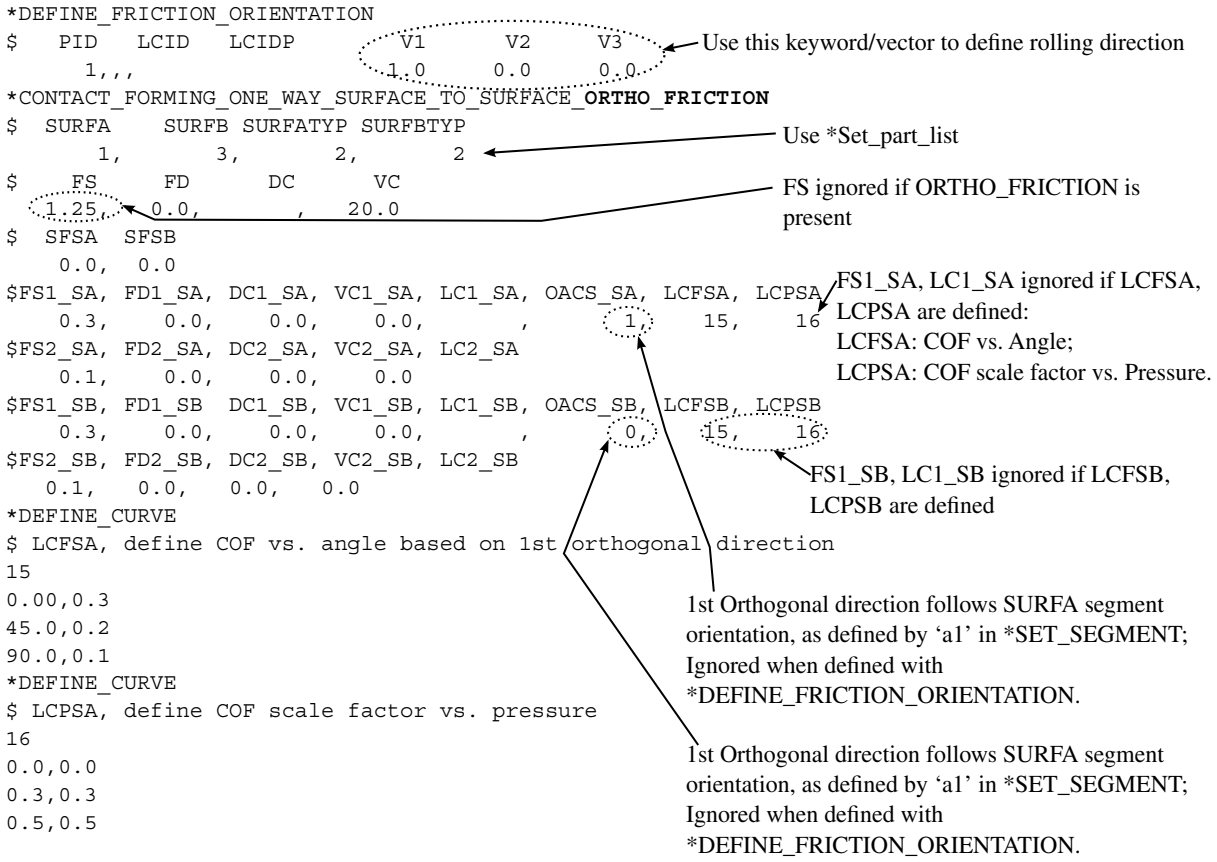


Figure 17-55. Use of this keyword with _ORTHO_FRICTION for MPP.

*DEFINE_FRICTION_SCALING

Purpose: Define scale factors for contact friction for the inner and outer surfaces of shell contact segments. This option is currently only available for contact interfaces that have SOFT set to 2 on optional card A of the *CONTACT input data.

Include as many of Card 1 as desired. The next keyword ("*") card terminates this input. Default friction values are used for any part ID pair that is not defined.

Card 1	1	2	3	4	5	6	7	8
Variable	FSID	CID	PSID	SCALEI	SCALEO			
Type	I	F	F	F	F			
Default	none	0.0	0.0	1.0	1.0			

VARIABLE	DESCRIPTION
FSID	Friction scaling ID number. Each friction scaling definition should have a unique ID which is used for output messages only.
CID	Contact ID. Optional input to limit friction scaling to one contact interface with this ID number.
PSID	Part set ID. Optional input to limit friction scaling to parts in the set.
SCALEI	Friction scale factor for the inner surface of shell segments
SCALEO	Friction scale factor for the outer surface of shell segments

Remarks:

Some materials, such as fabrics, have friction coefficients on one side that differ from the other side. To enable the contact friction to mimic this behavior, separate scale factors can be defined for the inner and outer surface of contact segments that are attached to thin shell elements. The inner and outer surfaces are defined by using the right hand rule with node numbering of the segments. Segment node numbering is consistent with shell element numbering if the contact surface is defined by part or part set.

When modeling an airbag, the inner and outer surface may not be consistent with element or segment numbering, but will depend on how the airbag is defined. When an airbag is

defined using the particle method (see *AIRBAG_PARTICLE), then the SCALEO parameter will scale the friction on the inner surface of the bag, and SCALEI will scale the outer surface. When defined by control volumes (see *AIRBAG), the SCALEI parameter will scale friction on the inner surface of the bag and SCALEO the scale the outer surface.

This option is compatible with all other options to define the friction coefficients as the scale factors are applied after the Coulomb coefficient, μ_c, has been calculated.

*DEFINE_FUNCTION

Purpose: Define a function that can be referenced by a limited number of keyword options. The function arguments are different for each keyword that references *DEFINE_FUNCTION. Unless stated otherwise, all the listed argument(s) in their correct order must be included in the argument list. Some usages of *DEFINE_FUNCTION allow random ordering of arguments and argument dropouts. See the individual keywords for the correct format. Some examples are shown below.

> **NOTE:** Unless the type of the function is explicitly given (as in Example 2 below), the function will return an integer value if the name of the function starts with a letter in the range of i - n. Otherwise it will return a real value. For example, if a function is defined as "ifunc(x) = sqrt(x)," then ifunc(2.0) will return 1, not 1.414.

The TITLE option is not allowed with *DEFINE_FUNCTION.

Card 1	1	2	3	4	5	6	7	8
Variable	FID	HEADING						
Type	I	A70						

Function Cards. Insert as many cards as needed. These cards are combined to form a single line of input. The next keyword ("*") card terminates this input.

Card 2	1	2	3	4	5	6	7	8
Variable	FUNCTION							
Type	A80							

VARIABLE	DESCRIPTION
FID	Function ID. Functions, tables (see *DEFINE_TABLE), and load curves may not share common IDs. A unique number must be defined.
HEADING	An optional descriptive heading
FUNCTION	Arithmetic expression involving a combination of independent variables and other functions, that is, "f(a,b,c)=a*2+b*c+sqrt(a*c),"

VARIABLE	DESCRIPTION
	where a, b, and c are the independent variables. The function name, "f(a,b,c)," must be unique since other functions can then use and reference this function. For example, "g(a,b,c,d)=f(a,b,c)**2+d." In this example, two *DEFINE_FUNCTION definitions are needed to define functions f and g.

Remarks:

1. **Reserved names for variables and constants.** Certain useful constants are intrinsic and available through reserved names. For example, PI is the proportionality constant relating the circumference of a circle to its diameter, while DTOR and RTOD are, respectively, used to convert from degrees to radians and from radians to degrees. Also, TIME stores the current simulation time. See Appendix U for a complete list of variable names reserved internally by LS-DYNA.

2. **Trigonometric and other intrinsic functions.** Most trigonometric and other mathematical functions available in the Fortran and C programming languages are valid in *DEFINE_FUNCTION, such as SIN, COS, ABS, and MAX. Please see a more complete list under "Intrinsic Functions" in *DEFINE_CURVE_FUNCTION.

3. **Units of angular measures.** Unless otherwise noted units of radians are always used for the arguments and output of functions involving angular measures.

4. **Dynamic relaxation.** Unlike *DEFINE_CURVE and *DEFINE_CURVE_FUNCTION, *DEFINE_FUNCTION is always active in dynamic relaxation phase.

5. **Local user variables.** Some keywords, such as *EM_EOS_TABULATED2 and *EM_RANDLES_BATMAC (with RDLTYPE = -1), support *DEFINE_FUNCTIONs that include local user variables as arguments. These are variables that you can define under certain argument names that will be saved during the run and can be modified during each call to the *DEFINE_FUNCTION. They allow you to have your own trackable local variable. Note that for the variable value to be saved from the function, a "*" must proceed the variable name in the function. See Example 3 below.

The following examples serve only as an illustration of syntax.

Example 1:

Prescribe sinusoidal *x*-velocity and *z*-velocity for some nodes.

```
*BOUNDARY_PRESCRIBED_MOTION_SET
$#     nsid       dof       vad      lcid        sf
         1         1         0         1
         1         3         0         2
*DEFINE_FUNCTION
1,x-velo
x(t)=1000*sin(100*t)
*DEFINE_FUNCTION
2,z-velo
z(t)=x(t)+200
```

Example 2:

Ramp up a hydrostatic pressure on a submerged surface.

```
*comment
units: mks
Apply a hydrostatic pressure ramped up over a finite time = trise.
pressure on segment = rho * grav * depth of water
where depth of water is refy - y-coordinate of segment
and refy is the y-coordinate of the water surface

*DEFINE_FUNCTION
10
float hpres(float t, float x, float y, float z, float x0, float y0, float z0)
{
  float  fac, trise, refy, rho, grav;
  trise=0.1; refy=0.5; rho=1000.; grav=9.81;
  fac=1.0;
  if(t<=trise) fac=t/trise;
  return fac*rho*grav*(refy-y);
}
*LOAD_SEGMENT_SET
1,10
```

Example 2 illustrates that a programming language resembling C can be used to define a function. Before a variable or function is used, its type must be declared. For example, variable *t* is real, so it has "float" preceding it in the function declaration. The braces indicate the beginning and end of the function being programmed. Semicolons must appear after each statement but several statements may appear on a single line. Please refer to a C programming guide for more detailed information.

Example 3:

For certain keywords, local user variables are supported as arguments, where the variable value of a local user variable is saved between calls to the function and can be modified in the function. The user variable name must have * before it to be saved. The following partial keyword input for a BatMac Randles circuit provides an example:

```
*EM_RANDLES_BATMAC
$--------1---------2---------3---------4---------5---------6---------7---------8
$#   rdlid   rdltype   rdlarea   ccppart   ccnpart   seppart poselpart negelpart
         4        -1         2         8
$#       q        cq   socinit    soctou

$#   r0cha     r0dis    r10cha    r10dis    c10cha    c10dis
       -10       -11
$ The following two define functions show how a Randles circuit order 2
$ can be defined using a user-defined-circuit.  Define function 10
$ defines the term sent to the left hand side of the stiffness matrix
$ (unit : resistance)
$ while Define function 11 defines the term sent to the right hand side
$ of the stiffness matrix (unit: current)
*DEFINE_FUNCTION
10
float randleLHS(float time,float efstrain,
                            float x_sep, float y_sep, float z_sep,
                            float x_sen, float y_sen, float z_sen,
                            float x_ccp, float y_ccp, float z_ccp,
                            float x_ccn, float y_ccn, float z_ccn,
                            float areaCircuit, float areaCell,float soc)
{
  float r00;
  float factor;
  r00=0.02;
  factor=areaCircuit/areaCell;
  return 1./(factor*1./r00) ;
}
*DEFINE_FUNCTION
11
$ save1, save2 and save3 are internal variables
$ that are locally saved by the define_function
$ (up to save10)
$ save1 represents the SOC
$ save2 represents the Vc
$ save3 represents the Vc2
float randleRHS(float time,float efstrain,
                            float x_sep, float y_sep, float z_sep,
                            float x_sen, float y_sen, float z_sen,
                            float x_ccp, float y_ccp, float z_ccp,
                            float x_ccn, float y_ccn, float z_ccn,
                            float areaCircuit, float areaCell,
                            float emdt, float current, float soc,
                            float *save1, float *save2, float *save3)
{
  float u0;
  float volt;
  float factor;
  float Q;
  float Cq;
  float r00;
  float c10;
  float r10;
  float c20;
  float r20;
  float rhs;
  u0=4.15;
  r00=0.02;
  Q=20.;
  r10=0.002;
  c10=500.;
  r20=0.001;
  c20=1000.;
  Cq=1./36.;
  float vc=0.;
```

```
  float vc2=0.;
  factor=areaCircuit/areaCell;
  Q=factor*Q;
  r10=r10/factor;
  c10=factor*c10;
  r20=r20/factor;
  c20=factor*c20;
  r00=r00/factor;
  *save1=*save1-emdt*current*cq/Q;
  if(time<emdt){
  *save1=100;
  *save2=0.;
  *save3=0.;
  rhs=u0/(r00);
  return rhs;}
  *save2=(*save2-r10*current)*exp(-emdt/(r10*c10))+current*r10;
  *save3=(*save3-r20*current)*exp(-emdt/(r20*c20))+current*r20;
  if (*save1>10.){
   volt=3.6+0.55*(*save1)/100.;
   rhs=(volt-*save2-*save3)/(r00);
   return rhs;}
   if (*save1<0.){
     volt=2.8;
     return 0.;}
   volt=2.8+0.855*(*save1)/10.;
   rhs=(volt-*save2-*save3)/(r00);
   return rhs;
}
```

*DEFINE_FUNCTION_TABULATED

Purpose: Define a function of one variable using two columns of input data (in the manner of *DEFINE_CURVE) that can be referenced by a limited number of keyword options or by other functions defined via *DEFINE_FUNCTION. This command must appear in the keyword deck before the function it defines is used.

The TITLE option is not allowed with *DEFINE_FUNCTION_TABULATED.

Card 1	1	2	3	4	5	6	7	8
Variable	FID	HEADING						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	FUNCTION							
Type	A80							

Point Cards. Put one pair of points per card (2E20.0). Add as many cards as necessary. Input is terminated when a keyword ("*") card is found.

Cards 3	1	2	3	4	5	6	7	8
Variable	A1		01					
Type	F		F					
Default	0.0		0.0					

VARIABLE	DESCRIPTION
FID	Function ID. Functions, tables (see *DEFINE_TABLE), and load curves may not share common ID's. A unique number has to be defined.
HEADING	An optional descriptive heading.

VARIABLE	DESCRIPTION
FUNCTION	Function name.
A1, A2, …	Abscissa values.
O1, O2, …	Ordinate (function) values.

Example:

```
*BOUNDARY_PRESCRIBED_MOTION_SET
$  function 300 prescribes z-acceleration of node set 1000
1000,3,1,300
*DEFINE_FUNCTION_TABULATED
201
tabfunc
0., 200
0.03, 2000.
1.0, 2000.
*DEFINE_FUNCTION
300
a(t)=tabfunc(t)*t
$ for t < 0.03, function 300 is equivalent to a(t)=(200. + 60000.*t)*t
$ for t >= 0.03, function 300 is equivalent to a(t)=(2000.)*t
```

*DEFINE_GROUND_MOTION

Purpose: Define an earthquake ground motion history using ground motion records provided as load curves, for use in conjunction with *LOAD_SEISMIC_SSI for dynamic earthquake analysis including nonlinear soil-structure interaction.

Card 1	1	2	3	4	5	6	7	8
Variable	GMID	ALCID	VLCID					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
GMID	Ground motion ID. A unique number has to be defined.
ALCID	Load curve ID of ground acceleration history.
VLCID	Load curve ID of ground velocity history.

Remarks:

1. **Earthquake Ground Motion Data.** Earthquake ground motion data is typically available either only as ground accelerations, or as a triple of ground accelerations, velocities and displacements. Usually, the velocities and the displacements are computed from the accelerations using specialized filtering and baseline correction techniques. Either input is accepted, with each quantity specified as a load curve. Only the acceleration and the velocity are required in the latter case; LS-DYNA does not require the ground displacement.

2. **Generated Velocity Load Curve.** If only the ground acceleration data is provided for a particular ground motion, LS-DYNA generates a corresponding load curve for the velocity by integrating the acceleration numerically. The generated load curves are printed out to the d3hsp file. It is up to the user to ensure that these generated load curves are satisfactory for the analysis.

*DEFINE_HAZ_PROPERTIES

Purpose: To model the heat affected zone in a welded structure, the yield stress and failure strain are scaled in shell models as a function of their distance from spot welds and the nodes specified in *DEFINE_HAZ_TAILOR_WELDED_BLANK. *DEFINE_HAZ_-PROPERTIES currently supports spot welds defined with *CONSTRAINED_SPR2 and *CONSTRAINED_INTERPOLATION_SPOTWELD. It also supports spot welds defined with beam or solid elements using *MAT_SPOTWELD. *CONSTRAINED_SPOTWELD is not currently supported.

Card 1	1	2	3	4	5	6	7	8
Variable	ID_HAZ	IOP	PID	PID_TYP				
Type	I	I	I	I				
Default	0	0	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	ISS	IFS	ISB	IFB	ISC	IFC	ISW	IFW
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
ID_HAZ	Property set ID. A unique ID number must be used.
IOP	Activate scaling flag: EQ.0: Scaling not activated EQ.1: Scaling activated
PID	Part or part set ID
PID_TYP	PID type: EQ.0: Part ID EQ.1: Part set ID

VARIABLE	DESCRIPTION
ISS	Curve ID for scaling the yield stress based on the distance to the closest solid element spot weld.
IFS	Curve ID for scaling the failure strain based on the distance to the closest solid element spot weld.
ISB	Curve ID for scaling the yield stress based on the distance to the closest beam element spot weld. Use a negative ID for curves normalized by the spot weld diameter as described in Remark 2.
IFB	Curve ID for scaling the failure strain based on the distance to the closest beam element spot weld. Use a negative ID for curves normalized by the spot weld diameter as described in Remark 2.
ISC	Curve ID for scaling the yield stress based on the distance to the closest constrained spot weld.
IFC	Curve ID for scaling the failure strain based on the distance to the closest constrained spot weld.
ISW	Curve ID for scaling the yield stress based on the distance to the closest tailor-welded blank node.
IFW	Curve ID for scaling the failure strain based on the distance to the tailor-welded blank node.

Remarks:

1. **Scaling.** The yield stress and failure strain are assumed to vary radially as a function of the distance of a point to its neighboring spot welds. Since larger spot welds may have a larger radius of influence, the smallest scale factor for the yield stress from all the neighboring spot welds is chosen to scale the yield stress at a particular point. The failure strain uses the scaling curve for the same weld.

2. **Weld diameter normalization.** Curve IDs may be input as negative values to indicate that they are normalized by the diameter of the spot weld to compensate for the effects of the spot weld size. When this option is used, the scale factor is calculated based on the distance divided by the spot weld diameter for the spot weld that is closest to the element.

3. **Distance measurement.** The distance from a spot weld (or node for the blank) is measured along the surface of the parts in the part set. This prevents the heat-softening effects of a weld from jumping across empty space.

4. **Material restrictions.** The HAZ capability only works with parts with materials using the STOCHASTIC option. It may optionally be simultaneously used with *DEFINE_STOCHASTIC_VARIATION to also account for the spatial variations in the material properties. See *DEFINE_STOCHASTIC_VARIATION for more details.

5. **Extra history variables.** The scale factors on yield stress and failure strain may be stored as extra history variables as follows:

Material Model	History Variable # for Scaling Factor on Yield Stress	History Variable # for Scaling Factor on Failure Strain
10	5	6
15	7	8
24	6	7
81	6	7
98	7	8
123 (shells only)	6	7
GISSMO	n/a	ND+20

When the HAZ capability is used in combination with *DEFINE_STOCHASTIC_VARIATION, the history variables shown in the table above represent the net scale factors due to HAZ and stochastic variation.

*DEFINE_HAZ_TAILOR_WELDED_BLANK

Purpose: Specify nodes of a line weld such as in a Tailor Welded Blank. The yield stress and failure strain of the shell elements in the heat affected zone (HAZ) of this weld are scaled according to *DEFINE_HAZ_PROPERTIES.

Card 1	1	2	3	4	5	6	7	8
Variable	IDTWB	IDNS	IDP	IPFLAG	IMONFLAG			
Type	I	I	I	I	I			
Default	0	0	0	0	0			

VARIABLE	DESCRIPTION
IDTWB	Tailor Welded Blank ID
IDNS	Node Set ID defining the location of the line weld.
IDP	Part or part set ID. Applies to all HAZ parts if IDP = 0 (default).
IPFLAG	IDP type: EQ.0: Part ID (default) EQ.1: Part set ID
IMONFLAG	Monotonicity flag for load curves ISW and IFW on *DEFINE_HAZ_PROPERTIES: EQ.0: ISW and IFW increase monotonically. EQ.1: ISW and IFW are allowed to be arbitrary load curves.

*DEFINE_HEX_SPOTWELD_ASSEMBLY_{OPTION}

Purpose: Define a list of hexahedral elements that make up a single spot weld for computing the force and moment resultants that are written into the swforc output file. A maximum of 16 elements may be used to define an assembly representing a single spot weld. See Figure 17-56. This table of element IDs is generated automatically when beam elements are converted to solid elements. See the input parameter RPBHX associated with the keyword *CONTROL_SPOTWELD_BEAM.

Available options for this command are:

<BLANK>

N

For the <BLANK> option, all solid elements specified on Card 2 make up the spot weld and no additional card is read. For the N option, N is an integer representing the total number of solid elements making up the spot weld. If N is greater than 8, the additional card beyond Card 2 is read. N may not exceed 16.

Card 1	1	2	3	4	5	6	7	8
Variable	ID_SW							
Type	I							
Default	0							

Card 2	1	2	3	4	5	6	7	8
Variable	EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

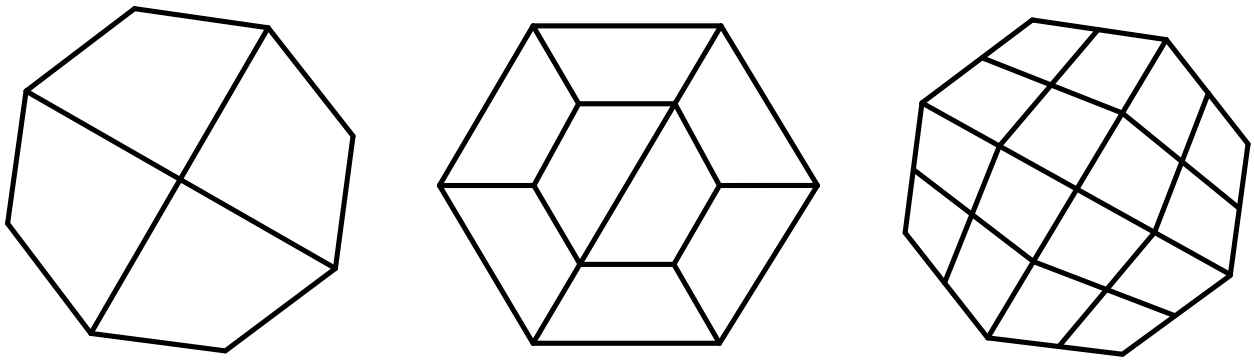


Figure 17-56. Illustration of four, eight, and sixteen element assemblies of solid hexahedron elements forming a single spot weld.

Additional card for N > 8.

Optional	1	2	3	4	5	6	7	8
Variable	EID9	EID10	EID11	EID12	EID13	EID14	EID15	EID16
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
ID_SW	Spot weld ID. A unique ID number must be used.
EID*n*	Element ID *n* for up to 16 solid hexahedral elements.

Remarks:

The elements comprising a spot weld assembly may share a part ID (PID) with elements in other spot weld assemblies defined using *DEFINE_HEX_SPOTWELD_ASSEMBLY but may not share a PID or even a material ID (MID) with elements that are not included in a *DEFINE_HEX_SPOTWELD_ASSEMBLY.

*DEFINE_ISPG_TO_SURFACE_COUPLING

Purpose: Define a coupling interface between fluid particles modeled with incompressible smoothed particle Galerkin (ISPG) and a surface. The coupling can be tied except on the edges, entirely tied, or free slip.

> **NOTE:** This keyword is only available for ISPG parts with sections specified with *SECTION_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	FP	SURF	FPTYPE	SURFTYPE	CPL_ID			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

Card 2	1	2	3	4	5	6	7	8
Variable	SBC	SCA	ACA	RCA	VCA	SFPN	THK	LSLP
Type	I	F	F	F	F	F	F	F
Default	0	none	SCA	SCA	1.0	1.0	0.5	0.0

Card 3	1	2	3	4	5	6	7	8
Variable	HTC							
Type	F							
Default	2.0E-5							

VARIABLE	DESCRIPTION
FP	Part or part set ID for the fluid particles

VARIABLE	DESCRIPTION
SURF	Segment set ID specifying the surface. Currently, the segment set should be generated from 8-node hexahedral elements.
FPTYPE	Type for FP: EQ.0: Part set ID EQ.1: Part ID
SURFTYPE	Type for SURF: EQ.0: Segment set ID
CPL_ID	ID of the coupling
SBC	Boundary condition type for the contact edges (see Figure 17-57) and contact treatment for the non-contact-edge ISPG nodes: EQ.0: Contact edges can move on the structure's surface, while non-contact-edge ISPG nodes in contact with the structure are tied to the structure, modeling a non-slip boundary condition. EQ.1: All ISPG nodes in contact with the structure, including those on the contact edges, are tied to the structure. EQ.2: A free slip condition applies to all ISPG nodes in contact with the structure, including those on the contact edges.
SCA	Static (equilibrium) contact angle in radians. See Remark 1.
ACA	Advancing contact angle in radians. If set to zero, then the value of SCA is used. See Remark 1.
RCA	Receding contact angle in radians. If set to zero, then the value of SCA is used. See Remark 1.
VCA	Velocity of the fluid contact edge for measurement of the contact angles. See Remark 1.
SFPN	Scale factor for the thickness of the solid elements coupled with the ISPG fluid. This parameter can increase the contact detection thickness if the length of the solid element along the contact normal direction is much less than the length in the in-plane direction.
THK	Scale factor on the average ISPG element edge length for contact detection

VARIABLE	DESCRIPTION
LSLP	Slip length, L_{slip}, for the Navier slip model (see Figure 17-58). A smaller slip length causes more friction. By default, LSLP = 0.0, meaning no Navier slip force is calculated. This field is ignored for SBC = 1.
HTC	Heat transfer coefficient for the thermal boundary condition between the ISPG nodes and the structural surface if performing a coupled thermal-flow analysis. The default value is 2.0×10^{-5} kW/(mm^{2}K) with its base units assumed to be kg, mm, ms, and K.

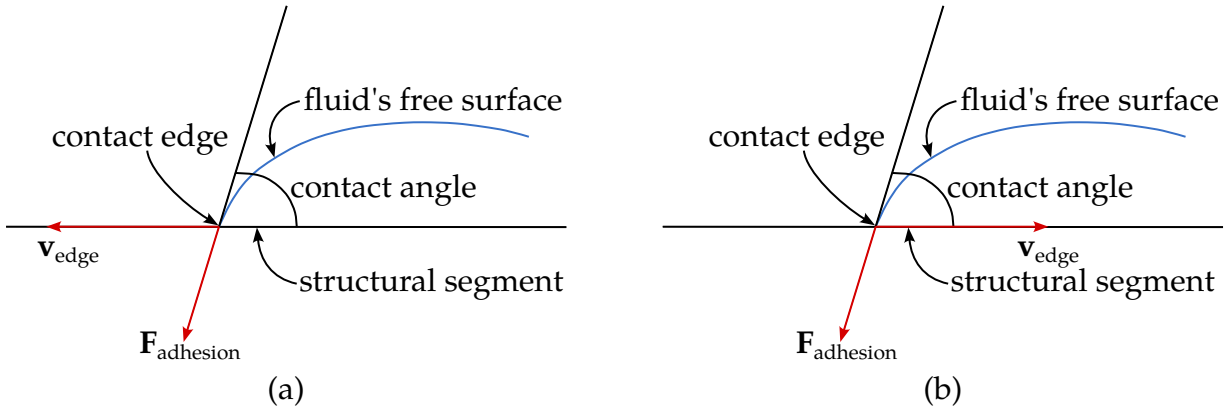


Figure 17-57. Example showing contact between an ISPG fluid and structural segments. When the fluid moves as shown in (a), the fluid moves "forward" or advances along the structural segment. If the fluid moves as shown in (b), the fluid moves "backward" or recedes along the structure.

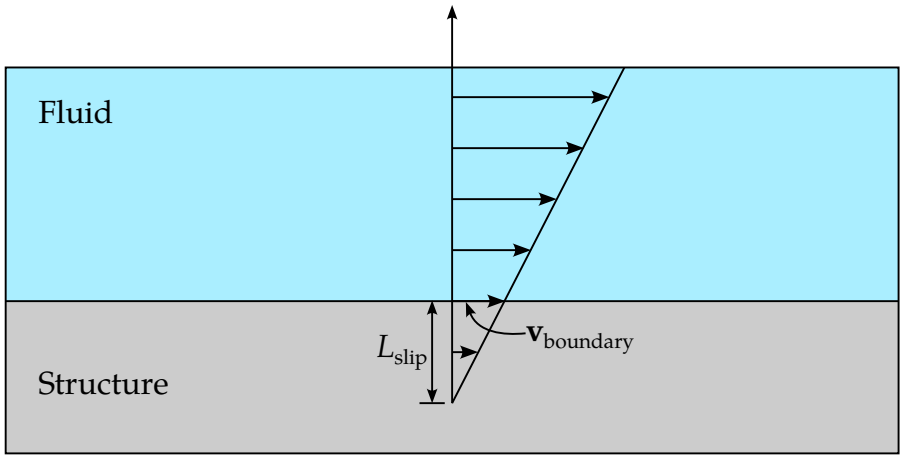


Figure 17-58. Navier slip boundary condition. The shear stress is given by $\boldsymbol{\tau}_{\text{friction}} = \frac{\mu}{L_{slip}}\mathbf{v}_{\text{boundary}}$, where μ is the viscosity, L_{slip} is slip length, and $\mathbf{v}_{\text{boundary}}$ is the relative tangential velocity of the fluid at the boundary.

Remarks:

1. **Adhesion force and contact angles.** For SBC ≠ 1, the wall adhesion force, $\mathbf{F}_{\text{adhesion}}$, acts on the contact edge of the ISPG fluid. $\mathbf{F}_{\text{adhesion}}$ is towards the structure's interior, and the contact angle determines the direction of the $\mathbf{F}_{\text{adhesion}}$, as illustrated in Figure 17-57.

   The contact angle depends on the ISPG fluid edge velocity relative to the structural segment. When the ISPG fluid moves forward, as shown in the portion of Figure 17-57 labeled (a), the contact angle is:

$$\begin{cases} \text{ACA} & \text{if } |\mathbf{v}_{\text{edge}}| > \text{VCA} \\ \text{ACA} + \dfrac{(\text{SCA} - \text{ACA})(\text{VCA} - |\mathbf{v}_{\text{edge}}|)}{\text{VCA}} & \text{if } |\mathbf{v}_{\text{edge}}| \le \text{VCA} \end{cases}$$

   When it moves backward, as illustrated by (b) in Figure 17-57, the contact angle is:

$$\begin{cases} \text{RCA} + \dfrac{(\text{SCA} - \text{RCA})(\text{VCA} - |\mathbf{v}_{\text{edge}}|)}{\text{VCA}} & \text{if } |\mathbf{v}_{\text{edge}}| \le \text{VCA} \\ \text{RCA} & \text{if } |\mathbf{v}_{\text{edge}}| > \text{VCA} \end{cases}$$

*DEFINE_LANCE_SEED_POINT_COORDINATES

Purpose: Define seed points that determine the portion of a part(s) that is not removed (not scraps) during the trimming portion of lancing. This keyword is to be used in conjunction with *ELEMENT_LANCING. See *ELEMENT_LANCING.

Card 1	1	2	3	4	5	6	7	8
Variable	NSEED	X1	Y1	Z1	X2	Y2	Z2	
Type	I	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
NSEED	Number of seed points. Maximum value of "2" is allowed.
X1, Y1, Z1	Location coordinates of seed point #1.
X2, Y2, Z2	Location coordinates of seed point #2.

Remarks:

1. **Trimming Restrictions.** This keyword will remove all scraps during or after lancing, depending on how the parameter AT is defined in *ELEMENT_LANCING. Lancing curves must form a closed loop, meaning the first and last point coordinates must be coincident. Scraps are the portions that are exclusive of the portions whose seed points are defined by this keyword.

2. **Example.** The following input defines two sets of seed point coordinates, where a double-attached part may be lanced and trimmed:

```
*DEFINE_LANCE_SEED_POINT_COORDINATES
$    NSEED         X1        Y1          Z1        X2         Y2         Z2
         2     -289.4     98.13    2354.679    -889.4     91.13     255.679
```

Revision Information

This feature is available starting with LS-DYNA Revision 107262.

*DEFINE_MATERIAL_HISTORIES_{OPTION}

The available options include:

<BLANK>

NAMES

Purpose: Control the content of the history variables in the d3plot and d3part databases. This feature is supported for solid, shell, and integrated beam elements. See Motivation below for the rationale behind this keyword and How this Keyword Works for a description of how the history variables are output. Note that all mentions of d3plot also apply to d3part which is excluded for clarity.

The NAMES keyword option enables giving each specified history variable a corresponding name. These names are reflected in d3hsp and a post-processing file containing the history variable names if HISNOUT > 0 on *CONTROL_OUTPUT.

> **NOTE:** Whenever the input file includes this keyword, the effective plastic strain output is zero for all materials for which there is no plastic strain (see Remark 3 for more details).

Card Summary:

Card Sets. Define as many instantiations of Card 1 or sets of Card 1 and Card 2 for the NAMES keyword option as needed to define the extra history variables. This input ends at the next keyword ("*") card. This keyword can be defined without any data cards; see Remark 3 below for details of when this should be used.

Card 1. Include as many of this card or sets of this card with Card 2 for the names keyword option as needed.

LABEL	A1	A2	A3	A4

Card 2. Include this card in sets with Card 1 when the NAMES keyword option is used.

NAME	

Data Card Definitions:

History Variable Card. Define as many cards as needed to define the extra history variables. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	LABEL				A1	A2	A3	A4
Type	A40				F	F	F	F
Default	none				0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
LABEL	String identifying history variable type. Labels are *case-sensitive*. See section LABEL below and Remark 4. By prefixing the label with (capital letter) 'X', the current history variable will not be printed to the binary database. See Example 4.
A*n*	Attributes. See section LABEL below.

User-Defined Name Card. Additional card for the NAMES keyword option. Define this card in sets with Card 1. See Example 2.

Card 2	1	2	3	4	5	6	7	8
Variable	NAME							
Type	A70							

VARIABLE	DESCRIPTION
NAME	User-defined name for history variable

LABEL:

The permitted LABELs (*case-sensitive*) are

Instability	A number between 0 and 1 that indicates how close an element or integration point is to failure or to initiate damage, whatever happens first (see Remark 1). Attributes apply to the DIEM

damage model (see *MAT_ADD_DAMAGE_DIEM), the GISSMO damage model (see *MAT_ADD_DAMAGE_GISSMO), and add erosion model (see *MAT_ADD_EROSION) only. In those cases, the following apply:

A1 value	Description
-12	Instability value for FAILTM on *MAT_ADD_EROSION. FAILTM is assumed nonzero.
-10	Instability value for SIGTH and IMPULSE on *MAT_ADD_EROSION. SIGTH and IMPULSE are assumed nonzero.
-9	Instability value for EPSSH on *MAT_ADD_EROSION. EPSSH is assumed nonzero.
-8	Instability value for MXEPS on *MAT_ADD_EROSION. MXEPS is assumed nonzero.
-7	Instability value for SIGVM on *MAT_ADD_EROSION. SIGVM is assumed nonzero.
-6	Instability value for SIGP1 on *MAT_ADD_EROSION. SIGP1 is assumed nonzero.
-5	Instability value for MNPRES on *MAT_ADD_EROSION. MNPRES is assumed nonzero.
-4	Instability value for VOLEPS on *MAT_ADD_EROSION. VOLEPS is assumed nonzero.
-3	Instability value for EFFEPS on *MAT_ADD_EROSION. EFFEPS is assumed nonzero.
-2	Instability value for MNEPS on *MAT_ADD_EROSION. MNEPS is assumed nonzero.
-1	Instability value for MXPRES on *MAT_ADD_EROSION. MXPRES is assumed nonzero.
1	Max instability value in any integration point and any criterion for DIEM
2	Location (isoparametric z-coordinate between -1 and 1) of max instability value for DIEM
3	Criterion attaining max instability value for DIEM (integer value)
4	Stress triaxiality η for evaluating instability criterion value for the ductile (DITYP = 0) criterion in DIEM

5	Shear influence θ for evaluating instability criterion value for the shear (DITYP = 1) criterion in DIEM
6	Ratio of principal strain rates α for evaluating instability criterion value for the MSFLD (DITYP = 2) criterion in DIEM
7	Ratio of principal strain rates α for evaluating instability criterion value for the FLD (DITYP = 3) criterion in DIEM
8	Stress state parameter β for evaluating instability criterion value for the weighted ductile (DITYP = 4) criterion in DIEM
9	GISSMO's instability measure F (ND + 8)

Damage

A number between 0 and 1 indicating the damage level of the element or integration point (see Remark 1). Attributes apply to the GISSMO damage model (see *MAT_ADD_DAMAGE_GISSMO) only. In those cases, the following apply:

A1 value	Description
0	GISSMO's effective damage value $\widetilde{D}$ (ND+18)
1	GISSMO's damage value D (ND)
2	GISSMO's alternative damage value $D^{1/\mathrm{DMGEXP}}$ (ND+13)

Plastic Strain Rate

Effective plastic strain rate, calculated generically for "all" plastic materials from the evolution of effective plastic strain. No attributes apply to this label.

Effective Stress

Effective, or equivalent, stress. If not specifically stated in the material section for the material in question, this will be the von Mises stress. For selected anisotropic materials (such as the Barlat models), it is the stress value that is conjugate to the effective plastic strain rate with respect to energy density rate.

Effective Tresca Stress

Effective, or equivalent, stress according to Tresca (maximum shear stress):

$$\sigma_e^T = \max[|\sigma_1 - \sigma_2|, |\sigma_2 - \sigma_3|, |\sigma_3 - \sigma_1|] \ ,$$

where σ_1, σ_2 and σ_3 are the principal stresses.

Effective Strain

Effective, or equivalent, (von Mises) strain:

$$\varepsilon_e = \sqrt{\frac{2}{3}\left(\varepsilon_{dev,11}^2 + \varepsilon_{dev,22}^2 + \varepsilon_{dev,33}^2 + 2\varepsilon_{12}^2 + 2\varepsilon_{23}^2 + 2\varepsilon_{31}^2\right)} \ ,$$

where ε_{ij} are the standard LS-DYNA output strains (similar to logarithmic strain, see Chapter 22.10 in the Theory Manual) and ε_{dev} denotes the deviatoric component.

Note, using this label activates (integration point) strain output in *DATABASE_EXTENT_BINARY (i.e. STRFLG(L) = 1 and NINTSLD = 8).

Plastic Energy Density

Plastic work per unit volume. It is only nonzero for materials deemed as plastic materials. This is evaluated as $\int_0^t \sigma \dot{\varepsilon}_{eff}^p$, where σ is the von Mises effective stress and ε_{eff}^p is the work equivalent plastic strain. Thus, the result will not be entirely correct for anisotropic materials. Upon request, we will improve the calculation for the asked for anisotropic materials.

Effective Creep Strain

Effective creep strain for material models with creep. In particular, it will provide the effective creep strain for *MAT_ADD_INELASTICITY if creep is one of the selected inelasticity laws; see this keyword for more details.

History

Fetch a history variable at a known place for a given material type, element type and part set (see Example 1). The following attributes apply:

Attribute	Description
A1	Mandatory (no defaults) GT.0: History variable location LT.0: Load curve ID = (-A1); see below for a complete description
A2	Material type. By default, it applies to all materials.
A3	Element type: 0 for all element types, 1 for solids, 2 for shells and 3 for beams
A4	Part set. History will only be fetched from the parts in the given part set. The default is all parts in the model.

When A1 < 0, then |A1| refers to a *DEFINE_CURVE. In this curve definition you specify an 'operation' that you would like

to perform over a certain list of history variables within an element.

Operation	Description
1	Maximum value
2	Layer where the maximum value occurs
3	History variable that attains the max value
4	Sum the history variables

Only the ordinate values of the load curve are used, but we suggest defining the abscissa value using increasing integer numbers starting with 0. Furthermore, you should use DATTYP = 6 in the curve definition to make sure that the data is not transformed in any way. Here is an example:

```
*DEFINE_CURVE
ID, 0, 1., 1., 0., 0., 6, 0
0, 1 (operation)
1, 12 (location of 1st history variable)
2, 17 (location of 2nd history variable)
3, 22 (location of 3rd history variable)
```

In this particular case, you will receive the maximum value of the history variables 12, 17 and 22 at any integration point within one element.

Operator

Perform an operation on a given history variable or quantity. The following attributes apply:

Attribute	Description
A1	Type of operator: EQ.0: Maximum value over time EQ.1: Point in time when maximum value over time was attained. EQ.2: Minimum value over time EQ.3: Point in time when minimum value over time was attained. EQ.5: Absolute value, |A2| EQ.6: Addition, A2 + A3 EQ.7: Subtraction, A2 - A3 EQ.8: Multiplication, A2 × A3 EQ.9: Division, A2/A3 (assuming A3 > 0) EQ.10: Exponentiation, $A2^{A3}$

	EQ.11: Maximum, max(A2, A3) EQ.12: Minimum, min(A2, A3) EQ.13: Natural logarithm, ln|A2|
A2 / A3	First and second argument to operator: EQ.-16: Real number given by A4 EQ.-9: *x*-stress EQ.-8: *y*-stress EQ.-7: *z*-stress EQ.-6: *xy*-stress EQ.-5: *yz*-stress EQ.-4: *zx*-stress EQ.-3: Mid principal stress EQ.-2: Minor principal stress EQ.-1: Major principal stress EQ.0: Effective plastic strain GT.0: History variable # in list specified here

A4	Parameter for operator A1:
	If A2 or A3 are EQ.-16 then
	Real number (see Example 4)
	Else
	If 0.LE.A1.LE.3 then
	GT.0: Time at which min/max will be reset
	LT.0: |A4| is ID to load curve whose ordinate contains time points when min/max will be reset. Even though only the ordinate values are used, DATTYP = 6 is recommended.
	Else
	LT.0: Shell stress will be rotated to global coordinates.
	Endif
	Endif

The stress components used as arguments (A2 and A3) are for solid elements in global coordinates and for shell elements in element-local (co-rotated) coordinates. However, by setting A4 to a negative value, shell element stresses will be rotated to global coordinates.

See Examples 3 and 4.

Principal Stress Range

Compute the maximum (principal) stress amplitude. This feature is a scanning tool intended for fatigue analysis. Currently, only shell elements are supported. See Remark 5 for details about how it is determined.

The following attributes apply:

Attribute	Description
A1	GT.0: Reset time LT.0: |A1| is ID to load curve whose ordinate contains time points when min/max will be reset. Even though only the ordinate values are used, DATTYP = 6 is recommended.
A2	Number of buckets (NBKTS) each 90° sector is divided into. Default value is 65 buckets.
A3	(not used)
A4	Part set. Stress range will only be computed in the parts in the given part set. The default is all parts in the model.

Max Stress Range

As label *Principal Stress Range* above but not specifically considering the principal stresses; see Remark 5. The drawback of this approach is that close to twice the amount of storage is needed. Therefore, the default number of buckets (NBKTS) is 45.

The same attributes apply as for label *Principal Stress Range.*

Motivation:

Material models in LS-DYNA have history variables that are specific to the constitutive model being used. For most materials, 6 are reserved for the Cauchy stress components and 1 for the effective plastic strain, but many models have more than that. The history variables may include physical quantities like material damage, material phase compositions, strain energy density and strain rate, as well as nonphysical quantities like material direction cosines and scale factors.

By using NEIPS, NEIPB and NEIPH on *DATABASE_EXTENT_BINARY, these extra history variables can be exported to the d3plot database *in the order that they are stored*, and in LS-PrePost the variables may then be plotted (Hist button) or fringed (Misc menu). This approach has a few drawbacks. You must, for instance, have knowledge of the

storage location of a certain history variable for a given material model and element type. While this information can be retrieved either in the LS-DYNA manual, on LS-DYNA support sites or in LS-PrePost itself, it is not always convenient.

Furthermore, the same physical quantity may be stored in different locations for different materials and different element types, meaning that history variable #1 will correspond to different things in different parts which complicates post-processing of large models. You may also be interested in a certain material specific quantity that is not necessarily stored as a history variable; this quantity is not retrievable using this approach. Finally, if the history variable of interest happens to be stored in a bad location, that is, among the last ones in a long list, it would be necessary to set NEIPS, NEIPB and/or NEIPH large enough to access this variable in LS-PrePost. This could result in unnecessarily large binary plot files.

How this Keyword Works:

This keyword attempts to organize the extra history variables with the goal of obtaining an output that is reasonably small and easy to interpret. The input is very simple: use the keyword *DEFINE_MATERIAL_HISTORIES in the keyword input deck, followed by lines that specify the history variables of interest using predetermined labels and attributes. *NEIPS, NEIPB and NEIPH on *DATABASE_EXTENT_BINARY are then overridden by the number of history variables, that is, number of lines, requested on this card*. As an example:

```
*DEFINE_MATERIAL_HISTORIES
Instability
Damage
```

means that two extra history variables are output to the d3plot database, so NEIPS, NEIPB and NEIPH are internally set to 2 regardless of the user input. History variable #1 corresponds to an instability measure (between 0 and 1) and history variable #2 corresponds to a material damage (between 0 and 1). Thus, the history variables are output in the order they are listed. If there are several instances of this keyword in an input deck, the order of the history variables follows the order that the cards are read by the keyword reader.

In the d3hsp file, search for the string "M a t e r i a l H i s t o r y L i s t" to find the complete list. For a material that does not store or calculate an "instability" or "damage" history variable, the output is zero and thus the output is not cluttered by unwanted data. Note that this keyword does not necessarily require that the history variable be stored, as long as it can be calculated when LS-DYNA outputs a plot state (see Remark 7 regarding storage). Thus, you can possibly request quantities that are not available by just using NEIPS, NEIPB and/or NEIPH on *DATABASE_EXTENT_BINARY.

Remarks:

1. **Damage and instability.** For large models with many different parts and materials, the *Instability* and *Damage* variables should provide a comprehensive overview and understanding of the critical areas in terms of failure that otherwise may not be assessable. Note that *Damage* can only be nonzero after *Instability* reaches 1, and thereafter the integration point fails when *Damage* reaches 1. If the material has no damage evolution law (which is the case in many of the *MAT_ADD_EROSION features for which failure of the integration point occurs immediately upon reaching the failure criterion without any stress softening), then in general *Damage* will jump from 0 to 1 exactly when *Instability* reaches 1.

2. **History variable requests from *DATABASE_EXTENT_BINARY.** As discussed in How this Keyword Works, NEIPS, NEIPB, and NEIPH set in *DATABASE_EXTENT_BINARY are ignored when the input deck includes *DEFINE_MATERIAL_HISTORIES. The number of extra history variables output to d3plot is determined by how many instantiations of Card 1 are included in the input deck. For instance, if only instability and damage are specified with this keyword, then two extra history variables are output.

3. **Effective plastic strain.** Whenever *DEFINE_MATERIAL_HISTORIES is present in the input file, the effective plastic strain output will be zero for all materials for which there is no plastic strain. Thus, this slot in the d3plot database is in this case *not* taken over by something else, but rather occupied by zero data. For instance, for *MAT_ELASTIC, it will be a null field while it will be the effective plastic strain field for *MAT_PIECEWISE_LINEAR_PLASTICITY. For a part that uses *MAT_ADD_INELASTICITY with a plastic model, the effective plastic strain will be the one pertaining to this inelasticity law. Thus, *DEFINE_MATERIAL_HISTORIES may be used without any cards if this behavior is desired for the effective plastic strain.

4. **Requestable history variables types.** At the end of the remarks for a material model in Volume II, the history variable types (other than *History*) available are listed in a table similar to the one below. *History* is available for all materials, but retrievable ones depend on the model. Also, whether mentioned in such a table or not, *Plastic Strain Rate* and *Plastic Energy Density* available for any material model that calculates plastic strain as the 7th standard history variable.

**DEFINE_MATERIAL_HISTORIES Properties*

Label	Attributes	Description
Instability	- - - -	Failure indicator $\varepsilon^p_{\text{eff}}/\varepsilon^p_{\text{fail}}$, see FAIL
Plastic Strain Rate	- - - -	Effective plastic strain rate $\dot{\varepsilon}^p_{\text{eff}}$

*DEFINE_MATERIAL_HISTORIES Properties					
Label	**Attributes**				**Description**
Plastic Energy Density	-	-	-	-	Plastic energy density $\int_0^t \sigma \dot{\varepsilon}^p_{\text{eff}}$

Label in the table states what the string LABEL on *DEFINE_MATERIAL_HISTORIES must be, *a1* to *a4* will list attributes A1 to A4 if necessary, and *Description* will be a short description of what is output with this option, including possible restrictions.

5. **Calculating principal stress range.** Assuming plane stress ($\sigma_z = 0$), the normal stress at an angle φ to the local element x-axis is

$$\sigma(\varphi) = \frac{\sigma_x + \sigma_y}{2} + \frac{\sigma_x - \sigma_y}{2}\cos 2\varphi + \tau_{xy}\sin 2\varphi.$$

Specifically, the major principal stress and its direction are

$$\sigma_1 = \frac{\sigma_x + \sigma_y}{2} \pm \sqrt{\left(\frac{\sigma_x - \sigma_y}{2}\right)^2 + \tau_{xy}^2}$$
$$\varphi_1 = \frac{1}{2}\tan^{-1}\frac{2\tau_{xy}}{\sigma_x - \sigma_y}$$

Using the equations above, the range of angles $\varphi \in [-90°, 90°]$ is divided into $2 \times$ NBKTS buckets with each bucket covering $90°/\text{NBKTS}$ degrees. In each bucket the minimum stress over time is stored. At the same time the over-time maximum major principal stress and its direction is stored. The stress range is then computed as the difference between the major principal stress and the minimum stress in the corresponding direction (i.e., bucket). Output to the binary database is the over-time maximum stress range.

The computed principal stress range is exact in the case when the angle of the minimum minor principal stress is the same as the angle of the maximum major principal stress. In most other cases the computed stress range is an overestimate. It is estimated that the worst overestimate would be close to $2\cos(90° - 90°/\text{NBKTS}) \times 100\%$ of the minimum minor principal stress in size.

Please note that besides the maximum stress range, $2 \times \text{NBKTS} + 2$ values will be stored for each integration point. Therefore, we recommend limiting the number of elements for which this label is used.

The alternative label *Max Stress Range* does not consider the direction of the major principal stress. Instead, in each bucket the stress range is computed as the difference of the over-time maximum and minimum normal stress. Output to the binary database is the over-time maximum stress range of all buckets. For

this label, besides the maximum stress range, 4 × NBKTS values will be stored for each integration point.

6. **Future developments and feature requests.** Further development of this keyword will mainly be driven by customer requests submitted to *LstSuggestions@ansys.com*. Currently only solid, shell, and integrated beam elements are supported for the binary d3plot format. Future developments may include adding output for thick shells to d3plot and supporting ASCII/binout.

7. **Storage allocation.** In many cases (e.g, *Instability*, *History*, *Effective Stress*), no additional storage (by the element or material) beyond the already occupied slots in the internal data array is needed. However, some requested quantities on this card will require additional storage. In most cases only one slot, but in other cases more. For example, requesting the *Operator* 'Point in time when maximum value over time was attained' (A1 = 1) will require two slots: one containing the maximum value and one for the time it occurred. Note that only one value (the time) will be output to the d3plot database. Below is a list of the *additional* data storage required for each label.

Label	Additional storage slots
Instability	0
Damage	0
Plastic Strain Rate	1
Effective Stress	0
Effective Tresca Stress	0
Plastic Energy Density	1
Effective Creep Strain	0
History	1
Operator	2 if A1 = 1 or 3 1 otherwise
Principal Stress Range	3 + NBKTS × 2
Max Stress Range	1 + NBKTS × 4

Examples:

1. **History label.** The History label is for when you know where history variables of interest are stored and want to use this to compress or simplify the output. For example, the following input

```
*DEFINE_MATERIAL_HISTORIES
History,4,272,1,23
History,1,81
*SET_PART_LIST
23
2,3
```

makes a list of two history variables in the output. History variable #1 fetches the 4th history variable and outputs it only for the RHT concrete model (material 272) solid elements in parts 2 and 3. History variable #2 fetches the 1st history variable and outputs it only for the plasticity with damage model (material 81) but for any element and part. Both of these requested variables happen to be the damage in the respective materials, so an alternative to do something similar would be to use

```
*DEFINE_MATERIAL_HISTORIES
Damage
```

for which the damage for all materials will be displayed in history variable #1.

2. **History variable names.** The NAMES keyword option allows you to give the history variables labels in d3hsp and the post-processing file (if HISNOUT > 0 on *CONTROL_OUTPUT). For example, the following

```
*DEFINE_MATERIAL_HISTORIES_NAMES
History,1,24
Effective strain rate
History,4,47
my_own_history
```

gives the first history variable corresponding to history variable 1 of *MAT_024 the label “Effective strain rate” and the second history variable corresponding to the fourth history variable of user material 47 the label “my_own_history”.

3. **Operator label.** The Operator label is for performing specific operations on stress or a history variable listed in *DEFINE_MATERIAL_HISTORIES. For example,

```
*DEFINE_MATERIAL_HISTORIES
Effective Stress
Operator,0,1,0,0.1
Operator,1,1,0,0.1
Operator,0,-1,0,-99
Operator,1,-1,0,-99
Operator,2,-2
Operator,3,-2
*DEFINE_CURVE
-99
```

```
0.0,0.1
0.0,0.3
0.0,0.7
```

leads to the following history variables in d3plot:

History Variable #	Description
1	Effective stress
2	Maximum value over time of effective stress, reset at time 0.1.
3	Time when maximum value in time of effective stress was attained, reset at time 0.1.
4	Maximum value in time of major principal stress, reset at times 0.1, 0.3, and 0.7.
5	Time when maximum value in time of major principal stress was attained, reset at times 0.1, 0.3, and 0.7.
6	Minimum value in time of minor principal stress
7	Time when minimum value in time of minor principal stress was attained.

4. **Operator label for creating expressions.** The Operator label can be used to create user-defined expressions. For example,

```
*DEFINE_MATERIAL_HISTORIES
$ 1 (1):
Effective Tresca Stress
$ 2 (-): s1-s2
XOperator,7,-1,-3
$ 3 (-): s2-s3
XOperator,7,-3,-2
$ 4 (-): s3-s1
XOperator,7,-2,-1
$ 5 (-): abs(s1-s2)
XOperator,5,2
$ 6 (-): abs(s2-s3)
XOperator,5,3
$ 7 (-): abs(s3-s1)
XOperator,5,4
$ 8 (-): max(abs(s1-s2),abs(s2-s3))
XOperator,11,5,6
$ 9 (2): max(abs(s1-s2),abs(s2-s3),abs(s3-s1))
Operator,11,7,8
$ 10 (-): hsv 1 + hsv 9
XOperator,6,1,9
```

```
$ 11 (3): hsv 10 * 0.5
Operator,8,10,-16,0.5
```

leads to the following history variables in d3plot:

History Variable #	Description
1	Effective Tresca stress
2	Effective Tresca stress (computed using operators)
3	Mean value of Tresca expressions above

*DEFINE_MULTI_DRAWBEADS_IGES

Purpose: Simplify the definition and creation of draw beads, which previously required the use of many keywords.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	DBID	VID	PID	BLKID	NCUR			
Type	I	I	I	I	I			
Default	none	none	1	1	none			

IGES Curve ID Cards. For multiple draw bead curves include as many cards as necessary. Input is terminated at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	CRVID	BFORCE						
Type	I	F						
Default	none	0.0						

VARIABLE	DESCRIPTION
FILENAME	IGES file that has the draw bead curve segment definitions
DBID	Draw bead set ID, which may consist of many draw bead segments.

VARIABLE	DESCRIPTION
VID	Vector ID, as defined by *DEFINE_VECTOR. This vector is used to project the supplied curves to the rigid tool, defined by PID.
PID	Part ID of a rigid tool to which the curves are projected and attached.
BLKID	Part ID of the sheet blank with which created draw beads will contact.
NCUR	Number of draw bead curve segments (in the IGES file defined by FILENAME) to be defined.
CVRID	IGES curve ID for each segment.
BFORCE	Draw bead force for each segment.

Revision information:

This feature is available in LS-DYNA R5 Revision 62840 and later releases.

*DEFINE_MULTI_SHEET_CONNECTORS

Purpose: Define a multi-sheet connection with up to four shell element sheets. The individual sheets can be joined with up to three connection elements. Currently, for the connection elements, only single hexahedron elements with *MAT_SPOTWELD_DAIMLERCHRYSLER (*MAT_100_DA) are supported. Also, PRUL must be ≥ 2 in *DEFINE_CONNECTION_PROPERTIES. In addition to the standard definition of *MAT_100_DA between two sheets, it is possible with this keyword to define the material behavior of the joining elements as functions of the geometric and material properties of all sheets involved.

Card Summary:

Card Sets. Include as many sets of the following cards as desired. This input ends with the next keyword ("*") card.

Card 1. This card is required.

ID	ITYP	NSHEETS					

Card 2. This card is required.

PID1	JNT12	PID2	JNT23	PID3	JNT34	PID4	

Card 3. This card is required.

PARM1	PARM2	PARM3	PARM4	PARM5	PARM6	PARM7	PARM8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ID	ITYP	NSHEETS					
Type	I	I	I					
Default	none	1	none					

VARIABLE	DESCRIPTION
ID	Multi sheet connector ID

VARIABLE	DESCRIPTION
ITYP	Material model of joint elements: EQ.1: *MAT_SPOTWELD_DAIMLERCHRYSLER.
NSHEETS	Number of sheets connected with this multi sheet connector

Card 2	1	2	3	4	5	6	7	8
Variable	PID1	JNT12	PID2	JNT23	PID3	JNT34	PID4	
Type	I	I	I	I	I	I	I	
Default	none	none	none	optional	optional	optional	optional	

VARIABLE	DESCRIPTION
PID1	Part ID of sheet number one
JNT12	ID of joining element between sheet one and two
PID2	Part ID of sheet number two
JNT23	ID of joining element between sheet two and three (optional)
PID3	Part ID of sheet number three (optional)
JNT34	ID of joining element between sheet three and four (optional)
PID4	Part ID of sheet number four (optional)

Card 3	1	2	3	4	5	6	7	8
Variable	PARM1	PARM2	PARM3	PARM4	PARM5	PARM6	PARM7	PARM8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PARM*i*	Set of user parameters additionally available in *DEFINE_FUNCTION. See Remark 1.

Remarks:

1. **Failure Rule from *DEFINE_FUNCTION.** The elements referenced through JNT12, JNT23 and JNT34 should be hexahedral solid elements with material *MAT_100_DA and PRUL must be ≥ 2on *DEFINE_CONNECTION_PROPERTIES. Therefore, the following eleven variables in the respective *DEFINE_CONNECTION_PROPERTIES must be defined as function IDs: DSIGY, DETAN, DDGPR, DSN, DSB, DSS, DEXSN, DEXSB, DEXSS, DGFAD, and DSCLMRR.

   These functions depend on:

   (t1,t2,t3,t4) = thicknesses of the sheets
   (sy1,sy2,sy3,sy4) = initial yield stresses at plastic strain
   (sm1,sm2,sm3,sm4) = maximum engineering yield stresses
   r = strain rate of joining element
   a = area of joining element
   s = number of sheets
   n = internal number of joining element (1, 2, or 3)
   (p1, p2, p3, p4, p5, p6, p7, p8) = parameters defined in card 3
   fn = normal term in failure function
   fb = bending term in failure function
   fs = shear term in failure function
   (ym1,ym2,ym3,ym4) = Young's moduli of the sheets

   For DSIGY = 100, such a function could look like:

```
*DEFINE_FUNCTION
                100
func(t1,t2,t3,t4,sy1,sy2,sy3,sy4,sm1,sm2,sm3,sm4,r,a,s,n,p1,p2,p3,
p4,p5,p6,p7,p8,fn,fb,fs,ym1,ym2,ym3,ym4)=0.5*(sy1+sy2)
```

   All the listed arguments in their correct order must be included in the argument list, even if you are not using 4 sheets. Note that this argument list is longer than the one listed in the remarks for the manual page of *DEFINE_CONNECTION_PROPERTIES. Also, PRUL = 2 and 3 are the same for this keyword because the order of the sheets is determined by this keyword. Since material parameters must be identified from the weld partners during initialization, this feature is only available for a subset of material models now, namely material types 3, 24, 36, 81, 120, 123, 124, 133, 187, 224, 243, 251, and 258.

*DEFINE_MULTISCALE

Purpose: Associate beam sets with multiscale local model IDs for modeling detailed local model failure through the multiscale method.

Local Model/Beam Set and Coupling Type Association Cards. Provide as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	BSET	CTYPE					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
ID	Multiscale local model ID to use. See *INCLUDE_MULTISCALE.
BSET	Beam set which uses this multiscale local model ID for failure modeling.
CTYPE	Coupling type (see Remark 2): EQ.0: Weak coupling (default) EQ.1: Strong coupling

Remarks:

1. **Explanation of capability.** See *INCLUDE_MULTISCALE for a detailed explanation of this capability.

2. **Coupling type.** When CTYPE is 0 or not specified, this multiscale feature generates the solid local models for the defined beams and uses a weak coupling algorithm (used in *INCLUDE_MULTISCALE) for the multiscale calculation. When CTYPE equals to 1, it generates the solid local models for the defined beams and uses a strong coupling algorithm (algorithm used in *INCLUDE_COSIM; see *INCLUDE_COSIM for details) for the calculation.

*DEFINE_NURBS_CURVE

Purpose: Define a NURBS curve using a univariate knot vector, a control polygon, and optionally a set of control weights. The knot vector defines the necessary shape functions and parameterizes the curve.

There is no limit on the size of the input data. Hence, the total number of keyword cards depends on the parameters defined on the first card. The total number of cards is 1 + ceil[(N + P + 1)/8] + N, where N and P designate the number of control points forming the control polygon and the polynomial degree, respectively.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	N	P		TYPE	WFL		
Type	I	I	I		I	I		
Default	none	none	none		0	0		

Knot Vector Cards. The knot vector of length N+P+1 is given below, requiring ceil[(N + P + 1)/8] cards in total. The knot vector must be normalized to the [0,1] interval.

Card 2	1	2	3	4	5	6	7	8
Variable	K1	K2	K3	K4	K5	K6	K7	K8
Type	F	F	F	F	F	F	F	F

Control Point Cards. The spatial coordinates of the control points and the control weights are listed on N cards. Control weight entries are disregarded unless WFL = 1 on Card 1.

Card 3	1	2	3	4	5	6	7	8
Variable	X	Y	Z	W				
Type	F	F	F	F				

VARIABLE	DESCRIPTION
ID	Curve ID. A unique number must be chosen.
N	Number of control points
P	Polynomial degree
TYPE	Coordinate type: EQ.0: Spatial EQ.1: Parametric
WFL	Flag for user defined control weights: EQ.0: Control weights are assumed to be uniform and positive, that is, the curve is a B-spline curve. The fourth column of Card 3 is disregarded. EQ.1: Control weights are defined on the fourth entry of Card 3.
Kn	Values of the univariate knot vector define in Card 2 with $n = 1, \ldots, \text{N} + \text{P} + 1$.
Xk	Spatial coordinates in the global X-direction defined in Card 3 with $k = 1, \ldots, \text{N}$.
Yk	Spatial coordinates in the global Y-direction defined in Card 3 with $k = 1, \ldots, \text{N}$.
Zk	Spatial coordinates in the global Z-direction defined in Card 3 with $k = 1, \ldots, \text{N}$.
Wk	Control weights defined in Card 3 with $k = 1, \ldots, \text{N}$.

Remarks:

1. **Trimming Curves.** While the keyword is meant to store NURBS curves in two or three spatial dimensions, it is also employed to describe trimming curves that define trimmed NURBS elements/surfaces; see *ELEMENT_SHELL_NURBS_-PATCH for further details. In the latter case, the control point coordinates are in fact parametric coordinates of the surface to be trimmed, that is, $(x, y, z, w) = (r, s, 0, w)$ and TYPE = 1 on Card 1.

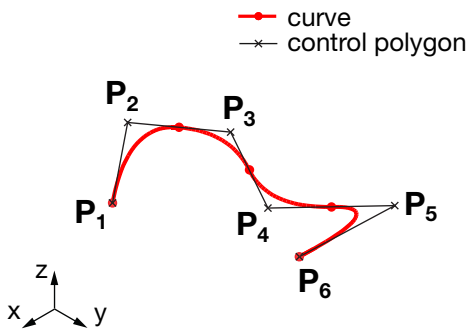


Figure 17-59. An example quadratic NURBS curve.

Example:

See Figure 17-59.

```
$ Define quadratic nurbs curve.
*DEFINE_NURBS_CURVE
$ CARD 1
$---+---ID----+----N----+----P--+---PERI--+---TYPE---+---WFL
         1          6          2          0          0          1
$ CARD 2
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
       0.0        0.0        0.0       0.25        0.5       0.75        1.0        1.0
       1.0
$ CARD 3
$---+----X----+----Y----+----Z----+----W
    -0.353      0.471      0.850      1.000
     0.050      0.600      1.000      1.000
     0.444      0.444      0.900      0.995
     0.125      0.000      0.800      1.000
     0.222     -0.556      0.900      1.010
    -0.353     -0.529      0.850      1.000
```

*DEFINE_PART_FROM_LAYER

Purpose: This keyword creates new blank(s) from an existing sheet blank. It is often used to model composite layers. This keyword applies to shell elements only. The new blanks cannot be adaptively refined.

Multiple layers can be defined. Input ends with the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LAYER	PIDSRCE	LAYOLD	MID	THICK		
Type	I	I	I	I	I	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
PID	The part ID to be created.
LAYER	The layer ID of the PID, see Figure 17-60.
PIDSRCE	The part ID of the existing blank to be copied from.
LAYOLD	The layer ID of the existing blank.
MID	The material ID of the PID.
THICK	The thickness of the PID.

Remarks:

1. **Required Input Deck Cards.** *PART and *SECTION_SHELL do not need to be defined for the to-be created composite layer, but *MAT and contact cards still need to be defined.

2. **Contact.** Contact between outer layer of composites and the tools can be modeled using *CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE, while *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_COMPOSITE, or *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE can be used to model the interactions between two composite layers.

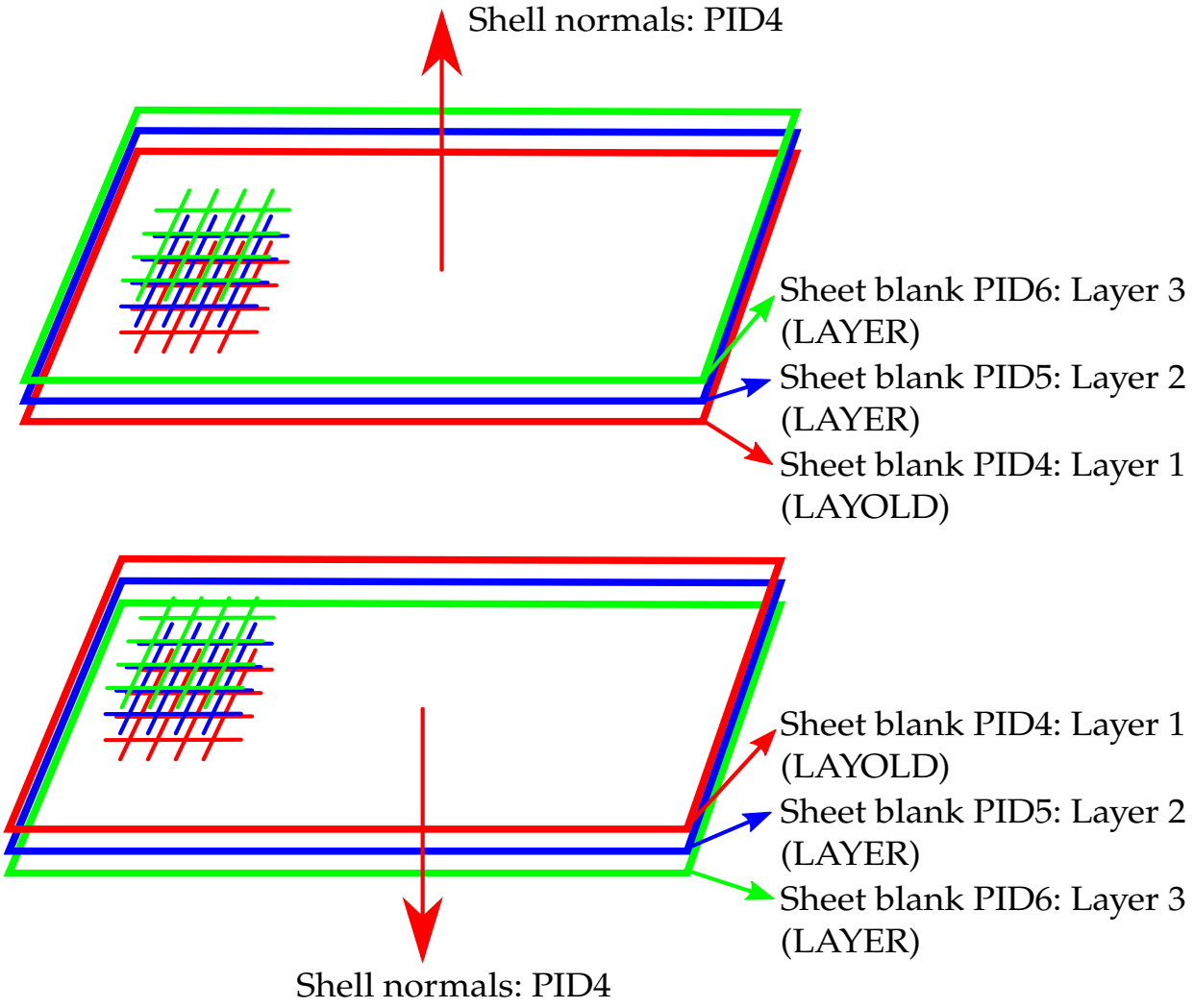


Figure 17-60. Defining the variables LAYER and LAYOLD.

Starting in Dev Revision 124300, contact definitions will be automatically defined between all adjacent layers using *CONTACT_SURFACE_TO_SURFACE during the simulation.

3. The following keyword defines new sheet metal blanks with part IDs 5 and 6 from an existing sheet blank with PID 4. It positions PID 5 on top of PID4 and PID6 on top of PID5.

```
*DEFINE_PART_FROM_LAYER
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$       PID     lAYER   PIDSRCE    LAYOLD       MID     THICK
          5         2         4         1         5      0.10
          6         3         4         1         5      0.10
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

Revision Information:

This feature is available starting from Dev Revision 117590. Automatic contact is available in Dev Revision 124300.

*DEFINE_PARTICLE_BLAST

Purpose: Define control parameters for particle-based blast loading. This keyword was formerly called *PARTICLE_BLAST in versions R11 and earlier.

Card Summary:

Card 1. This card is required.

LAGSID	LAGSTYPE	NODID	NODTYPE	HECID	HECTYPE	AIRCID	

Card 2. This card is required.

NPHE	NPAIR	IUNIT					

Card 3. This card is required.

IHETYPE	DENSITY	ENERGY	GAMMA	COVOL	DETO_V	SMOOTH	

Card 4. This card is required.

DETX	DETY	DETZ	TDET	BTEND	NID		

Card 5. This card is required.

BCX0	BCX1	BCY0	BCY1	BCZ0	BCZ1		

Card 6. This card is required.

IBCX0	IBCX1	IBCY0	IBCY1	IBCZ0	IBCZ1	BC_P	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	LAGSID	LAGSTYPE	NODID	NODTYPE	HECID	HECTYPE	AIRCID	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

VARIABLE	DESCRIPTION
LAGSID	Structure ID for particle structure interaction

VARIABLE	DESCRIPTION
LAGSTYPE	Structure type: EQ.0: Part Set EQ.1: Part
NODID	Discrete element sphere (DES) or smooth particle hydrodynamics (SPH) ID for the interaction between particles and nodes
NODTYPE	Nodal type: EQ.0: Node Set EQ.1: Node EQ.2: Part Set EQ.3: Part
HECID	Initial container for high explosive particle
HECTYPE	Structure type: EQ.0: Part Set EQ.1: Part EQ.2: Geometry; see *DEFINE_PBLAST_GEOMETRY
AIRCID	Initial geometry for air particles: EQ.0: Fill the domain defined by Card 5 with air particles GT.0: ID of an instance of *DEFINE_PBLAST_AIRGEO

Card 2	1	2	3	4	5	6	7	8
Variable	NPHE	NPAIR	IUNIT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
NPHE	Number of high explosive particles

VARIABLE	DESCRIPTION
NPAIR	Number of air particles
IUNIT	Unit system: EQ.0: kg-mm-ms-K EQ.1: SI Units EQ.2: Ton-mm-s-K EQ.3: g-cm-us-K EQ.4: blob-in-s-K, where $[\text{blob}] = [\text{lbf}][\text{s}]^2/[\text{in}]$

Card 3	1	2	3	4	5	6	7	8
Variable	IHETYPE	DENSITY	ENERGY	GAMMA	COVOL	DETO_V	SMOOTH	
Type	I	F	F	F	F	F	F	
Default	0	0.	0.	0.	0.	0.	1.0	

VARIABLE	DESCRIPTION
IHETYPE	High explosive type (see Remark 1): EQ.0: User-defined explosive EQ.1: TNT EQ.2: C4
DENSITY	High explosive density for a user-defined explosive (see Remark 1).
ENERGY	High explosive energy per unit volume for a user-defined explosive (see Remark 1).
GAMMA	High explosive fraction between C_p and C_v for a user-defined explosive (see Remark 1).
COVOL	High explosive co-volume for a user-defined explosive (see Remark 1).
DETO_V	High explosive detonation velocity for a user-defined explosive (see Remark 1).

VARIABLE	DESCRIPTION
SMOOTH	Smoothing factor for the blast load: EQ.1.0: The blast load is not smoothed. GT.1.0: The blast load is smoothed by this factor around the contact area.

Card 4	1	2	3	4	5	6	7	8
Variable	DETX	DETY	DETZ	TDET	BTEND	NID		
Type	F	F	F	F	F	I		
Default	0.	0.	0.	0.	0.	0		

VARIABLE	DESCRIPTION
DETX	Detonation point x
DETY	Detonation point y
DETZ	Detonation point z
TDET	Detonation time
BTEND	Blast end time
NID	An optional node ID defining the position of the detonation point. If defined, its coordinates will overwrite the DETX, DETY, and DETZ defined above.

Card 5	1	2	3	4	5	6	7	8
Variable	BCX0	BCX1	BCY0	BCY1	BCZ0	BCZ1		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
BCX0	Global domain x-min
BCX1	Global domain x-max
BCY0	Global domain y-min
BCY1	Global domain y-max
BCZ0	Global domain z-min
BCZ1	Global domain z-max

Card 6	1	2	3	4	5	6	7	8
Variable	IBCX0	IBCX1	IBCY0	IBCY1	IBCZ0	IBCZ1	BC_P	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

VARIABLE	DESCRIPTION
IBCX0	Boundary conditions for global domain x-min: EQ.0: Free EQ.1: Rigid reflecting boundary
IBCX1	Boundary conditions for global domain x-max: EQ.0: Free EQ.1: Rigid reflecting boundary
IBCY0	Boundary conditions for global domain y-min: EQ.0: Free EQ.1: Rigid reflecting boundary
IBCY1	Boundary conditions for global domain y-max: EQ.0: Free EQ.1: Rigid reflecting boundary

VARIABLE	DESCRIPTION
IBCZ0	Boundary conditions for global domain z-min: EQ.0: Free EQ.1: Rigid reflecting boundary
IBCZ1	Boundary conditions for global domain z-max: EQ.0: Free EQ.1: Rigid reflecting boundary
BC_P	Pressure ambient boundary condition for global domain: EQ.0: Off (Default) EQ.1: On (Remark 2)

Remarks:

1. **Material constants for commonly used high explosives.** If the explosive material is TNT or C4, then the parameters DENSITY (ρ), ENERGY (e_0), GAMMA (γ), COVOL (COV), and DET_V (D) are set automatically to the values in the table below. For other explosive materials, you must provide those values.

IHETYPE	ρ	e_0	γ	COV	D
TNT	$1630\,\frac{\text{kg}}{\text{m}^3}$	$7\,\frac{\text{GJ}}{\text{m}^3}$	1.35	0.6	$6930\,\frac{\text{m}}{\text{s}}$
C4	$1601\,\frac{\text{kg}}{\text{m}^3}$	$9\,\frac{\text{GJ}}{\text{m}^3}$	1.32	0.6	$8193\,\frac{\text{m}}{\text{s}}$

2. **Pressure boundary conditions.** If pressure boundary conditions are used, particles will not escape the global domain when the domain pressure is lower than the ambient pressure. The ambient pressure is assumed to be 1 ATM by default.

*DEFINE_PBLAST_AIRGEO

Purpose: To define a simple geometry for initial air domain.

Card 1	1	2	3	4	5	6	7	8
Variable	GID	GTYPE1	GTYPE2					
Type	I	I	I					
Default	0	0	0					

Card 2	1	2	3	4	5	6	7	8
Variable	XA	YA	ZA	XB	YB	ZB		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

Card 3	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	G1	G2	G3		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

Card 4	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	G4	G5	G6		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
GID	ID of a GEOMETRY defining initial air particle domain.
GTYPE1 GTYPE2	Geometry type EQ.1: box EQ.2: sphere EQ.3: cylinder EQ.4: ellipsoid EQ.5: hemisphere (see Remark 1)
XA, YA, ZA	(XA, YA, ZA) defines a vector of the *x*-axis
XB, YB, ZB	(XB, YB, ZB) defines a vector of the *y*-axis
X0, Y0, Z0	Center coordinates of air domain
G1	Dimension value depending on GTYPE. GTYPE.EQ.1: length of *x* edge GTYPE.EQ.2: Radius of sphere GTYPE.EQ.3: Radius of cross section GTYPE.EQ.4: length of *x*-axes GTYPE.EQ.5: Radius of hemisphere
G2	Dimension value depending on GTYPE. GTYPE.EQ.1: length of *y* edge GTYPE.EQ.3: length of cylinder GTYPE.EQ.4: length of *y*-axes
G3	Dimension value depending on GTYPE. GTYPE.EQ.1: length of *z* edge GTYPE.EQ.4: length of *z*-axes
XC, YC, ZC	Center coordinates of domain excluded from the air domain
G4, G5, G6	See definition of G1, G2, G3

Remarks:

1. If GTYPE1/GTYPE2 is 5, the hemisphere is defined in negative z direction defined by the cross product of the y and x axis.

*DEFINE_PBLAST_GEOMETRY

Purpose: To define a simple geometry for high explosives domain.

Card 1	1	2	3	4	5	6	7	8
Variable	GID	GTYPE						
Type	I	I						
Default	0	none						

Card 2	1	2	3	4	5	6	7	8
Variable	XA	YA	ZA	XB	YB	ZB	ITYPE	
Type	F	F	F	F	F	F	I	
Default	0.	0.	0.	0.	0.	0.	0	

Card 3	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC					
Type	F	F	F					
Default	0.	0.	0.					

Card 4	1	2	3	4	5	6	7	8
Variable	G1	G2	G3					
Type	F	F	F					
Default	0.	0.	0.					

VARIABLE	DESCRIPTION
GID	ID of a GEOMETRY defining high explosive particle domain.
GTYPE	Geometry type EQ.1: Box EQ.2: Sphere EQ.3: Cylinder EQ.4: Ellipsoid EQ.5: Hemisphere (see Remark 1)
XA, YA, ZA	ITYPE.EQ.0: Local x-axis ITYPE.EQ.0: Starting coordinate of the charge used to internally calculate the local coordinate system
XB, YB, ZB	ITYPE.EQ.0: Local y-axis ITYPE.EQ.1: Ending coordinate of the charge used to internally calculate the local coordinate system
ITYPE	Flag for determining how the domain geometry of the high explosive particles is generated: EQ.0: Particles are generated in the given geometry with the local coordinate system provided with XA, YA, ZA, XB, YB, and ZB (default). EQ.1: Particles are generated in the given geometry using starting and ending coordinates given with XA, YA, ZA, XB, YB, and ZB.
XC	X-coordinate of charge center
YC	Y-coordinate of charge center
ZC	Z-coordinate of charge center
G1	GTYPE.EQ.1: Length of x edge GTYPE.EQ.2: Radius of sphere GTYPE.EQ.3: Radius of cross section GTYPE.EQ.4: Length of x-axes

VARIABLE	DESCRIPTION
	GTYPE.EQ.5: Radius of hemisphere
G2	GTYPE.EQ.1: Length of y edge GTYPE.EQ.3: Length of cylinder GTYPE.EQ.4: Length of y-axes
G3	GTYPE.EQ.1: Length of z edge GTYPE.EQ.4: Length of z-axes

Remarks:

1. **Hemisphere.** If GTYPE is 5, the hemisphere is defined in the negative z-direction given by the cross product of the y- and x-axes.

*DEFINE_PLANE

Purpose: Define a plane with three non-collinear points. The plane can be used to define a reflection boundary condition for problems like acoustics.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	X1	Y1	Z1	X2	Y2	Z2	CID
Type	I	F	F	F	F	F	F	I
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	X3	Y3	Z3					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
PID	Plane ID. A unique number has to be defined.
X1	*X*-coordinate of point 1.
Y1	*Y*-coordinate of point 1.
Z1	*Z*-coordinate of point 1.
X2	*X*-coordinate of point 2.
Y2	*Y*-coordinate of point 2.
Z2	*Z*-coordinate of point 2.
CID	Coordinate system ID applied to the coordinates used to define the current plane. The coordinates X1, Y1, Z1, X2, Y2, Z2, X3, Y3 and Z3 are defined with respect to the coordinate system CID.
X3	*X*-coordinate of point 3.

VARIABLE	DESCRIPTION
Y3	*Y*-coordinate of point 3.
Z3	*Z*-coordinate of point 3.

Remarks:

1. **Setup Suggestions.** The coordinates of the points must be separated by a reasonable distance and not collinear to avoid numerical inaccuracies.

*DEFINE_POINT_CLOUD_{OPTION}

Available options include:

<BLANK>

TITLE

Purpose: Define a set of points in the physical space. One or more fields reference a point cloud to define spatially varying data on a set of points; see *DEFINE_FIELD.

Card Summary:

Card Title. This card is included if the TITLE keyword option is used.

TITLE

Card 1. This card is required.

PCID							

Card 2. This card is required. Include as many instantiations of this card as needed. This input ends at the next keyword ("*") card. See Remarks 1 and 2.

X	Y	Z		

Data Card Definitions:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

VARIABLE	DESCRIPTION
TITLE	Name or description of the point cloud defined in this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	PCID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PCID	Point cloud ID. A unique ID number must be used.

Point Coordinates Card. Include as many instantiations of this card as needed, one for each point in the point cloud. This input ends at the next keyword ("*") card. See Remarks 1 and 2.

Card 2	1	2	3	4	5	6	7	8
Variable	X		Y		Z			
Type	F		F		F			
Default	0.0		0.0		0.0			

VARIABLE	DESCRIPTION
X	x-coordinate of the i^{th} point in the point cloud in the global coordinate system
Y	y-coordinate of the i^{th} point in the point cloud in the global coordinate system
Z	z-coordinate of the i^{th} point in the point cloud in the global coordinate system

Remarks:

1. **Data consistency**. Each instance of Card 2 enables defining a single point in the point cloud. The input reader permits an arbitrary number of instantiations of Card 2. However, the number of points in the point cloud must be consistent with the number of field data included in *DEFINE_FIELD, meaning that the

number of points × NV must equal the total number of input field data (NV is the number of data input for each point). See *DEFINE_FIELD for details.

2. **Point ID**. Points defined with *DEFINE_POINT_CLOUD are not associated with an ID. Therefore, these points cannot be explicitly referenced by other keywords, such as *SET.

***DEFINE_POROUS**_OPTION

Available options include:

ALE

LAGRANGIAN

Purpose: The *DEFINE_POROUS_ALE card defines the Ergun porous coefficients for ALE elements. It is to be used with *LOAD_BODY_POROUS. This card with the LAGRANGIAN option, *DEFINE_POROUS_LAGRANGIAN, defines the porous coefficients for Lagrangian elements and is to be used with *CONSTRAINED_LAGRANGE_IN_SOLID (Lagrangian structure parts with CTYPE = 11 or 12).

Card 1	1	2	3	4	5	6	7	8
Variable	EIDBEG	EIDEND	LOCAL	VECID1	VECID2	USERDEF		
Type	I	I	I	I	I	I		
Default	none	0	0	0	0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	AXX	AXY	AXZ	BXX	BXY	BXZ		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Card 3	1	2	3	4	5	6	7	8
Variable	AYX	AYY	AYZ	BYX	BYY	BYZ		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

Card 4	1	2	3	4	5	6	7	8
Variable	AZX	AZY	AZZ	BZX	BZY	BZZ		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
EIDBEG, EIDEND	EIDBEG, EIDEND > 0: Range of thick porous element IDs. These are solids in 3D and shells in 2D. EIDBEG, EIDEND < 0: Range of thin porous element IDs. These are shells in 3D and beams in 2D. The ALE option does *not* support thin porous elements. EIDBEG > 0, EIDEND $= 0$: EIDBEG is a set of thick porous elements EIDBEG > 0, EIDEND < 0: EIDBEG is a set of thin porous elements
LOCAL	Flag to activate an element coordinate system: EQ.0: The forces are applied in the global directions. EQ.1: The forces are applied in a local system attached to the element. The system is consistent with DIREC = 1 and CTYPE = 12 in *CONSTRAINED_LAGRANGE_IN_SOLID. For CTYPE = 11, LOCAL is always 1 and the x-axis is aligned with the element normal while the y-axis passes through the element center and the first node in the element connectivity (*ELEMENT_BEAM in 2D or *ELEMENT_SHELL in 3D)
VECID1, VECID2	*DEFINE_VECTOR IDs to define a specific coordinate system. VECID1 and VECID2 give the x- and y-direction respectively. The z-vector is a cross product of VECID1 and VECID2. If this latter is not orthogonal to VECID1, its direction will be corrected with a cross-product of z- and x-vectors. The vectors are stored as isoparametric locations to update their directions if the element deforms or rotates.

VARIABLE	DESCRIPTION
USERDEF	Flag to compute A*ij* and B*ij* with a user defined routine in the file dyn21.F called lagpor_getab_userdef. The file is part of the general package usermat.
A*ij*	Viscous matrix for the porous flow Ergun equation (see Remark 1)
B*ij*	Inertial matrix for the porous flow Ergun equation (see Remark 1)

Remarks:

1. **Ergun equation.** The Ergun equation computing the pressure gradient along each direction $i = x, y, z$ can be written as follows:

$$\frac{dP}{dx_i} = \sum_{j=1}^{3} \left[\mu A_{ij} V_j + \rho B_{ij} |V_j| V_j \right]$$

where,

a) V_i is the relative velocity of the flow in the porous media

b) A_{ij} are the viscous coefficients of the Ergun-type porous flow equation in the i^{th} direction.. This matrix is similar to the viscous coefficients used in *LOAD_BODY_POROUS.

c) B_{ij} are the inertial coefficient of the Ergun-type porous flow equation in the i^{th} direction. This matrix is similar to the inertial coefficients used in *LOAD_BODY_POROUS.

If this keyword defines the porous properties of Lagrangian elements in *CONSTRAINED_LAGRANGE_IN_SOLID, the porous coupling forces are computed with the pressure gradient as defined above instead of the equations used for CTYPE = 11 and 12.

*DEFINE_PRESSURE_TUBE

Purpose: Define a gas-filled tube to simulate interior pressure waves resulting from changes in the tube's cross-sectional area over time. The tube is specified with tubular beam elements, and the initial gas volume is determined by the beam cross-sectional area and initial element lengths. Area changes are either given by beam contact penetration (only mortar contacts are currently supported) or by deformation of automatically generated shell/solid elements. The pressure calculation is not coupled with the beam deformation (except for from contact penetration) and does not use any data from the material card. Pressure and tube area at the beam nodes are output through *DATABASE_PRTUBE.

Card Summary:

Card 1. This card is required.

PID	WS	PR	MTD	TYPE	GAMMA	CFAC	CUTOFF

Card 2. This card is optional.

VISC	CFL	DAMP	BNDL	BNDR	CAVL	CAVR	SNODE

Card 3a. This card is only read if |TYPE| = 1.

NSHL	ELFORM	NIP	SHRF	BPID	ISAVE	IORIEN	

Card 3b. This card is only read if |TYPE| = 2.

NSLD	ELFORM	NTHK		BPID	ISAVE	IORIEN	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PID	WS	PR	MTD	TYPE	GAMMA	CFAC	CUTOFF
Type	I	F	F	I	I	F	F	I
Default	0	0.0	0.0	0	0	1.0	0.0	0

VARIABLE	DESCRIPTION
PID	Part ID of the tube. All connected beam elements in the part will

VARIABLE	DESCRIPTION
	model a tube. Only tubular beam elements are allowed, that is, ELFORM = 1, 4, 5, and 11 with CST = 1 on *SECTION_BEAM. The initial tube cross-sectional area is calculated using the beam inner diameter TT1/TT2 fields in *SECTION_BEAM. The outer diameter TS1/TS2 fields in *SECTION_BEAM are used if no inner diameter is given. The beam elements may not contain junctions. Two different parts on which *DEFINE_PRESSURE_TUBE is defined may not share nodes. For MPP, all elements in the part will be on a single processor.
WS	Speed of sound c_0 in the gas
PR	Initial gas pressure p_0 inside tube
MTD	Solution method (see Remark 2): EQ.0: Standard Galerkin FEM EQ.1: Discontinuous Galerkin EQ.2: Discontinuous Galerkin on isentropic Euler equations
TYPE	Tube element type (see Figure 17-61): EQ.0: The tube is entirely simulated with beam elements. The contact penetration of the beam elements gives the cross-sectional area. Contact stiffness governs the mechanical response in the radial direction of the beam elements. Only mortar contacts are supported. EQ.1: The tube is simulated by the automatic generation of shell elements, which are assigned the beam part ID and the beam material model. A new part ID is given to the beam elements, which are no longer part of the mechanical solution. Contacts and other properties associated with the old beam part ID apply to the new shell part. The shell element nodes give the cross-sectional area. The shells entirely govern the mechanical response. All contact definitions are supported. Constraints defined by *BOUNDARY_SPC, *BOUNDARY_PRESCRIBED_MOTION, *CONSTRAINED_EXTRA_NODES, and *CONSTRAINED_NODAL_RIGID_BODY and nodes that are shared with a rigid body are moved to the new shell tube. EQ.2: The tube is simulated by the automatic generation of solid elements, similarly to TYPE = 1 above.

VARIABLE	DESCRIPTION
	LT.0: Automatic generation of elements as above, but the beam nodes are given new nodal IDs. The old beam NIDs are moved to the automatically generated tube (one row of nodes along the length). Any nodal constraints thus apply to the new tube instead of the beam element tube. See Figure 17-61 for an example of different values of TYPE and how they affect nodal constraints.
GAMMA	Adiabatic index, γ. It must be ≥ 1. It is only used for MTD = 2.
CFAC	Cavity correction factor; see Remark 5. This post-processing correction overwrites the pressure on cavity interface nodes (for CAVi $\neq 0$ on Card 2).
CUTOFF	Flag for enabling tube cutoff: EQ.0: Inactive EQ.1: Active When a part of the tube is fully compressed, air can no longer pass through, and the equations become invalid. When CUTOFF = 0, this issue is avoided by assuming that the minimum tube cross-sectional area under compression is a small fraction (1%) of the initial area for stability reasons. This assumption makes the equations valid but can lead to small pressure waves passing through a closed-off tube. With CUTOFF = 1, nodes are temporarily removed from the simulation if the area is less than 1% of the initial area and if the flow velocity is sufficiently slow (0.1% of WS). Thus, the solver is not used in the fully compressed area at all. Waves cannot pass through that part of the tube and instead are reflected with a reflective boundary condition.

Card 2	1	2	3	4	5	6	7	8
Variable	VISC	CFL	DAMP	BNDL	BNDR	CAVL	CAVR	SNODE
Type	F	F	F	F	F	F	F	I
Default	1.0	0.9	0.0	0.0	0.0	0.0	0.0	0

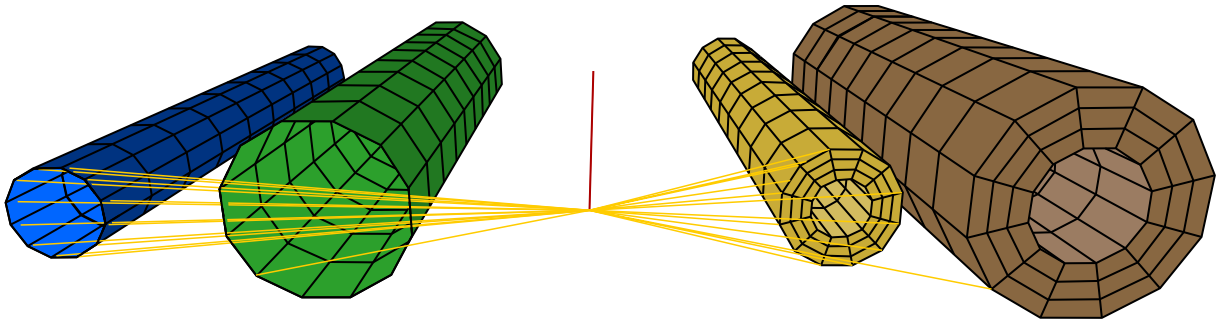


Figure 17-61. Automatically generated tubes using TYPE = 1, -1, 0, 2, and -2, from left to right. The beam tube ends are connected with *CONSTRAINED_NODAL_RIGID_-BODY, which applies differently to the generated shell/solid tubes, depending on the sign of TYPE.

VARIABLE	DESCRIPTION
VISC	MTD.EQ.0: Artificial viscosity multiplier (VISC > 0.0); see Remark 2. A smaller value gives a more resolved pulse at shorter wavelengths but may lead to instabilities. We recommend the default value for typical automotive crash applications (tube length ~2m, diameter ~5mm, pressure pulse width ~5ms). MTD.GT.0: Slope limiter smoothing factor; see Remark 2. Smaller values give a more resolved pulse at shorter wavelengths but may lead to instabilities. Larger values lead to a smeared pulse.
CFL	Stability factor (CFL > 0.0); see Remark 2. A smaller value increases stability at the expense of increased computational cost. For typical automotive crash applications, the default value is recommended.
DAMP	Linear damping (DAMP ≥ 0.0); see Remark 2.
BND*i*	Left/right boundary condition (0 ≤ BND*i* ≤ 1); see Remark 3. Special cases are: EQ.0.0: Closed tube end, that is, zero velocity boundary condition EQ:0.5: Non-reflecting boundary condition EQ:1.0: Open tube end, that is, constant pressure boundary condition Left/right tube end is automatically assigned to the lowest/highest beam node number on the tube, respectively.

VARIABLE	DESCRIPTION
CAV*i*	Left/right cavity; see Remark 4.
	GT.0.0: A cavity replaces the elements near the end of the tube. The integer part of CAV*i* determines the number of beam elements that belong to the cavity. The remainder of CAV*i* determines the boundary condition on the interface between the tube and the cavity.
	LT:0.0: A cavity extends the tube by adding new beam elements. The length of the added cavity is given by $L = \text{int}(\lvert\text{CAV}i\rvert)/100$ where $\text{int}(x)$ truncates the decimal portion of x (leaving an integer). The remainder of $\lvert\text{CAV}i\rvert$ determines the boundary condition on the interface between the tube and the cavity.
SNODE	Optional starting node. This node determines the left end of the tube. If not set, the tube starts at the lowest numbered beam node.

Automatically Generated Shells Card. This card is only read if $|\text{TYPE}| = 1$.

Card 3a	1	2	3	4	5	6	7	8
Variable	NSHL	ELFORM	NIP	SHRF	BPID	ISAVE	IORIEN	
Type	F	F	F	F	I	I	I	
Default	12.0	16.0	3.0	1.0	optional	0	0	

Automatically Generated Solids Card. This card is only read if $|\text{TYPE}| = 2$.

Card 3b	1	2	3	4	5	6	7	8
Variable	NSLD	ELFORM	NTHK		BPID	ISAVE	IORIEN	
Type	F	F	F		I	I	I	
Default	12.0	1.0	3.0		optional	0	0	

VARIABLE	DESCRIPTION
NSHL/NSLD	Number of automatically generated shells/solids on the circumference of the tube
ELFORM	ELFORM for automatically generated shells/solids; see *SECTION_SHELL/SOLID.
NIP	Number of through-thickness integration points for automatically generated shells; see NIP in *SECTION_SHELL.
NTHK	Number of solid elements in the thickness of the tube for automatically generated solids
SHRF	Shear correction factor for automatically generated shells; see SHRF in *SECTION_SHELL.
BPID	Optional PID given to the beam elements when automatically generating shells/solids.
ISAVE	Save shell/solid geometry and connectivity to a keyword file: EQ.0: No saving (default) EQ.1: Save to keyword file prtube.k Subsequent runs can include the generated file, in which case TYPE and Card 3 are ignored for saved tubes. Boundary conditions and contacts are not saved. Thus, beam boundary conditions and contacts are not transferred to the saved shell/solid tubes in subsequent runs. Therefore, boundary conditions, contacts, etc., must still be supplied to the saved shell/solid geometry.
IORIEN	Control circumferential orientation of automatically generated shells/solids: EQ.0: Orient the first end segment with the global coordinate system. Subsequent segments are created in a way to minimize twisting along the length of the tube. (default) EQ.1: Orient each segment using the third beam node if it exists. Otherwise, use the vector from *ELEMENT_BEAM_ORIENTATION to orient each segment.

Remarks:

1. **Pressure tube model.** The pressure tube is modeled with an acoustic approximation of the one-dimensional compressible Euler equations for pipes with varying thickness:

$$\frac{\partial}{\partial t}(\rho A) + \frac{\partial}{\partial x}(\rho u A) = 0$$
$$\frac{\partial}{\partial t}(\rho u A) + \frac{\partial}{\partial x}(\rho u^2 A + pA) = p\frac{\partial A}{\partial x}$$
$$\frac{\partial}{\partial t}(EA) + \frac{\partial}{\partial x}(u(E+p)A) = 0$$

where $A = A(x,t)$ is the cross-sectional area and ρ, p, u, and E are density, pressure, velocity, and energy per unit volume, respectively. The above system is closed under the constitutive relations

$$E = \rho e + \frac{\rho u^2}{2} \ , \qquad p = p(\rho, e) \ ,$$

where e is the internal energy per unit mass.

For an isentropic and isothermal flow, the pressure is proportional to the density, that is,

$$p = c_0^2 \rho \ ,$$

and the energy equation can be dropped. This is a good approximation of the Euler equations for acoustic flows where the state variables are smooth perturbations around a background state. No shocks develop over time for such flows but may be present from initial/boundary values or source terms.

Assuming small perturbations, linearization around $\rho_0, p_0, u_0 = 0$ gives the acoustic approximation

$$\frac{\partial}{\partial t}(A\rho) + \rho_0 \frac{\partial y}{\partial x} = 0$$
$$\rho_0 \frac{\partial y}{\partial t} + c_0^2 \frac{\partial}{\partial x}(A\rho) = p\frac{\partial A}{\partial x}$$

where $y = Au$. Expressed in y and p we have

$$\frac{\partial p}{\partial t} + \frac{\partial \ln A}{\partial t} p + \frac{p_0}{A}\frac{\partial y}{\partial x} = 0$$
$$\frac{\partial y}{\partial t} + A\frac{c_0^2}{p_0}\frac{\partial p}{\partial x} = 0$$

2. **Solution methods for the pressure tube.** We have two solution methods for the linearized system given in Remark 1 and one for the isothermal nonlinear system. MTD on Card 1 sets the method.

a) *Continuous Galerkin (MTD = 0).* The linearized system can be solved using the standard Continuous Galerkin finite element method, using piecewise linear basis functions and artificial viscosity. Linear damping is then added to model energy losses from friction between the gas and the tube walls. With artificial viscosity and damping, the linear system can be written as

$$\frac{\partial p}{\partial t} + \frac{\partial \ln A}{\partial t} p + \frac{p_0}{A}\frac{\partial y}{\partial x} = \epsilon \frac{\partial^2 p}{\partial x^2} - \text{DAMP} \times (p - p_0)$$
$$\frac{\partial y}{\partial t} + A\frac{c_0^2}{p_0}\frac{\partial p}{\partial x} = \epsilon \frac{\partial^2 y}{\partial x^2}$$

The artificial viscosity, ϵ, is proportional to the maximum initial beam element length, that is,

$$\epsilon = \text{VISC} \times c_0 \max_i \Delta x_i \ .$$

Small values of VISC may lead to convergence problems while large values smear out the solution. Time integration is independent of the mechanical solver and uses a step size less than or equal to the global time step, satisfying a CFL condition

$$\Delta t < \min_i \frac{\text{CFL} \times \Delta x_i}{\Delta x_i \left|\frac{\partial \ln A}{\partial t}\right| + 3c_0} \ .$$

b) *Discontinuous Galerkin (MTD = 1).* Alternatively, the linear system can be solved with the Discontinuous Galerkin method, using piecewise linear basis functions on each element, Lax-Friedrich flux, and a MUSCL flux limiter to limit spatial oscillations. Time integration is done with Heun's method which is a 2nd order TVD Runge Kutta method. In this case, linear damping is again added to the model.

$$\frac{\partial p}{\partial t} + \frac{\partial \ln A}{\partial t} p + \frac{p_0}{A}\frac{\partial y}{\partial x} = -\text{DAMP} \times (p - p_0)$$
$$\frac{\partial y}{\partial t} + A\frac{c_0^2}{p_0}\frac{\partial p}{\partial x} = 0$$

VISC > 0 is a smoothing factor for the MUSCL limiter with VISC = 0 indicating no smoothing. The Discontinuous Galerkin method gives less diffusion (smearing) than the Continuous Galerkin method.

c) *Discontinuous Galerkin with isentropic Euler equations (MTD = 2).* This method uses Discontinuous Galerkin to solve the damped nonlinear isentropic Euler equations

$$\frac{\partial}{\partial t}(\rho A) + \frac{\partial}{\partial x}(\rho u A) = -\text{DAMP} \times (\rho - \rho_0)A,$$
$$\frac{\partial}{\partial t}(\rho u A) + \frac{\partial}{\partial x}\left(\rho u^2 A + pA\right) = p\frac{\partial A}{\partial x},$$

with

$$p = \frac{{c_0}^2}{\gamma \rho_0^{\gamma-1}} \rho^\gamma,$$
$$\rho_0 = \frac{\gamma p_0}{{c_0}^2}.$$

VISC has the same meaning as for MTD = 1. This method may be beneficial for scenarios with large pressure/velocity perturbations, where the acoustic approximation is poor.

3. **Boundary conditions.** The boundary conditions are

$$(p - p_0) = \pm Z u \ ,$$

where the sign depends on whether it is the left or right boundary. The impedance is defined as

$$Z = \frac{p_0}{c_0} \frac{\text{BND}i}{(1 - \text{BND}i)} \ .$$

A closed end, BNDi = 0, thus corresponds to $Z = 0$ and $u = 0$, meaning a zero-velocity condition. An open end, BNDi = 1, corresponds to $Z = \infty$ and $p = p_0$, a constant-pressure condition. For the intermediate case, we note that if A is a constant, DAMP = 0, and $\epsilon = 0$, then the acoustic equations become

$$\frac{\partial p}{\partial t} + p_0 \frac{\partial u}{\partial x} = 0$$
$$\frac{\partial u}{\partial t} + \frac{c_0^2}{p_0} \frac{\partial p}{\partial x} = 0$$

which is equivalent to the acoustic wave equation

$$\frac{\partial^2 p}{\partial t^2} - c_0^2 \frac{\partial^2 p}{\partial x^2} = \left(\frac{\partial}{\partial t} - c_0 \frac{\partial}{\partial x} \right) \left(\frac{\partial}{\partial t} + c_0 \frac{\partial}{\partial x} \right) p = 0 \ .$$

Differentiation of the boundary condition (and assuming that the wave equation holds on the boundary) gives

$$\frac{\partial p}{\partial t} = \pm Z \frac{\partial u}{\partial t} = \pm \frac{p_0}{c_0} \frac{\text{BND}i}{(1 - \text{BND}i)} \frac{\partial u}{\partial t} = \mp \text{c}_0 \frac{\text{BND}i}{(1 - \text{BND}i)} \frac{\partial p}{\partial x} \ .$$

Setting BNDi = 0.5 implies

$$\frac{\partial p}{\partial t} = \mp \text{c}_0 \frac{\partial p}{\partial x} \ ,$$

which is the left/right traveling solution to the acoustic wave equation; that is, BNDi = 0.5 corresponds to $Z = Z_0 = p_0/c_0$ and a non-reflecting boundary condition.

The boundary condition can also be seen as a jump in the cross-sectional area of the tube, meaning if we have two tubes with area A_l and A_r, then a wave (p, u)

traveling from A_l to A_r can be split into a reflected part $(Rp, -Ru)$ and a transmitted part (Tp, Tu), with reflectance/transmission coefficients

$$R = \frac{A_l - A_r}{A_l + A_r} \quad , \qquad T = \frac{2A_l}{A_l + A_r} \quad .$$

Reflectance can also be expressed with respect to impedance at the interface:

$$R = \frac{Z_0 - Z}{Z_0 + Z} \quad .$$

Combining the expressions for the reflectance, the boundary condition can thus be seen as an infinite extension of the original tube, with a jump from the tube area A_{in} at the boundary to an outside area

$$A_{\text{out}} = \frac{\text{BND}i}{(1 - \text{BND}i)} A_{\text{in}} \quad .$$

BNDi $= 0.5$, thus, corresponds to $A_{\text{out}} = A_{\text{in}}$, BND$i$ $= 0$ gives $A_{\text{out}} = 0$, and BNDi $= 1$ gives $A_{\text{out}} = \infty$.

4. **Cavities.** Cavities close to the tube ends can be simulated in three ways:

   a) Altering tube thickness with *ELEMENT_BEAM_THICKNESS.

   b) Setting CAV$i > 0$.

      For CAV$i > 0$, the original tube geometry is intact, and $n = \text{int}(\text{CAV}i)$ (the integer portion of CAVi) elements from the end belong to a cavity, that is, a different cross-section area (that can only be seen in binout/prtubeout). The boundary condition, $\alpha = \text{CAV}i - n$, on the interface between the tube and the cavity determines the cavity's cross-sectional area. The n elements closest to the boundary have the new area

      $$A_{\text{cav}} = \frac{\alpha}{(1 - \alpha)} A_{n+1} \quad ,$$

      where A_{n+1} is the initial cross-section the area of element $n + 1$ from the boundary. For example, for a tube with initial cross-sectional area $A = 10$, CAVL $= 50.8$ means that the 50 elements closest to the left end will have the constant cross-sectional area

      $$A_{\text{cav}} = \frac{0.8}{(1 - 0.8)} 10 = 40 \quad .$$

   c) Setting CAV$i < 0$.

      For CAV$i < 0$, a cavity of length $L = \text{int}(|\text{CAV}i|)/100$ is added at the end by creating new beam elements (at least 5 beam elements are used). These new beam elements will not be converted to shells/solids regardless of TYPE; thus, they are not part of the mechanical solution. The cavity cross-section

area is calculated as above using the boundary condition, $\alpha = |\text{CAV}i| - \text{int}(|\text{CAV}i|)$.

When using CAVi, the area A_{cav} overrides any cross-sectional area updates for the cavity elements resulting from physical deformation. Also, the boundary conditions set by BNDi still hold at the outer ends of the (extended) tube.

5. **Cavity correction.** Since cavities may have large cross-sections, the pressure from the one-dimensional solver may not be accurate in the cavity. To correct this, we use a heuristic approach based on the velocity, u, from the one-dimensional solver and the polytropic gas law:

$$PV^{\gamma} = \text{const},$$

for air flowing in and out of the cavity during a time step.

First, assume that gas is flowing from the cavity to the tube during a time step $\Delta t_n = t_{n+1} - t_n$. Initially, the gas occupies the cavity with volume $V_n = V_c$. After a time step, the gas occupies the cavity plus a small part of the tube, with total volume

$$V_{n+1} = V_c + A_{\text{tube}} \times \text{CFAC}|u_n|\Delta t_n.$$

From the polytropic gas law, we have the pressure change

$$P_{n+1} = P_n \frac{V_n^{\gamma}}{V_{n+1}^{\gamma}} = P_n \left(1 + \frac{A_{\text{tube}} \times \text{CFAC}|u_n|\Delta t_n}{V_c}\right)^{-\gamma}.$$

Similarly, inflow to the cavity gives a similar expression with a change of sign of the exponent with the assumption $V_{n+1} = V_c$:

$$V_{n+1} = V_c = V_n - A_{\text{tube}} \times \text{CFAC}|u_n|\Delta t_n$$
$$P_{n+1} = P_n \frac{V_n^{\gamma}}{V_{n+1}^{\gamma}} = P_n \left(1 + \frac{A_{\text{tube}} \times \text{CFAC}|u_n|\Delta t_n}{V_c}\right)^{\gamma}$$

*DEFINE_QUASAR_COUPLING

Purpose: Define LS-DYNA node/node set that interacts with Cadlm's QUASAR ROM model. Each coupling needs to have its own keyword card and will not accept multiple entries.

> **WARNING:** We added Card 2 in September of 2020. It breaks backward compatibility. Old input decks that include this keyword will not work with executables created September of 2020 and later.

Card 1	1	2	3	4	5	6	7	8
Variable	NODE	TYPE	ROMID	PID	PTYPE	IOPT	CID	EX_ND
Type	I	I	I	I	I	I	I	I

Card 2	1	2	3	4	5	6	7	8
Variable	FRCFRQ							
Type	I							

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME1							
Type	A80							

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME2							
Type	A80							

User Defined Constants Card. Optional card for user defined constants. This input ends at the next keyword ("*") card.

Card 5	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
NODE	Coupled node/node set
TYPE	Region type: EQ.0: Node ID EQ.1: Node set ID
ROMID	Cadlm's ROM ID
PID	LS-DYNA part/part set ID
PTYPE	Type for PID: EQ.0: Part ID (Default) EQ.1: Part set ID
IOPT	Option for exchanging data between LS-DYNA/Cadlm Quasar: EQ.0: Default. LS-DYNA outputs nodal translational and rotational coordinates and receives nodal translational and rotational forces. EQ.1: LS-DYNA outputs nodal translational and rotational displacements and receives nodal translational and rotational forces. EQ.2: LS-DYNA outputs nodal translational coordinates and receives nodal translational forces. EQ.3: LS-DYNA outputs nodal translational displacements and receives nodal translational forces.
CID	Reference coordinate system needed to transform the data from the LS-DYNA global system to the Quasar local system

VARIABLE	DESCRIPTION
EX_ND	Node set to exclude from Quasar output. LS-DYNA still expects the complete set of data. (Quasar can predict the forces from a reduced data set.)
FRCFRQ	Number of cycles between QUASAR force updates for the coupling interface
FILENAME1	LS-DYNA output file to QUASAR
FILENAME2	QUASAR output file to LS-DYNA
VAR*i*	User defined constant

*DEFINE_REGION

Purpose: Define a volume of space, optionally in a local coordinate system.

Card Summary:

Card 1. This card is required.

ID	TITLE

Card 2. This card is required.

TYPE	CID	MOVE					

Card 3a. This card is included if and only if TYPE = 0.

XMN	XMX	YMN	YMX	ZMN	ZMX		

Card 3b. This card is included if and only if TYPE = 1.

XC1	YC1	ZC1	RMIN1	RMAX1			

Card 3c.1. This card is included if and only if TYPE = 2.

XC2	YC2	ZC2	AX2	AY2	AZ2	RMIN2	RMAX2

Card 3c.2. This card is included if and only if TYPE = 2.

L2							

Card 3d.1. This card is included if and only if TYPE = 3.

XC3	YC3	ZC3	AX3	AY3	AZ3	BX3	BY3

Card 3d.2. This card is included if and only if TYPE = 3.

BZ3	RA3	RB3	RC3				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Region ID
TITLE	Title for this region

Card 2	1	2	3	4	5	6	7	8
Variable	TYPE	CID	MOVE					
Type	I	I	0					

VARIABLE	DESCRIPTION
TYPE	Region type: EQ.0: Box EQ.1: Sphere or spherical shell EQ.2: Cylinder or cylindrical shell, infinite or finite in length EQ.3: Ellipsoid
CID	Optional local coordinate system ID. If given, all the following input parameters will be interpreted in this coordinate system.
MOVE	Flag to specify whether the region moves: EQ.0: Region is stationary. EQ.1: Region moves to follow the local origin and rotates with the local coordinate system (see CID).

Rectangular Prism. Use when TYPE = 0.

Card 3a	1	2	3	4	5	6	7	8
Variable	XMN	XMX	YMN	YMX	ZMN	ZMX		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
XMN	Lower *x* limit of box
YMN	Lower *y* limit of box
ZMN	Lower *z* limit of box
XMX	Upper *x* limit of box
YMX	Upper *y* limit of box
ZMX	Upper *z* limit of box

Sphere. Use when TYPE = 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	XC1	YC1	ZC1	RMIN1	RMAX1			
Type	R	R	R	R	R			
Default	0.0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
XC1, YC1, ZC1	Coordinates of the center of the sphere
RMIN1, RMAX1	The inner and outer radii of the spherical shell. Set RMIN1 = 0 for a solid sphere

Cylinder. Use when TYPE = 2.

Card 3c.1	1	2	3	4	5	6	7	8
Variable	XC2	YC2	ZC2	AX2	AY2	AZ2	RMIN2	RMAX2
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Cylinder. Use when TYPE = 2.

Card 3c.2	1	2	3	4	5	6	7	8
Variable	L2							
Type	F							
Default	∞							

VARIABLE	DESCRIPTION
XC2, YC2, ZC2	A point on the cylindrical axis
AX2, AY2, AZ2	A vector which defines the direction of the axis of the cylinder
RMIN2, RMAX2	The inner and outer radii of the cylindrical shell. Set RMIN2 = 0 for a solid cylinder.
L2	Length of the cylinder. If L2 = 0.0, an infinite cylinder is defined. Otherwise the cylinder has one end at the point (XC2, YC2, ZC2) and the other at a distance L2 along the axis in the direction of the vector (AX2, AY2, AZ2).

Ellipsoid. Use when TYPE = 3.

Card 3d.1	1	2	3	4	5	6	7	8
Variable	XC3	YC3	ZC3	AX3	AY3	AZ3	BX3	BY3
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Ellipsoid. Use when TYPE = 3.

Card 3d.2	1	2	3	4	5	6	7	8
Variable	BZ3	RA3	RB3	RC3				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
XC3, YC3, ZC3	Coordinates of the center of the ellipsoid
AX3, AY3, AZ3	A vector in the direction of the first axis of the ellipsoid (axis **a**)
BX3, BY3, BZ3	A vector, $\tilde{\mathbf{b}}$, in the plane of the first and second axes of the ellipsoid. The third axis of the ellipsoid (axis **c**) will be in the direction of $\mathbf{a} \times \tilde{\mathbf{b}}$ and finally the second axis $\mathbf{b} = \mathbf{c} \times \mathbf{a}$
RA3, RB3, RC3	The semi-axis lengths of the ellipsoid

*DEFINE_SALEAB_PARAMETERS

Purpose: Serve as one of the component keywords of *AIRBAG_SALE. *AIRBAG_SALE is a wrapper keyword that the keyword reader internally translates into a series of component keywords. Please refer to the manual page on *AIRBAG_SALE for a detailed description. This card holds miscellaneous parameters only used for airbag simulations. They are defined in *AIRBAG_SALE but do not fit into any other existing S-ALE keywords designed for general applications.

The initialization phase generates this keyword along with some others and outputs them into saleabcvrt.inc. The converted keywords provide exactly the same parameters and results as the original *AIRBAG_SALE card. saleabcvrt.inc is provided for experienced ALE users to fine-tune certain flags in subsequent runs. However, this keyword should be kept unchanged.

Card 1	1	2	3	4	5	6	7	8
Variable	NABSALE	IUNIT	PATM					
Type	I	I	F					
Default	none	0	none					

Card 2. Define NABSALE cards; one for each original *AIRBAG_SALE instantiation.

Card 2	1	2	3	4	5	6	7	8
Variable	FSIEXT	FSIINT	FSILCK	CVID	PAIR	TSWT		
Type	I	I	I	I	F	F		
Default	0	0	0	none	none	1.0e16		

VARIABLE	DESCRIPTION
NABSALE	Number of original *AIRBAG_SALE cards
IUNIT	IUNIT in original *AIRBAG_SALE cards
PATM	PATM in original *AIRBAG_SALE cards

VARIABLE	DESCRIPTION
FSIEXT	FSI ID of *ALE_STRUCTURED_FSI_ABEXT card coupling to exterior parts. See Remark 1.
FSIINT	FSI ID of *ALE_STRUCTURED_FSI_ABINT card coupling to interior parts. See Remark 1.
FSILCK	FSI ID of *ALE_STRUCTURED_FSI card for preventing contact locking. See Remark 1.
CVID	ID of *AIRBAG_HYBRID to perform switch if prescribed in the original *AIRBAG_SALE card.
PAIR	PAIR in the original *AIRBAG_SALE card.
TSWT	TSWT in the original *AIRBAG_SALE card.

Remarks:

1. **FSIEXT, FSIINT, FSILCK.** The conversion process of *AIRBAG_SALE generates and outputs FSI IDs FSIEXT, FSIINT, and FSILCK. Do not change any of these IDs because these IDs, combined with the *ALE_STRUCTURED_FSI cards, determine the interior and exterior portions of each airbag in subsequent runs in the absence of the original *AIRBAG_SALE card.

*DEFINE_SD_ORIENTATION

Purpose: Define orientation vectors for discrete springs and dampers. These orientation vectors are optional for this element class. Four alternative options are possible. With the first two options, IOP = 0 or 1, the vector is defined by coordinates and is fixed permanently in space. The third and fourth option orients the vector based on the motion of two nodes, so that the direction can change as the line defined by the nodes rotates.

Card	1	2	3	4	5	6	7	8
Variable	VID	IOP	XT	YT	ZT	NID1	NID2	
Type	I	I	F	F	F	I	I	
Default	0	0	0.0	0.0	0.0	0	0	

VARIABLE	DESCRIPTION
VID	Orientation vector ID. A unique ID number must be used.
IOP	Option (see Remark 1): EQ.0: deflections/rotations are measured and forces/moments applied along the following orientation vector. EQ.1: deflections/rotations are measured and forces/moments applied along the axis between the two spring/damper nodes projected onto the plane normal to the following orientation vector. EQ.2: deflections/rotations are measured and forces/moments applied along a vector defined by the following two nodes. EQ.3: deflections/rotations are measured and forces/moments applied along the axis between the two spring/damper nodes projected onto the plane normal to the a vector defined by the following two nodes.
XT	x-value of orientation vector. Define if IOP = 0 or 1.
YT	y-value of orientation vector. Define if IOP = 0 or 1.
ZT	z-value of orientation vector. Define if IOP = 0 or 1.
NID1	Node 1 ID. Define if IOP = 2 or 3.

VARIABLE	DESCRIPTION
NID2	Node 2 ID. Define if IOP = 2 or 3.

Remarks:

1. **Orientation Vectors.** The orientation vectors defined by options 0 and 1 are fixed in space for the duration of the simulation. Options 2 and 3 allow the orientation vector to change with the motion of the nodes. Generally, the nodes should be members of rigid bodies, but this is not mandatory. When using nodes of deformable parts to define the orientation vector, care must be taken to ensure that these nodes will not move past each other. If this happens, the direction of the orientation vector will immediately change with the result that initiate severe instabilities can develop.

*DEFINE_SET_ADAPTIVE

Purpose: To control the adaptive refinement level by element or part set.

Card 1	1	2	3	4	5	6	7	8
Variable	SETID	STYPE	ADPLVL	ADPSIZE				
Type	I	I	I	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
SETID	Element set ID or part set ID
STYPE	Set type for SETID: EQ.1: element set EQ.2: part set
ADPLVL	Adaptive refinement level for all elements in SETID set.
ADSIZE	Minimum element size to be adapted based on element edge length for all elements in SETID set.

Remarks:

1. **Element Restrictions.** This option is for 3D-shell *h*-adaptivity only at the present time.

2. **Keyword Priority.** Three keywords control refinement level: *CONTROL_ADAPTIVE, *DEFINE_BOX_ADAPTIVE, and *DEFINE_SET_ADAPTIVE. *CONTROL_ADAPTIVE must be included in the keyword deck for any adaptivity to be done. If all three keywords are defined in a keyword deck, *DEFINE_BOX_ADAPTIVE defines the refinement within the box; it unlike the other two keywords does not control the minimum element size to be adapted. *DEFINE_SET_ADAPTIVE has the next highest priority for defining refinement level while *CONTROL_ADAPTIVE has the least priority. This keyword is useful for defining different refinement levels for different parts.

3. **Multiple Refinement Definitions.** If there are multiple definitions of refinement level or element size for any elements, the latter one will be used.

*DEFINE_SHELL_SWITCH

Purpose: Define a region in which the shell formulation and/or number of through-thickness integration points are changed. Note that this change only occurs during initialization.

Card 1	1	2	3	4	5	6	7	8
Variable	BOXID	PSID	SHLFORM	NIP	PERCELIN			
Type	I	I	I	I	I			
Default	none	0	0	0	100			

VARIABLE	DESCRIPTION
BOXID	Box ID (see *DEFINE_BOX) of a box region in which the shell formulation and number of through-thickness integration points are switched. The change applies only to elements within the box at initialization.
PSID	Optional part set ID (see *SET_PART). If defined, only shell elements belonging to this part set are switched if the shell elements are in the box. If zero or undefined, all shell elements in the box are switched.
SHLFORM	Shell formulation to switch to. See ELFORM on *SECTION_SHELL for a list of available shell formulations. See Remark 1 for a list of switching restrictions. If set to 0 or left undefined, the element formulation remains unchanged.
NIP	Number of through-thickness integration points to switch to. If set to 0 or left undefined, the number of integration points remains unchanged.
PERCELIN	Percentage of elements belonging to a part that must be inside the box for the switch to occur for that part. By default, the whole part must be inside the box.

Remarks:

1. **Shell formulation switching restrictions.** The following restrictions apply when switching between shell formulations:

a) 2D plane strain shells cannot be switched to axisymmetric shells and vice versa. Also, 2D shells cannot be switched to 3D shells and vice versa.

b) Shell formulations without midside nodes cannot be changed to shell formulations with midside nodes. However, shell formulations with midside nodes can be changed to shell formulations without midside nodes.

c) User-defined shells cannot be switched.

d) EFG, IGA, and normal 3D shells cannot be switched between each other. In other words, IGA shells cannot be switched to normal 3D shells, and vice versa; EFG shells cannot be switched to normal 3D shells, and vice versa; and EFG cannot be switched to IGA shells and vice versa.

e) Composite shells cannot be switched.

f) Shells with a rigid (*MAT_020) or null material (*MAT_009) are not switched.

*DEFINE_SPG_TO_SPG_COUPLING

Purpose: Define a coupling interface between two Smoothed Particle Galerkin (SPG) parts.

Card 1	1	2	3	4	5	6	7	8
Variable	SURFA	SURFB	SPGPA	SPGPB				
Type	I	I	I	I				
Default	none	none	none	none				

Card 2	1	2	3	4	5	6	7	8
Variable	SBC	FS	FD	DC	SFP	THK	FRQ	
Type	I	F	F	F	F	F	I	
Default	0	0.	FS	0.	0.1	0.5	50	

VARIABLE	DESCRIPTION
SURFA	Segment set ID for the surface of SPG part SPGPA
SURFB	Segment set ID for the surface of SPG part SPGPB
SPGBA	SPG part ID
SPGPB	SPG part ID
SBC	Type of boundary condition: EQ.0: Free-slip boundary EQ.1: No-slip boundary (tied after contact)
FS	Static coefficient of friction. If SBC = 1, then FS is not used.
FD	Dynamic coefficient of friction. It is set to the value of FS by default.

VARIABLE	DESCRIPTION
DC	Exponential decay of the friction coefficient
SFP	Scaling factor of the penetration stiffness coefficient along the normal direction of the contact interface. The default value is 0.1.
THK	Thickness scaling factor for contact search. During initialization, the code searches for the particle that is closest to the surface. The distance of this particle from the surface at the beginning of the simulation is scaled by THK. This scaled distance is used as the contact thickness for the surface's segments. The default value of THK is 0.5.
FRQ	Bucket sorting frequency. The default value is 50.

*DEFINE_SPG_TO_SURFACE_COUPLING

Purpose: Define a coupling interface between smoothed particle Galerkin (SPG) particles and a surface.

Card 1	1	2	3	4	5	6	7	8
Variable	SPGP	SURF	SPTYPE	SFTYPE				
Type	I	I	I	I				
Default	none	none	1	0				

Card 2	1	2	3	4	5	6	7	8
Variable	SBC	FS	FD	DC	SFP	THK	FRQ	
Type	I	F	F	F	F	F	I	
Default	0	0.	FS	0.	0.1	0.5	50	

VARIABLE	DESCRIPTION
SPGP	Part ID/Part set ID for the SPG particles
SURF	Segment set ID specifying the surface
SPTYPE	Type for SPGP: EQ.0: Part set ID EQ.1: Part ID
SFTYPE	Type for SURF: EQ.0: Segment set ID
SBC	Type of boundary condition: EQ.0: Free-slip boundary EQ.1: Non-slip boundary (tied after contact)

VARIABLE	DESCRIPTION
FS	Static coefficient of friction. If SBC = 1, then FS is not used.
FD	Dynamic coefficient of friction. It is set to the value of FS by default.
DC	Exponential decay of the friction coefficient
SFP	Scaling factor of the penetration stiffness coefficient along the normal direction of the contact interface. Default value is 0.1.
THK	Thickness scaling factor for contact search. During initialization, LS-DYNA searches for the particle that is closest to the surface. The distance of this particle from the surface at the beginning of the simulation is scaled by THK. This scaled distance is used as the contact thickness for the surface's segments. The default value of THK is 0.5.
FRQ	Bucket sorting frequency. Default is 50.

*DEFINE_SPH_ACTIVE_REGION

Purpose: The purpose of this keyword is to increase the efficiency of the SPH method's neighborhood search algorithm by specifying an *active region*. All SPH elements located outside of the active region are deactivated. This card supports active regions consisting of the volume bounded by two closed surfaces (boxes, centered cylinders, and centered spheres are currently supported). Once the SPH particle is deactivated, it will stay inactive.

Card Summary:

Card 1. This card is required.

ID	TYPE	STYPE	CYCLE	NID	ICID	IBUFF	

Card 2a.1. Include this card if STYPE = 0.

XIMIN	YIMIN	ZIMIN	XIMAX	YIMAX	ZIMAX		

Card 2a.2. Include this card if STYPE = 0.

XOMIN	XOMIN	ZOMIN	XOMAX	YOMAX	ZOMAX		

Card 2b.1. Include this card if STYPE = 1.

X0	Y0	Z0	XH	YH	ZH		

Card 2b.2. Include this card if STYPE = 1.

RMIN	ZMIN	RMAX	ZMAX				

Card 2c.1. Include this card if STYPE = 2.

X0	Y0	Z0					

Card 2c.2. Include this card if STYPE = 2.

RMIN	RMAX						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	STYPE	CYCLE	NID	ICID	IBUFF	
Type	I	I	I	I	I	I	I	
Default	none	0	0	1	0	0	0	

VARIABLE	DESCRIPTION
ID	Part Set ID/Part ID
TYPE	EQ.0: Part set EQ.1: Part
STYPE	Type of the region. EQ.0: Rectangular box EQ.1: Cylinder EQ.2: Sphere
CYCLE	Number of cycles between each check
NID	Referential nodal ID. The SPH box will move with this node.
ICID	Local coordinate system ID
IBUFF	Buffer zone flag (only used when STYPE = 0): EQ.0: Particles on the edge of the outer box do not get any special treatment. EQ.1: Particles on the edge of the outer box are frozen in space and act as neighbors for active particles inside the box. This option is mainly used for fluid simulations to prevent the fluid from spilling out of the activation box.

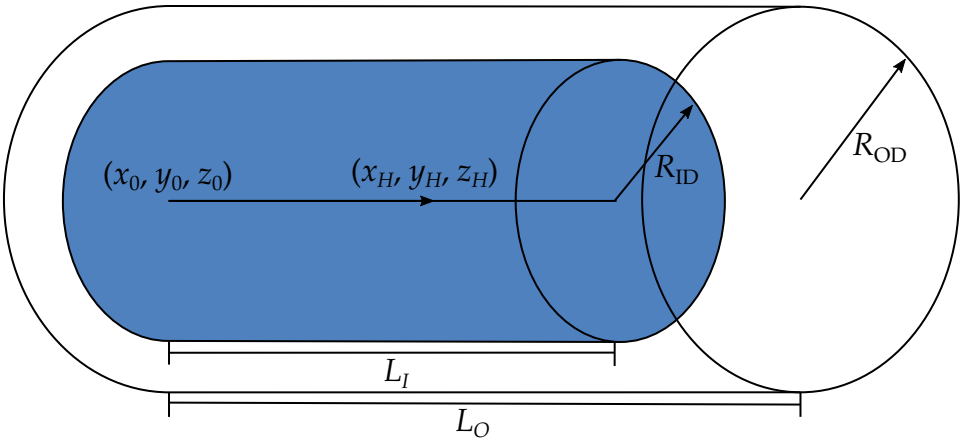
Interior Rectangular Box Card. This card is included if STYPE = 0.

Card 2a.1	1	2	3	4	5	6	7	8
Variable	XIMIN	YIMIN	ZIMIN	XIMAX	YIMAX	ZIMAX		
Type	F	F	F	F	F	F		
Default	none	none	none	None	none	none		

Outer Rectangular Box Card. This card is included if STYPE = 0.

Card 2a.2	1	2	3	4	5	6	7	8
Variable	XOMIN	XOMIN	ZOMIN	XOMAX	YOMAX	ZOMAX		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
XIMIN, YIMIN, ZIMIN	Minimum x, y, z coordinate of the inner box
XIMAX, YIMAX, ZIMAX	Maximum x, y, z coordinates of the inner box
XOMIN, YOMIN, ZOMIN	Minimum x, y, z coordinate of the outer box
XOMAX, YOMAX, ZOMAX	Maximum x, y, z coordinates of the outer box




```
*DEFINE_SPH_ACTIVE_REGION
1,1,1,100
x0,y0,z0,xH,yH,zH
RID,LI,ROD,LO
```

Figure 17-62. Example DEFINE_SPH_ACTIVE_REGION

Cylinder Axis Card. This card is included if STYPE = 1.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	XH	YH	ZH		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

Cylinder Radii Card. This card is included if STYPE = 1.

Card 2b.2	1	2	3	4	5	6	7	8
Variable	RMIN	ZMIN	RMAX	ZMAX				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
X0, Y0, Z0	Coordinates of the center of the cylinder base. The nested cylinders share the same starting base plane. This point also serves as the tail for the vector specifying the direction of the cylinders' axis. See Figure 17-62.
XH, YH, ZH	Coordinates for the head of the cylinders' axial direction vector.
RMIN, ZMIN	Radius and length of the interior cylinder.
RMAX, ZMAX	Radius and length of the outer cylinder.

Center of Sphere Card. This card is included if STYPE = 2.

Card 2c.1	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0					
Type	F	F	F					
Default	none	none	none					

Sphere Radii Card. This card is included if STYPE = 2.

Card 2c.2	1	2	3	4	5	6	7	8
Variable	RMIN	RMAX						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
X0, Y0, Z0	The spheres' center.
RMIN	Radius of the interior sphere
RMAX	Radius of the outer sphere

*DEFINE_SPH_AMBIENT_DRAG

Purpose: Add drag forces to SPH particles that use the implicit incompressible formulation (FORM = 13 in *CONTROL_SPH). For a wading simulation, this keyword sets air properties.

Card 1	1	2	3	4	5	6	7	8
Variable	ICPL	VX	VY	VZ	RHOA	MUA	SFTENS	
Type	I	F	F	F	F	F	F	
Default	0	0.0	0.0	0.0	none	none	none	

VARIABLE	DESCRIPTION
ICPL	Coupling with ICFD: EQ.0: No coupling EQ.1: Import ambient velocity field from a steady-state CFD analysis in a profile file format. Provide the file name in Card 2. See Example 1 below for a sample profile file.
VX, VY, VZ	Velocity components of the ambient material
RHOA	Density of the ambient material
MUA	Viscosity of the ambient material
SFTENS	Surface tension coefficient for the interface between the SPH fluid and ambient material

ICPL Card. Include this card if the ICPL = 1.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A60							

VARIABLE	DESCRIPTION
FILENAME	Name of the file containing steady-state CFD data for the ambient air velocity field.

Example 1:

The following example illustrates the profile file format, which can be exported directly by Fluent. Note that the first line is empty.

```
[Name]
cfd_result

[Data]
          x,          y,          z,         vx,         vy,         vz
        0.1,        0.0,        0.0,        1.0,        0.0,        0.0
        0.1,        0.0,        0.1,        1.0,        0.0,        0.0
        0.1,        0.1,        0.1,        1.0,        0.0,        0.0
        0.1,        0.1,        0.0,        1.0,        0.0,        0.0
        0.2,        0.0,        0.0,        0.9,        0.0,        0.0
        0.2,        0.0,        0.1,        0.9,        0.0,        0.0
        0.2,        0.1,        0.1,        0.9,        0.0,        0.0
        0.2,        0.1,        0.0,       0.85,        0.0,        0.0
```

***DEFINE_SPH_DE_COUPLING**_*{OPTION}*

Purpose: Define a penalty based contact. This option is to be used for the node to node contacts to couple the SPH solver and the discrete element sphere (DES) solver.

The available options include:

<BLANK>

ID

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	DID	HEADING						
Type	I	A80						
Default	none	none						

SPH Part Cards. Include as many of this card as desired. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SPHID	DESID	SPHTYP	DESTYP	PFACT	DFACT	SPHBOX	
Type	I	I	I	I	F	F	I	
Default	none	none	none	none	1.0	0.	none	

VARIABLE	DESCRIPTION
DID	Definition ID. This must be a unique number.
HEADING	Definition descriptor. It is suggested that unique descriptions be used.
SPHID	SPH part or part set ID.
DESID	DES part or part set ID.

VARIABLE	DESCRIPTION
SPHTYP	SPH part type: EQ.0: Part set ID, EQ.1: Part ID.
DESTYP	DES part type: EQ.0: Part set ID, EQ.1: Part ID.
PFACT	Penalty scale factor
DFACT	Penalty scale factor for contact damping coefficient
SPHBOX	BOX ID for SPH parts, See Remarks.

Remarks:

SPHBOX is used to define the box IDs for the SPH parts. Only the particles inside the boxes are defined for the node to node contacts.

*DEFINE_SPH_INJECTION

Purpose: Inject SPH elements from user defined grid points.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NSID	CID	VX	VY	VZ	AREA	VMAG
Type	I	I	I	F	F	F	F	I
Default	none	none	none	0.0	0.0	0.0	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable	TBEG	TEND	NID	RADIUS	XLEN	YLEN	PSIZE	
Type	F	F	I	F	F	F	F	
Default	0.0	10^{20}	0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
PID	Part ID of newly generated SPH elements.
NSID	Node set ID. Nodes are used for initial injection position for the SPH elements. Ignored if RADIUS, XLEN or YLEN are defined below.
CID	Local coordinate system ID; see *DEFINE_COORDINATE_SYSTEM for example. The local x and y-directions define the injection plane, while the local z-direction defines the normal to the injection plane. A CID is always required. This coordinate system is used for the definition of injection velocity (VX,VY,VZ) as well.
VX, VY, VZ	Velocity of the injected particles: $\mathbf{v}$ = (VX, VY, VZ), defined in the local coordinate system from CID.
AREA	The area of initial injection surface. The density of injection flow comes from the material models; see *MAT definition. This parameter can be left blank if the nodes provided in NSID are uniformly

VARIABLE	DESCRIPTION
	distributed, or if RADIUS, XLEN or YLEN are defined below, in which case AREA will be automatically computed.
VMAG	Injected particle velocity multiplier: GT.0: The velocity of the injected particles is multiplied by VMAG. LT.0: |VMAG| is a curve ID defining the magnitude of the velocity vector with respect to time, for variable injection speed.
TBEG	Birth time
TEND	Death time
NID	Optional Node ID. If defined, the injection plane follows the motion of this node.
RADIUS	If this option is nonzero, the node set provided in NSID is ignored. Instead, this parameter defines the radius of a circular injection surface centered at the origin of the coordinate system defined in CID, using the local Z-axis of CID as normal to the injection plane. Particles are uniformly generated with a spacing of PSIZE.
XLEN	If this option is nonzero and RADIUS is zero, the node set provided in NSID is ignored. Instead, a rectangular injection surface is automatically created, centered at the origin of the coordinate system defined in CID, using the local Z-axis of CID as normal to the injection plane. This rectangular injection has dimensions of XLEN and YLEN in the local coordinate system defined by CID. Particles are uniformly generated with a spacing of PSIZE.
YLEN	See XLEN.
PSIZE	If RADIUS or XLEN / YLEN is defined, the particles are automatically placed on the defined injection geometry, with a spacing of PSIZE.

*DEFINE_SPH_INJECTION_SIMPLIFIED_{OPTION1}

For *OPTION1*, available options are:

<BLANK>

FLOWRATE

Purpose: Inject SPH elements like *DEFINE_SPH_INJECTION but only for a circular injection area. This keyword has simplified syntax compared to *DEFINE_SPH_INJECTION for this geometry.

Card Summary:

Card 1a. Include this card when not using the keyword option (<BLANK>).

PID	VEL	RADIUS	PSIZE	OFFST	CID	NID	

Card 1b. Include this card when using the FLOWRATE keyword option.

PID	FLWRT	RADIUS	PSIZE	OFFST	CID	NID	

Card 2. This card is required.

XC	YC	ZC	DNX	DNY	DNZ	TBEG	TEND

Data Card Definitions:

Include this card when not using the keyword option (<BLANK>).

Card 1a	1	2	3	4	5	6	7	8
Variable	PID	VEL	RADIUS	PSIZE	OFFST	CID	NID	
Type	I	F	F	F	F	I	I	
Default	none	none	none	0.0	0.0	0	0	

VARIABLE	DESCRIPTION
PID	Part ID of newly generated SPH elements
VEL	Injected particle velocity in the injection direction: GT.0.0: The velocity of the injected particles is constant, equal to

VARIABLE	DESCRIPTION
	VMAG. LT.0.0: |VMAG| is a curve ID defining the injection velocity with respect to time for variable injection speed.
RADIUS	Radius of the circular injection surface. Particles are uniformly generated across the surface at a spacing of PSIZE.
PSIZE	Spacing between particles on the injection surface
OFFST	Option to offset the centroid of the injection surface, (XC, YC, ZC), along the normal vector, (DNX, DNY, DNZ),` by a distance of OFFST. Only valid when CID is zero.
CID	Local coordinate system ID; see *DEFINE_COORDINATE_SYSTEM for example. The origin of this local coordinate system defines the centroid of the injection area, and the local *z*-direction defines the normal to the injection area, which is the direction in which particles will be injected.
NID	Optional Node ID. If defined, the injection plane follows the motion of this node.

Include this card when using the FLOWRATE keyword option.

Card 1b	1	2	3	4	5	6	7	8
Variable	PID	FLWRT	RADIUS	PSIZE	OFFST	CID	NID	
Type	I	F	F	F	F	I	I	
Default	none	none	none	0.0	0.0	0	0	

VARIABLE	DESCRIPTION
PID	Part ID of newly generated SPH elements
FLWRT	Volumetric flow rate of injection: GT.0.0: The volumetric flow rate is constant, equal to FLWRT. LT.0.0: |FLWRT| is a curve ID defining the volumetric flow rate with respect to time for variable injection flow rate.

VARIABLE	DESCRIPTION
RADIUS	Radius of the circular injection surface. Particles are uniformly generated across the surface at a spacing of PSIZE.
PSIZE	Spacing between particles on the injection surface
OFFST	Option to offset the centroid of the injection surface, (XC, YC, ZC), along the normal vector, (DNX, DNY, DNZ),` by a distance of OFFST. Only valid when CID is zero.
CID	Local coordinate system ID; see *DEFINE_COORDINATE_SYSTEM for example. The origin of this local coordinate system defines the centroid of the injection area, and the local *z*-direction defines the normal to the injection area, which is the direction in which particles will be injected.
NID	Optional Node ID. If defined, the injection plane follows the motion of this node.

Card 2	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	DNX	DNY	DNZ	TBEG	TEND
Type	F	F	I	F	F	F	F	F
Default	0.0	10^{20}	0	0.0	0.0	0.0	0.0	10^{20}

VARIABLE	DESCRIPTION
XC,YZ,ZC	Centroid of the injection surface. Ignored if CID is nonzero.
DNX, DNY, DNZ	Components of the normal vector to the injection surface. Ignored if CID is nonzero.
TBEG	Birth time
TEND	Death time

*DEFINE_SPH_MASSFLOW_PLANE

Purpose: To measure SPH mass flow across a defined plane. See also the accompanying keyword *DATABASE_SPHMASSFLOW which controls the output frequency.

Card 1	1	2	3	4	5	6	7	8
Variable	PRTCLSID	SURFSID	PTYPE	STYPE				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
PRTCLSID	Node set ID, node ID, part set ID or part ID specifying the SPH particles to be measured. PTYPE below indicates the ID type specified by PRTCLSID.
SURFSID	Part set ID or part ID defining the surface across which the flow is measured. STYPE below indicates the ID type specified by SURFSID.
PTYPE	PRTCLSID type: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
STYPE	SURFSID type: EQ.0: Part set EQ.1: Part

*DEFINE_SPH_MESH_BOX

Purpose: Generate SPH particles inside a given box.

Card 1	1	2	3	4	5	6	7	8
Variable	XMIN	YMIN	ZMIN	XLEN	YLEN	ZLEN		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

Card 2	1	2	3	4	5	6	7	8
Variable	IPID	NX	NY	NZ		IDSEG	SFSP	
Type	I	I	I	I		I	F	
Default	none	none	none	none		0	1.0	

VARIABLE	DESCRIPTION
XMIN	Minimum x-coordinate
YMIN	Minimum y-coordinate
ZMIN	Minimum z-coordinate
XLEN	Box length in the x-direction
YLEN	Box length in the y-direction
ZLEN	Box length in the z-direction
IPID	Part ID for generated SPH elements
NX	Number of SPH particles in the x-direction
NY	Number of SPH particles in the y-direction

VARIABLE	DESCRIPTION
NZ	Number of SPH particles in the z-direction
IDSEG	Segment set ID that can be used to removed generated SPH elements. segment set is used to split the box into two regions, one that has SPH elements and one without SPH (see Remark 2). The sign of IDSEG determines which region keeps the SPH elements. Also, to avoid sudden movement, elements that are "too close" to the segment set will be removed, regardless of the sign of IDSEG. Too close means the normal distance from the center of the SPH element to the nearest segment is smaller than the SPH smoothing length scaled by SFSP. EQ.0: No generated elements are removed. GT.0: Keep the SPH element if it lies nominally in the normal direction of the segments in the segment set. LT.0: Keep the SPH element if it lies nominally in the reverse normal direction of segments in the segment set.
SFSP	Scale factor for SPH smoothing length (active only when IDSEG ≠ 0). If the normal distance between an SPH particle and the nearest segment is smaller than this distance, the SPH element is removed.

Remarks:

1. **Interparticle Distances.** Interparticle distances are calculated as XLEN/NX, YLEN/NY, and ZLEN/NZ

2. **Region for Element Removal.** The region created by the segment set, IDSEG, must either define a closed volume in the box or intersect the box. If it defines an open surface in the box, the elements that are to be kept cannot be uniquely determined.

*DEFINE_SPH_MESH_OBJ

Purpose: Import an OBJ file whose geometry represents the same surface as an incompressible SPH part comprised of structure particles. Both the OBJ geometry and incompressible SPH part follow the motion of a rigid body. Note that the ISPH part is not internally generated but must be generated with a pre-processor. A script is provided with LS-PrePost to generate these particles. The OBJ geometry is included in d3plot and (if requested) ISPHFOR output files. This feature enables some field variables for incompressible SPH to be evaluated on the OBJ geometry.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A							

Card 1	1	2	3	4	5	6	7	8
Variable	PPID	PID	SPHPID					
Type	I	I	I					
Default	none	optional	none					

VARIABLE	DESCRIPTION
FILENAME	Name of the OBJ file to be included
PPID	Parent part ID. The OBJ geometry will follow the motion of the rigid part with ID PPID.
PID	Optional part ID to identify this OBJ file in d3plot and ISPHFOR outputs. An ID will be automatically generated if this parameter is left blank.
SPHPID	SPH part ID. ISPH particles from this part will follow the motion of the rigid part with ID PPID.

Remarks:

1. **Data compression.** When this keyword is included in the input deck, DCOMP = 5 is automatically set for *DATABASE_EXTENT_BINARY.

2. **Required ISPH material.** Part ISPHPID must use *MAT_SPH_INCOMPRESSIBLE_STRUCTURE (*MAT_SPH_03) as its material.

*DEFINE_SPH_MESH_SURFACE

Purpose: Generate and place SPH elements on the surface of triangular shell elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	TYPE	SPHPID	SPHXID	NSID	SPACE	IOUT	
Type	I	I	I	I	I	F	I	
Default	none	none	Rem 1	Rem 1	0	none	0	

VARIABLE	DESCRIPTION
SID	Part or part set ID for the region of the mesh upon which the SPH elements will be placed
TYPE	SID type: EQ.0: part set ID EQ.1: part ID
SPHPID	Part ID for generated SPH elements
SPHXID	Section ID for generated SPH elements
NSID	If defined, this card creates a node set with ID NSID (see *SET_NODE) for the nodes generated by this card.
SPACE	Maximum space between SPH elements
IOUT	Keyword file output: EQ.0: no output (default) EQ.1: output generated nodes, SPH elements and node set to a keyword file with SPH_surface_ prefix.

Remarks:

1. **SPHPID and SPHXID.** Part ID and/or section ID will be generated by this card if they are not provided in the input.

*DEFINE_SPH_TO_SPH_COUPLING

Purpose: Define a penalty-based, node-to-node contact for particles of SPH parts. Note that this contact type is an alternative to inter-part particle interaction by "particle approximation;" see the field CONT in *CONTROL_SPH and the INTERACTION option of *SECTION_SPH.

Card Sets: Each set consists of a Card 1 and may include an additional Card 2. Unless the card following Card 1 contains an "&" in its first column, the optional card is not read. Provide as many sets as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SIDA	SIDB	SATYP	SBTYP	IBOXA	IBOXB	PFACT	SRAD
Type	I	I	I	I	I	I	F	F
Default	none	none	none	none	none	none	1.0	1.0

Optional. The keyword reader identifies this card by an "&" in the first column.

Card 2	1	2	3	4	5	6	7	8
Variable	DFACT	ISOFT						
Type	F	I						
Default	0.0	0						

VARIABLE	DESCRIPTION
SIDA	Part or part set ID for one set of particles in the contact
SIDB	Part or part set ID for the other set of particles in the contact
SATYP	SIDA part type: EQ.0: Part set ID EQ.1: Part ID

VARIABLE	DESCRIPTION
SBTYP	SIDB part type: EQ.0: Part set ID EQ.1: Part ID
IBOXA	Box ID for the A parts; see Remark 1.
IBOXB	Box ID for the B parts; see Remark 1.
PFACT	Penalty scale factor; see Remark 2.
SRAD	Scale factor for nodes to nodes contact criteria; see Remark 3.
DFACT	Penalty scale factor for contact damping coefficient; see Remark 4.
ISOFT	Soft constraint option: EQ.0: Penalty formulation (default) EQ.1: Soft constraint formulation The soft constraint may be necessary if the material constants of the parts in contact have a wide variation in the elastic bulk moduli. In the soft constraint option, the interface stiffness is based on the nodal mass and the global time step size.

Remarks:

1. **Contact Particles.** IBOXA and IBOXB are used to define the box IDs for the A parts and the B parts, respectively. Only the particles inside the boxes are defined for the node to node contacts.

2. **Impact Velocity.** For High Velocity Impact problems, a smaller value ranging from 0.01 to 10^{-4} for the PFACT field is recommended. A value ranging from 0.1 to 1 is recommended for low velocity contact between two SPH parts.

3. **Contact Detection.** Contact between two SPH particles from different parts is detected when the distance of two SPH particles I and J is less than SRAD $\times \left(h_I + h_J\right)/2.0$. If SRAD < 0, the contact distance is based on the nodal volumes instead of the smoothing lengths: SRAD $\times \left(\sqrt[3]{V_I} + \sqrt[3]{V_J}\right)/2.0$.

4. **DFACT.** The default value, DFACT = 0.0, is recommended. For DFACT > 0.0, interaction between SPH parts includes a viscous effect, providing some stickiness similar to the particle approximation invoked when CONT = 0 in *CON-

TROL_SPH. At present, no recommendation can be given for a value of DFACT other than the value should be less than 1.0.

*DEFINE_SPH_VICINITY_SENSOR

Purpose: Measure the mass of specified SPH particles in the vicinity of a surface. The list of all particles flagged as within the vicinity of the surface is exported along with time at which particle moved within the vicinity. See also the accompanying keyword *DATABASE_SPHVICINITY which controls the output frequency.

Card 1	1	2	3	4	5	6	7	8
Variable	PRTCLSID	SURFSID	PTYPE	STYPE	DIST			
Type	I	I	I	I	F			
Default	0	0	0	0	0.0			

VARIABLE	DESCRIPTION
PRTCLSID	Node set ID, node ID, part set ID or part ID specifying the checked SPH particles. PTYPE below indicates the ID type specified by PRTCLSID.
SURFSID	Part set ID or part ID defining the surface. STYPE below indicates the ID type specified by SURFSID.
PTYPE	PRTCLSID type: EQ.0: Node set EQ.1: Node EQ.2: Part set EQ.3: Part
STYPE	SURFID type: EQ.0: Part set EQ.1: Part
DIST	Distance criteria. Any particle closer than this distance to the surface is considered in the vicinity.

*DEFINE_SPOTWELD_FAILURE_{OPTION}

The available options are

<BLANK>

ADD

PID

Purpose: Define spot weld failure data for the failure criterion developed by Lee and Balur (2011). This card provides the failure data for OPT = 10 on *MAT_SPOTWELD. It is available for spot welds consisting of beam elements, solid elements, or solid assemblies.

*DEFINE_SPOTWELD_FAILURE requires that the weld nodes be tied to shell elements using tied constraint based contact options:

1. For beam element welds, only *CONTACT_SPOTWELD is valid.
2. For solid element welds or solid assembly welds, valid options are the following:

   *CONTACT_TIED_SURFACE_TO_SURFACE

   *CONTACT_SPOTWELD

   *CONTACT_TIED_SHELL_EDGE_TO_SURFACE

Other tied contact types cannot be used.

The ADD keyword option adds materials to a previously defined spot weld failure data set. The PID option changes the Card 3 input to use part set ID's rather than material ID's.

Card Summary:

Card 1. This card is required.

ID	TFLAG	DC1	DC2	DC3	DC4	EXN	EXS

Card 2. This card is included unless the ADD option is used.

NAVG	D_SN	D_SS	R_SULT	TSCALE			

Card 3a. This card is included if the PID option is used. Include one card for each pair of weld parts associated with the data set. The next keyword ("*") card terminates the keyword.

PID1	PID2	SN	SS				

Card 3b. This card is included if the PID option is *not* used. Include one card for each material associated with the data set. The next keyword ("*") card terminates the keyword.

MID	SN	SS					

Data Card 1. This card contains the data set's ID and the first 7 parameters. When the ADD option is active *leave the 7 parameters blank*.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TFLAG	DC1	DC2	DC3	DC4	EXN	EXS
Type	I	I	F	F	F	F	F	F
Default	none	0	1.183	0.002963	0.0458	0.1	2.0	2.0

VARIABLE	DESCRIPTION
ID	Identification number of data set, input as FVAL on *MAT_SPOTWELD
TFLAG	Thickness flag for nominal stress calculation EQ.0: Use minimum sheet thickness EQ.1: Use average sheet thickness EQ.2: Use maximum sheet thickness EQ.3: Use sum of sheet thicknesses
DC1	Dynamic coefficient, c_1
DC2	Dynamic coefficient, c_2
DC3	Dynamic coefficient, c_3
DC4	Dynamic coefficient, c_4

VARIABLE	DESCRIPTION
EXN	Exponent on the normal term, n_n
EXS	Exponent on the shear term, n_s

Data Card 2. This card contains 3 spot weld failure data parameters. *Do not include this card when the ADD option is active*

Card 2	1	2	3	4	5	6	7	8
Variable	NAVG	D_SN	D_SS	R_SULT	TSCALE			
Type	I	F	F	F	F			
Default	0	0.0	0.0	Rem 2	1.0			

VARIABLE	DESCRIPTION
NAVG	Number of points in the time average of the load rates. See Remark 4.
D_SN	Reference value of the static normal strength, $S_{n,\text{stat}}$. See Remark 2.
D_SS	Reference value of the static shear strength, $S_{s,\text{stat}}$. See Remark 2.
R_SULT	Reference ultimate strength. See Remark 2.
TSCALE	Scale factor for thickness used in nominal stress calculations

Material-Specific Strength Data Cards with PID option. Include one card for each pair of weld parts associated with the data set. The next keyword ("*") card terminates the keyword.

Card 3a	1	2	3	4	5	6	7	8
Variable	PID1	PID2	SN	SS				
Type	I	I	F	F				
Default	none	none	Rem 2	Rem 2				

VARIABLE	DESCRIPTION
PID1	Part ID of welded shell part
PID2	Part ID of part welded to PID1
SN	Static normal strength of material MID. $S_{n,\text{stat}}$. See Remark 2.
SS	Static shear strength of material MID, $S_{s,\text{stat}}$. See Remark 2.

Material-Specific Strength Data Cards without PID option. Include one card for each material associated with the data set. The next keyword ("*") card terminates the keyword.

Card 3b	1	2	3	4	5	6	7	8
Variable	MID	SN	SS					
Type	I	F	F					
Default	none	Rem 2	Rem 2					

VARIABLE	DESCRIPTION
MID	Material ID number of welded shell material
SN	Static normal strength of material MID. $S_{n,\text{stat}}$. See Remark 2.
SS	Static shear strength of material MID, $S_{s,\text{stat}}$. See Remark 2.

Remarks:

1. **Failure Model.** This stress based failure model, which was developed by Yung-Li Lee and Santosh Balur (2011) of FCA, compares nominal stress to dynamical strengths for the weld. The weld fails when the stresses are outside of the failure surface defined as

$$\left(\frac{s_n}{S_{n,\text{dyn}}}\right)^{n_n} + \left(\frac{s_s}{S_{s,\text{dyn}}}\right)^{n_s} = 1\,,$$

where s_n and s_s are nominal stresses in the normal and tangential directions, $S_{n,\text{dyn}}$ and $S_{s,\text{dyn}}$ are dynamical strengths in the normal and tangential directions, and n_n and n_s are the exponential factors EXN and EXS, respectively. The nominal stresses are defined as

$$s_n = \frac{P_n}{Dct}, \qquad s_s = \frac{P_s}{Dct},$$

where P_n and P_s are the loads carried by the weld in the normal and tangential directions, D is the weld diameter, c is scale factor TSCALE, and t is the thickness of the welded sheets calculated according to the value of TFLAG. The dynamical strength terms in the denominator are load-rate dependent and are derived from static strength:

$$S_{n,\text{dyn}} = S_{n,\text{stat}}\left[c_1 + c_2\left(\frac{\dot{P}_n}{c_4}\right) + c_3 \log\left(\frac{\dot{P}_n}{c_4}\right)\right]$$
$$S_{s,\text{dyn}} = S_{s,\text{stat}}\left[c_1 + c_2\left(\frac{\dot{P}_s}{c_4}\right) + c_3 \log\left(\frac{\dot{P}_s}{c_4}\right)\right]$$

where the constants c_1 to c_4 are the input in the fields DC1 to DC4, $\dot{P}_n$ and $\dot{P}_s$ are the load rates, and $S_{n,\text{stat}}$ and $S_{s,\text{stat}}$ are the static strengths of the welded sheet materials (see Remark 2).

2. **Static Strengths.** When two different materials are welded, the material having the smaller normal strength (SN) determines the static strengths used for the weld. Materials that do not have SN and SS values default to D_SN and D_SS from Card 2.

   If R_SULT is defined on Card 2 and the PID keyword option is not used, then D_SN and D_SS are interpreted to be reference values of the normal and shear static strength, and the SN field on Card 3 is interpreted as a material specific ultimate strength. These values are then used to calculate material specific strength values by

$$S_{n,\text{stat}} = S_{n,\text{ref}}\left(\frac{S_u}{S_{u,\text{ref}}}\right), \qquad S_{s,\text{stat}} = S_{s,\text{ref}}\left(\frac{S_u}{S_{u,\text{ref}}}\right)$$

   where $S_{n,\text{ref}}$, $S_{s,\text{ref}}$, and $S_{u,\text{ref}}$, are D_SN, D_SS, and R_SULT on Card 2, and S_u is SN on Card 3. The SS values are ignored.

   If the PID keyword option is used, then R_SULT is ignored, and SN and SS are static strength values.

3. **Default Values.** The default values for DC1 to DC4, and EXN and EXS are based on the work Chao, Wang, Miller and Zhu (2010). and Wang, Chao, Zhu, and Miller (2010). These parameters are unitless except for DC4 which has units of force per unit time. The default value of 0.1 has units of MN/sec.

4. **Load Rate.** The load rate, $\dot{P}$, can be time averaged to reduce the effect of high frequency oscillations on the dynamic weld strength. NAVG is the number of terms in the time average.

*DEFINE_SPOTWELD_FAILURE_RESULTANTS

Purpose: Define failure criteria between part pairs for predicting spot weld failure. This table is implemented for *solid* element spot welds, which are used with the tied, constraint based, contact option: *CONTACT_TIED_SURFACE_TO_SURFACE. *Note that other tied contact types cannot be used.* The input in this section continues until then next "*" card is encountered. Default values are used for any part ID pair that is not defined. Only one table can be defined. See *MAT_SPOTWELD where this option is used whenever *OPT* = 7.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	DSN	DSS	DLCIDSN	DLCIDSS			
Type	I	F	F	I	I			
Default	0	0.0	0.0	0	0			

Failure Cards. Provide as many as necessary. The next keyword ("*") card terminates the table definition.

Card 2…	1	2	3	4	5	6	7	8
Variable	PID_I	PID_J	SNIJ	SSIJ	LCIDSNIJ	LCIDSSIJ		
Type	I	I	F	F	I	I		
Default	none	none	0.0	0.0	0	0		

VARIABLE	DESCRIPTION
ID	Identification number. Only one table is allowed.
DSN	Default value of the normal static stress at failure.
DSS	Default value of the transverse static stress at failure.
DLCIDSN	Load curve ID defining a scale factor for the normal stress as a function of strain rate. This factor multiplies DSN to obtain the failure value at a given strain rate.

VARIABLE	DESCRIPTION
DLCIDSS	Load curve ID defining a scale factor for static shear stress as a function of strain rate. This factor multiplies DSS to obtain the failure value at a given strain rate.
PID_I	Part ID I
PID_J	Part ID J
SNIJ	The maximum axial stress at failure between parts I and J. The axial stress is computed from the solid element stress resultants, which are based on the nodal point forces of the solid element.
DSSIJ	The maximum shear stress at failure between parts I and J. The shear stress is computed from the solid element stress resultants, which are based on the nodal point forces of the solid element.
LCIDSNIJ	Load curve ID defining a scale factor for the normal stress as a function of strain rate. This factor multiplies SNIJ to obtain the failure value at a given strain rate.
LCIDSSIJ	Load curve ID defining a scale factor for static shear stress as a function of strain rate. This factor multiplies SSIJ to obtain the failure value at a given strain rate.

Remarks:

The stress based failure model, which was developed by *Toyota Motor Corporation,* is a function of the peak axial and transverse shear stresses. The entire weld fails if the stresses are outside of the failure surface defined by:

$$\left(\frac{\sigma_{rr}}{\sigma_{rr}^F}\right)^2 + \left(\frac{\tau}{\tau^F}\right)^2 - 1 = 0$$

where σ_{rr}^F and τ^F are specified in the above table by part ID pairs. LS-DYNA automatically identifies the part ID of the attached shell element for each node of the spot weld solid and checks for failure. If failure is detected the solid element is deleted from the calculation.

If the effects of strain rate are considered, then the failure criteria becomes:

$$\left[\frac{\sigma_{rr}}{f_{dsn}(\dot{\varepsilon}^p)\sigma_{rr}^F}\right]^2 + \left[\frac{\tau}{f_{dss}(\dot{\varepsilon}^p)\tau^F}\right]^2 - 1 = 0$$

*DEFINE_SPOTWELD_MULTISCALE

Purpose: Associate beam sets with multi-scale spot weld types for modeling spot weld failure via the multi-scale spot weld method.

Spot Weld/Beam Set Association Cards. Provide as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPE	BSET	TYPE	BSET	TYPE	BSET	TYPE	BSET
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
TYPE	MULTISCALE spot weld type to use. See *INCLUDE_MULTI-SCALE_SPOTWELD.
BSET	Beam set which uses this multi-scale spot weld type for failure modeling.

Remarks:

See *INCLUDE_MULTISCALE_SPOTWELD for a detailed explanation of this capability.

*DEFINE_SPOTWELD_RUPTURE_PARAMETER

Purpose: Define a parameter by part ID for shell elements attached to spot weld *beam* elements using the constrained contact option: *CONTACT_SPOTWELD. *This table will not work with other contact types*. Only one table is permitted in the problem definition. Data, which is defined in this table, is used by the stress based spot weld failure model developed by *Toyota Motor Corporation*. See *MAT_SPOTWELD where this option is activated by setting the parameter OPT to a value of 9. This spot weld failure model is a development of *Toyota Motor Corporation*.

Card 1	1	2	3	4	5	6	7	8
Variable	PID							
Type	I							

Card 2	1	2	3	4	5	6	7	8
Variable	C11	C12	C13	N11	N12	N13		SIG_PF
Type	F	F	F	F	F	F		F

Card 3	1	2	3	4	5	6	7	8
Variable	C21	C22	C23	N2				SIG_NF
Type	F	F	F	F				F

Card 4	1	2	3	4	5	6	7	8
Variable	LCDPA	LCDPM	LCDPS	LCDNA	LCDNM	LCDNS		NSMT
Type	I	I	I	I	I	I		I
Default	0	0	0	0	0	0		0

VARIABLE	DESCRIPTION
PID	Part ID for the attached shell.
C11-N2	Parameters for model, see Remarks below.
SIG_PF	Nugget pull-out stress, σ_P
SIG_NF	Nugget fracture stress, σ_F
LCDPA	Curve ID defining dynamic scale factor of spot weld axial load rate for nugget pull-out mode.
LCDPM	Curve ID defining dynamic scale factor of spot weld moment load rate for nugget pull-out mode.
LCDPS	Curve ID defining dynamic scale factor of spot weld shear load rate for nugget pull-out mode.
LCDNA	Curve ID defining dynamic scale factor of spot weld axial load rate for nugget fracture mode.
LCDNM	Curve ID defining dynamic scale factor of spot weld moment load rate for nugget fracture mode.
LCDNS	Curve ID defining dynamic scale factor of spot weld shear load rate for nugget fracture mode.
NSMT	The number of time steps used for averaging the resultant rates for the dynamic scale factors.

Remarks:

This failure model incorporates two failure functions, one for nugget pull-out and the other for nugget fracture. The nugget pull-out failure function is

$$F_p = \frac{\mathrm{C11}\times\frac{A}{D^{\mathrm{N11}}} + \mathrm{C12}\times\frac{M}{D^{\mathrm{N12}}} + \mathrm{C13}\times\frac{S}{D^{\mathrm{N13}}}}{\sigma_P\left[1+\left(\frac{\dot{\varepsilon}^p}{C}\right)^{1/p}\right]},$$

where A, M, and S are the axial force, moment, and shear resultants respectively, D is the spot weld diameter, and the Cowper-Symonds coefficients are from the attached shell material model. If the Cowper-Symonds coefficients aren't specified, the term within the square brackets, [ ], is 1.0. The fracture failure function is

$$F_n = \frac{\sqrt{(\text{C21} \times A + \text{C22} \times M)^2 + 3(\text{C23} \times S)^2}}{D^{\text{N2}}\sigma_F \left[1 + \left(\frac{\dot{\varepsilon}^p}{C}\right)^{1/p}\right]} .$$

When the load curves for the rate effects are specified, the failure criteria are

$$F_p = \frac{\text{C11} \times f_{\text{dpa}}(\dot{A}) \times \frac{A}{D^{\text{N11}}} + \text{C12} \times f_{\text{dpa}}(\dot{M}) \times \frac{M}{D^{\text{N12}}} + \text{C13} \times f_{\text{dpa}}(\dot{S}) \times \frac{S}{D^{\text{N13}}}}{\sigma_P}$$

$$F_n = \frac{\sqrt{\left[\text{C21} \times f_{\text{dna}}(\dot{A}) \times A + \text{C22} \times f_{\text{dnm}}(\dot{M}) \times M\right]^2 + 3\left[\text{C23} \times f_{dns}(\dot{S}) \times S\right]^2}}{D^{\text{N2}}\sigma_F}$$

where f is the appropriate load curve scale factor. The scale factor for each term is set to 1.0 for when no load curve is specified. No extrapolation is performed if the rates fall outside of the range specified in the load curve to avoid negative scale factors. A negative load curve ID designates that the curve abscissa is the $\log_{10}$ of the resultant rate. This option is recommended when the curve data covers several orders of magnitude in the resultant rate. Note that the load curve dynamic scaling replaces the Cowper-Symonds model for rate effects.

Failure occurs when either of the failure functions is greater than 1.0.

*DEFINE_SPOTWELD_RUPTURE_STRESS

Purpose: Define a static stress rupture table by part ID for shell elements connected to spot weld *beam* elements using the constrained contact option: *CONTACT_SPOTWELD. *This table will not work with other contact types*. Only one table is permitted in the problem definition. Data, which is defined in this table, is used by the stress based spot weld failure model developed by *Toyota Motor Corporation*. See *MAT_SPOTWELD where this option is activated by setting the parameter *OPT* to a value of 6.

Part Cards. Define rupture stresses part by part. The next keyword ("*") card terminates this input.

Card	1	2	3	4	5	6	7	8
Variable	PID	SRSIG	SIGTAU	ALPHA				
Type	I	F	F	F				

VARIABLE	DESCRIPTION
PID	Part ID for the attached shell.
SRSIG	Axial (normal) rupture stress, σ_{rr}^F.
SRTAU	Transverse (shear) rupture stress, τ^F.
ALPHA	Scaling factor for the axial stress as defined by Toyota. The default value is 1.0.

Remarks:

The stress based failure model, which was developed by *Toyota Motor Corporation,* is a function of the peak axial and transverse shear stresses. The entire weld fails if the stresses are outside of the failure surface defined by:

$$\left(\frac{\sigma_{rr}}{\sigma_{rr}^F}\right)^2 + \left(\frac{\tau}{\tau^F}\right)^2 - 1 = 0$$

where σ_{rr}^F and τ^F are specified in the above table by part ID. LS-DYNA automatically identifies the part ID of the attached shell element for each node of the spot weld beam and independently checks each end for failure. If failure is detected in the end attached to the shell with the greatest plastic strain, the beam element is deleted from the calculation.

If the effects of strain rate are considered, then the failure criteria becomes:

$$\left[\frac{\sigma_{rr}}{\sigma_{rr}^F(\dot{\varepsilon}^p)}\right]^2 + \left[\frac{\tau}{\tau^F(\dot{\varepsilon}^p)}\right]^2 - 1 = 0$$

where $\sigma_{rr}^F(\dot{\varepsilon}^p)$ and $\tau^F(\dot{\varepsilon}^p)$ are found by using the Cowper and Symonds model which scales the static failure stresses:

$$\sigma_{rr}^F(\dot{\varepsilon}^p) = \sigma_{rr}^F\left[1 + \left(\frac{\dot{\varepsilon}^p}{C}\right)^{1/p}\right]$$

$$\tau^F(\dot{\varepsilon}^p) = \tau^F\left[1 + \left(\frac{\dot{\varepsilon}^p}{C}\right)^{1/p}\right]$$

where $\dot{\varepsilon}^p$is the average plastic strain rate which is integrated over the domain of the attached shell element, and the constants p and C are uniquely defined at each end of the beam element by the constitutive data of the attached shell. The constitutive model is described in the material section under keyword: *MAT_PIECEWISE_LINEAR_PLASTICITY.

The peak stresses are calculated from the resultants using simple beam theory.

$$\sigma_{rr} = \frac{N_{rr}}{A} + \frac{\sqrt{M_{rs}^2 + M_{rt}^2}}{\alpha Z} \quad \tau = \frac{M_{rr}}{2Z} + \frac{\sqrt{N_{rs}^2 + N_{rt}^2}}{A}$$

where the area and section modulus are given by:

$$A = \pi\frac{d^2}{4}$$

$$Z = \pi\frac{d^3}{32}$$

and d is the diameter of the spot weld beam.

*DEFINE_STAGED_CONSTRUCTION_PART_{OPTION}

Available options include:

<BLANK>

SET

Purpose: This keyword offers a simple way to define parts that are removed or added during the analysis. It also handles gravity load. It is available for solid, shell, thick shell, and beam element parts.

Part Cards. Provide as many as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	STGA	STGR					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part ID (or Part Set ID for the SET keyword option)
STGA	Construction stage at which part is added. See Remark 2.
STGR	Construction stage at which part is removed. See Remark 2.

Remarks:

1. **Mandatory related keywords.** If this keyword is used, the model must also contain *CONTROL_STAGED_CONSTRUCTION and *DEFINE_CONSTRUCTION_STAGES. The latter keyword defines the times at which stages STGA and STGR begin and end. See Remarks under *CONTROL_STAGED_CONSTRUCTION.

2. **Meaning of STGA and STGR.** If STGA = 0, the part is present ("active") from the start of the analysis, and the gravity load is applied with its full value from time zero onwards. If STGA > 0, the part is initially inactive ("dormant") and becomes active at the beginning of stage STGA, with the gravity load being ramped up from zero to its full value. The ramp begins at the start of stage

STGA, and the duration of the ramp is given by ATR on *DEFINE_CONSTRUCTION_STAGES.

If STGR = 0, the part is never removed and is still present at the end of the analysis. If STGR > 0, the part is removed at the beginning of stage STGR. To avoid sudden shocks that would be caused if the elements were removed instantaneously, the removal of the elements is preceded by a period during which the stresses in the elements and the gravity load acting on the elements are ramped down to zero. The ramp begins at the start of stage STGR, and the duration of the ramp is given by ATR on *DEFINE_CONSTRUCTION_STAGES.

If STGA = 0 and STGR = 0, the effect of this keyword is to apply a gravity load to the part, which is present continuously throughout the analysis.

3. **Application examples.** When modeling the construction of a building, this keyword would be applied as follows:

   a) Soil or other parts that are present throughout the analysis would have STGA = 0 and STGR = 0.

   b) Soil that is excavated would have STGA = 0 but STGR > 0.

   c) New construction would have STGA > 0 and STGR = 0.

   d) Temporary works would have STGA > 0 and STGR > STGA.

4. **Gravity load.** The acceleration due to gravity is defined by ACCEL on *CONTROL_STAGED_CONSTRUCTION. For a part to receive a gravity load by this mechanism, it must be referenced on *DEFINE_STAGED_CONSTRUCTION_PART. Using this keyword ensures that the gravity load is not applied to parts while they are dormant or after they have been removed. It also automatically ramps the gravity load up from zero when a part is added and ramps it down to zero prior to the removal of a part.

   The gravity loading can be disabled for the whole model by setting ACCEL to zero on *CONTROL_STAGED_CONSTRUCTION, and it can be overridden for specific parts by defining *LOAD_GRAVITY_PART for those parts. Note that if gravity loading is defined by a different keyword (such as *LOAD_BODY) in the same model, this loading would be additional to the gravity loading applied by *DEFINE_STAGED_CONSTRUCTION_PART. In most cases, you will not want to duplicate the gravity load in this manner.

   For this keyword, the gravity load value is calculated based on the mass of the elements of the referenced parts only (density times volume). This calculation does not include attached lumped masses (*ELEMENT_MASS).

5. **Relationship with *LOAD cards.** *DEFINE_STAGED_CONSTRUCTION_-PART is a convenience feature that reduces the amount of input data needed for many typical construction models. Internally, LS-DYNA automatically creates *LOAD_REMOVE_PART, *LOAD_GRAVITY_PART, and *LOAD_STIFFEN_-PART. referencing the same PID and using the default gravity and pre-construction stiffness factors from *CONTROL_STAGED_CONSTRUCTION. If the same PID is also referenced in the input file by any of the three types of *LOAD cards mentioned above, the load curves entered on the *LOAD cards will override the STGA and STGR inputs on *DEFINE_STAGED_CONSTRUCTION_PART.

6. **Use of ramp time.** Ramp time ATR on *DEFINE_CONSTRUCTION_STAGES is intended to reduce dynamic effects that would occur if loading or stresses were changed suddenly. It applies to gravity loading during the addition and removal of parts, meaning *LOAD_GRAVITY_PART automatically ramps up to its full value during the first ATR of the stage when a part is added (STGA), and ramped down to zero during the first ATR of the stage when a part is removed (STGR). For the stiffness, strength, and stresses, ATR applies during the removal of a part but not during the addition. Starting with R11, when a part is added, the stiffness jumps immediately to its full value at the start of STGA. This behavior is different from versions R10 and previous, where ATR was also applied to stiffness during part addition. The disadvantage of the R10 behavior was that significant deformations could occur before the part reached its full stiffness.

DEFINE_STOCHASTIC_ELEMENT_*OPTION

Options:

SOLID_VARIATION for solid elements

SHELL_VARIATION for shell elements

Purpose: Define the stochastic variation in the yield stress, damage/failure models, density, and elastic moduli for solid material models with the STOCHASTIC keyword option on an element basis. This keyword works currently with materials 10, 15, 24, 81, and 98. This option overrides values assigned by *DEFINE_STOCHASTIC_VARIATION.

Card 1	1	2	3	4	5	6	7	8
Variable	IDE	VARSY	VARF	VARRO	VARE			
Type	I	F	F	F	F			
Default	none	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
IDE	Element ID
VARSY	The yield stress and its hardening function are scaled by 1.0 + VARSY.
VARF	The failure criterion is scaled by 1.0 + VARF.
VARRO	The density is scaled by 1.0 + VARRO. This is intended to be used with topology optimization. This option is not available for shell elements.
VARE	The elastic moduli are scaled by 1.0 + VARE. This is intended to be used with topology optimization.

*DEFINE_STOCHASTIC_VARIATION

Purpose: Define the stochastic variation in the yield stress and damage/failure models for material models with the STOCHASTIC keyword option. This keyword works with materials 10, 15, 24, 81, 98, 251, 280, and the shell version of material 123. It is also possible to vary the failure strain in GISSMO with this keyword.

Card Summary:

Card 1. This card is required

ID_SV	PID	PID_TYP	ICOR	VAR_S	VAR_F	IRNG	IEXTRA

Card 1.1. Include this card if IEXTRA = 1.

ISHLFLG							

Card 2a. Include this card if VAR_S = 0, 1, or 2.

R1	R2	R3					

Card 2b. Include this card if VAR_S = 3 or 4.

LCID							

Card 3a. Include this card if VAR_F = 0, 1, or 2.

R1	R2	R3					

Card 3b. Include this card if VAR_S = 3 or 4.

LCID							

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	ID_SV	PID	PID_TYP	ICOR	VAR_S	VAR_F	IRNG	IEXTRA
Type	I	I	I	I	I	I	I	I
Default	none	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
ID_SV	Stochastic variation ID. A unique ID number must be used.
PID	Part, part set, or material ID depending on PID_TYP
PID_TYP	Flag for PID type. If PID and PID_TYP are both 0, then the properties defined here apply to all shell and solid parts using materials with the STOCHASTIC option. EQ.0: PID is a part (*PART) ID. EQ.1: PID is a part set (*SET_PART) ID. EQ.2: PID is a material ID (MID defined on *MAT_...).
ICOR	Correlation between the yield stress and failure strain scaling. EQ.0: Perfect correlation. EQ.1: No correlation. The yield stress and failure strain are independently scaled.
VAR_S	Variation type for scaling the yield stress (or tensile strength for material 280). EQ.0: The scale factor is 1.0 everywhere. EQ.1: The scale factor is a random number in the uniform random distribution in the interval defined by R1 and R2. EQ.2: The scale factor is a random number obeying the Gaussian distribution defined by R1, R2, and R3. EQ.3: The scale factor is defined by the probability distribution function defined by curve LCID. EQ.4: The scale factor is defined by the cumulative distribution function defined by curve LCID.
VAR_F	Variation type for scaling failure strain. EQ.0: The scale factor is 1.0 everywhere. EQ.1: The scale factor is a random number in the uniform random distribution in the interval defined by R1 and R2. EQ.2: The scale factor is a random number obeying the Gaussian distribution defined by R1, R2, and R3. EQ.3: The scale factor is defined by the probability distribution function defined by curve LCID. EQ.4: The scale factor is defined by the cumulative distribution

VARIABLE	DESCRIPTION
	function defined by curve LCID.
IRNG	Flag for random number generation. EQ.0: Use a deterministic (pseudo-) random number generator. The same input always leads to the same distribution. EQ.1: Use a non-deterministic (true) random number generator. With the same input, a different distribution is achieved in each run.
IEXTRA	Flag to include Card 1.1: EQ.0: Do not include Card 1.1. EQ.1: Include Card 1.1.

Shell Layer Flag Card. Include this card if IEXTRA = 1

Card 1.1	1	2	3	4	5	6	7	8
Variable	ISHLFLG							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ISHLFLG	Flag to use the same random number for all in-plane integration points in a shell layer, such as when using shell type 16: EQ.0: Do not use the same number. EQ.1: Use the same number.

Yield Stress Card for Built-in Distribution. Card 2 for VAR_S set to 0, 1, or 2.

Card 2a	1	2	3	4	5	6	7	8
Variable	R1	R2	R3					
Type	F	F	F					

Yield Stress Card for Load Curve. Card 2 for VAR_S set to 3 or 4.

Card 2b	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							

Failure Strain Card for Built-in Distribution. Card 3 for VAR_F set to 0, 1, or 2.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1	R2	R3					
Type	F	F	F					

Failure Strain Card for Load Curve. Card 3 for VAR_F set to 3 or 4.

Card 3b	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							

VARIABLE	DESCRIPTION
R1, R2, R3	Real values to define the stochastic distribution. See below.
LCID	Curve ID defining the stochastic distribution. See below.

Remarks:

Each integration point x_g in the parts specified by PID is assigned the random scale factors R_S and R_F. These scale factors are applied to the values calculated by the material model for the yield stress and failure strain:

$$\sigma_\mathrm{y} = R_\mathrm{S}(x_g)\sigma_\mathrm{y}(\bar{\varepsilon}^\mathrm{p}, \dots)$$
$$\bar{\varepsilon}^\mathrm{p}_\mathrm{FAIL} = R_\mathrm{F}(x_g)\bar{\varepsilon}^\mathrm{p}_\mathrm{FAIL}(\dot{\varepsilon}, \bar{\varepsilon}^\mathrm{p}, \dots)$$

The scale factors vary spatially over the model according to the chosen statistical distributions defined in this section and are independent of time. The scale factors may be

completely correlated or uncorrelated, with the default being completely correlated, because the failure strain is generally reduced as the yield stress increases. The scale factors R_S and R_F may be stored as extra history variables as follows:

Material Model	History Variable # for R_S	History Variable # for R_F
10	5	6
15	7	8
24	6	7
81	6	7
98	7	8
123 (shells only)	6	7
251	9	not applicable
280	13	not applicable
GISSMO	not applicable	ND + 20

You are responsible for defining the distributions to be physically meaningful and restricted to a realistic range. For example, because neither the yield stress nor the failure strain may be negative, the minimum values of the distributions must always be greater than zero.

The probability that a particular value, R, will occur defines the *probability distribution function*, $P(R)$. Since a value must be chosen from the distribution, the integral from the minimum to the maximum value of R of the probability distribution function must be 1.0,

$$\int_{R_{\text{MIN}}}^{R_{\text{MAX}}} P(R)\mathrm{d}R = 1.0 \ .$$

Another way to characterize a distribution is the *cumulative distribution function*, $C(R)$, which defines the probability that a value will lie between R_{MIN} and R,

$$C(R) = \int_{R_{\text{MIN}}}^{R} P\left(\hat{R}\right)\mathrm{d}\hat{R}.$$

By definition $C(R_{\text{MIN}}) = 0$ and $C(R_{\text{MAX}}) = 1$. An inverse cumulative probability function D gives the number for a cumulative probability of $C(R)$,

$$D\big(C(R)\big) = R.$$

A random variable satisfying the probability distribution function $P(R)$ can be generated from a sequence of uniformly distributed numbers, $\hat{R}_I$, for $I = 1, \ldots, N$, using the inverse cumulative distribution function D as

$$R_I = D(\hat{R}_I).$$

The scale factors for the yield stress and the failure strain may be generated using the same value of $\hat{R}_I$ for both (ICOR = 0) or by using independent values for each one (ICOR = 1). If the same values are used, there is perfect correlation, and the failure strain scale factor becomes an implicit value of the yield stress scale factor.

VAR = 0. No Scaling.

The corresponding yield stress or failure strain is not scaled.

VAR = 1. Scaling from Uniform Distribution

A uniform distribution is specified by setting VAR = 1. The input variable R1 is interpreted as R_{MIN} and R2 as R_{MAX}. If R1 = R2, the yield stress or failure strain will be scaled by R1.

When using the uniform random distribution, the probability of a particular value is given by

$$P(R) = \frac{1}{R_{\text{MAX}} - R_{\text{MIN}}}$$

and the cumulative probability function is given by

$$C(R) = \frac{R - R_{\text{MIN}}}{R_{\text{MAX}} - R_{\text{MIN}}}.$$

VAR = 2. Gaussian Distribution

The Gaussian distribution, VAR = 2, is smoothly varying with a peak at μ and 68% of the values occurring within the interval of one standard deviation σ, $[\mu - \sigma, \mu + \sigma]$. The input parameter R1 is interpreted as the mean, μ, while R2 is interpreted as the standard deviation, σ. There is a finite probability that the values of R will be outside of the range that are physically meaningful in the scaling process, and R3, which is interpreted as δ, restricts the range of R to $[\mu - \delta, \mu + \delta]$. The resulting truncated Gaussian distribution is rescaled such that,

$$D(\mu + \delta) = 1 \, .$$

VAR = 3 or 4. Distribution from a Load Curve

The user may directly specify the probability distribution function or the cumulative probability distribution function with *DEFINE_CURVE by setting VAR = 3 or VAR = 4, respectively, and then specifying the required curve ID on the next data card.

Stochastic variations may be used simultaneously with the heat affected zone (HAZ) options (see *DEFINE_HAZ_PROPERTIES). The effects of the scale factors from stochastic variation and HAZ options are multiplied together to scale the yield stress and failure strain:

$$\sigma_{\mathrm{y}} = R_{\mathrm{S}}(x_g) R_{\mathrm{S}}^{\mathrm{HAZ}} \sigma_{\mathrm{y}}(\bar{\varepsilon}^{\mathrm{p}}, \dots)$$
$$\bar{\varepsilon}^{\mathrm{p}}_{\mathrm{FAIL}} = R_{\mathrm{F}}(x_g) R_{\mathrm{F}}^{\mathrm{HAZ}} \bar{\varepsilon}^{p}_{\mathrm{FAIL}}(\dot{\varepsilon}, \bar{\varepsilon}^{\mathrm{p}}, \dots).$$

*DEFINE_STOCHASTIC_VARIATION_PROPERTIES

Purpose: Define the stochastic variation in predefined material model history variables. The predefined history variables to which the stochastic variation can be applied are listed in the documentation of the material model. It is an extension of *DEFINE_STOCHASTIC_VARIATION to more material models and additional material properties.

This keyword currently works with materials 10, 15, 24, 81, 98, and 213 as well the shell version of material 123.

Card Summary:

Card 1. This card is required.

ID_SV	MTYPE	PID	PID_TYP	IRNG	NUMV	NUM_BEG	

Card 2a. This card is included if and only if VARTYP = 0, 1, or 2 (built-in distributions). For each predefined history variable, include a combination of Cards 2a and 2b in the order of the history variables.

VARTYP	CORLGRP	R1	R2	R3			

Card 2b. This card is included if and only if VARTYP = 3 or 4 (distributions defined by curves). For each predefined history variable, include a combination of Cards 2a and 2b in the order of the history variables.

VARTYP	CORLGRP	LCID					

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	ID_SV	MTYPE	PID	PID_TYP	IRNG	NUMV	NUM_BEG	
Type	I	I	I	I	I	I	I	
Default	none	none	0	0	0	0	0	

VARIABLE	DESCRIPTION
ID_SV	Stochastic variation ID. A unique ID number must be used.

VARIABLE	DESCRIPTION
MTYPE	Material type. The available types are 10, 15, 24, 81, 98, and 213. This variation only works for this material.
PID	*PART ID or *SET_PART ID.
PID_TYP	Flag for PID type. If PID and PID_TYP are both 0, then the properties defined here apply to all shell and solid parts using the materials defined in the *OPTION* string. EQ.0: PID is a *PART ID. EQ.1: PID is a *SET_PART ID.
IRNG	Flag for random number generation. EQ.0: Use deterministic (pseudo-) random number generator. The same input always leads to the same distribution. EQ.1: Use non-deterministic (true) random number generator. With the same input, a different distribution is achieved in each run.
NUMV	Number of variations for a user material
NUM_BEG	The location of the first variation in the history variables for a user material. The remaining variations are added sequentially.

Card for Built-in Distributions. Card 2 for VARTYP set to 0, 1, or 2.

Card 2a	1	2	3	4	5	6	7	8
Variable	VARTYP	CORLGRP	R1	R2	R3			
Type	I	I	F	F	F			

Card for Distributions Defined by a Curve. Card 2 for VARTYP set to 3 or 4.

Card 2b	1	2	3	4	5	6	7	8
Variable	VARTYP	CORLGRP	LCID					
Type	I	I	I					

VARIABLE	DESCRIPTION
VARTYP	Variation type for scaling the material property: EQ.0: The scale factor is 1.0 everywhere. EQ.1: The scale factor is a random number in the uniform random distribution in the interval defined by R1 and R2. EQ.2: The scale factor is a random number obeying the Gaussian distribution defined by R1, R2, and R3. EQ.3: The scale factor is defined by the probability distribution function defined by curve LCID. EQ.4: The scale factor is defined by the cumulative distribution function defined by curve LCID.
CORLGRP	Correlation group number. If CORLGRP is 0, then the random number for the distribution is uncorrelated with all the other distributions. The same random number is used for evaluating all the distributions having the same positive integer value for CORLGRP.
R1, R2, R3	Real values to define the stochastic distribution. See Remark 6.
LCID	Curve ID defining the stochastic distribution. See Remark 6.

Remarks:

1. **Physically Meaningful Distributions.** The user is responsible for defining the distributions so that they are physically meaningful and are restricted to a realistic range. A certain history variable may not be negative, for example, so the minimum values of the distributions must always be greater than zero.

2. **Compatibility.** This implementation is compatible with *DEFINE_STOCHASTIC_VARIATION for materials 10, 15, 24, 81, 98, and the shell version of material 123 in that it scales the yield stress and the failure strain. This implementation also supports stochastic variations with user materials but does not support GISSMO. The results of the two implementations will not be identical because the generality of the correlation groups results in the random number generator being called in a different order.

3. **Correlation Group Number.** The correlation group number allows the user to selectively correlate different history variables for a given material model. For a material model that is scaling five history variables with the group numbers being 0, 0, 1, 2, 2, each of the first two variables will be uncorrelated with the other

four, the third variable is also uncorrelated with the rest, while the last two are perfectly correlated with each other but not with any others.

4. **Number of Variations.** If more variations are defined in the input than are permitted by the material model, the excess will be ignored. If fewer variations are defined, the unspecified variations will be set to 1.0.

5. **Distribution Functions.** The probability that a particular value R will occur defines the *probability distribution function*, $P(R)$. Since a value must be chosen from the distribution, the integral from the minimum to the maximum value of R of the probability distribution function must be 1.0,

$$\int_{R_{\text{MIN}}}^{R_{\text{MAX}}} P(R)dR = 1.0 \ .$$

Another way to characterize a distribution is the *cumulative distribution function*, $C(R)$, which defines the probability that a value will lie between R_{MIN} and R,

$$C(R) = \int_{R_{\text{MIN}}}^{R} P(\hat{R})d\hat{R} \ .$$

By definition $C(R_{\text{MIN}}) = 0$ and $C(R_{\text{MAX}}) = 1$. An inverse cumulative probability function D gives the number for a cumulative probability of $C(R)$,

$$D\big(C(R)\big) = R \ .$$

A random variable satisfying the probability distribution function $P(R)$ can be generated from a sequence of uniformly distributed numbers, $\hat{R}_I$, for $I = 1, \ldots, N$, using the inverse cumulative distribution function D as

$$R_I = D(\hat{R}_I).$$

6. **VARTYP.** The applied distribution and input fields for Card 2 depends on VARTYP. Descriptions of each distribution follows.

   a) *No Scaling (VARTYP = 0).* The distribution value is 1.0.

   b) *Scaling from Uniform Distribution (VARTYP = 1).* A uniform distribution is specified by setting VAR = 1. The input variable R1 is interpreted as R_{MIN} and R2 as R_{MAX}. If R1 = R2, then the material property will be scaled by R1.

      When using the uniform random distribution, the probability of a particular value is given by

$$P(R) = \frac{1}{R_{\text{MAX}} - R_{\text{MIN}}} \ ,$$

      and the cumulative probability function is given by

$$C(R) = \frac{R - R_{\text{MIN}}}{R_{\text{MAX}} - R_{\text{MIN}}} \ .$$

c) *Gaussian Distribution (VARTYP = 2).* The Gaussian distribution, VAR = 2, is smoothly varying with a peak at μ and 63 percent of the values occurring within the interval of one standard deviation σ, $[\mu - \sigma, \mu + \sigma]$. The input parameter R1 is interpreted as the mean, μ,, while R2 is interpreted as the standard deviation, σ. There is a finite probability that the values of R will be outside of the range that are physically meaningful in the scaling process, and R3 which is interpreted as δ restricts the range of R to $[\mu - \delta, \mu + \delta]$. The resulting truncated Gaussian distribution is rescaled such that,

$$D(\mu + \delta) = 1 \ .$$

d) *Distribution from a Load Curve (VARTYP = 3 or 4).* The user may directly specify the probability distribution function or the cumulative probability distribution function with *DEFINE_CURVE by setting VAR = 3 or VAR = 4, respectively, and then specifying the required curve ID.

7. **Heat Affected Zone.** Stochastic variations may be used simultaneously with the heat affected zone (HAZ) options in LS-DYNA (see *DEFINE_HAZ_PROPERTIES). The effect of the scale factors from stochastic variation and HAZ options are multiplied together to scale the predefined history variables.

*DEFINE_TABLE

Purpose: To interpolate from point data a continuously indexed family of nonintersecting curves. The family of curves, $\mathcal{F}$, consists of x-y curves, $f_s(x)$, indexed by a parameter, s.

$$\mathcal{F} = \{f_s(x) | \forall s \in [s_{\min}, s_{\max}]\}.$$

The interpolation is built up by sampling functions in $\mathcal{F}$ at discrete parameter values, s_i,

$$f_{s_i}(x) \in \mathcal{F}.$$

The points, s_i, are input to LS-DYNA on the data cards for the *DEFINE_TABLE keyword. LS-DYNA requires that they be ordered from least to greatest. The curves, $f_{s_i}(x)$, must be defined as lists of (x, y) pairs in a collection of *DEFINE_CURVE sections that directly follow the *DEFINE_TABLE section. Each *DEFINE_CURVE section is paired to its corresponding s_i value by list position (and not load curve ID, for that see *DEFINE_TABLE_2D).

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	SFA	OFFA					
Type	I	F	F					
Default	none	1.	0.					

Points Cards. Place one point per card. The values must be in ascending order. Input is terminated when a "*DEFINE_CURVE" keyword card is found.

Card 2	1	2	3	4	5	6	7	8
Variable	VALUE							
Type	F							
Default	0.0							

Include one ***DEFINE_CURVE** input section here for each point defined above. The i^{th} *DEFINE_CURVE card contains the curve at the i^{th} *DEFINE_TABLE value.

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and Load curves may not share common ID's. LS-DYNA allows load curve ID's and table ID's to be used interchangeably.
SFA	Scale factor for VALUE.
OFFA	Offset for VALUE, see explanation below.
VALUE	Load curve will be defined corresponding to this value, e.g., this value could be a strain rate, see purpose above.

Motivation:

This capability was implemented with strain-rate dependent stress-strain relations in mind. To define such a function, the first step is to tabulate stress-strain curves at known strain-rate values. Then, the list of strain-rates is written in ascending order to the data cards following *DEFINE_TABLE. Following *DEFINE_TABLE, the tabulated stress-strain curves must be input to LS-DYNA as a set of *DEFINE_CURVE sections ordered so that the i^{th} curve corresponds to the i^{th} strain-rate point. This section is structured as:

***DEFINE_TABLE**
strain-rate point 1
strain-rate point 2
⋮
strain-rate point n
***DEFINE_CURVE**
[stress-strain curve at strain-rate 1]
***DEFINE_CURVE**
[stress-strain curve at strain-rate 2]
⋮
***DEFINE_CURVE**
[stress-strain curve at strain-rate n]

Details, Features and Limitations:

1. All the curves in a table must start from the same abscissa value and end at the same abscissa value. This limitation is necessary to avoid slow indirect addressing in the inner loops used in the constitutive model stress evaluation. Curves must not intersect except at the origin and end points.

2. Each curve may have unique spacing and an arbitrary number of points in its definition.

3. All the curves in a table must share the same value of LCINT.

4. In most applications, curves can only be extrapolated in one direction, that is, to the right of the last data point. An example would be curves representing effective stress vs. effective plastic strain. For cases when extrapolation is only to the right, the curves comprising a table are allowed to intersect *only* at their starting point but the curves *and their extrapolations* must not intersect elsewhere.

   For other applications in which the curves are extrapolated in both directions, the curves and their extrapolations are not allowed to intersect except at the origin (0,0). An example would be curves representing force vs. change in gage length where negative values are compressive and positive values are tensile.

5. Load curve IDs defined for the table may be referenced elsewhere in the input.

6. No keyword commands may come between *DEFINE_TABLE and the *DEFINE_CURVE commands that feed the table. The set of *DEFINE_CURVE commands must not be interrupted by any other keyword. This coupling between *DEFINE_TABLE and subsequent *DEFINE_CURVE commands is an exception to the general order-independence of the keyword format.

7. VALUE is scaled in the same manner as in *DEFINE_CURVE, i.e.,

$$\text{Scaled value} = \text{SFA} \times (\text{Defined value} + \text{OFFA}).$$

8. Unless stated otherwise in the description of a keyword command that references a table, there is no extrapolation beyond the range of VALUEs defined for the table. For example, if the table VALUE represents strain rate and the calculated strain rate exceeds the last/highest VALUE given by the table, the stress-strain curve corresponding to the last/highest table VALUE will be used.

*DEFINE_TABLE_2D

Purpose: Define a table. Unlike the *DEFINE_TABLE keyword above, a curve ID is specified for each value defined in the table. This allows the same curve ID to be referenced by multiple tables, and the curves may be defined anywhere in the input file. Other than these differences from *DEFINE_TABLE, the general rules given in the remarks of *DEFINE_TABLE still apply.

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	SFA	OFFA					
Type	I	F	F					
Default	none	1.	0.					

Points Cards. Place one point per card. The values must be in ascending order. Input is terminated when the next keyword card is found.

Card 2	1	2	3	4
Variable	VALUE	CURVE ID		
Type	F	I		
Default	0.0	none		

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and Load curves may not share common ID's. LS-DYNA allows load curve ID's and table ID's to be used interchangeably.
SFA	Scale factor for VALUE.
OFFA	Offset for VALUE; see remarks for *DEFINE_TABLE.
VALUE	Load curve will be defined corresponding to this value. The value could be, for example, a strain rate.
CURVEID	Load curve ID. See Remark 1.

Remarks:

1. **Curve IDs.** Though generally of no concern to the user, curve CURVEID is automatically duplicated during initialization and the duplicate curve is automatically assigned a unique curve ID. The generated curve IDs used by the table are revealed in d3hsp. It is generally only necessary to know the generated curve IDs when interpreting warning messages about those curves.

*DEFINE_TABLE_3D

Purpose: Define a three-dimensional table. For each value defined below, a table ID is specified. For example, in a thermally dependent material model, the value given below could correspond to temperature for a table ID defining effective stress versus strain curves for a set of strain rate values. Each table ID can be referenced by multiple three-dimensional tables, and the tables may be defined anywhere in the input.

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	SFA	OFFA					
Type	I	F	F					
Default	none	1.	0.					

Points Cards. Place one point per card. The values must be in ascending order. Input is terminated when the next keyword card is found.

Card 2	1	2	3	4
Variable	VALUE	TABLE ID		
Type	F	I		
Default	0.0	none		

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and load curves may not share common ID's. LS-DYNA allows load curve ID's and table ID's to be used interchangeably.
SFA	Scale factor for VALUE.
OFFA	Offset for VALUE, see explanation below.
VALUE	Load curve will be defined corresponding to this value, e.g., this value could be a strain rate for example.
TABLEID	Table ID.

Remarks:

1. **Scaling.** VALUE is scaled in the same manner as in *DEFINE_CURVE, that is,

$$\text{Scaled value} = \text{SFA} \times (\text{Defined value} + \text{OFFA}).$$

2. **Extrapolation.** Unless stated otherwise in the description of a keyword command that references a table, there is no extrapolation beyond the range of VALUEs defined for the table. For example, if the table VALUE represents strain rate and the calculated strain rate exceeds the last/highest VALUE given by the table, the stress-strain curve corresponding to the last/highest table VALUE will be used

*DEFINE_TABLE_*{X}*D

Available options for *X* are 4, 5, 6, 7, 8, or 9.

Purpose: Define an *X*-dimensional table. For each value defined below, a table ID of one lower dimension is specified. This keyword works in the same way as *DEFINE_TABLE_3D, but with a higher dimensionality (current maximum would be X = 9).

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	SFA	OFFA					
Type	I	F	F					
Default	none	1.	0.					

Points Cards. Place one point per card. The values must be in ascending order. Input is terminated when the next keyword card is found.

Card 2	1	2	3	4
Variable	VALUE	TABLE ID		
Type	F	I		
Default	0.0	none		

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and load curves may not share common IDs. LS-DYNA allows load curve IDs and table IDs to be used interchangeably.
SFA	Scale factor for VALUE
OFFA	Offset for VALUE; see explanation below.
VALUE	Value which corresponds to a $X-1$ dimension with ID TABLEID
TABLEID	Table ID of table with one dimension smaller

Remarks:

1. **Scaling.** VALUE is scaled in the same manner as in *DEFINE_CURVE, that is,

$$\text{Scaled value} = \text{SFA} \times (\text{Defined value} + \text{OFFA}).$$

2. **Extrapolation.** Unless stated otherwise in the description of a keyword command that references a table, there is no extrapolation beyond the range of values defined for the table. For example, if the table VALUE represents strain rate and the calculated strain rate exceeds the last/highest VALUE given by the table, the stress-strain curve corresponding to the last/highest table VALUE will be used.

3. **Example.** The following input shows the possible definition of stress-strain curves for *MAT_251 with 3 points each for two values of history variable #7 (80 and 200), two values for history variable #6 (20 and 400), and two strain rates (0 and 100) using a TABLE_4D:

```
*DEFINE_TABLE_4D
     10000
              80.0                1000
             200.0                2000
$~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*DEFINE_TABLE_3D
      1000
              20.0                1100
             400.0                1200
$---------------------------------------------------
*DEFINE_TABLE
      1100
               0.0
             100.0
*DEFINE_CURVE
      1101
               0.0               360.0
               0.3               570.0
               1.0               780.0
*DEFINE_CURVE
      1102
               0.0               470.0
               0.3               680.0
               1.0               860.0
$---------------------------------------------------
*DEFINE_TABLE
      1200
               0.0
             100.0
*DEFINE_CURVE
      1201
               0.0               180.0
               0.3               285.0
               1.0               390.0
*DEFINE_CURVE
      1202
               0.0               235.0
               0.3               340.0
               1.0               430.0
$---------------------------------------------------
```

```
$~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*DEFINE_TABLE_3D
      2000
              20.0                2100
             400.0                2200
$---------------------------------------------------
*DEFINE_TABLE
      2100
               0.0
             100.0
*DEFINE_CURVE
      2101
               0.0               540.0
               0.3               855.0
               1.0              1170.0
*DEFINE_CURVE
      2102
               0.0               705.0
               0.3              1020.0
               1.0              1290.0
$---------------------------------------------------
*DEFINE_TABLE
      2200
               0.0
             100.0
*DEFINE_CURVE
      2201
               0.0               270.0
               0.3               427.5
               1.0               585.0
*DEFINE_CURVE
      2202
               0.0               352.5
               0.3               510.0
               1.0               645.0
$---------------------------------------------------
```

*DEFINE_TABLE_COMPACT

Purpose: Define a multi-dimensional table, meaning a functional dependence on several variables. Each ordinate value (for example, stress) can depend on several abscissa values (such as strain, strain rate, history variables, …).

Card Summary:

Card 1. This card is required.

TBID	NVAR	LCINT	MATHIS	INEXETC	ISCALE		

Card 2. This card is included if MATHIS = 1.

	HIS1	HIS2	HIS3	HIS4	HIS5	HIS6	HIS7

Card 2.1. This card is included if MATHIS = 1 and NVAR > 7.

	HIS8	HIS9					

Card 3. This card is included if INEXETC = 1.

	IXE1	IXE2	IXE3	IXE4	IXE5	IXE6	IXE7

Card 3.1. This card is included if INEXETC = 1 and NVAR > 7.

	IXE8	IX89					

Card 4. This card is included if ISCALE = 1.

SF0	SFA1	SFA2	SFA3	SFA4	SFA5	SFA6	SFA7

Card 4.1. This card is included if ISCALE = 1 and NVAR > 7.

	SFA8	SFA9					

Card 5. Include one of this card for each ordinate value if NVAR ≤ 7 or a set of this card and Card 4.1 for each ordinate value if NVAR > 7. This input ends with the next keyword (“*”) card.

01	A1.1	A1.2	A1.3	A1.4	A1.5	A1.6	A1.7

Card 5.1. Include this in a set of this card with Card 4 if NVAR > 7 for each ordinate value. This input ends with the next keyword (“*”) card,

	A1.8	A1.9					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	NVAR	LCINT	MATHIS	INEXETC	ISCALE		
Type	I	I	I	I	I	I		
Default	none	none	0	0	0	0		

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and load curves may not share common IDs. LS-DYNA allows load curve IDs and table IDs to be used interchangeably.
NVAR	Number of variables (dimension of the table). Current maximum is 9.
LCINT	Number of discretization points. EQ.0: Value of LCINT from *CONTROL_SOLUTION will be used.
MATHIS	Material history flag. Option to identify the abscissa variables as a specified history variable number(s) (see Remarks 3 and 6). Additional Card 2 (and possibly Card 2.1) is read if this option is active. EQ.0: Off EQ.1: On
INEXETC	Extra curve settings flag. Option to assign settings about curve discretization, extrapolation and interpolation for each abscissa variable. Additional Card 3 (and possibly Card 3.1) is read if this option is active. EQ.0: Off EQ.1: On
ISCALE	Variable scaling flag. Option to scale ordinate and abscissa values. Additional Card 4 (and possibly Card 4.1) is read if this option is active. EQ.0: Off

VARIABLE	DESCRIPTION
	EQ.1: On

Material History Card. Additional card for MATHIS = 1 only.

Card 2	1	2	3	4	5	6	7	8
Variable		HIS1	HIS2	HIS3	HIS4	HIS5	HIS6	HIS7
Type		I	I	I	I	I	I	I

Optional Material History Card. Include up to 2 additional values if MATHIS = 1 and NVAR > 7.

Card 2.1	1	2	3	4	5	6	7	8
Variable		HIS8	HIS9					
Type		I	I					

VARIABLE	DESCRIPTION
HIS1, HIS2, …	History variable numbers which indicate that variable X is HISX. For instance, setting HIS2 = 6 indicates that variable 2 is history variable 6 for the material. A value of 0 indicates that the abscissa variable is not a history variable. See Remarks 3 and 6.

Extra Settings Card. Additional card for INEXETC = 1 only.

Card 3	1	2	3	4	5	6	7	8
Variable		IXE1	IXE2	IXE3	IXE4	IXE5	IXE6	IXE7
Type		I	I	I	I	I	I	I

Optional Extra Settings Card. Include up to 2 additional values if INEXETC = 1 and NVAR > 7.

Card 3.1	1	2	3	4	5	6	7	8
Variable		IXE8	IXE9					
Type		I	I					

VARIABLE	DESCRIPTION
IXE1, IXE2, …	Extra settings assigned to abscissa values. See Remark 3. IXE is interpreted digit-wise, namely as, $\text{IXE} = [MLK] = K + 10 \times L + 100 \times M$. with the following interpretations: K.EQ.0: Discretized curve using LCINT is used. K.EQ.1: Original curve as defined in this keyword is used. L.EQ.0: Extrapolation at lower and upper end is used. L.EQ.1: No extrapolation is used, meaning the first and last values are kept constant. M.EQ.0: Linear interpolation between data points is used. M.EQ.1: Logarithmic interpolation between data points is used.

Variable Scaling Card. Additional card for ISCALE = 1 only.

Card 4	1	2	3	4	5	6	7	8
Variable	SF0	SFA1	SFA2	SFA3	SFA4	SFA5	SFA6	SFA7
Type	F	F	F	F	F	F	F	F

Optional Variable Scaling Card. Include up to 2 additional values if ISCALE = 1 and NVAR > 7.

Card 4.1	1	2	3	4	5	6	7	8
Variable		SFA8	SFA9					
Type		F	F					

VARIABLE	DESCRIPTION
SFO	Scale factor for ordinate value. Default set to 1.0.
SFA1, SFA2, …	Scale factors for abscissa values. Defaults set to 1.0.

Point Cards. Place one ordinate value with NVAR abscissa values per card. If NVAR > 7, include sets of Card 5 and Card 5.1 for each ordinate value. The values must be placed in a special ascending order; see Remarks. Input ends with the next keyword ("*") card.

Card 5	1	2	3	4	5	6	7	8
Variable	01	A1.1	A1.2	A1.3	A1.4	A1.5	A1.6	A1.7
Type	F	F	F	F	F	F	F	F

Optional Point Cards. Include this card if NVAR > 7 in a set with Card 5 for the same ordinate value to define up to 2 additional abscissa values.

Card 5.1	1	2	3	4	5	6	7	8
Variable		A1.8	A1.9					
Type		F	F					

VARIABLE	DESCRIPTION
O1, O2, …	Ordinate (function) values. See Remarks 2, 4 and 5.
A1.X, A2.X, …	Abscissa values of variable X. See Remarks 2, 4 and 5.

Remarks:

1. **Internal Conversion.** The input from this keyword is internally converted into a *DEFINE_TABLE_{X}D with dimension X = NVAR. Corresponding sub-tables and curves will automatically be created, and all rules/properties described in *DEFINE_CURVE / *DEFINE_TABLE hold.

2. **Input Syntax.** The abscissa values of the first variable (second column) should increase first, while values in the following columns are kept constant. Then, values of variable 2 should increase next while the remaining variables to the right are kept constant. Next, the same for variable 3, and so on.

3. **Support Material Models for Material History and Extra Settings.** The options MATHIS and INEXETC are currently supported for *MAT_024 with VP = 3 and *MAT_ADD_DAMAGE_DIEM (or *MAT_ADD_EROSION with IDAM < 0).

4. **Basic Example.** The following input shows the possible definition of stress-strain curves for *MAT_251 with 3 points each for two different values of history variable #7 (80 and 200), two values for history variable #6 (20 and 400), and two strain rates (0 and 100), as an alternative to *DEFINE_TABLE_4D (see remarks there for comparison):

```
*DEFINE_TABLE_COMPACT
$     tbid    numvar     lcint
     10000         4
$    value      var1      var2      var3      var4
$      sig       eps      rate     his#6     his#7
     360.0       0.0       0.0      20.0      80.0
     570.0       0.3       0.0      20.0      80.0
     780.0       1.0       0.0      20.0      80.0
     470.0       0.0     100.0      20.0      80.0
     680.0       0.3     100.0      20.0      80.0
     860.0       1.0     100.0      20.0      80.0
     180.0       0.0       0.0     400.0      80.0
     285.0       0.3       0.0     400.0      80.0
     390.0       1.0       0.0     400.0      80.0
     235.0       0.0     100.0     400.0      80.0
     340.0       0.3     100.0     400.0      80.0
     430.0       1.0     100.0     400.0      80.0
     540.0       0.0       0.0      20.0     200.0
     855.0       0.3       0.0      20.0     200.0
    1170.0       1.0       0.0      20.0     200.0
     705.0       0.0     100.0      20.0     200.0
    1020.0       0.3     100.0      20.0     200.0
    1290.0       1.0     100.0      20.0     200.0
     270.0       0.0       0.0     400.0     200.0
     427.5       0.3       0.0     400.0     200.0
     585.0       1.0       0.0     400.0     200.0
     352.5       0.0     100.0     400.0     200.0
     510.0       0.3     100.0     400.0     200.0
```

5. **Empty Columns.** It is possible to leave individual columns without defined values. For instance, if history variable #6 were to be left out in the above example, the table would reduce to:

```
*DEFINE_TABLE_COMPACT
$     tbid    numvar     lcint
     10000         4
$    value      var1      var2      var3      var4
```

```
$       sig       eps      rate     his#6     his#7
     360.0       0.0       0.0                80.0
     570.0       0.3       0.0                80.0
     780.0       1.0       0.0                80.0
     470.0       0.0     100.0                80.0
     680.0       0.3     100.0                80.0
     860.0       1.0     100.0                80.0
     540.0       0.0       0.0               200.0
     855.0       0.3       0.0               200.0
    1170.0       1.0       0.0               200.0
     705.0       0.0     100.0               200.0
    1020.0       0.3     100.0               200.0
    1290.0       1.0     100.0               200.0
```

6. **MATHIS Example.** MATHIS = 1 causes the reading of one to two additional cards depending on NVAR which indicate which history variable a given abscissa variable is. Usually there is a fixed order to the history variables used in this table, but MATHIS = 1 gives you more freedom when assigning the variables to the table. In the following example, variable 2 is history variable #1, variable 3 is history variable #6, variable 4 is history variable #7, variable 5 is history variable #8, and variable 6 is history variable #9.

```
*DEFINE_TABLE_COMPACT
$     tbid    numvar     lcint    mathis   inexetc
     10000         6       201         1         1
$              his1      his2      his3      his4      his5      his6
                  0         1         6         7         8         9
$              ixe1      ixe2      ixe3      ixe4      ixe5      ixe6
                  0       111        10        10        10        10
$    value      var1      var2      var3      var4      var5      var6
$      sig       eps     his#1     his#6     his#7     his#8     his#9
     360.0       0.0       0.0      20.0      80.0       0.0       1.0
     570.0       0.3       0.0      20.0      80.0       0.0       1.0
     780.0       1.0       0.0      20.0      80.0       0.0       1.0
                                      ⋮
```

*DEFINE_TABLE_MATRIX

Purpose: This is an alternative input format for *DEFINE_TABLE that allows for reading data from an unformatted text file containing a matrix with data separated by comma delimiters. The purpose is to use data saved directly from excel sheets without having to convert it to keyword syntax.

Card 1	1	2	3	4	5	6	7	8
Variable	TBID	FILENAME						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	NROW	NCOL	SROW	SCOL	SVAL	OROW	OCOL	OVAL
Type	I	I	F	F	F	F	F	F
Default	none	none	1.	1.	1.	0.	0.	0.

VARIABLE	DESCRIPTION
TBID	Table ID. Tables and Load curves may not share common ID's. LS-DYNA allows load curve ID's and table ID's to be used interchangeably.
FILENAME	Name of file containing table data (stored as a matrix).
NROW	Number of rows in the matrix, same as number of rows in the file FILENAME. A negative value of NROW switches the interpretation of rows and columns in the read matrix, see Remarks 1 and 2.
NCOL	Number of columns in the matrix, same as number of data entries per row in the file FILENAME
SROW	Scale factor for row data, see Remark 2.
SCOL	Scale factor for column data, see Remark 2.
SVAL	Scale factor for matrix values, see Remark 2.

VARIABLE	DESCRIPTION
OROW	Offset for row data, see Remark 2.
OCOL	Offset for column data, see Remark 2.
OVAL	Offset for matrix values, see Remark 2.

Remarks:

1. **Example.** The use of this keyword allows for inputting a table in form of a matrix from a file, exemplified here by a 4 × 5 matrix.

	C1	C2	C3	C4
R1	V11	V12	V13	V14
R2	V21	V22	V23	V24
R3	V31	V32	V33	V34
⋮	⋮	⋮	⋮	⋮

The unformatted file representing this matrix would contain the following data

```
,C1,C2,C3,C4
R1,V11,V12,V13,V14
R2,V21,V22,V23,V24
R3,V31,V32,V33,V34
```

Note that the first entry in the matrix is a dummy and delimited by an initial comma in the file. The keyword card for this matrix is: (TBID = 1000 and the filename is file.txt)

```
*DEFINE_TABLE_MATRIX
1000,file.txt
4,5
```

This is equivalent to using:

```
*DEFINE_TABLE
1000
C1
C2
C3
C4
*DEFINE_CURVE
1001
R1,V11
R2,V21
R3,V31
*DEFINE_CURVE
```

```
1002
R1,V12
R2,V22
R3,V32
*DEFINE_CURVE
1003
R1,V13
R2,V23
R3,V33
*DEFINE_CURVE
1004
R1,V14
R2,V24
R3,V34
```

2. **Scaling, Offsets, and Transposition.** All entries in the matrix can be scaled and offset following the convention for other tables and curves:

$$\text{Scaled Value} = \text{S[ROW/COL]} \times (\text{Value} + \text{O[ROW/COL]})$$

Finally, the matrix can be transposed by setting NROW to a negative value. In the example above (Remark 1) this would mean that

```
*DEFINE_TABLE_MATRIX
1000,file.txt
-4,5
```

is equivalent to using:

```
*DEFINE_TABLE
1000
R1
R2
R3
*DEFINE_CURVE
1001
C1,V11
C2,V12
C3,V13
C4,V14
*DEFINE_CURVE
1002
C1,V21
C2,V22
C3,V23
C4,V24
*DEFINE_CURVE
1003
C1,V31
C2,V32
```

```
C3,V33
C4,V34
```

In this case, any scaling applies to the matrix entries before transposing the data, i.e., for row entries the scaled value is

$$\text{Scaled Value} = \text{SROW} \times (R + \text{OROW}),$$

and for column entries

$$\text{Scaled Value} = \text{SCOL} \times (C + \text{OCOL})$$

regardless the sign of NROW.

*DEFINE_TARGET_BOUNDARY

Purpose: This keyword is used to define the desired boundary of a formed part. This boundary provides the criteria used during blank size development. The definitions associated with this keyword are used, exclusively, by the *INTERFACE_BLANKSIZE_DEVELOPMENT feature.

Point Cards. Include one card for each point in the curve. These points are interpolated to form a closed curve. This input is terminated with *END.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	X		Y		Z					
Type	E16.0		E16.0		E16.0					
Default	none		none		none					

VARIABLE	DESCRIPTION
X, Y, Z	Location coordinates of a target node.

Remarks:

1. **INTERFACE_BLANKSIZE_DEVELOPMENT.** The keyword file specified on the second data card for the *INTERFACE_BLANKSIZE_DEVELOPMENT keyword must contain a *DEFINE_TARGET_BOUNDARY keyword.

2. **Example.** A partial keyword input is shown below. Note that the input is in a 3E16.0 FORTRAN format. Also note that the first and last curve points coincide.

```
*KEYWORD
*DEFINE_TARGET_BOUNDARY
    -1.83355e+02    -5.94068e+02    -1.58639e+02
    -1.80736e+02    -5.94071e+02    -1.58196e+02
    -1.78126e+02    -5.94098e+02    -1.57813e+02
    -1.75546e+02    -5.94096e+02    -1.57433e+02
    -1.72888e+02    -5.94117e+02    -1.57026e+02
           ⋮               ⋮               ⋮
    -1.83355e+02    -5.94068e+02    -1.58639e+02
*END
```

3. **IGES to Keyword Format.** Typically, these boundary nodes are obtained from the boundary curves for a final (trimmed) piece or from a draw blank edge at a certain distance outside of the draw beads. LS-PrePost 4.1 can generate the

points for this keyword from IGES data. To use IGES data select *Curve → Convert→ Method (To Keyword) → Select *DEFINE_TARGET_BOUNDARY*; pick the curves; and then select *To Key*. To output in keyword format, choose *File → Save as → Save Keyword As*, and select "Output Version" as "V971_R7".

Revisions:

This feature is available in LS-DYNA R6 Revision 74560 and later releases.

*DEFINE_TRACER_PARTICLES_2D

Purpose: Define tracer particles that follow the deformation of a material. This is useful for visualizing the deformation of a part that is being adapted in a metal forming operation. Nodes used as tracer particles should only be used for visualization and not associated with anything in the model that may alter the response of the model, e.g., they should not be used in any elements except those with null materials.

Card 1	1	2	3	4	5	6	7	8
Variable	NSET	PSET						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
NSET	The node set ID for the nodes used as tracer particles.
PSET	Optional part set ID. If this part set is specified, only tracer particles in these parts are updated and the others are stationary. If this part set is not specified, all tracer particles are updated.

*DEFINE_TRANSFORMATION

Purpose: Define a transformation for the *INCLUDE_TRANSFORM keyword option. The *DEFINE_TRANSFORMATION command must be defined before the *INCLUDE_-TRANSFORM command can be used.

Card 1	1	2	3	4	5	6	7	8
Variable	TRANID							
Type	I							
Default	none							

Transformation Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	OPTION	A1	A2	A3	A4	A5	A6	A7
Type	A	F	F	F	F	F	F	F

Matrix Card 1. Cards 3 and 4 are only included when OPTION = MATRIX.

Card 3	1	2	3	4	5	6	7	8
Variable	M11	M12	M13	M14	M21	M22	M23	M24
Type	F	F	F	F	F	F	F	F

Matrix Card 2. Cards 3 and 4 are only included when OPTION = MATRIX.

Card 4	1	2	3	4	5	6	7	8
Variable	M31	M32	M33	M34	M41	M42	M43	M44
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
TRANID	Transform ID
OPTION	For the available options, see the table below.
A1-A7	Parameters. See Table 17-63 below.
M*ij*	Parameters for OPTION = MATRIX. See Table 17-63 below.

Table 17-63. List of allowed transformations.

OPTION	PARAMETERS	FUNCTION
MIRROR	a1, a2, a3, a4, a5, a6, a7	Reflect about a mirror plane defined to contain the point (a1, a2, a3) having its normal pointing from point (a1, a2, a3) toward (a4, a5, a6). Setting a7 = 1 reflects the coordinate system as well; that is, the mirrored coordinate system uses the left-hand rule to determine the local z-axis Setting a7 = 2 changes the element connectivity or segment node numbering sequence to mirror the element/segment normal. The initial stress/strain will be reordered as well to match the mirrored element coordinate system. Currently, only linear shell, solid, and thick shell elements are supported for a7 = 2.
MATRIX	M*ij* in Cards 3 & 4	Direct input of a 4x4 space transformation matrix. A node with coordinates (x, y, z) will be transformed according to $(x, y, z, 1)M$. In most cases M14 = M24 = M34 = 0 and M44 = 1.0. The matrix can be equivalent to other options. For example, TRANSL is equivalent to M41 = a1, M42 = a2 and M43 = a3, and SCALE is the same as M11 = a1, M22 = a2 and M33 = a3.
POINT	a1,a2,a3,a4	Define a point with ID, a1, with the initial coordinates a2, a3, and a4.
POS6P	a1, a2, a3, a4, a5, a6	Positioning by 6 points. Affine transformation (rotation and translation, no scaling) given by three start points a1, a2, and a3 and three target points a4, a5, and a6. The six POINTs must be defined before they are referenced. Only one POS6P option is

OPTION	PARAMETERS	FUNCTION
		permitted within a *DEFINE_TRANSFORMATION definition.
POS6N	a1, a2, a3, a4, a5, a6	Positioning by 6 nodes. Affine transformation (rotation and translation, no scaling) given by three start nodes a1, a2, and a3 and three target nodes a4, a5, and a6. The six nodes must be defined before they are referenced. Only one POS6N option is permitted within a *DEFINE_TRANSFORMATION definition.
ROTATE	a1, a2, a3, a4, a5, a6, a7	Rotate through an angle (deg), a7, about a line with direction cosines a1, a2, and a3 passing through the point with coordinates a4, a5, and a6. If a4 through a7 are zero, then a1 and a2 are the IDs of two POINTs, and a3 defines the rotation angle. The axis of rotation is defined by a vector going from point with ID a1 to point with ID a2.
ROTATE3NA	a1, a2, a3, a4	Rotate through an angle (deg), a4. The axis of rotation is defined by a vector going from the node with ID a1 to the node with ID a2 and passing through the node with ID a3 (a3 could be the same as a1 or a2). The three nodes must be defined before they are referenced. Only one ROTATE3NA option is permitted within a *DEFINE_TRANSFORMATION definition.
SCALE	a1, a2, a3	Scale the global x, y, and z coordinates of a point by a1, a2, and a3, respectively. If zero, a default of unity is set.
TRANSL	a1, a2, a3	Translate the x, y, and z coordinates of a point by a1, a2, and a3, respectively.
TRANSL2ND	a1, a2, a3	Translate by distance a3. The direction is defined by a vector going from node with ID a1 to node with ID a2. The two nodes must be defined before they are referenced. If a3 is set to zero, then the distance between nodes a1 and a2 is used directly for the translation. Only one TRANSL2ND option is permitted

OPTION	PARAMETERS	FUNCTION
		within a *DEFINE_TRANSFORMATION definition.

Each option represents a transformation matrix. When more than one option is used, the transformation matrix defined by MIRROR, SCALE, ROTATE, ROTATE3NA, TRANSL, or TRANSL2ND is applied to the previously defined existing matrix to form the new global transformation matrix. Therefore the ordering of the MIRROR, SCALE, ROTATE, ROTATE3NA, TRANSL, and TRANSL2ND commands is important. We generally recommend first scaling, then rotating, and finally translating the model. When used together with *INCLUDE_TRANSFORM, we advise defining the six nodes referred to by POS6N and the *DEFINE_TRANSFORMATION in the same file.

The POINT option in ROTATE provides a means of defining rotations about axes defined by the previous transformations. The coordinates of the two POINTs are transformed by all the transformations up to the transformation where they are referenced. The POINTs must be defined before they are referenced, and their identification numbers are local to each *DEFINE_TRANSFORMATION. The coordinates of a POINT are transformed using all the transformations before it is referenced, not just the transformations between its definition and its reference. To put it another way, while the ordering of the transformations is important, the ordering between the POINTs and the transformations is not important.

> **NOTE:** When *DEFINE_TRANSFORMATION is called from within the target of an *INCLUDE_TRANSFORM keyword, the result will involve stacked transformations.

In the following example, the *DEFINE_TRANSFORMATION command is used 3 times to input the same dummy model and position it as follows:

1. Transformation ID 1000 imports the dummy model (dummy.k) and rotates it 45 degrees about z-axis at the point (0.0,0.0,0.0). Transformation ID 1001 performs the same transformation using the POINT option.

2. Transformation ID 2000 imports the same dummy model (dummy.k) and translates 1000 units in the x direction.

3. Transformation ID 3000 imports the same dummy model (dummy.k) and translates 2000 units in the x direction. For each *DEFINE_TRANSFORMATION, the commands TRANSL, SCALE, and ROTATE are available. The transformations are applied in the order in which they are defined in the file. For instance, transformation ID 1000 in this example would translate, scale and then rotate the model. *INCLUDE_TRANSFORM uses a transformation ID defined by a *DE-

FINE_TRANSFORMATION command to import a model and perform the associated transformations. It also allows the user upon importing the model to apply offsets to the various entity IDs and perform unit conversion of the imported model.

```
*KEYWORD
*DEFINE_TRANSFORMATION
      1000
$ option &       dx&       dy&       dz&
TRANSL        0000.0       0.0       0.0
$ option &       dx&       dy&       dz&
SCALE           1.00       1.0       1.0
$ option &       dx&       dy&       dz&       px&       py&       pz&
angle&
ROTATE          0.00       0.0       1.0      0.00      0.00       0.0
45.00
*DEFINE_TRANSFORMATION
      1001
POINT              1       0.0       0.0       0.0
POINT              2       0.0       0.0       1.0
ROTATE             1         2      45.0
*DEFINE_TRANSFORMATION
      2000
$ option &       dx&       dy&       dz&
TRANSL        1000.0       0.0       0.0
*DEFINE_TRANSFORMATION
$ tranid &
     3000
$ option &       dx&       dy&       dz&

TRANSL        2000.0       0.0       0.0
*INCLUDE_TRANSFORM
dummy.k
$idnoff  &   ideoff&   idpoff& idmoff  &  idsoff &  iddoff&  iddoff &
         0         0         0         0         0        0         0
$  idroff&   ilctmf&
         0         0
$  fctmas&   fcttim&   fctlen&  fcttem &   incout&
    1.0000    1.0000      1.00       1.0         1
$ tranid &
      1000
*INCLUDE_TRANSFORM
dummy.k
$idnoff  &   ideoff&   idpoff& idmoff  &  idsoff &  iddoff&  iddoff &
   1000000   1000000   1000000   1000000   1000000  1000000   1000000
$  idroff&   ilctmf&
   1000000   1000000
$  fctmas&   fcttim&   fctlen&  fcttem &   incout&
    1.0000    1.0000      1.00       1.0         1
$ tranid &
      2000
*INCLUDE_TRANSFORM
dummy.k
$idnoff  &   ideoff&   idpoff& idmoff  &  idsoff &  iddoff&  iddoff &
   2000000   2000000   2000000   2000000   2000000  2000000   2000000
$  idroff&   ilctmf&
   2000000   2000000
$  fctmas&   fcttim&   fctlen&  fcttem &   incout&
    1.0000    1.0000      1.00       1.0         1
$ tranid &
      3000
*END
```

*DEFINE_TRIM_SEED_POINT_COORDINATES

Purpose: The keyword facilitates blank trimming during a stamping line die simulation. It allows for the trimming process and inputs to be defined independent of the previous process simulation results and is applicable to shell, solid and laminate.

Card 1	1	2	3	4	5	6	7	8
Variable	NSEED	X1	Y1	Z1	X2	Y2	Z2	
Type	I	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
NSEED	Number of seed points. Maximum value of "2" is allowed.
X1, Y1, Z1	Location coordinates of seed point 1.
X2, Y2, Z2	Location coordinates of seed point 2.

Remarks:

1. **Associated Keywords.** This keyword is used in conjunction with keywords *ELEMENT_TRIM and *DEFINE_CURVE_TRIM, where variables NSEED1 and NSEED2 should be left as blank. For detailed usage, refer to **Seed Node Definition** section in *DEFINE_CURVE_TRIM.

2. **Number of Seed Points.** Variable NSEED is set to the number of seed points desired. For example, in a double attached drawn panel trimming, NSEED would equal to 2.

3. **Example Input Deck.** A partial keyword inputs for a single drawn panel trimming is listed below.

```
*INCLUDE_TRIM
drawn.dynain
*ELEMENT_TRIM         1
*DEFINE_CURVE_TRIM_NEW
$#    TCID    TCTYPE      TFLG      TDIR     TCTOL      TOLN     NSEED
         1         2                11     0.250
trimlines.iges
*DEFINE_TRIM_SEED_POINT_COORDINATES
$    NSEED        X1        Y1         Z1        X2        Y2        Z2
         1    -271.4     89.13   1125.679
*DEFINE_VECTOR
```

```
11,0.0,0.0,0.0,0.0,0.0,10.0
```

Typically, seed point coordinates can be selected from the stationary post in punch home position.

4. **Revision Information.** This feature is available in LS-DYNA R4 Revision 53048 and later releases.

*DEFINE_TWINBUILDER_COUPLING

Purpose: Define a node/node set that interacts with an Ansys Twin Builder™ ROM model. Each coupling requires its own keyword card instantiation. Note that this feature is *not* supported on Windows.

TwinSDK runtime libraries are required for the coupling. Add the path to libTwinRuntimeSDK.so and its accompanying dependencies to the LD_LIBRARY_PATH environment.

Card 1	1	2	3	4	5	6	7	8
Variable	TWINPATH							
Type	A80							

Card 2	1	2	3	4	5	6	7	8
Variable	ROMNAME							
Type	A80							

Card 3	1	2	3	4	5	6	7	8
Variable	LOGNAME							
Type	A80							

Card 4	1	2	3	4	5	6	7	8
Variable	NODE	TYPE	IOPT	CID	FRCFRQ	FSWITCH		DEBUG
Type	I	I	I	I	I	I		I
Default	none	0	0	0	10	0		0

Card 5	1	2	3	4	5	6	7	8
Variable	FNOPT	ALPHA						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
TWINPATH	Full path, including the filename, to the Twin Builder .twin file. See Remark 1.
ROMNAME	Twin Builder ROM name within the .twin file
LOGNAME	Twin Builder log file name. This log file is generated in the current working directory.
NODE	Coupled node/node set. See Remark 2.
TYPE	Type for NODE: EQ.0: Node ID EQ.1: Node set ID
IOPT	Option for exchanging data: EQ.0: Output nodal translational and rotational coordinates and receive nodal translational and rotational forces from Twin Builder (default). EQ.1: Output nodal translational and rotational displacements and receive nodal translational and rotational forces from Twin Builder. EQ.2: Output nodal translational coordinates and receive nodal translational forces from Twin Builder. EQ.3: Output nodal translational displacements and receive nodal translational forces from Twin Builder.
CID	Reference coordinate system needed to transform the data from the global system to the Twin Builder local system. If zero or blank, the global coordinate system is used.

VARIABLE	DESCRIPTION
FRCFRQ	Number of cycles between Twin Builder force updates for the coupling interface
FSWITCH	Flag to flip the sign of the Twin Builder predicted forces if needed for consistency with ROM training: EQ.0: Add ROM-predicted force to the internal force of the node/nodes (default). EQ.1: Add ROM-predicted force to the internal force in the negative direction for the node/nodes.
DEBUG	Debug level for identifying issues in the ROM or coupling: EQ.0: No debug output (default). EQ.1: Debug level I. This level outputs tbModalInputs.csv, tbModalOutputs.csv, tbdisp.bin, and tb.forc.bin. It also prints extra data to stdout.
FNOPT	Option for function smoothing. The predicted force is applied at the interface every cycle according to the following (see Remark 3): EQ.0: No smoothing (default) EQ.1: Power law EQ.2: C1 transition
ALPHA	Parameter for the smoothing function. See Remark 3.

Remarks:

1. **Input file.** TWINPATH is limited to 236 characters spread over up to three 80-character lines. When 2 or 3 lines are needed to specify the filename or pathname, end the preceding line with " ␣+" (space followed by a plus sign) to signal that a continuation line follows. Note that the " ␣+" combination is, itself, part of the 80-character line; hence, the maximum number of allowed characters is 78 + 78 + 80 = 236

2. **Coupled node/node set.** NODE is a node or node set ID for the node(s) belonging to the substructure (finite element model) at the ROM interface. The substructure interface node(s) must be shared, or coincident, with the ROM interface node(s). For a node set, the order of the nodes in the node set must exactly match the order of the ROM interface nodes used during the ROM's training, as specified in the "ids" field of the Twin Builder settings.json file.

3. **Function smoothing.** By default, the Twin Builder predicted force is applied every simulation cycle. The ROM updates the predicted force every FRCFRQ cycles. The change in force during the transition may be abrupt, which can lead to instabilities. To mitigate this, a smoothing function can be used to gradually transition from the current predicted force, F_1, to the newly predicted one, F_0. The applied interface force at cycle n is given by:

$$F(n) = F_0 + S(n) \times (F_1 - F_0)$$

Here, $S(n)$ is a smoothing factor. When no smoothing is applied, $S(n) = 1$. For power law smoothing, $S(n)$ is defined as:

$$S(n) = \left(\frac{C(n)}{\text{FRCFRQ}}\right)^{\text{ALPHA}}$$

For C1 transition smoothing, $S(n)$ is defined as:

$$S(n) = 0.5 + \frac{0.5 \times \tan^{-1}\left(\text{ALPHA} \times \left(\frac{C(n)}{\text{FRCFRQ}} - 0.5\right)\right)}{\tan^{-1}(\text{ALPHA} \times 0.5)}$$

In the above equations, $C(n)$ is the position of the current cycle within the FRCFRQ update period, taking values from 1 to FRCFRQ.

*DEFINE_VECTOR

Purpose: Define a vector by defining the coordinates of two points.

Card	1	2	3	4	5	6	7	8
Variable	VID	XT	YT	ZT	XH	YH	ZH	CID
Type	I	F	F	F	F	F	F	I
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	global

VARIABLE	DESCRIPTION
VID	Vector ID
XT	*X*-coordinate of tail of vector
YT	*Y*-coordinate of tail of vector
ZT	*Z*-coordinate of tail of vector
XH	*X*-coordinate of head of vector
YH	*Y*-coordinate of head of vector
ZH	*Z*-coordinate of head of vector
CID	Coordinate system ID to define vector in local coordinate system. All coordinates, XT, YT, ZT, XH, YH, and ZH are in respect to CID. EQ.0: global (default).

Remarks:

1. **Numerical Inaccuracies.** The coordinates should differ by a certain margin to avoid numerical inaccuracies.

*DEFINE_VECTOR_NODES

Purpose: Define a vector with two nodal points.

Card	1	2	3	4	5	6	7	8
Variable	VID	NODET	NODEH					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
VID	Vector ID
NODET	Nodal point to define tail of vector
NODEH	Nodal point to define head of vector

EXAMPLES

The following examples demonstrate the input for these options:

*DEFINE_BOX

*DEFINE_COORDINATE_NODES,

*DEFINE_COORDINATE_SYSTEM,

*DEFINE_COORDINATE_VECTOR

*DEFINE_CURVE

*DEFINE_SD_ORIENTATION

*DEFINE_VECTOR commands.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_BOX
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define box number eight which encloses a volume defined by two corner
$  points: (-20.0, -39.0, 0.0) and (20.0, 39.0, 51.0).  As an example, this
$  box can be used as an input for the *INITIAL_VELOCITY keyword in which
$  all nodes within this box are given a specific initial velocity.
$
*DEFINE_BOX
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    boxid       xmm       xmx       ymn       ymx       zmn       zmx
         8     -20.0      20.0     -39.0      39.0       0.0      51.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_COORDINATE_NODES
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define local coordinate system number 5 using three nodes: 10, 11 and 20.
$  Nodes 10 and 11 define the local x-direction.  Nodes 10 and 20 define
$  the local x-y plane.
$
$  For example, this coordinate system (or any coordinate system defined using
$  a *DEFINE_COORDINATE_option keyword) can be used to define the local
$  coordinate system of a joint, which is required in order to define joint
$  stiffness using the *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED keyword.
$
*DEFINE_COORDINATE_NODES
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      cid        n1        n2        n3
         5        10        11        20
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$



$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_COORDINATE_SYSTEM
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define local coordinate system number 3 using three points.  The origin of
$  local coordinate system is at (35.0, 0.0, 0.0).  The x-direction is defined
$  from the local origin to (35.0, 5.0, 0.0).  The x-y plane is defined using
$  the vector from the local origin to (20.0, 0.0, 20.0) along with the local
$  x-direction definition.
$
*DEFINE_COORDINATE_SYSTEM
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      cid        Xo        Yo        Zo        Xl        Yl        Zl
         3      35.0       0.0       0.0      35.0       5.0       0.0
$
$       Xp        Yp        Zp
      20.0       0.0      20.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_COORDINATE_VECTOR
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define local coordinate system number 4 using two vectors.
$    Vector 1 is defined from (0.0, 0.0, 0.0) to (1.0, 1.0, 0.0)
$    Vector 2 is defined from (0.0, 0.0, 0.0) to (1.0, 1.0, 1.0)
$  See the corresponding keyword command for a description.
$
*DEFINE_COORDINATE_VECTOR
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      cid        Xx        Yx        Zx        Xv        Yv        Zv
         4       1.0       1.0       0.0       1.0       1.0       1.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$



$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_CURVE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define curve number 517.  This particular curve is used to define the
$  force-deflection properties of a spring defined by a *MAT_SPRING_INELASTIC
$  keyword.  The abscissa value is offset 25.0 as a means of modeling a gap
$  at the front of the spring.  This type of spring would be a compression
$  only spring.
$
*DEFINE_CURVE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     lcid      sidr      scla      sclo      offa      offo
       517                                     25.0
$
$           abscissa            ordinate
                0.0                 0.0
               80.0                58.0
               95.0                35.0
              150.0                44.5
              350.0                45.5
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *DEFINE_SD_ORIENTATION
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A discrete spring is defined with two nodes in 3-D space.  However, it is
$  desired to have the force of that spring to act only in the z-direction.
$  The following definition makes this happen.  Additionally, vid = 7
$  must be specified in the *ELEMENT_DISCRETE keyword for this spring.
$
*DEFINE_SD_ORIENTATION
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      vid       iop        xt        yt        zt      nid1      nid2
         7         0       0.0       0.0       1.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$



$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *DEFINE_VECTOR
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define vector number 5 from (0,0,0) to (0,1,1).  As an example, this vector
$  can be used to define the direction of the prescribed velocity of a node
$  using the *BOUNDARY_PRESCRIBED_MOTION_NODE keyword.
$
*DEFINE_VECTOR
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      vid        xt        yt        zt        xh        yh        zh
         3       0.0       0.0       0.0       0.0       1.0       1.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*DEFORMABLE_TO_RIGID

The cards in this section are defined in alphabetical order and are as follows:

*DEFORMABLE_TO_RIGID

*DEFORMABLE_TO_RIGID_AUTOMATIC

*DEFORMABLE_TO_RIGID_INERTIA

If one of these cards is defined, then any deformable part defined in the model may be switched to rigid during the calculation. Parts that are defined as rigid (*MAT_RIGID) in the input are permanently rigid and cannot be changed to deformable.

Deformable parts may be switched to rigid at the start of the calculation by specifying them on the *DEFORMABLE_TO_RIGID card.

Part switching may be specified on a restart (see RESTART section of this manual) or it may be performed automatically by use of the *DEFORMABLE_TO_RIGID_AUTOMATIC cards.

The *DEFORMABLE_TO_RIGID_INERTIA cards allow inertial properties to be defined for deformable parts that are to be swapped to rigid at a later stage.

It is not possible to perform part material switching on a restart if it was not flagged in the initial analysis. The reason for this is that extra memory needs to be set up internally to allow the switching to take place. If part switching is to take place on a restart, but no parts are to be switched at the start of the calculation, no inertia properties for switching and no automatic switching sets are to be defined, then just define one *DEFORMABLE_TO_RIGID card without further input.

*DEFORMABLE_TO_RIGID

Purpose: Define parts for which the material will be switched to rigid at the start of the calculation.

WARNING: This keyword is only supported for switching between finite elements and rigid bodies.

WARNING: *PART_MODES is not supported when using this keyword.

Card	1	2	3	4	5	6	7	8
Variable	PID	LRB	PTYPE					
Type	I	I	A					
Default	none	0	PART					

VARIABLE	DESCRIPTION
PID	Part ID for the part that will be switched to a rigid material; also see *PART.
LRB	Part ID of the lead rigid body to which the part is merged. If zero, the part becomes either an independent or lead rigid body.
PTYPE	Type of PID: EQ."PART": PID is a part ID. EQ."PSET": PID is a part set ID. All parts included in part set PID will be switched to rigid at the start of the calculation.

*DEFORMABLE_TO_RIGID_AUTOMATIC

Purpose: Define a set of parts to be switched to rigid or to deformable at some stage during the calculation.

> **WARNING:** This keyword is only supported for switching between finite elements and rigid bodies.

> **WARNING:** *PART_MODES is not supported when using this keyword.

Card Summary:

Card 1. This card is required

SWSET	CODE	TIME1	TIME2	TIME3	ENTNO	RELSW	PAIRED

Card 2. This card is required.

NRBF	NCSF	RWF	DTMAX	D2R	R2D	OFFSET	

Card 3. Include D2R of this card.

PID	LRB	PTYPE					

Card 4. Include R2D of this card.

PID	PTYPE						

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	SWSET	CODE	TIME1	TIME2	TIME3	ENTNO	RELSW	PAIRED
Type	I	I	F	F	F	I	I	I
Default	none	0	0.	10^{20}	0.	0.	0	0
Remark		1				1, 2		3

VARIABLE	DESCRIPTION
SWSET	Set number for this automatic switch set. Must be unique.
CODE	Activation switch code. Defines the test to activate the automatic material switch of the part: EQ.0: Switch takes place at time 1. EQ.1: Switch takes place between time 1 and time 2 if rigid wall force (specified below) is zero. EQ.2: Switch takes place between time 1 and time 2 if contact surface force (specified below) is zero. EQ.3: Switch takes place between time 1 and time 2 if rigid wall force (specified below) is nonzero. EQ.4: Switch takes place between time 1 and time 2 if contact surface force (specified below) is nonzero. EQ.5: Switch is controlled by *SENSOR_CONTROL with TYPE = DEF2RIG; see *SENSOR_CONTROL. When CODE = 5, inputs of column 3 to column 8, TIME1 to PAIRED, are ignored.
TIME1	Switch will not take place before this time
TIME2	Switch will not take place after this time: EQ.0.0: Time 2 set to 10^{20}
TIME3	Delay period. After this part switch has taken place, another automatic switch will not take place for the duration of the delay period. If set to zero, a part switch may take place immediately after this switch.
ENTNO	Rigid wall / contact surface number for switch codes 1, 2, 3, 4
RELSW	Related switch set. The related switch set is another automatic switch set paired to this one so the switches can be activated more than once. EQ.0: No related switch set
PAIRED	Specify how the related switch sets are paired (if there are paired switches): EQ.0: SWSET is not paired to another switch set.

VARIABLE	DESCRIPTION
	EQ.1: SWSET is paired with switch set RELSW and is the first switch set to be activated.
	EQ.-1: SWSET is paired with switch set RELSW and is the second switch to be activated.

Card 2	1	2	3	4	5	6	7	8
Variable	NRBF	NCSF	RWF	DTMAX	D2R	R2D	OFFSET	
Type	I	I	I	F	I	I	F	
Default	0	0	0	0.	0	0	0.	
Remark	4	4	4					

VARIABLE	DESCRIPTION
NRBF	Nodal rigid body flag. For all values of NRBF, nodal rigid bodies defined using *CONSTRAINED_NODAL_RIGID_BODY and *CONSTRAINED_GENERALIZED_WELD_OPTION, which share any nodes with a rigid body created by deformable-to-rigid switching, are merged with the latter to form a single rigid body. Other actions dependent upon the value of NRBF are: EQ.0: No further action EQ.1: Delete all remaining nodal rigid bodies, that is, delete those nodal rigid bodies that do not share any nodes with a rigid body created by deformable-to-rigid switching. EQ.2: Activate nodal rigid bodies.
NCSF	Nodal constraint set flag. If nodal constraint / spot weld definitions are active in the deformable bodies that are switched to rigid, then the definitions should be deleted to avoid instabilities. EQ.0: No change EQ.1: Delete EQ.2: Activate
RWF	Flag to delete or activate rigid walls:

VARIABLE	DESCRIPTION
	EQ.0: No change EQ.1: Delete EQ.2: Activate
DTMAX	Maximum permitted time step size after switch
D2R	Number of deformable parts to be switched to rigid plus number of rigid parts for which new merged (lead/constrained) rigid body combinations will be defined. EQ.0: No parts defined
R2D	Number of rigid parts to be switched to deformable. EQ.0: No parts defined
OFFSET	Optional contact thickness for switch to deformable. For contact, its value should be set to a value greater than the contact thickness offsets to ensure the switching occurs prior to impact. This option applies if and only if CODE is set to 3 or 4. For CODE = 3 all rigid wall options are implemented. For CODE = 4, the implementation works for the contact type CONTACT_AUTOMATIC when the options: ONE_WAY_SURFACE_TO_SURFACE, NODES_TO_-SURFACE, and SURFACE_TO_SURFACE are invoked.

Deformable to Rigid Cards. D2R additional cards with one for each part.

Card 3	1	2	3	4	5	6	7	8
Variable	PID	LRB	PTYPE					
Type	I	I	A					
Default	none	0	PART					

VARIABLE	DESCRIPTION
PID	Part ID of the part which is switched to a rigid material. When PID is merged to another rigid body by the LRB field, this part is allowed to be rigid before the switch.

VARIABLE	DESCRIPTION
LRB	Part ID of the lead rigid body to which part PID is merged. If zero, part PID becomes either an independent or lead rigid body.
PTYPE	Type of PID: EQ.PART: PID is a part ID. EQ.PSET: PID is a part set ID.

Rigid to Deformable Cards. R2D additional cards with one for each part.

Card 4	1	2	3	4	5	6	7	8
Variable	PID	PTYPE						
Type	I	A						
Default	none	PART						

VARIABLE	DESCRIPTION
PID	Part ID of the part which is switched to a deformable material
PTYPE	Type of PID: EQ.PART: PID is a part ID. EQ.PSET: PID is a part set ID.

Remarks:

1. **Allowed Contact Types.** Only surface to surface and node to surface contacts can be used to activate an automatic part switch.

2. **Rigid Wall Numbering.** Rigid wall numbers are the order in which they are defined in the deck. The first rigid wall and the first contact surface encountered in the input deck will have an entity number of 1. The contact surface ID is that as defined on the *CONTACT_…_ID card.

3. **Paired Switches.** Switch sets may be paired together to allow a pair of switches to be activated more than once. Each pair of switches should use consistent values for CODE, meaning 1 & 3 or 2 & 4. Within each pair of switches, the related switch, RELSW, should be set to the ID of the other switch in the pair. The first

switch (PAIRED = 1) will be activated before the second switch (PAIRED = -1). Pairing allows multiple switches to take place as for example when contact is made and lost several times during an analysis.

```
$  Define a pair of related switches that will be activated by force/no force on
$  Contact 3.  To start with, switch set 20 will be activated (PAIRED = 1),
$  swapping the PARTS to RIGID. When the contact force is non-zero, switch set
$  10 will be activated, swapping the PARTS to DEFORMABLE.  If the contact force
$  returns to zero, switch set 20 will be activated again, making the PARTS
$  RIGID.
$
*DEFORMABLE_TO_RIGID_AUTOMATIC
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    swset      code    time 1    time 2    time 3     entno     relsw    paired
        20         2                                       3        10         1
$     nrbf      ncsf       rwf     dtmax       D2R       R2D
                                                 1
*DEFORMABLE_TO_RIGID_AUTOMATIC
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    swset      code    time 1    time 2    time 3     entno     relsw    paired
        10         4                                       3        20        -1
$     nrbf      ncsf       rwf     dtmax       D2R       R2D
                                                   1
```

4. **Delete Switch.** If the delete switch is activated, *all* corresponding constraints are deactivated regardless of their relationship to a switched part. By default, constraints which are directly associated with a switched part are deactivated/activated as necessary.

*DEFORMABLE_TO_RIGID_INERTIA

Purpose: Inertial properties can be defined for the new rigid bodies that are created when the deformable parts are switched. These can only be defined in the initial input if they are needed in a later restart. Unless these properties are defined, LS-DYNA will recompute the new rigid body properties from the finite element mesh. The latter requires an accurate mesh description. *When rigid bodies are merged to a lead rigid body, the inertial properties specified for the lead rigid body apply to all members of the merged set.*

Card 1	1	2	3	4	5	6	7	8
Variable	PID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	TM				
Type	F	F	F	F				

Card 3	1	2	3	4	5	6	7	8
Variable	IXX	IXY	IXZ	IYY	IYZ	IZZ		
Type	F	F	F	F	F	F		
Default	none	0.0	0.0	none	0.0	none		

VARIABLE	DESCRIPTION
PID	Part ID, see *PART.
XC	x-coordinate of center of mass
YC	y-coordinate of center of mass

VARIABLE	DESCRIPTION
ZC	z-coordinate of center of mass
TM	Translational mass
IXX	I_{xx} (the xx component of inertia tensor)
IXY	I_{yy}
IXZ	I_{xz}
IYY	I_{yy}
IYZ	I_{yz}
IZZ	I_{zz}

*ELEMENT

The element cards in this section are defined in alphabetical order:

*ELEMENT_BEAM

*ELEMENT_BEAM_PULLEY

*ELEMENT_BEAM_SOURCE

*ELEMENT_BEARING

*ELEMENT_BLANKING

*ELEMENT_DIRECT_MATRIX_INPUT

*ELEMENT_DISCRETE

*ELEMENT_DISCRETE_SPHERE

*ELEMENT_GENERALIZED_SHELL

*ELEMENT_GENERALIZED_SOLID

*ELEMENT_INERTIA

*ELEMENT_INTERPOLATION_SHELL

*ELEMENT_INTERPOLATION_SOLID

*ELEMENT_LANCING

*ELEMENT_MASS

*ELEMENT_MASS_MATRIX

*ELEMENT_MASS_PART

*ELEMENT_PLOTEL

*ELEMENT_SEATBELT

*ELEMENT_SEATBELT_ACCELEROMETER

*ELEMENT_SEATBELT_PRETENSIONER

*ELEMENT_SEATBELT_RETRACTOR

*ELEMENT_SEATBELT_SENSOR

*ELEMENT_SEATBELT_SLIPRING

*ELEMENT_SHELL

*ELEMENT_SHELL_NURBS_PATCH

*ELEMENT_SHELL_SOURCE_SINK

*ELEMENT_SOLID

*ELEMENT_SOLID_NURBS_PATCH

*ELEMENT_SOLID_PERI

*ELEMENT_SOLID_REFERENCE

*ELEMENT_SPH

*ELEMENT_TRIM

*ELEMENT_TSHELL

The ordering of the element cards in the input file is completely arbitrary. An arbitrary number of element blocks can be defined preceded by a keyword control card.

***ELEMENT_BEAM**_{OPTION}_{OPTION}

Available options include:

<BLANK>

THICKNESS, SCALAR, SCALR or SECTION

PID

OFFSET

ORIENTATION

WARPAGE

ELBOW (*beta*)

Purpose: Define two node elements including 3D beams, trusses, 2D axisymmetric shells, and 2D plane strain beam elements. The type of the element and its formulation is specified through the part ID (see *PART) and the section ID (see *SECTION_BEAM).

Two alternative methods are available for defining the cross sectional property data. The THICKNESS and SECTION options are provided for the user to override the *SECTION_BEAM data which is taken as the default if the THICKNESS or SECTION option is not used. The SECTION option applies only to resultant beams (ELFORM = 2 on *SECTION_BEAM). End release conditions are imposed using constraint equations, and caution must be used with this option as discussed in Remark 2.

The SCALAR and SCALR options apply only to material model type 146, *MAT_1DOF_GENERALIZED_SPRING.

The PID option is used by the type 9 spot weld element only and is ignored for all other beam types. When the PID option is active, you provide two part IDs that are tied by the spot weld element in an additional card. If the PID option is inactive for the type 9 element, the nodal points of the spot weld are located to the two nearest reference surface segments. In either case, *CONTACT_SPOTWELD must be defined with the spot weld beam part as the tracked surface and the shell parts (including parts PID1 and PID2) as the reference surface. The surface of each segment should project to the other and in the most typical case the node defining the weld, assuming only one node is used, should lie in the middle; however, this is not a requirement. Note that with the spot weld elements only one node is needed to define the weld, and two nodes are optional.

The OFFSET option is not available for discrete beam elements or spotweld beam elements. Neither the OFFSET option nor the ORIENTATION option is available for truss elements or 2D beam element forms 7 or 8.

The ELBOW option is a 3-node beam element with quadratic interpolation that is tailored for the piping industry. It includes 12 degrees of freedom, including 6 ovalization degrees of freedom for describing the ovalization, per node. That is a total of 36 DOFs for each element. An internal pressure can also be given that tries to stiffen the pipe. The pressure, if activated accordingly, can also contribute to the elongation of the pipe. The control node must be given, but it is only used for initially straight elbow elements. For curved elements the curvature center is used as the control node. See *SECTION_BEAM for more information about the physical properties such as pressure and output options.

Card Summary:

Card 1. This card is required,

EID	PID	N1	N2	N3	RT1	RR1	RT2	RR2	LOCAL

Card 2. This card is included if the THICKNESS keyword option is used.

PARM1	PARM2	PARM3	PARM4	PARM5

Card 3. This card is included if the SECTION keyword option is used.

STYPE	D1	D2	D3	D4	D5	D6	

Card 4. This card is included if the SCALAR keyword option is used.

VOL	INER	CID	DOFN1	DOFN2

Card 5. This card is included if the SCALR keyword option is used.

VOL	INER	CID1	CID2	DOFNS

Card 6. This card is included if the PID keyword option is used.

PID1	PID2								

Card 7. This card is included if the OFFSET keyword option is used.

WX1	WY1	WZ1	WX2	WY2	WZ2		

Card 8. This card is included if the ORIENTATION keyword option is used.

VX	VY	VZ					

Card 9. This card is included if the WARPAGE keyword option is used.

SN1	SN2						

Card 10. This card is included if the ELBOW keyword option is used.

MN							

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	N3	RT1	RR1	RT2	RR2	LOCAL
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	0	0	0	0	2
Remarks					1	2	2	2	2	

VARIABLE	DESCRIPTION
EID	Element ID. A unique ID is generally required, that is, EID must be different from the element IDs also defined under *ELEMENT_-DISCRETE and *ELEMENT_SEATBELT. If the field BEAM is set to 1 on the keyword input for *DATABASE_BINARY_D3PLOT, the null beams used for visualization are not created for the latter two types, and the IDs used for the discrete elements and the seat-belt elements can be identical to those defined here.
PID	Part ID; see *PART.
N1	Nodal point (end) 1
N2	Nodal point (end) 2. This node is optional for the spot weld, beam type 9, since if it not defined it will be created automatically and given a non-conflicting nodal point ID. Nodes N1 and N2 are automatically positioned for the spot weld beam element. For the zero length discrete beam elements where one end is attached to ground, set N2 = -N1. For this case, a fully constrained nodal point will be created with a unique ID for node N2.
N3	Nodal point 3 for orientation. The third node, N3, is optional for beam types 3, 6, 7, and 8; if the cross-section is circular, it is optional for beam types 1 and 9. The third node is used for the discrete beam, type 6, if and only if SCOOR is set to 2.0 in the *SECTION_-

VARIABLE	DESCRIPTION
	BEAM input, but even in this case it is optional. An orientation vector can be defined directly by using the option, ORIENTATION. For this case N3 can be defined as zero.
RT1, RT2	Release conditions for translations at nodes N1 and N2, respectively: EQ.0: No translational degrees of freedom are released. EQ.1: *x*-translational degree of freedom EQ.2: *y*-translational degree of freedom EQ.3: *z*-translational degree of freedom EQ.4: *x* and *y*-translational degrees of freedom EQ.5: *y* and *z*-translational degrees of freedom EQ.6: *z* and *x*-translational degrees of freedom EQ.7: *x*, *y*, and *z*-translational degrees of freedom (3DOF) This option does not apply to the spot weld, beam type 9.
RR1, RR2	Release conditions for rotations at nodes N1 and N2, respectively: EQ.0: No rotational degrees of freedom are released. EQ.1: *x*-rotational degree of freedom EQ.2: *y*-rotational degree of freedom EQ.3: *z*-rotational degree of freedom EQ.4: *x* and *y*-rotational degrees of freedom EQ.5: *y* and *z*-rotational degrees of freedom EQ.6: *z* and *x*-rotational degrees of freedom EQ.7: *x*, *y*, and *z*-rotational degrees of freedom (3DOF) This option does not apply to the spot weld, beam type 9.
LOCAL	Coordinate system option for release conditions: EQ.1: Global coordinate system EQ.2: Local coordinate system (default)

Thickness Card. Additional card required for THICKNESS keyword option.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	PARM1		PARM2		PARM3		PARM4		PARM5	
Type	F		F		F		F		F	
Remarks	4		4		4		4		4, 5	

VARIABLE	DESCRIPTION
PARM1	Based on beam type: Type.EQ.1: Beam thickness, *s*-direction at node 1 Type.EQ.2: Area Type.EQ.3: Area Type.EQ.4: Beam thickness, *s*-direction at node 1 Type.EQ.5: Beam thickness, *s*-direction at node 1 Type.EQ.6: Volume, see description for VOL below. Type.EQ.7: Beam thickness, *s*-direction at node 1 Type.EQ.8: Beam thickness, *s*-direction at node 1 Type.EQ.9: Beam thickness, *s*-direction at node 1
PARM2	Based on beam type: Type.EQ.1: Beam thickness, *s*-direction at node 2 Type.EQ.2: I_{ss} Type.EQ.3: Ramp-up time for dynamic relaxation Type.EQ.4: Beam thickness, *s*-direction at node 2 Type.EQ.5: Beam thickness, *s*-direction at node 2 Type.EQ.6: Inertia, see description for INER below. Type.EQ.7: Beam thickness, *s*-direction at node 2 Type.EQ.8: Beam thickness, *s*-direction at node 2 Type.EQ.9: Beam thickness, *s*-direction at node 2
PARM3	Based on beam type:

VARIABLE	DESCRIPTION
	Type.EQ.1: Beam thickness, t-direction at node 1
	Type.EQ.2: I_{tt}
	Type.EQ.3: Initial stress for dynamic relaxation
	Type.EQ.4: Beam thickness, t-direction at node 1
	Type.EQ.5: Beam thickness, t-direction at node 1
	Type.EQ.6: Local coordinate ID
	Type.EQ.7: Not used
	Type.EQ.8: Not used
	Type.EQ.9: Beam thickness, t-direction at node 1
PARM4	Based on beam type:
	Type.EQ.1: Beam thickness, t-direction at node 2
	Type.EQ.2: I_{rr}
	Type.EQ.3: Not used
	Type.EQ.4: Beam thickness, t-direction at node 2
	Type.EQ.5: Beam thickness, t-direction at node 2
	Type.EQ.6: Area
	Type.EQ.7: Not used
	Type.EQ.8: Not used
	Type.EQ.9: Beam thickness, t-direction at node 2
PARM5	Based on beam type:
	Type.EQ.1: Not used
	Type.EQ.2: Shear area
	Type.EQ.3: Not used
	Type.EQ.4: Not used
	Type.EQ.5: Not used
	Type.EQ.6: Offset
	Type.EQ.7: Not used
	Type.EQ.8: Not used
	Type.EQ.9: Print flag to SWFORC file. The default is taken from the *SECTION_BEAM input. To override, set PAR-

VARIABLE	DESCRIPTION
	M5 to 1.0 to suppress printing and to 2.0 to print.

Section Card. Additional card required for SECTION keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	STYPE	D1	D2	D3	D4	D5	D6	
Type	A	F	F	F	F	F	F	

VARIABLE	DESCRIPTION
STYPE	Section type (A format) of resultant beam, see Figure 41-1:

EQ.SECTION_01: I-Shape
EQ.SECTION_02: Channel
EQ.SECTION_03: L-Shape
EQ.SECTION_04: T-Shape
EQ.SECTION_05: Tubular box
EQ.SECTION_06: Z-Shape
EQ.SECTION_07: Trapezoidal
EQ.SECTION_08: Circular
EQ.SECTION_09: Tubular
EQ.SECTION_10: I-Shape 2
EQ.SECTION_11: Solid box
EQ.SECTION_12: Cross
EQ.SECTION_13: H-Shape
EQ.SECTION_14: T-Shape 2
EQ.SECTION_15: I-Shape 3
EQ.SECTION_16: Channel 2
EQ.SECTION_17: Channel 3
EQ.SECTION_18: T-Shape 3
EQ.SECTION_19: Box-Shape 2
EQ.SECTION_20: Hexagon
EQ.SECTION_21: Hat-Shape
EQ.SECTION_22: Hat-Shape 2

D1-D6	Input parameters for section option using STYPE above.

Scalar card. Additional card for SCALAR keyword option.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	VOL		INER		CID		DOFN1		DOFN2	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
VOL	Volume of discrete beam and scalar (MAT_146) beam. If the mass

VARIABLE	DESCRIPTION
	density of the material model for the discrete beam is set to unity, the magnitude of the lumped mass can be defined here instead. This lumped mass is partitioned to the two nodes of the beam element. The translational time step size for the type 6 beam depends on the volume, mass density, and translational stiffness values, so it is important to define this parameter. Defining the volume is also essential for mass scaling if the type 6 beam controls the time step size.
INER	Mass moment of inertia for the six degrees of freedom discrete beam and scalar (*MAT_146) beam. This lumped inertia is partitioned to the two nodes of the beam element. The rotational time step size for the type 6 beam depends on the lumped inertia and the rotational stiffness values, so it is important to define this parameter if the rotational springs are active. Defining the rotational inertia is also essential for mass scaling if the type 6 beam rotational stiffness controls the time step size.
CID	Coordinate system ID for orientation, material type 146; see *DEFINE_COORDINATE_SYSTEM. If CID = 0, a default coordinate system is defined in the global system.
DOFN1	Active degree of freedom at node 1, a number between 1 to 6 where 1, 2, and 3 are the x, y, and z-translations and 4, 5, and 6 are the x, y, and z-rotations. This degree of freedom acts in the local system given by CID above. This input applies to material model type 146.
DOFN2	Active degree of freedom at node 2, a number between 1 to 6 as described in DOF1 above. This degree of freedom acts in the local system given by CID above. This input applies to material model type 146.

Scalar Card (alternative). Additional card for SCALR keyword option.

Card 5	1	2	3	4	5	6	7	8	9	10
Variable	VOL		INER		CID1		CID2		DOFNS	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
VOL	Volume of discrete and scalar (*MAT_146) beams. If the mass

VARIABLE	DESCRIPTION
	density of the material model for the discrete beam is set to unity, the magnitude of the lumped mass can be defined here instead. This lumped mass is partitioned to the two nodes of the beam element. The translational time step size for the type 6 beam depends on the volume, mass density, and translational stiffness values, so it is important to define this parameter. Defining the volume is also essential for mass scaling if the type 6 beam controls the time step size.
INER	Mass moment of inertia for the six degrees of freedom discrete beam and scalar (*MAT_146) beam. This lumped inertia is partitioned to the two nodes of the beam element. The rotational time step size for the type 6 beam depends on the lumped inertia and the rotational stiffness values, so it is important to define this parameter if the rotational springs are active. Defining the rotational inertia is also essential for mass scaling if the type 6 beam rotational stiffness controls the time step size.
CID1	Coordinate system ID at node 1 for orientation for material type 146; see *DEFINE_COORDINATE_SYSTEM. If CID1 = 0, a default coordinate system is defined in the global system.
CID2	Coordinate system ID at node 2 for orientation for material type 146; see *DEFINE_COORDINATE_SYSTEM. If CID2 = 0, a default coordinate system is defined in the global system.
DOFNS	Active degrees of freedom at node 1 and node 2. A two-digit number, the first for node 1 and the second for node 2, between 11 and 66 is expected where 1, 2, and 3 are the x, y, and z-translations and 4, 5, and 6 are the x, y, and z-rotations. These degrees of freedom act in the local system given by CID1 and CID2 above. This input applies to material model type 146. For example, if DOFNS = 12, node 1 has an x-translation and node 2 has a y-translation.

Spot Weld Part Card. Additional card for PID keyword option.

Card 6	1	2	3	4	5	6	7	8	9	10
Variable	PID1	PID2								
Type	I	I								

VARIABLE	DESCRIPTION
PID1	Optional part ID for spot weld element type 9
PID2	Optional part ID for spot weld element type 9

Offset Card. Additional card for OFFSET keyword option.

Card 7	1	2	3	4	5	6	7	8
Variable	WX1	WY1	WZ1	WX2	WY2	WZ2		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
WX1 - WZ1	Offset vector at nodal point N1. See Remark 7.
WX2 - WZ2	Offset vector at nodal point N2. See Remark 7.

Orientation Card. Additional card for ORIENTATION keyword option.

Card 8	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
VX, VY, VZ	Components of an orientation vector relative to node N1. In this case, the orientation vector points to a virtual third node, so the field N3 should be left undefined.

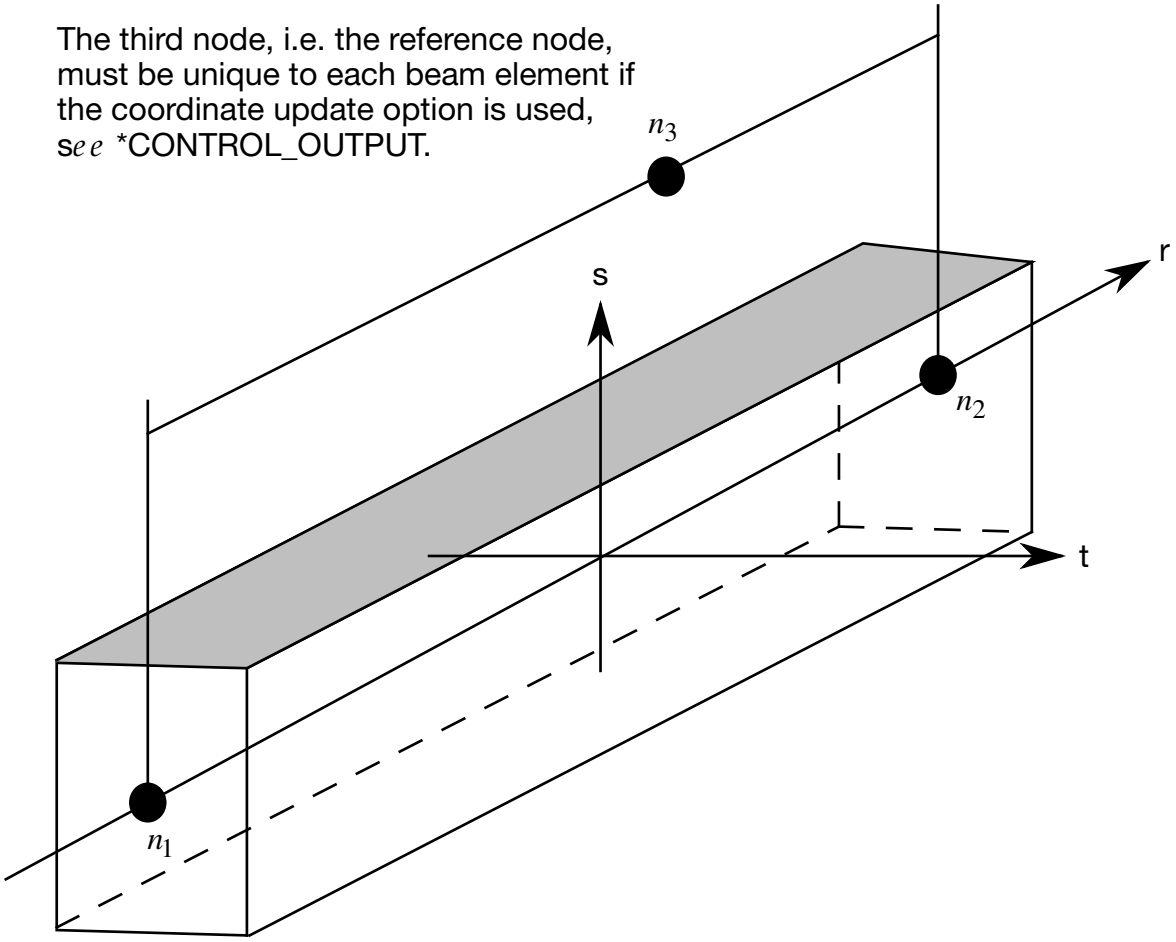


Figure 19-1. LS-DYNA beam elements. Node n_3 determines the initial orientation of the cross section.

Warpage Card. Additional card for WARPAGE keyword option.

Card 9	1	2	3	4	5	6	7	8
Variable	SN1	SN2						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
SN1	Scalar nodal point (end) 1. This node is required.
SN2	Scalar nodal point (end) 2. This node is required.

Elbow Card. Additional card for ELBOW keyword option.

Card 10	1	2	3	4	5	6	7	8
Variable	MN							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
MN	Middle node for the ELBOW element. See Remark 8.

Remarks:

1. **Beam orientation.** A plane through N_1, N_2, and N_3 defines the orientation of the principal *r-s* plane of the beam; see Figure 19-1.

2. **Release conditions.** The option to specify release conditions applies to all three-dimensional beam elements. The released degrees of freedom can be either global, or local relative to the local beam coordinate system as shown in Figure 19-1. A local coordinate system is stored for each node of the beam element, and the orientation of the local coordinate systems rotates with the node. To properly track the response, the nodal points with a released resultant are automatically replaced with new nodes to accommodate the added degrees of freedom. Then constraint equations are used to join the nodal points together with the proper release conditions imposed.

   > Nodal points belonging to beam elements that have release conditions applied cannot be subjected to other constraints, such as applied displacement/velocity/acceleration boundary conditions, nodal rigid bodies, nodal constraint sets, or any of the constraint type contact definitions.

   Force-type loading conditions and penalty-based contact algorithms may be used with this option.

   Please note that specification of translational release conditions may lead to non-physical constraints, but this should not be a problem if the displacements are infinitesimal.

3. **Beam properties.** If the THICKNESS option is not used, or if THICKNESS is used but essential PARMx values are not provided, beam properties are taken from *SECTION_BEAM.

4. **Discrete beam elements.** In the case of the THICKNESS option for type 6, discrete beam elements, PARM1 through PARM5 replace the first five fields on Card 2 of *SECTION_BEAM. Cables are a subset of type 6 beams. PARM1 is for non-cable discrete beams and is optional for cables, PARM2 and PARM3 apply only to non-cable discrete beams, and PARM4 and PARM5 apply only to cables.

5. **PARM5.** In the THICKNESS option, PARM5 applies only to beam types 2, 6 (cables only), and 9.

6. **Stress resultants.** The stress resultants are output in the local coordinate system for the beam. Stress information is optional and is also output in the local system for the beam.

7. **Offsets.** Beam offsets are sometimes necessary for correctly modeling beams that act compositely with other elements such as shells or other beams. When the OFFSET option is specified, global X, Y, and Z components of two offset vectors are given, one vector for each of the two beam nodes. The offset vector extends from the beam node (N1 or N2) to the reference axis of the beam. The beam reference axis lies at the origin of the local s and t-axes. For beam formulations 1 and 11, this origin is halfway between the outermost surfaces of the beam cross-section. Note that for cross-sections that are not doubly symmetric, such as a T-section, the reference axis does not pass through the centroid of the cross-section. For beam formulation 2, the origin is at the centroid of the cross-section.

8. **Elbow.** The Elbow beam is defined with 4 nodes; see Figure 19-2. Nodes n_1 and n_2 are the end nodes, and node n_3 is the middle node. It is custom to set n_3 at the midpoint of the beam. Node n_4 is an orientation node that should be at the curvature center of the beam. If a straight beam is defined initially, the orientation node must be defined and should be on the convex side of the beam. If a curved beam is defined initially, the orientation node is automatically calculated as the center of the beam curvature. However, an orientation node is still required at the input.

The extra nodes that include the ovalization degree of freedom are written to the messag file during initialization. These extra nodes have 3 DOFs each. That means that there are 2 extra nodes for each physical node. For example, it can look something like this:

```
ELBOW BEAM:            1
n1-n3-n2:              1      3      2
ovalization nodes:     5      7      6
                       8     10      9
```

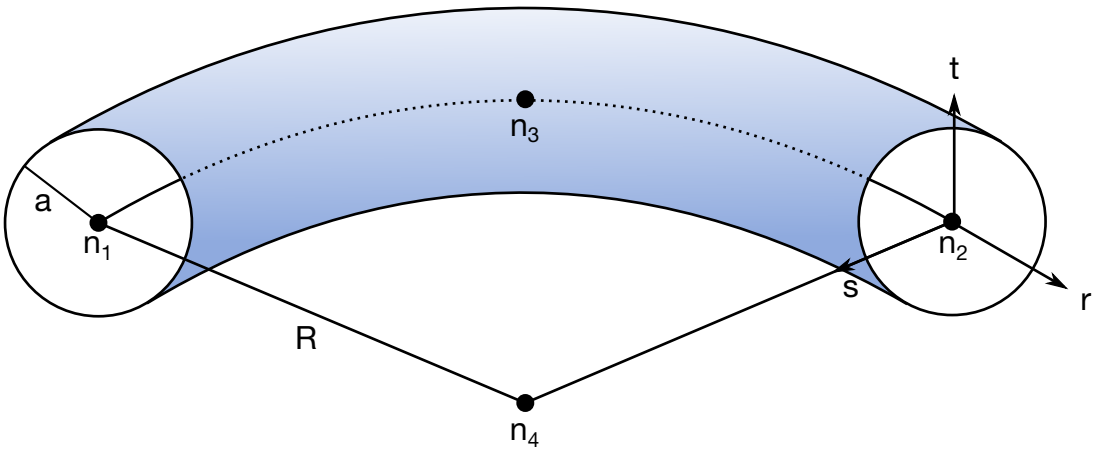


Figure 19-2. Elbow element. Node n_4 is the control node and is given as the beam center of curvature.

And it means that node 1 has the ovalization extra nodes 5 and 8. The first line of ovalization nodes includes the c_1, c_2 and c_3 parameters, and the second line includes the d_1, d_2 and d_3 parameters. That means that node 5 includes c_1, c_2 and c_3, and node 8 includes d_1, d_2 and d_3 for beam node 1. All ovalization dofs can be written to the ASCII file "elbwov" if the correct print flag IOVPR is set to 1 on *SECTION_BEAM. These extra nodes can be constrained as usual nodes. For example, for a cantilever beam that is mounted at node 1, the nodes 1, 5 and 8 should be constrained. The ovalization is approximated with the following trigonometric function:

$$w(r,\theta) = \sum_{k=1}^{3} \sum_{m=1}^{3} h_k(r)\left(c_m^k \cos 2m\theta + d_m^k \sin 2m\theta\right), \qquad -1 \le r \le 1, 0 \le \theta < 2\pi$$

where h_k is the interpolation function at the physical node k.

The Elbow beam only supports tubular cross sections and the pipe outer radius, a, should be smaller than the pipe bend radius, R. That is $a/R \ll 1$. Moreover, the ELBOW beams have 4 stresses: axial *rr*-, shear *rs*-, shear *rt*- and loop-stresses. The loop stress is written at each integration point and can be visualized in LS-PrePost with the user fringe plot file "elbwlp.k". NOTE that the loop-stress is not written to d3plot as default! The NEIPB flag on *DATABASE_EXTENT_BINARY must be set to enable d3plot support.

Right now there is only basic support from the material library. The following materials are currently supported for the ELBOW beam (if requested more materials might be added in the future):

*MAT_ELASTIC (MAT_001)

*MAT_PLASTIC_KINEMATIC (MAT_003)

*MAT_ELASTIC_PLASTIC_THERMAL (MAT_004)

*MAT_VISCOELASTIC (MAT_006)

*MAT_PIECEWISE_LINEAR_PLASTICITY (MAT_024)

*MAT_DAMAGE_3 (MAT_153, explicit only)

*MAT_CONCRETE_BEAM (MAT_195)

*ELEMENT_BEAM_PULLEY

Purpose: Define pulley for beam elements. This feature is implemented for truss beam elements (*SECTION_BEAM, ELFORM = 3) using materials *MAT_001 and *MAT_156 or discrete beam elements (ELFORM = 6) using *MAT_CABLE_DISCRETE_BEAM.

Card 1	1	2	3	4	5	6	7	8
Variable	PUID	BID1	BID2	PNID	FD	FS	LMIN	DC
Type	I	I	I	I	F	F	F	F
Default	none	0	0	0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PUID	Pulley ID. A unique number has to be used.
BID1	Truss beam element 1 ID.
BID2	Truss beam element 2 ID.
PNID	Pulley node, NID.
FD	Coulomb dynamic friction coefficient.
FS	Optional Coulomb static friction coefficient.
LMIN	Minimum length, see notes below.
DC	Optional decay constant to allow smooth transition between the static and dynamic friction coefficient, i.e., $\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}$

Remarks:

Elements 1 and 2 should share a node which is coincident with the pulley node. The pulley node should not be on any beam elements.

Pulleys allow continuous sliding of a truss beam element through a sharp change of angle. Two elements (1 & 2 in Figure 19-23 of *ELEMENT_SEATBELT_SLIPRING) meet at the pulley. Node *B* in the beam material remains attached to the pulley node, but beam material (in the form of unstretched length) is passed from element 1 to element 2 to

achieve slip. The amount of slip at each time step is calculated from the ratio of forces in elements 1 and 2. The ratio of forces is determined by the relative angle between elements 1 and 2 and the coefficient of friction, FD. The tension in the beams are taken as T_1 and T_2, where T_2 is on the high tension side and T_1 is the force on the low tension side. Thus, if T_2 is sufficiently close to T_1, no slip occurs; otherwise, slip is just sufficient to reduce the ratio T_2/T_1 to $e^{FC\times\theta}$, where θ is the wrap angle; see Figures 19-24 of *ELEMENT_SEATBELT_SLIPRING. The out-of-balance force at node B is reacted on the pulley node; the motion of node B follows that of pulley node.

If, due to slip through the pulley, the unstretched length of an element becomes less than the minimum length LMIN, the beam is remeshed locally: the short element passes through the pulley and reappears on the other side (see Figure 19-23). The new unstretched length of e_1 is 1.1 × minimum length. The force and strain in e_2 and e_3 are unchanged; while the force and strain in e_1 are now equal to those in e_2. Subsequent slip will pass material from e_3 to e_1. This process can continue with several elements passing in turn through the pulley.

To define a pulley, the user identifies the two beam elements which meet at the pulley, the friction coefficient, and the pulley node. If BID1 and BID2 are defined as 0 (zero), adjacent beam elements are automatically detected. The two elements must have a common node coincident with the pulley node. No attempt should be made to restrain or constrain the common node for its motion will automatically be constrained to follow the pulley node. Typically, the pulley node is part of a structure and, therefore, beam elements should not be connected to this node directly, but any other feature can be attached, including rigid bodies.

*DATABASE_PLLYOUT can be used to write a time history output database pllyout for the pulley which records beam IDs, slip, slip rate, resultant force, and wrap angle.

*ELEMENT_BEAM_SOURCE

Purpose: Define a nodal source for beam elements. This feature is implemented *only* for truss beam elements (*SECTION_BEAM, ELFORM = 3) with material *MAT_001, for discrete beam elements (ELFORM = 6) with material *MAT_071 or for Belytschko-Schwer resultant beam elements (ELFORM = 2) using material *MAT_001 or *MAT_028.

Card 1	1	2	3	4	5	6	7	8
Variable	BSID	BSNID	BSEID	NELE	LFED	FPULL	LMIN	NCUT
Type	I	I	I	I	F	F	F	I
Default	none	none	none	0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
BSID	Beam Source ID. A unique number has to be used.
BSNID	Source node ID.
BSEID	Source element ID.
NELE	Number of elements to be pulled out.
LFED	Beam element fed length (typical element initial length).
FPULL	Pull-out force. GT.0: Constant value, LT.0: Load curve ID = |FPULL| which defines pull-out force as a function of time. Either *DEFINE_FUNCTION or *DEFINE_ CURVE_FUNCTION (with argument TIME) can be used.
LMIN	Minimum beam element length, see notes below. One to two tenth of the fed length LFED is usually a good choice.
NCUT	Option to cut the last element from the source node: EQ.0: Last element remains attached to the source node (default). EQ.1: Last element gets cut from the source node along with sudden release of the pull-out force, FPULL.

VARIABLE	DESCRIPTION
	GT.1: Similar to NCUT = 1 but the pull-out force, FPULL, decreases linearly to zero over the last NCUT elements to avoid the sudden release of the pull-out force.

Remarks:

The source node BSNID can be defined by itself or it can be part of another structure. It is free to move in space during the simulation process. Initially, the source node should have the same coordinates as one of the source element (BSEID) nodes, but not having the same ID. If the pre-defined pull-out force FPULL is exceeded in the element next to the source, beam material gets drawn out by increasing the length of the beam without increasing its axial force (equivalent to ideal plastic flow at a given yield force). If more than the pre-defined length LFED is drawn out, a new beam element is generated. A new beam element has an initial undeformed length of 1.1 × LMIN. The maximum number of elements, NELE, times the fed length, LFED, defines the maximum cable length that can be pulled out from the source node.

***ELEMENT_BEARING**_OPTION

The available option is:

TITLE

Purpose: Define a bearing between two nodes. See Carney, Howard, Miller, and Benson [2014] for a description of this model.

Title Card. Additional card for title keyword option.

Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A70							
Default	none							
Remarks	1							

Card 1	1	2	3	4	5	6	7	8
Variable	ID	ITYPE	N1	CID1	N2	CID2	NB	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Material Properties Card.

Card 2	1	2	3	4	5	6	7	8
Variable	EBALL	PRBALL	ERACE	PRRACE	STRESL			
Type	F	F	F	F	F			
Default	0.0	0.0	0.0	0.0	0.0			

Geometry Card.

Card 3	1	2	3	4	5	6	7	8
Variable	D	DI	DO	DM				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

Geometry Card 2.

Card 4	1	2	3	4	5	6	7	8
Variable	A0	BI	BO	PD				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

Preloading Card.

Card 5	1	2	3	4	5	6	7	8
Variable	IPFLAG	XTRAN	YTRAN	ZTRAN	XROT	YROT		
Type	I	F	F	F	F	F		
Default	00	0.0	0.0	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ID	Bearing ID
ITYPE	Bearing type: EQ.1: Ball bearing EQ.2: Roller bearing
N1	Node on the centerline of the shaft (the shaft rotates)

VARIABLE	DESCRIPTION
CID1	Coordinate system ID orienting the shaft. The local *z*-axis defines the axis of rotation. See Remark 2.
N2	Node on the centerline of the bearing (the bearing does not rotate). It should initially coincide with N1.
CID2	Coordinate system ID orienting the bearing. The local *z*-axis defines the axis of rotation. See Remark 2.
NB	Number of balls or rollers
EBALL	Young's modulus for balls or rollers
PRBALL	Poisson's ratio for balls or rollers
ERACE	Young's modulus for races
PRRACE	Poisson's ratio for races
STRESL	Value of the bearing stress that causes LS-DYNA to print a warning message when it has been reached. If it is 0.0, then no message is printed. See Remark 1.
D	Diameter of balls or rollers
DI	Bore inner diameter
DO	Bore outer diameter
DM	Pitch diameter. If DM is not specified, it is calculated as the average of DI and DO.
A0	Initial contact angle in degrees
BI	Inner groove radius to ball diameter ratio for ball bearings and the roller length for roller bearings
BO	Outer race groove radius to ball diameter ratio. Unused for roller bearings.
PD	Total radial clearance between the ball bearings and races when no load is applied
IPFLAG	Preload flag (see Remark 3): EQ.0: No preload

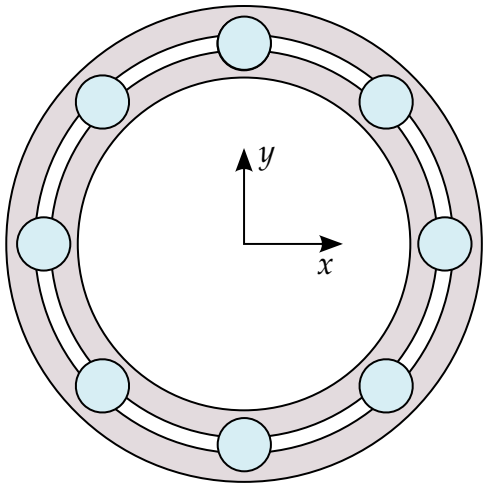


Figure 19-3. Schematic indicating the bearing's local coordinate system determined internally by LS-DYNA. LS-DYNA applies the preload in this coordinate system. The bearing rotates about the local z-axis determined with the right-hand rule.

VARIABLE	DESCRIPTION
	EQ.1: Displacement preload specified
	EQ.2: Force preload specified
XTRAN	Displacement or force preload in the local x-direction (see Remark 3)
YTRAN	Displacement or force preload in the local y-direction (see Remark 3)
ZTRAN	Displacement or force preload in the local z-direction (see Remark 3)
XROT	Angle (in radians) or moment preload in local x-direction (see Remark 3)
YROT	Angle (in radians) or moment preload in local y-direction (see Remark 3)

Remarks:

1. **Exceeding stress limit parameter.** If the bearing stress limit parameter, STRESL, is exceeded, LS-DYNA writes to the messag and d3hsp files. The element's behavior does not change when this value is surpassed.

2. **Local coordinate systems.** The coordinate systems, CID1 and CID2, must initially be aligned. Define these coordinate systems with *DEFINE_COORDINATE_NODES using FLAG = 1 because their orientations need to be updated.

3. **Preload coordinate system.** When LS-DYNA applies the preload, it applies values input on Card 5 exactly in the internally determined local, bearing coordinate system shown in Figure 19-3. In other words, LS-DYNA does not transform the values from the coordinate system into this system. LS-DYNA orients this local coordinate system such that the bearing rotates about the local z-direction and matches the system manufacturers use. Note that this bearing coordinate is not exactly the same as the coordinate system input with CID2. However, it will be very similar to CID2.

4. **Simulation precision.** We strongly suggest using double precision for solution stability.

5. **Bearing damping.** A realistic level of bearing damping (which can be included using *ELEMENT_DISCRETE and *MAT_DAMPER_VISCOUS) may be needed for solution stability.

6. **Bearing forces.** Bearing forces can be output in the brngout file, using *DATABASE_BEARING. This output includes the effects of the preload.

*ELEMENT_BLANKING

Purpose: Define a part set to be used by keyword *DEFINE_FORMING_BLANKMESH to generate a mesh on a sheet blank for metal forming simulations.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PSID	Part set ID. See *SET_PART.

Revisions:

This feature is available in LS-DYNA R5 Revision 59165 or later releases.

***ELEMENT_DIRECT_MATRIX_INPUT**_{OPTION}

Available options include:

<BLANK>

BINARY

Purpose: Define an element consisting of mass, damping, stiffness, and inertia matrices in a specified file, which follows the format used in the direct matrix input, DMIG, of NASTRAN. The supported format is the type 6 symmetric matrix in real double precision. We support both the standard and the extended precision formats. Binary format is only supported when LS-DYNA generates the superelement. The mass and stiffness matrices are required. The inertia matrix is required when using *LOAD_BODY_*OPTION* to correctly compute the action of a prescribed base acceleration on the superelement; otherwise, the inertia matrix is unused. The damping matrix is optional. The combination of these matrices is referred to as a superelement. Three input cards are required for each superelement.

The degrees of freedom for this superelement may consist of generalized coordinates as well as nodal point quantities. Degrees of freedom, defined using *NODE input, are called attachment nodes. Only attachment nodes are included in the output to the ASCII and binary databases.

The matrices for a given superelement can be of different order. However, the explicit integration scheme requires the inversion of the union of the element mass matrix and nodal masses associated with attachment nodes. Any degree of freedom included in the other (stiffness, damping, inertia) matrices but without nonzero columns in the combined mass matrix will be viewed as massless and constrained not to move. After deleting zero rows and columns, the combined mass matrix must be positive definite.

The inertia matrix must have 3 columns, corresponding to the 3 global coordinates. It is used to compute the forces acting on the superelement by multiplying the inertia matrix times the gravitational acceleration specified using *LOAD_BODY_*OPTION*.

There is no assumption made on the order of the matrices nor the sparse matrix structure of the element matrices, except that they are symmetric and the combined mass matrix is invertible as described above.

Multiple elements may be input using *ELEMENT_DIRECT_MATRIX_INPUT. They may share attachment nodes with other direct matrix input elements. Only *BOUNDARY_PRESCRIBED_MOTION and global constraints imposed using *NODE or *BOUNDARY_SPC on attachment nodes can be applied in explicit applications. Implicit applications can have additional constraints on attachment nodes.

Damping is included via the damping matrix; see the variable DAMP. That damping matrix may be computed by including *DAMPING_GLOBAL or *DAMPING_PART_-

MASS when constructing the elemental matrices; see SE_DAMP in *CONTROL_IMPLICIT_MODES.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	IFRMT	NID	PID	ICOROT			
Type	I	I	I	I	I			

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

Card 3	1	2	3	4	5	6	7	8
Variable	MASS	DAMP	STIF	INERT				
Type	A	A	A	A				

VARIABLE	DESCRIPTION
EID	Superelement ID
IFRMT	Format: EQ.0: Standard format NE.0: Extended precision format
NID	Node ID for the node corresponding to the placement of the center of mass. This node is internally generated and corresponds to the hub of beam elements representing the super element in the output files. If zero, this ID is internally generated and takes a value that is currently unused. If nonzero, then it is important that this ID is not in conflict with an ID of an existing node in the model.
PID	Part ID of the superelement. This is part ID of the beam elements internally generated to represent the superelement in the output files. If zero, this ID is internally generated and takes a value that

VARIABLE	DESCRIPTION
	is currently unused. If nonzero, then it is important that this ID is not in conflict with an ID of an existing part in the model.
ICOROT	Set to 1 to invoke a corotational option, meaning that the superelement can undergo large rotations with small overlayed displacements. Without this option, the superelement is restricted to small-displacement analysis.
FILENAME	Name of file that has the element matrices
MASS	Name of mass matrix in the file defined by FILENAME. This filename should be no more than eight characters to be compatible with NASTRAN.
DAMP	Name of damping matrix in the file defined by FILENAME. This filename should be no more than eight characters to be compatible with NASTRAN.
STIF	Name of stiffness matrix in the file defined by FILENAME. This filename should be no more than eight characters to be compatible with NASTRAN.
INERT	Name of inertia matrix in the file defined by FILENAME. This filename should be no more than eight characters to be compatible with NASTRAN. This file must be present when *LOAD_BODY is used to put gravitational forces on the model.

*ELEMENT_DISCRETE_{OPTION}

Available options include:

<BLANK>

LCO

Purpose: Define a discrete (spring or damper) element between two nodes or a node and ground.

An option, LCO, is available for using a load curve(s) to initialize the offset to avoid the excitation of numerical noise that can sometimes result with an instantaneous imposition of the offset. This can be done using a single curve at the start of the calculation or two curves where the second is used during dynamic relaxation prior to beginning the transient part. In the latter case, the first curve would simply specify the offset as constant during time. If the LCO option is active, a second card is read.

Beam type 6, see *ELEMENT_BEAM and SECTION_BEAM, may be used as an alternative to *ELEMENT_DISCRETE and *SECTION_DISCRETE, and is recommended if the discrete element's line of action is not node N1 to N2, i.e., if VID ≠ 0.

> **NOTE:** The discrete elements enter into the time step calculations. Care must be taken to ensure that the nodal masses connected by the springs and dampers are defined and unrealistically high stiffness and damping values must be avoided. **All rotations are in radians**.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	VID	S		PF	OFFSET	
Type	I	I	I	I	I	F		I	F	
Default	none	none	none	none	0	1.0		0	0.0	

Offset Load Curve Card. Additional card for LCO keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	LCID	LCIDDR						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
EID	Element ID. A unique number is required. Since null beams are created for visualization, this element ID should not be identical to element ID's defined for *ELEMENT_BEAM and *ELEMENT_SEATBELT.
PID	Part ID, see *PART.
N1	Nodal point 1.
N2	Nodal point 2. If zero, the spring/damper connects node N1 to ground.
VID	Orientation option. The orientation option should be used cautiously since forces, which are generated as the nodal points displace, are not orthogonal to rigid body rotation unless the nodes are coincident. The type 6, 3D beam element, is recommended when orientation is required with the absolute value of the parameter SCOOR set to 2 or 3, since this option avoids rotational constraints. EQ.0: the spring/damper acts along the axis from node N1 to N2, NE.0: the spring/damper acts along the axis defined by the orientation vector, VID defined in the *DEFINE_SD_ORIENTATION section.
S	Scale factor on forces.
PF	Print flag: EQ.0: forces are printed in DEFORC file, EQ.1: forces are not printed DEFORC file.

VARIABLE	DESCRIPTION
OFFSET	Initial offset. The initial offset is a displacement or rotation at time zero. For example, a positive offset on a translational spring will lead to a tensile force being developed at time zero. Ignore this input if LCID is defined below.
LCID	Load curve ID defining the initial OFFSET as a function of time. Positive offsets correspond to tensile forces, and, likewise negative offsets result in compressive forces.
LCIDDR	Load curve ID defining OFFSET as a function of time during the dynamic relaxation phase.

*ELEMENT_DISCRETE_SPHERE_{OPTION}

Available options include:

<BLANK>

VOLUME

Purpose: Define a discrete spherical element for discrete element method (DEM) calculations. Currently, LS-DYNA's implementation of the DEM supports only spherical particles, as discrete element spheres (DES). Each DES consists of a single node with its mass, mass moment of inertia, and radius defined by the input below. Initial coordinates and velocities are specified using the nodal data. The element ID corresponds to the ID of the node. The discrete spherical elements are visualized in LS-PrePost using the same options as the SPH elements.

Please note that the DES part requires *PART, *SECTION, and *MAT keywords. The element type and formulation values in *SECTION are ignored. No *SECTION keyword specific to DES exists, so any type of *SECTION keyword can be used. *SECTION_SOLID is the most commonly used. DEM retrieves the bulk modulus from the *MAT input for coupling stiffness and time step size evaluation, and density from the *MAT input if VOLUME is used to calculate the proper mass. *MAT_ELASTIC and *MAT_RIGID are most commonly used, but other material models are also permissible.

Card Summary:

Card 1a. Include this card if not using the keyword option (<BLANK>).

NID	PID	MASS	INERTIA	RADIUS		IDIST	NID2

Card 1b. Include this card if using the VOLUME keyword option.

NID	PID	VOLUME	INERTIA	RADIUS		IDIST	NID2

Card 2a. Include this card for IDIST = -1 or 1.

		MSD	ISD	RSD			

Card 2b. Include this card for IDIST = -2 or 2.

		MM	MI	MR			

Data Card Definitions:

Card 1a	1	2	3	4	5	6	7	8
Variable	NID	PID	MASS	INERTIA	RADIUS		IDIST	NID2
Type	I	I	F	F	F		I	I
Default	none	none	none	none	none		0	0

VARIABLE	DESCRIPTION
NID	DES Node ID
PID	DES Part ID (see *PART)
MASS	Mass. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MM ≠ 0, this value is the scale parameter.
INERTIA	Mass moment of inertia. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MI ≠ 0, this value is the scale parameter.
RADIUS	Particle radius. Determining contact between particles requires the particle radius. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MR ≠ 0, this value is the scale parameter.
IDIST	Distribution of DES properties (see Remarks 1 and 2): EQ.-2: Weibull distribution (non-deterministic) EQ.-1: Gaussian distribution (non-deterministic) EQ.0: Single property (default) EQ.1: Gaussian distribution (deterministic) EQ.2: Weibull distribution (deterministic)
NID2	Define more than one element with the same PID by setting this field without requiring additional cards. For IDIST = 0, these elements share equal MASS, INERTIA, and RADIUS. For IDIST ≠ 0, these elements each have their MASS, INERTIA, and/or RADIUS randomly distributed according to the distribution and Card 2 parameters assigned. If set, NID2 is a node ID that must have a value greater than NID. Then, LS-DYNA associates a DES to each node

VARIABLE	DESCRIPTION
	with an ID between NID and NID2 (including NID and NID2). If 0 or left blank, then only NID has a DES associated with it.

Card 1b	1	2	3	4	5	6	7	8
Variable	NID	PID	VOLUME	INERTIA	RADIUS		IDIST	NID2
Type	I	I	F	F	F		I	I
Default	none	none	none	none	none		0	0

VARIABLE	DESCRIPTION
NID	DES Node ID.
PID	DES Part ID, see *PART.
VOLUME	Volume. The mass is calculated from material density, $M = \text{VOLUME} \times \rho_{\text{mat}}$. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MM ≠ `0, this value is the scale parameter.
INERTIA	Inertia per unit density. The actual inertia is calculated from material density, $I = \text{INERTIA} \times \rho_{\text{mat}}$. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MI ≠ 0, this value is the scale parameter.
RADIUS	Particle radius. Determining contact between particles requires the particle radius. For IDIST = -1 and 1, this value is the mean. For IDIST = -2 and 2 with MR ≠ 0, this value is the scale parameter.
IDIST	Distribution of DES properties (see Remarks 1 and 2): EQ.-2: Weibull distribution (non-deterministic) EQ.-1: Gaussian distribution (non-deterministic) EQ.0: Single property (default) EQ.1: Gaussian distribution (deterministic) EQ.2: Weibull distribution (deterministic)

VARIABLE	DESCRIPTION
NID2	Define more than one element with the same PID by setting this field without requiring additional cards. For IDIST = 0, these elements share equal VOLUME, INERTIA, and RADIUS. For IDIST ≠ 0, these elements each have their VOLUME, INERTIA, and/or RADIUS randomly distributed according to the distribution and Card 2 parameters assigned. If set, NID2 is a node ID that must have a value greater than NID. Then, LS-DYNA associates a DES to each node with an ID between NID and NID2 (including NID and NID2). If 0 or left blank, then only NID has a DES associated with it.

Standard Deviation for Gaussian Distribution Card. Include this card if IDIST = -1 or 1.

Card 2a	1	2	3	4	5	6	7	8
Variable			MSD	ISD	RSD			
Type			F	F	F			
Default			0.0	0.0	0.0			

VARIABLE	DESCRIPTION
MSD	Standard deviation for mass or volume (if using the VOLUME keyword option). The mass or volume on Card 1 is the mean. EQ.0.0: Uniform distribution
ISD	Standard deviation for mass moment of inertia or inertia per unit density (if using the VOLUME keyword option). The inertia (or inertia per unit density) on Card 1 is the mean. EQ.0.0: Uniform distribution
RSD	Standard deviation for radius. The radius on Card 1 is the mean. EQ.0.0: Uniform distribution

Shape Parameters for Weibull Distribution Card. Include this card if IDIST = -2 or 2.

Card 2b	1	2	3	4	5	6	7	8
Variable			MM	MI	MR			
Type			F	F	F			
Default			0.0	0.0	0.0			

VARIABLE	DESCRIPTION
MM	Shape parameter (heterogeneity index) for mass or volume (if using the VOLUME keyword option). The mass (or volume) on Card 1 is the associated scale parameter. EQ.0.0: No distribution, MASS / VOLUME sets the value of mass / volume.
MI	Shape parameter (heterogeneity index) for mass moment of inertia or inertia per unit density (if using the VOLUME keyword option). The inertia (or inertia per unit density) on Card 1 is the associated scale parameter. EQ.0.0: No distribution. INERTIA sets the value of inertia or inertia per unit density.
MR	Shape parameter (heterogeneity index) for radius. The radius on Card 1 is the associated scale parameter. EQ.0.0: No distribution. RADIUS sets the value of the radius.

Remarks:

1. **Distributions of DES properties.** IDIST determines the distribution of DES properties for the DES. It specifically affects the mass (or volume), inertia, and radius. IDIST = 0 means all the DES have the same properties.

   The distribution of DES properties for IDIST = -1 and 1 follows a Gaussian distribution:

$$f(x|\text{prop_mean}, \text{prop_sd}) = \frac{1}{\text{prop_sd}\sqrt{2\pi}} e^{\frac{(x-\text{prop_mean})^2}{2\times\text{prop_sd}^2}},$$

where prop_mean is the mean value of the property, prop_sd is the standard deviation, and x is the statistical distribution of the property. For this distribution, you input the following pairs to specify the mean and standard deviation for the three DES properties: (MASS or VOLUME, MSD), (INERTIA, ISD), and (RADIUS, RSD).

The distribution of DES properties for IDIST = -2 and 2 follows a Weibull distribution:

$$f(n|\text{prop_sf}, \text{mprop}) = \frac{\text{mprop}}{\text{prop_sf}} \left(\frac{n}{\text{prop_sf}} \right)^{\text{mprop}-1} e^{-\left(\frac{n}{\text{prop_sf}}\right)^{\text{mprop}}},$$

where prop_sf is the scale factor for the property, mprop is the shape factor for the property, and n is the statistical distribution of the property. For this distribution, you input the following pairs to specify the scale factor and shape factor for the three DES properties: (MASS or VOLUME, MM), (INERTIA, MI), and (RADIUS, MR).

2. **Deterministic versus non-deterministic distributions.** We have implemented both deterministic and non-deterministic versions for the distributions of DES properties (IDIST ≠ 0). By deterministic, we mean that each run uses the same seed and has the same sequence of numbers that fit the distribution. Thus, multiple runs with the same input deck yield the same results. In contrast, the non-deterministic algorithms obtain a random seed each time. Therefore, each run with the same input deck may have a different solution as the sequence of numbers changes each time.

*ELEMENT_GENERALIZED_SHELL

Purpose: Define a general 3D shell element with an arbitrary number of nodes. The formulation of this element is specified in *DEFINE_ELEMENT_GENERALIZED_-SHELL, which is specified through the part ID (see *PART) and the section ID (see *SECTION_SHELL). For an illustration of this referencing, see Figure 19-4. This generalized shell implementation allows for rapid prototyping of new shell element formulations without requiring further coding.

The element formulation used in *SECTION_SHELL needs to be greater or equal than 1000.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	PID	NMNP					
Type	I	I	I					
Default	none	none	none					

Connectivity Cards. Define the connectivity of the element by specifying NMNP-nodes (up to eight nodes per card). Include as many cards as needed. For example, for NMNP = 10, the deck should include two additional cards.

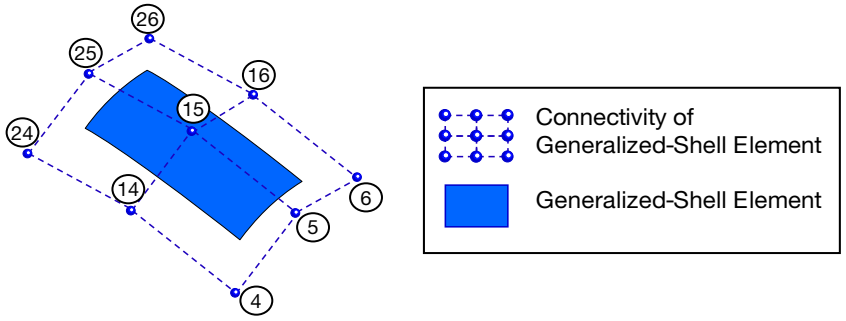
Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EID	Element ID. Chose a unique number with respect to other elements.
PID	Part ID, see *PART.
NMNP	Number of nodes to define this element.
N_i	Nodal point *i* (defined via *NODE) to define connectivity of this element.

Remarks:

1. **Interpolation Shell Elements.** For post-processing and the treatment of contact boundary conditions, the use of interpolation shell elements (see *ELEMENT_-INTERPOLATION_SHELL and *CONSTRAINED_NODE_INTERPOLATION) is necessary.

2. **Connectivity.** The definition of the connectivity of the element is basically arbitrary, but it has to correlate with the definition of the element formulation in *DEFINE_ELEMENT_GENERALIZED_SHELL.




```
*ELEMENT_GENERALIZED_SHELL
$---+--EID----+--PID----+-NMNP----+----4----+----5----+----6----+----7----+----8
         1         11          9
$---+---N1----+---N2----+---N3----+---N4----+---N5----+---N6----+---N7----+---N8
        26         25         24         16         15         14          6          5
$---+---N9----+--Etc----+--Etc----+--Etc----+--Etc----+--Etc----+--Etc----+--Etc
         4

*PART
Part for generalized shell
$---+--PID----+SECID----+--MID----+----4----+----5----+----6----+----7----+----8
         11         15          3
*SECTION_SHELL
$---+SECID----ELFORM----+-SHRF----+--NIP----+----5----+----6----+----7----+----8
         15       1001                    2
$---+---T1----+---T2----+---T3----+---T4----+----5----+----6----+----7----+----8
        1.0
*DEFINE_ELEMENT_GENERALIZED_SHELL
$---ELFORM---+---NGP----+-NMNP----+IMASS----+-FORM----+----6----+----7----+----8
        1001           4          9          0          1
...
```

Figure 19-4. Example of the connection between *ELEMENT_GENERALIZED_-SHELL and *DEFINE_ELEMENT_GENERALIZED_SHELL.

*ELEMENT_GENERALIZED_SOLID

Purpose: Define a general 3D solid element with an arbitrary number of nodes. The formulation of this element is specified in *DEFINE_ELEMENT_GENERALIZED_SOLID, which is referenced through the part ID (see *PART) and the section ID (see *SECTION_SOLID). For an illustration of this referencing, see Figure 19-5. This generalized solid implementation allows for rapid prototyping of new solid element formulations without further coding.

The element formulation used in *SECTION_SOLID needs to be greater or equal than 1000.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	PID	NMNP					
Type	I	I	I					
Default	none	none	none					

Connectivity Cards. Define the connectivity of the element by specifying NMNP nodes (up to eight nodes per card). Include as many cards as needed. For example, for NMNP = 10, the deck should include two additional cards.

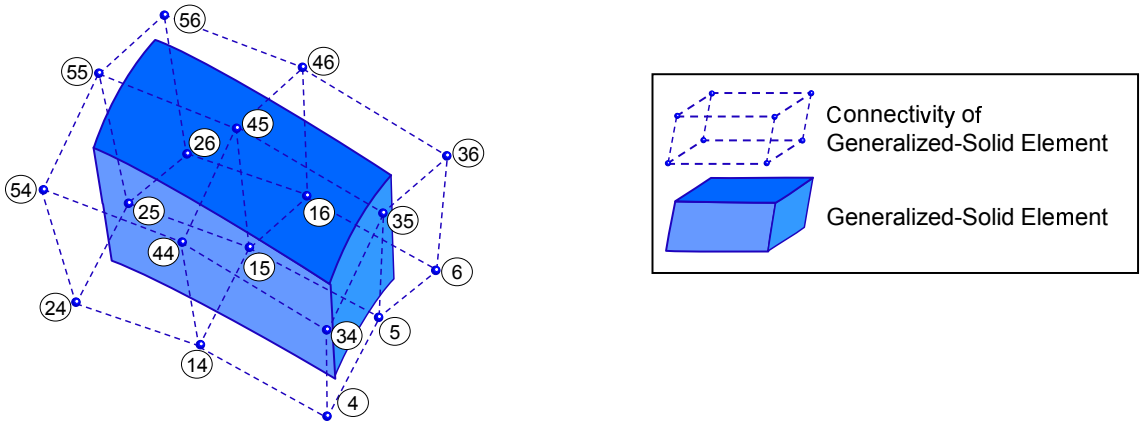
Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EID	Element ID. Chose a unique number with respect to other elements.
PID	Part ID, see *PART.
NMNP	Number of nodes to define this element.
N_i	Nodal point *i* (defined via *NODE) to define connectivity of this element.

Remarks:

1. **Post-Processing.** For post-processing the use of interpolation solid elements (see *ELEMENT_INTERPOLATION_SOLID and *CONSTRAINED_NODE_INTERPOLATION) is necessary.

2. **Connectivity.** The definition of the connectivity of the element is basically arbitrary, but it must be in correlation with the definition of the element formulation in *DEFINE_ELEMENT_GENERALIZED_SOLID.




```
*ELEMENT_GENERALIZED_SOLID
$---+--EID----+--PID----+-NMNP----+----4----+----5----+----6----+----7----+----8
         1         11         18
$---+---N1----+---N2----+---N3----+---N4----+---N5----+---N6----+---N7----+---N8
        56         55         54         46         45         44         36         35
$---+---N9----+--N10----+--N11----+--N12----+--N13----+--N14----+--N15----+--N16
        34         26         25         24         16         15         14          6
$---+--N17----+--N18----+--Etc----+--Etc----+--Etc----+--Etc----+--Etc----+--Etc
         5          4

*PART
Part for generalized solid
$---+--PID----+SECID----+--MID----+----4----+----5----+----6----+----7----+----8
         11         15          3
*SECTION_SOLID
$---+SECID----ELFORM----+--AET----+----4----+----5----+----6----+----7----+----8
         15       1001                    2
*DEFINE_ELEMENT_GENERALIZED_SOLID
$---ELFORM---+---NGP----+-NMNP----+IMASS----+----5----+----6----+----7----+----8
       1001           8         18         0
```

Figure 19-5. Example of the connection between *ELEMENT_GENERALIZED_SOLID and *DEFINE_ELEMENT_GENERALIZED_SOLID.

***ELEMENT_INERTIA**_{OPTION}_

Available options include:

<BLANK>

OFFSET

Purpose: Define a lumped inertia and, optionally, a lumped mass. This inertia/mass may be located at a nodal point or may be offset from a nodal point. The nodal point can belong to either a deformable body or a rigid body.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	NID	CSID							
Type	I	I	I							
Default	none	none	global							

Card 2	1	2	3	4	5	6	7	8
Variable	IXX	IXY	IXZ	IYY	IYZ	IZZ	MASS	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Offset Card. Additional card for offset keyword option.

Card 3	1	2	3	4	5	6	7	8
Variable	X-OFF	Y-OFF	Z-OFF					
Type	F	F	F					
Default	0.	0.	0.					

VARIABLE	DESCRIPTION
EID	Element ID. A unique number must be used.
NID	Node ID. Node to which the mass and inertia is assigned.
CSID	Coordinate system ID EQ.0: Components of the inertia tensor are input with orientation in the global coordinate system. GE.1: Principal moments of inertias are input with orientation of the principal axes defined by coordinate system CSID. See *DEFINE_COORDINATE_*OPTION*. See Remark 2.
IXX	*xx* component of inertia tensor
IXY	*xy* component of inertia tensor
IXZ	*xz* component of inertia tensor
IYY	*yy* component of inertia tensor
IYZ	*yz* component of inertia tensor
IZZ	*zz* component of inertia tensor
MASS	Lumped mass (optional)
X-OFF	*x*-offset from nodal point
Y-OFF	*y*-offset from nodal point
Z-OFF	*z*-offset from nodal point

Remarks:

1. **Inertia and Mass Contribution.** The inertia and mass specified by this keyword is added to the inertia and mass contributed by the elements.

2. **Principal Moments of Inertia.** If CSID is defined, then IXY, IXZ and IYZ are set to zero. Because its inverse is required, the nodal inertia tensor must be positive definite, meaning its determinant must be greater than zero. This check is done after the nodal inertia is added to the defined inertia tensor.

3. **Inertia Tensor for Deformable Bodies.** Nodes of deformable bodies do not support a full inertia tensor. For these bodies, only one value is needed. Specified off-diagonal terms will be ignored. If there is variation among terms on the

diagonal, the largest value will be used. Therefore, an easy way to input inertia for a deformable body node is to define only IXX.

*ELEMENT_INTERPOLATION_SHELL

Purpose: With the definition of interpolation shells, the stresses and other solution variables can be interpolated from the generalized shell elements (see *ELEMENT_GENERALIZED_SHELL and *DEFINE_ELEMENT_GENERALIZED_SHELL) permitting the solution to be visualized using standard 4-node shell elements with one integration point (one value of each solution variable per interpolation shell). The definition of the interpolation shells is based on interpolation nodes (see *CONSTRAINED_NODE_INTERPOLATION). The connections between these various keywords are illustrated in Figure 19-6.

Card 1	1	2	3	4	5	6	7	8
Variable	EIDS	EIDGS	NGP					
Type	I	I	I					
Default	none	none	none					

Weighting Factor Cards. These cards set the weighting factors used for interpolating the solution onto the center of *this* interpolation shell. Set one weight for each of the NGP integration points. Each card can accommodate 4 points; define as many cards as necessary. For example, if NGP = 10, three cards are required.

Card 2	1	2	3	4	5	6	7	8
Variable	IP1	W1	IP2	W2	IP3	W3	IP4	W4
Type	I	F	I	F	I	F	I	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EIDS	Element ID of the interpolation shell. This needs to coincide with a proper definition of a 4-node shell element (*ELEMENT_SHELL) using interpolation nodes (*CONSTRAINED_NODE_INTERPOLATION).
EIDGS	Element ID of the master element defined in *ELEMENT_GENERALIZED_SHELL.

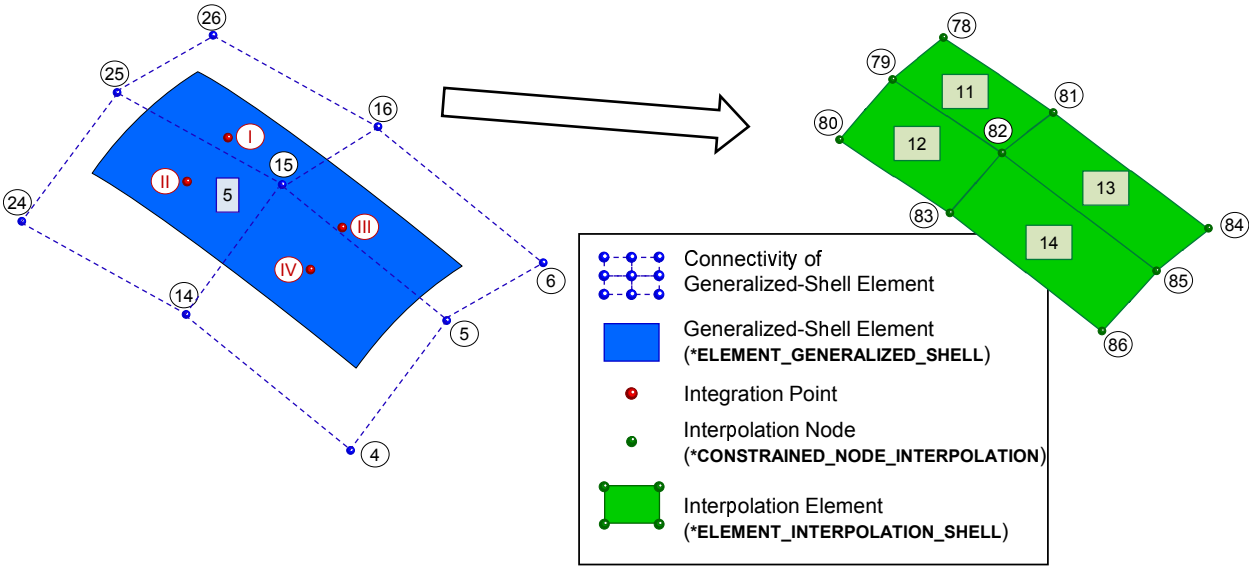
VARIABLE	DESCRIPTION
NGP	Number of in-plane integration points of the master element.
IP_i	Integration point number (1 to NGP) in the order they were defined in *DEFINE_ELEMENT_GENERALIZED_SHELL.
W_i	Interpolation weight of integration point *i*.

Remarks:

1. **Interpolation.** For each interpolation shell element, one single value (v_{IS}) of a solution variable is interpolated based on values at the integration points (v_i) of the master element (*ELEMENT_GENERALIZED_SHELL) and the appropriate weighting factors (w_i). The interpolation is computed as follows:

$$v_{IS} = \sum_{i=1}^{NGP} w_i v_i \ .$$

2. **Element Formulation.** To use *ELEMENT_INTERPOLATION_SHELL, ELFORM = 98 must be used in *SECTION_SHELL.




```
*CONSTRAINED_NODE_INTERPOLATION
$---+--NID----+NUMCN----+----3----+----4----+----5----+----6----+----7----+----8
        78         4
$---+--CN1----+---W1----+--CN2----+---W2----+--CN3----+---W3----+--CN4----+---W4
        26      0.35         25      0.32         15      0.18         16      0.15
*ELEMENT_SHELL
$--+-EID---+ PID---+- N1---+--N2---+--N3---+--N4---+--N5---+--N6---+--N7---+--N8
       11       33       78       79       82       81
*PART
Part for interpolation shell
$---+--PID----+SECID----+--MID----+----4----+----5----+----6----+----7----+----8
        33         45         3
*SECTION_SHELL
$---+SECID----ELFORM----+-SHRF----+--NIP----+----5----+----6----+----7----+----8
        45         98                   2
$---+---T1----+---T2----+---T3----+---T4----+----5----+----6----+----7----+----8
       1.0
*ELEMENT_INTERPOLATION_SHELL
$----+-EIDS---+-EIDGS----+--NGP----+----4----+----5----+----6----+----7----+----8
       11          5          4
$---+--IP1---+----W1----+--IP2----+---W2----+--IP3----+---W3----+--IP4----+---W4
         1       0.5          2       0.2          3       0.2          4       0.1
```

Figure 19-6. Example for *ELEMENT_INTERPOLATION_SHELL.

*ELEMENT_INTERPOLATION_SOLID

Purpose: With the definition of interpolation solids, the stresses and other solution variables can be interpolated from the generalized solid elements (see *ELEMENT_GENERALIZED_SOLID and *DEFINE_ELEMENT_GENERALIZED_SOLID) permitting the solution to be visualized using standard 8-noded solid elements with one integration point (one value of each solution variable per interpolation solid). The definition of the interpolation solids is based on interpolation nodes (see *CONSTRAINED_NODE_INTERPOLATION). The connection between these various keywords are illustrated in Figure 19-7.

Card 1	1	2	3	4	5	6	7	8
Variable	EIDS	EIDGS	NGP					
Type	I	I	I					
Default	none	none	none					

Weighting Factor Cards. These cards set the weighting factors used for interpolating the solution onto the center of *this* interpolation solid. Set one weight for each of the element's NGP integration points. Each card can accommodate 4 points; define as many cards as necessary. As an example, for NGP = 10 three cards are required.

Card 2	1	2	3	4	5	6	7	8
Variable	IP1	W1	IP2	W2	IP3	W3	IP4	W4
Type	I	F	I	F	I	F	I	F
Default	none	none	none	none	none	none	none	none

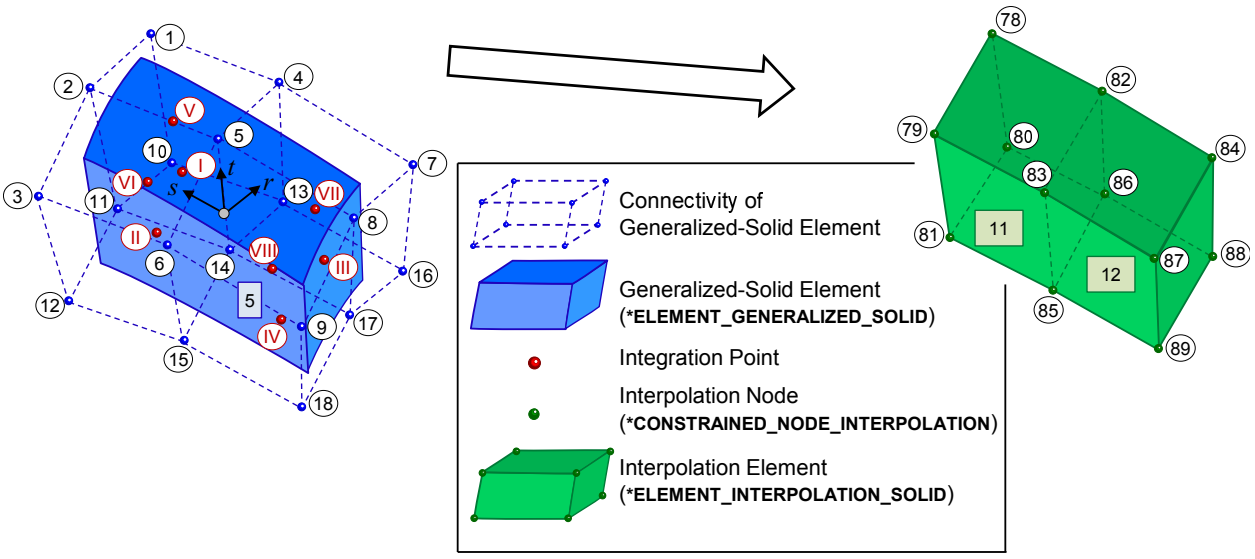
VARIABLE	DESCRIPTION
EIDS	Element ID of the interpolation solid. This needs to coincide with a proper definition of an 8-noded solid element (*ELEMENT_SOLID) using interpolation nodes (*CONSTRAINED_NODE_INTERPOLATION).
EIDGS	Element ID of the master element defined in *ELEMENT_GENERALIZED_SOLID.

VARIABLE	DESCRIPTION
NGP	Number of integration points of the master element.
IP_i	Integration point number (1 to NGP) in the order how they were defined in *DEFINE_ELEMENT_GENERALIZED_SOLID.
W_i	Interpolation weight of integration point *i*.

Remarks:

1. **Interpolation.** For each interpolation solid element, one single value (v_{IS}) of a solution variable is interpolated based on values at the integration points (v_i) of the master element (*ELEMENT_GENERALIZED_SOLID) and the appropriate weighting factors w_i. The interpolation is computed as follows: $v_{IS} = \sum_{i=1}^{NGP} w_i v_i$

2. **Element Formulation.** To use *ELEMENT_INTERPOLATION_SOLID, ELFORM = 98 must be used in *SECTION_SOLID.




```
*ELEMENT_SOLID
$--+-EID---+ PID---+- N1---+--N2---+--N3---+--N4---+--N5---+--N6---+--N7---+--N8
        11       33
$--+--N1---+--N2---+- N3---+--N4---+--N5---+--N6---+--N7---+--N8---+--N9---+-N10
        80       81       85       86       78       79       83       82

*PART
Part for interpolation solid
$---+--PID----+SECID----+--MID----+----4----+----5----+----6----+----7----+----8
          33         45          3

*SECTION_SOLID
$---+SECID----ELFORM----+--AET----+----4----+----5----+----6----+----7----+----8
          45         98

*ELEMENT_INTERPOLATION_SOLID
$-----EIDS---+-EIDGS----+--NGP----+----4----+----5----+----6----+----7----+----8
          11          5          8
$---+--IP1---+----W1----+--IP2----+---W2----+--IP3----+---W3----+--IP4----+---W4
          1       0.30          2       0.12          3       0.13          4       0.07
$---+--IP5---+----W5----+--IP6----+---W6----+--IP7----+---W7----+--IP8----+---W8
          5       0.20          6       0.08          7       0.07          8       0.03
```

Figure 19-7. Example for *ELEMENT INTERPOLATION SOLID.

*ELEMENT_LANCING

Purpose: This feature models a lancing process during a metal forming process by trimming along a curve. Two types of lancing, instant and progressive, are supported. This keyword is used together with *DEFINE_CURVE_TRIM_3D, and only applies to shell elements. The lanced scraps can be removed (trimming) during or after lancing when used in conjunction with *DEFINE_LANCE_SEED_POINT_COORDINATES; see Trimming remarks below. This feature is available for explicit dynamics for both SMP and MPP.

The element lancing feature is supported in LS-PrePost 4.3, under Application → Metal-Forming → Easy Setup.

For each trim include an additional card. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	IDPT	IDCV	IREFINE	SMIN	AT	ENDT	NTIMES	CIVD
Type	I	I	I	F	F	F	I	I
Default	none	none	1	none	none	↓	↓	↓

VARIABLE	DESCRIPTION
IDPT	A flag to indicate if a part to be lanced is a part or a part set. GT.0: IDPT is the PID of a part to be lanced; see *PART. LT.0: The absolute value is the part set ID as in *SET_PART_LIST. This option allows the lancing to be performed across a tailor-welded line.
IDCV	A trim curve ID (the variable TCID in *DEFINE_CURVE_TRIM_3D) defining a lancing route (see Remarks). XYZ format (TCTYPE = 1) has always been supported; however, IGES format (TCTYPE = 2) is not supported until Revision 110246.
IREFINE	Set IREFINE = 1 to refine elements along lancing route until no adapted nodes exist in the neighborhood. This feature results in a more robust lancing in the form of improved lancing boundary. Available starting in Revision 107708. Values greater than "1" are not allowed. See Figure 19-16 for an example of the mesh refinement.

VARIABLE	DESCRIPTION
SMIN	Minimum element characteristic length to be refined to (to be supported in the future). Currently, one level of refinement will be automatically done.
AT	Activation time for lancing operation. This variable needs to be defined for both instant and progressive lancing types (see Remarks). If CIVD is defined, AT becomes the distance from the punch home position.
ENDT	Lancing end time (for progressive lancing only). If CIVD is defined, ENDT becomes the distance from the punch home position. Do not define for instant lancing.
NTIMES	A progressive lancing operation is evenly divided into NTIMES segments between AT and ENDT; within each segment lancing is done instantly. Do not define for instant lancing.
CIVD	The load curve ID (LCID) to define the kinematics (time vs. velocity) of the tool, with VAD = 0; see *DEFINE_CURVE. When this variable is used, AT and ENDT will become the distances from the punch bottom position, unless the variables DT and BT are defined in *BOUNDARY_PRESCRIBED_MOTION_RIGID. See an example with CIVD.

Remarks:

Lancing the blank during forming at strategic locations under controlled conditions alleviates thinning and necking of sheet metal panels. Typically, the blank is lanced in the last few millimeters before the punch reaches its home position. Being an unstable process, lancing is not favored by all stampers; nevertheless, many users have devised processes which would be impossible without lancing.

The benefits of lancing are illustrated in Figure 19-8. In this figure two closed-loop holes are instantly lanced each along the C-pillar top (window opening area) and bottom (window regulator area) to improve the formability at those two corners. The right panel of Figure 19-8 is lanced and suffers less thinning compared to the no-lancing case, which is shown in the left-panel. This keyword offers two types of lancing operations:

1. **Instant lancing.** Instant lancing cuts the sheet metal once along the defined curve at a time specified in the AT field.

2. **Progressive lancing.** The cut is spatially divided into NTIMES sub-lances traveling along the curve in the direction of definition. See Figure 19-11. Progressive

lancing starts at AT and ends at ENDT thereby achieving a gradual and even release along the curve.

Modeling Guidelines:

Some modeling guidelines and limitations are listed below:

1. Both closed-loop (Figure 19-9) and open-loop (Figure 19-10) lancing curves are supported. A lancing curve may not cross another lancing curve or itself.

2. Since progressive lancing starts from the beginning of the curve and proceeds towards the end, the direction of the curve needs to be defined to match the direction of the physical cut (Figure 19-11). The direction can be set using LS-PrePost. The menu option GeoTol→ Measure with the Edge box checked can be used to show the direction of the curve. If the direction is not as desired, GeoTol → Rever can be used to reverse the direction.

3. The effect of NTIMES can be seen in Figure 19-12. Compared with NTIMES of 6, setting NTIMES to a value of 20 results in a smoother lancing boundary and less stress concentration along the separated route.

4. Although the IGES format curve is supported by the keyword *DEFINE_CURVE_TRIM_3D, curves defined with the keyword and used for lancing must be specified using only the XYZ format (TCTYPE = 1 or 0) for revisions prior to Revision 110246. The manual entry in the keyword manual for the *INTERFACE_BLANKSIZE_DEVELOPMENT keyword outlines a procedure for converting an IGES file into the required XYZ format. Note the IGES format support for lancing is enabled starting in Revision 110246.

5. The first two points as well as the last two points of any progressive lancing curve must be separated so that LS-DYNA can correctly determine the direction of the curve.

6. The lancing curve needs to be much longer than the element sizes in the lancing area.

7. To prevent mesh distortion at the end of the lancing route ENDT must be defined to be less than the simulation completion time (slightly less is sufficient).

8. Lancing can be in any direction. The direction is determined by *DEFINE_CURVE_TRIM_3D.

9. Tailor-welded blanks are supported; however, the lancing route should not cross the laser line, as currently only one part can be defined with one lancing curve.

10. Both *PARAMETER and *PARAMETER_EXPRESSION are supported for AT and ENDT as of Revision 92335. This makes it possible for users to input distance from punch home as onset of the lancing. Refer to Figure 19-14, where a punch's velocity profile is shown, the lancing activation time AT, is calculated based on the distance to home, "dhome" (the shaded area), punch travel velocity "vdraw", and total simulation time "ENDTIME". The variable CIVD implemented in Rev 110173 removes the need for the calculation of AT from punch distance to home.

11. Mesh adaptivity (*CONTROL_ADAPTIVE and the parameter "ADPOPT" under *PART) must be turned on during lancing.

12. All trim curves used to define lancing routes (*DEFINE_CURVE_TRIM_3D) must be placed after all other curves in the input deck. Furthermore, no curves defined by *DEFINE_CURVE_TRIM_3D that are not used for lancing should be present anywhere in the input deck. Note this restriction is removed starting in Revision 110316.

13. Only *DEFINE_CURVE_TRIM_3D is supported. If defined curve is far away from the blank, it will be projected onto the blank.

14. This keyword is input order sensitive: *DEFINE_CURVE_TRIM_3D must be placed after *ELEMENT_LANCING.

Application example:

A partial deck implementing instant lancing is listed below. A blank having a PID of 8 is being lanced along curves #119 and #202 instantly at 0.05 and 0.051 seconds, respectively.

```
*ELEMENT_LANCING
$     IDPT      IDCV   IREFINE      SMIN        AT      ENDT    NTIMES
         8       119                       0.0500
         8       202                       0.0510
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln     nseed
       119         1         1                0.100         1
$#                cx                cy                cz
           172.99310          42.632320          43.736160
           175.69769         -163.08299          46.547531
           177.46982         -278.03793          49.138161
           186.82404         -303.67191          51.217964
           205.16177         -315.33484          53.299248
               ⋮                 ⋮                 ⋮
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln     nseed
       202         1         1                0.100         1
$#                cx                cy                cz
           187.46982         -578.73793          89.238161
           168.88404         -403.97191          61.417964
           215.18177         -215.03484          73.899248
               ⋮                 ⋮                 ⋮
```

A partial keyword deck implementing two progressive lances is listed below. Both lances travel along paths starting at the same coordinate value. A sheet blank having a part ID of 9 is progressively lanced along both curves (IDCV = 1 and 2) as defined by *DEFINE_-CURVE_TRIM_3D. Both lancing operations commence at 0.05 seconds and finish at 0.053 seconds with 20 cuts along each curve in opposite directions. The lancing results are shown in Figure 19-13. Note that the termination time is 53.875 seconds, which is slightly larger than the ENDT.

```
*ELEMENT_LANCING
$     IDPT      IDCV   IREFINE      SMIN        AT      ENDT    NTIMES
         9         1                       0.0500   5.3E-02        20
         9         2                       0.0500   5.3E-02        20
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln     nseed
         1         1         1                0.100         1
$#                cx                cy                cz
           172.99310          42.632320          43.736160
           175.69769        -163.08299           46.547531
           177.46982        -278.03793           49.138161
           186.82404        -303.67191           51.217964
           205.16177        -315.33484           53.299248
           223.13152        -308.03534           54.193089
           234.96263        -290.49695           54.885273
           222.03900        -270.08289           53.163551
           199.31226        -251.27985           50.401234
               ⋮                 ⋮                   ⋮
*DEFINE_CURVE_TRIM_3D
         2         1         1                0.100         1
$#                cx                cy                cz
           172.99310          42.632320          43.736160
           171.33121           47.22141          42.513367
           171.28690          128.84601          43.032799
           176.89932          149.39539          43.495331
           192.41418          159.53757          44.756699
           208.39861          158.93469          45.878036
           218.10101          149.34409          47.128345
           218.34503          135.23810          47.682144
           209.05414          122.82422          46.616959
           190.19659          117.66074          44.858204
               ⋮                 ⋮                   ⋮
```

Trimming after lancing:

As shown in Figure 19-15 and the following partial keyword file, the lanced scraps can be removed (trimming) after a lancing simulation. An extra keyword, *DEFINE_-LANCE_SEED_POINT_COORDINATES is needed to define the portion that would remain after the lancing and trimming. It should be obvious that the lancing curve defined by *DEFINE_CURVE_TRIM_3D must form a closed loop. The following example will trim a part ID 9 with a fully enclosed lancing curve #1, at time = 0.049 seconds. Since the termination time is 0.0525 seconds, the scrap will be deleted (trimmed off) before the simulation ends. The scrap, enclosed by the curve, is located outside of the seed node, defined by *DEFINE_LANCE_SEED_POINT_COORDINATES (starting in Revision 107262).

```
*CONTROL_TERMINATION
0.0525
*ELEMENT_LANCING
$     IDPT      IDCV   IREFINE      SMIN        AT      ENDT    NTIMES
         9         1                       0.0490
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln     nseed
         1         1         1                0.100         1
$#                cx                cy                cz
           172.99310          42.632320          43.736160
           175.69769         -163.08299          46.547531
           177.46982         -278.03793          49.138161
           186.82404         -303.67191          51.217964
           205.16177         -315.33484          53.299248
           223.13152         -308.03534          54.193089
               ⋮                 ⋮                 ⋮
           172.99310          42.632320          43.736160
           172.99310          42.632320          43.736160
*DEFINE_LANCE_SEED_POINT_COORDINATES
$    NSEED        X1        Y1         Z1        X2        Y2        Z2
         1    -289.4    98.13   2354.679
```

This feature also makes it possible to combine the trimming process with a forming simulation, saving a trimming step in a line-die process simulation by skipping writing out a formed dynain file and reading in the same file for the trimming simulation.

Lance-trimming with a part set, IGES curve, a seed node and CIVD:

The following partial keyword input shows instant lance-trimming across the weld line of a tailor-welded blank using the part set ID "blksid" 100, which consists of PIDs 1 and 9. The part set ID used for *ELEMENT_LANCING input is "idpt", which is set as the negative of blksid (-100). The lance-trimming curve ID 1117 is defined using the file lance4.iges in IGES format (TCTYPE = 2). The defined variable CIVD refers to the load curve ID 1115 under *DEFINE_CURVE, which is the kinematic curve for the upper die. The lancing starts at 15.5 mm (AT = 15.5) away from upper die bottom position. A lance seed coordinate (-382.0, -17.0, 76.0) is defined using the keyword *DEFINE_LANCE_-SEED_POINT_COORDINATES, resulting in the lanced scrap piece being removed after lancing.

```
*PARAMETER
I blk1pid          1
I blk2pid          9
I blksid         100
*SET_PART_LIST
&blksid
&blk1pid    &blk2pid
*PARAMETER_EXPRESSION
I idpt        -1*blksid
*ELEMENT_LANCING
$     IDPT      IDCV   IREFINE      SMIN        AT      ENDT    NTIMES      CIVD
     &idpt      1117         1                15.5                          1115
*DEFINE_CURVE_TRIM_3D
$#    tcid    tctype      tflg      tdir     tctol      toln    nseed1    nseed2
      1117         2         1         0    0.1000         0
lance4.iges
*DEFINE_LANCE_SEED_POINT_COORDINATES
$     NSUM        X1        Y1        Z1        X2        Y2        Z2
```

```
         1  -382.000   -17.000      76.0
*BOUNDARY_PRESCRIBED_MOTION_RIGID
$    TYPEID       DOF       VAD      LCID        SF       VID     DEATH     BIRTH
  &udiepid         3         0      1115
  &bindpid         3         0      1116
*DEFINE_CURVE
1115
0.0, 0.0
0.001, 1000.0
0.211, 1000.0
```

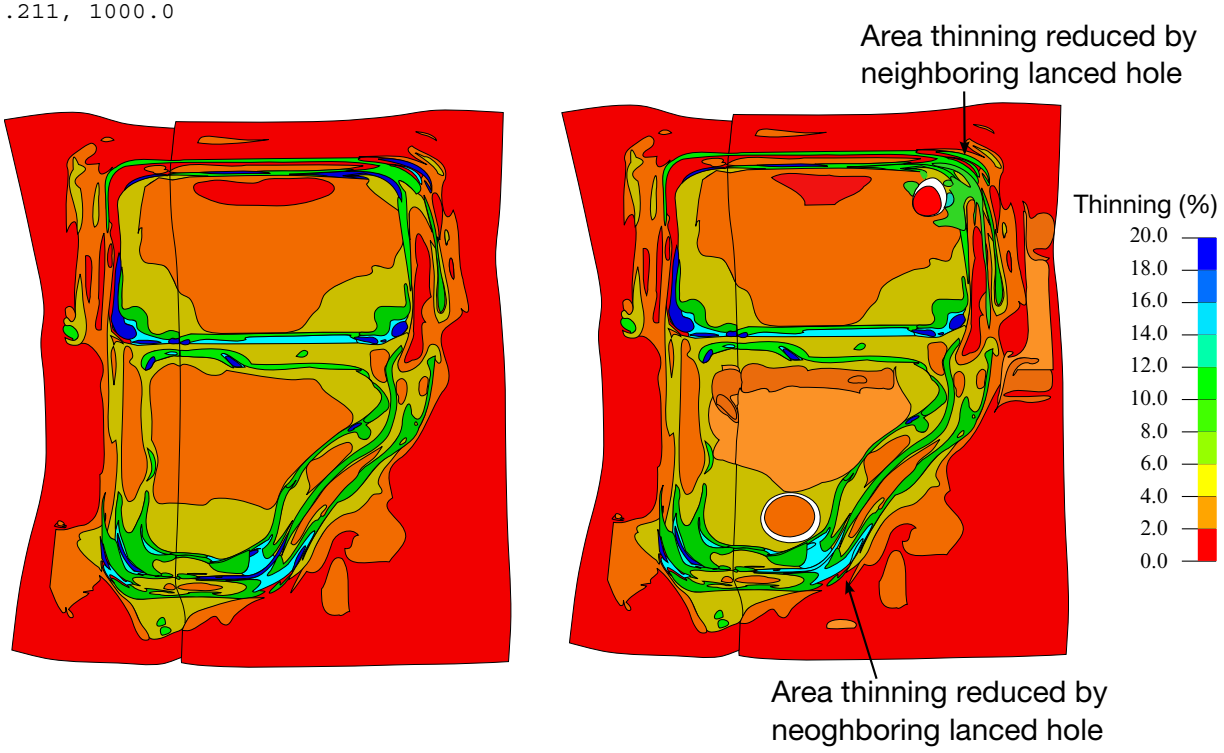


Figure 19-8. Thinning improvement on a door as a result of lancing at the upper and lower corner of the C-pillar.

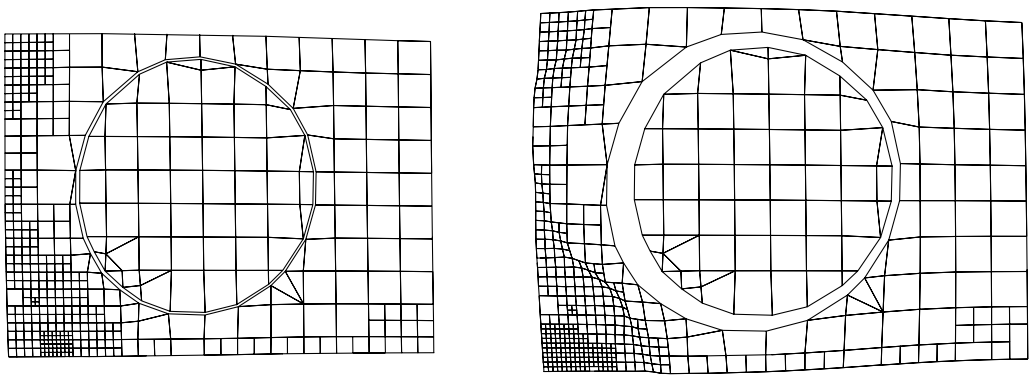


Figure 19-9. Instant lancing – closed-loop hole. The left mesh immediately after lancing at time AT while the right one when the punch is at punch home.

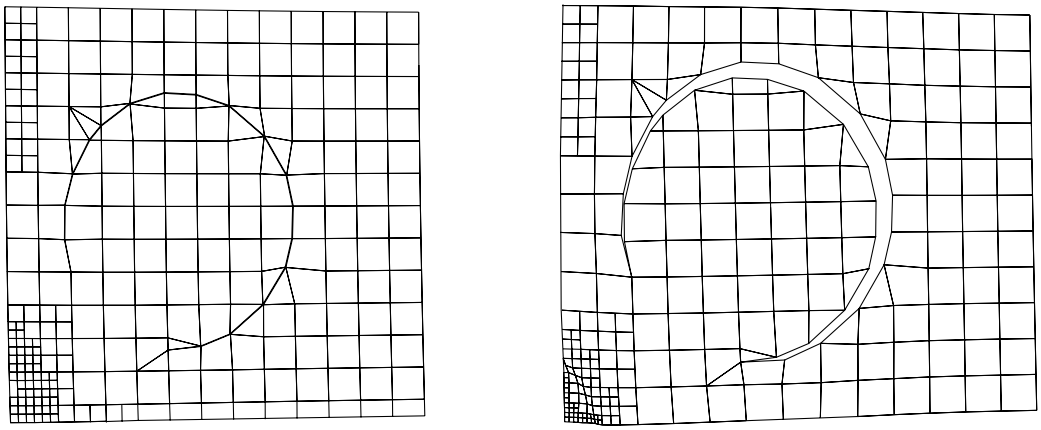


Figure 19-10. Instant lancing – open-loop hole. The left mesh immediately after lancing at time AT while the right one is when the punch is at punch home.

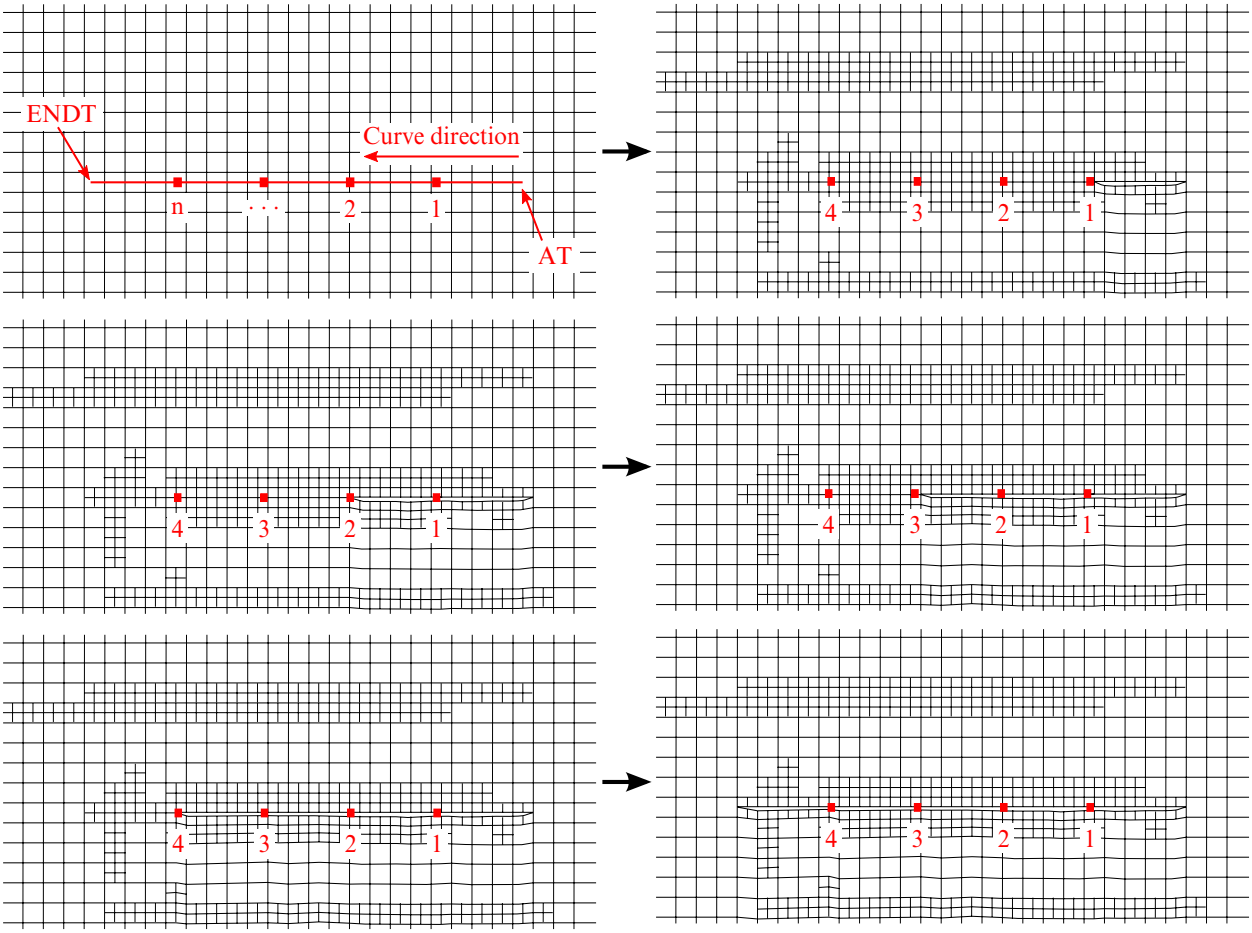


Figure 19-11. Progressive lancing - defining *AT*, *ENDT*, *NTIMES*, curve direction; mesh separation progression during progressive lancing.

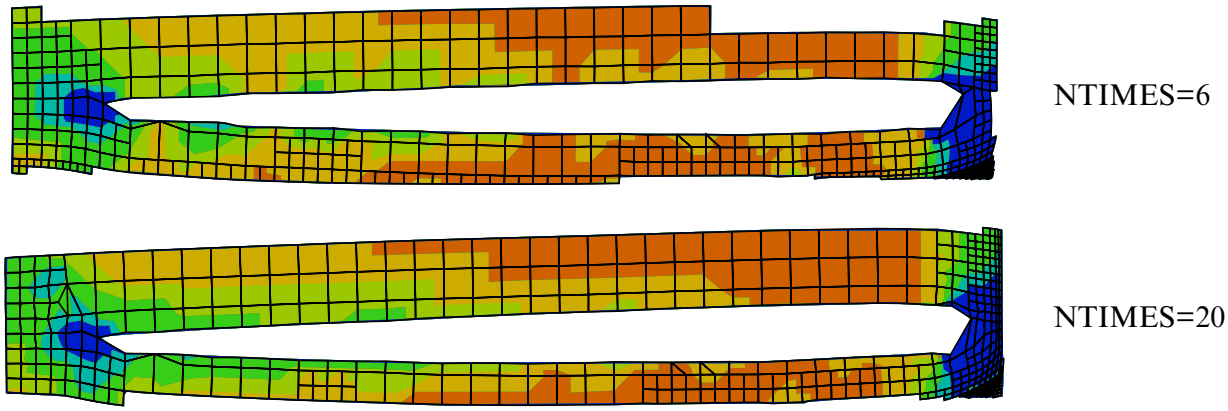


Figure 19-12. Larger NTIMES causes a smoother lancing boundary and less stress concentration.

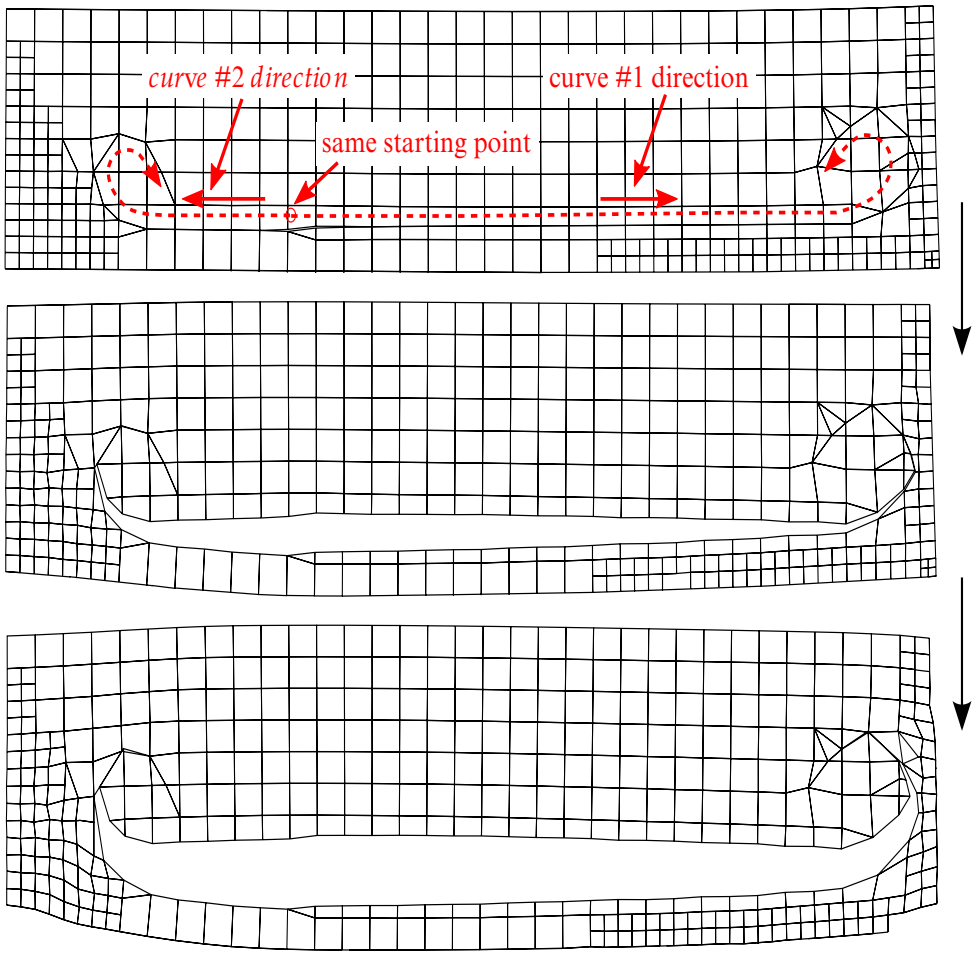
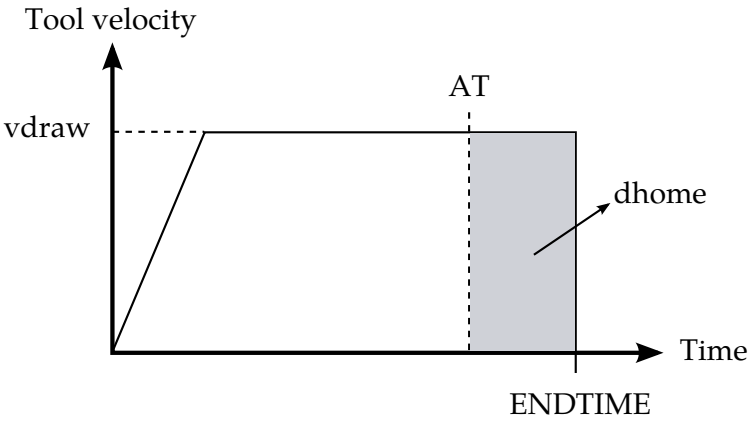


Figure 19-13. Progressive lancing – multiple lancing processes starting from the same coordinates.




```
*PARAMETER_EXPRESSION
R dhome      12.5
R at         ENDTIME-dhome/vdraw
*ELEMENT_LANCING
$ IDPT       IDCV       IREFINE       SMIN       AT
      9        112                                &at
```

Figure 19-14. An example of defining lancing activation time AT using tool's distance to home.

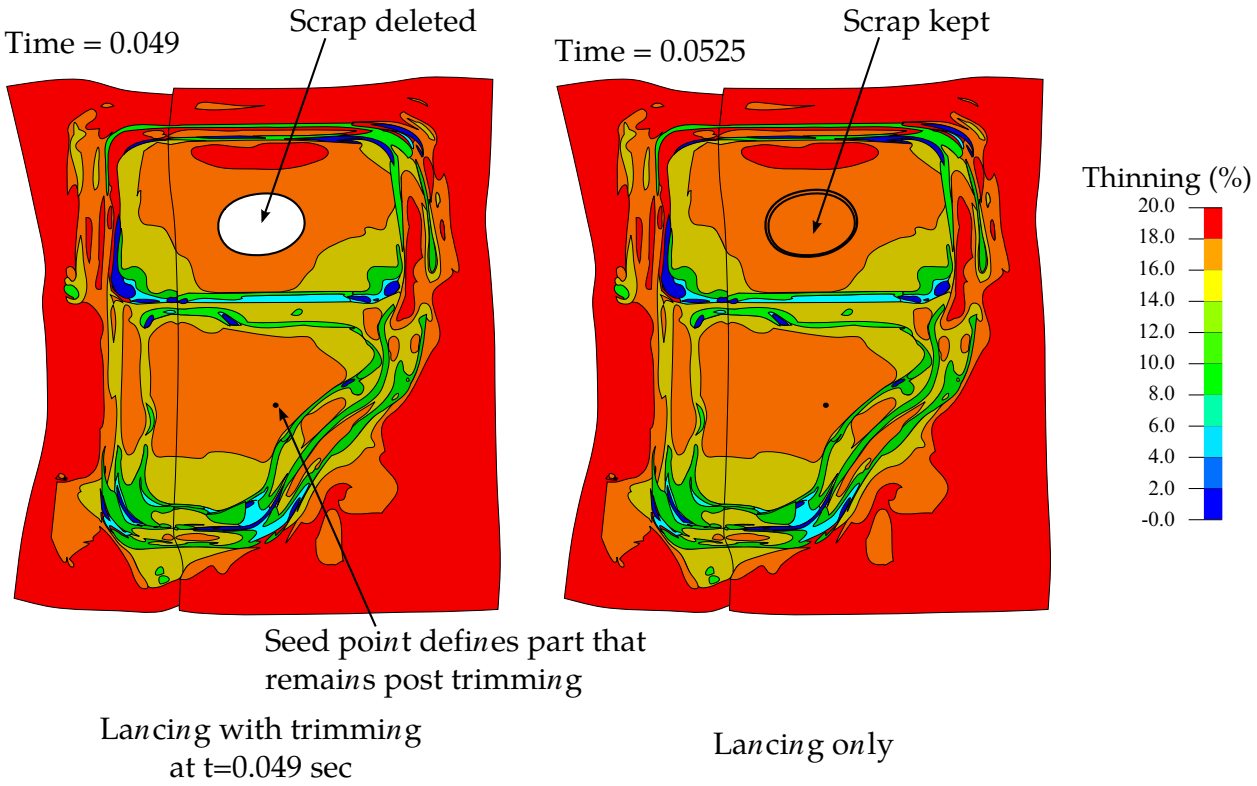
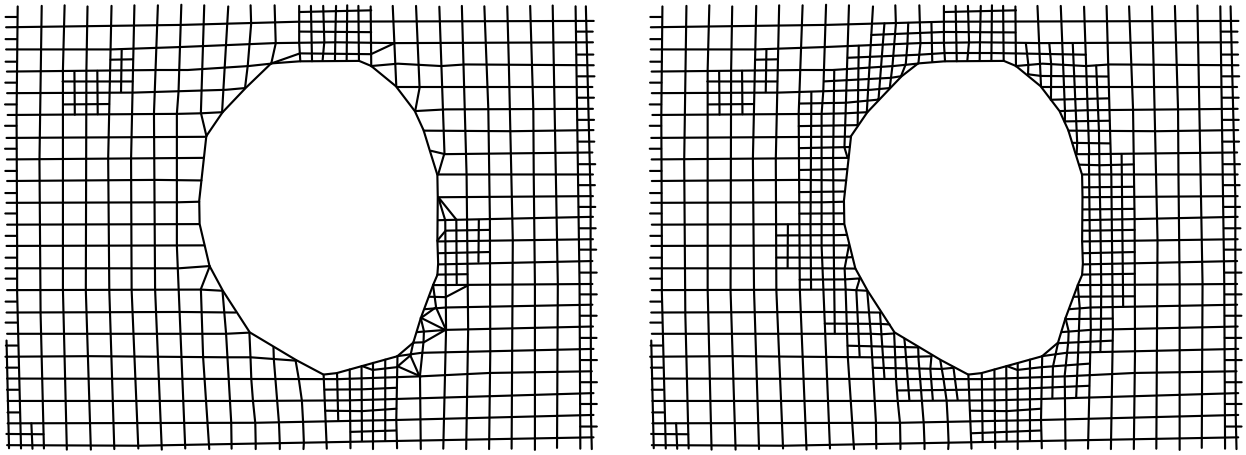


Figure 19-15. Lancing with trimming



Lanced mesh prior to Revision 107708

Improved lanced boundary mesh with IREFINE=1 after Revision 107708

Figure 19-16. Set IREFINE = 1 (Revision 107708 and later) for improved lanced boundary.

***ELEMENT_MASS**_*{OPTION}*

Available options include:

<BLANK>

NODE_SET

Purpose: Define a lumped mass element assigned to a nodal point or equally distributed to the nodes of a node set. The nodal point can belong to either a deformable body or a rigid body.

Card	1	2	3	4	5	6	7	8	9	10
Variable	EID	ID	MASS		PID					
Type	I	I	F		I					
Default	none	none	0.		0					

VARIABLE	DESCRIPTION
EID	Element ID. A unique number is recommended. The nodes in a node set share the same element ID.
ID	Node ID or node set ID if the NODE_SET option is active. This is the node or node set to which the mass is assigned.
MASS	Mass value. When the NODE_SET option is active, the mass is equally distributed to all nodes in a node set.
PID	Part ID. This input is optional.

Remarks:

1. **Kinetic Energy Output.** Kinetic energy of lumped mass elements is output as kinetic energy of part 0 in matsum (*DATABASE_MATSUM) if IERODE is set to 1 on *CONTROL_OUTPUT.

*ELEMENT_MASS_MATRIX_{OPTION}

Available options include:

<BLANK>

NODE_SET

Purpose: Define a 6 × 6 symmetric nodal mass matrix assigned to a nodal point or each node within a node set. A node may not be included in more than one *ELEMENT_MASS_MATRIX(_NODE_SET) command. The nodal point can only belong to a deformable body. This keyword cannot be applied to nodes that belong to rigid bodies.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	ID	CID					
Type	I	I	I					
Default	none	none	0					

Card 2	1	2	3	4	5	6	7	8
Variable	M11	M21	M22	M31	M32	M33	M41	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 3	1	2	3	4	5	6	7	8
Variable	M42	M43	M44	M51	M52	M53	M54	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 4	1	2	3	4	5	6	7	8
Variable	M55	M61	M62	M63	M64	M65	M66	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
EID	Element ID. A unique number is recommended. The nodes in a node set share the same element ID.
ID	Node ID or node set ID if the NODE_SET option is active. This is the node or node set to which the mass is assigned.
CID	Local coordinate ID which defines the orientation of the mass matrix
M*ij*	The ij^{th} term of the symmetric mass matrix. The lower triangular part of the matrix is defined.

*ELEMENT_MASS_PART_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Define additional non-structural mass to be distributed by an area (shell) / volume (solid) or mass weighted distribution to all nodes of a given part, or part set, ID. As an option, the total mass can be defined and the additional non-structural mass is computed. This option applies to all part IDs defined by shell, tshell and solid elements.

Card	1	2	3	4	5	6	7	8	9	10
Variable	ID	ADDMASS		FINMASS		LCID		MWD		
Type	I	F		F		I		I		
Default	none	0.		0.		0		0		

VARIABLE	DESCRIPTION
ID	Part or part set ID if the SET option is active. A unique number must be used.
ADDMASS	Added translational mass to be distributed to the nodes of the part ID or part set ID. Set to zero if FINMASS is nonzero. Since the additional mass is not included in the time step calculation of the elements in the PID or SID, ADDMASS must be greater than zero if FINMASS is zero. Multiple applications of mass using the SET option are treated by the additional masses being summed. For multiple mass applications to the same PID (<BLANK> option), only the last definition is used.
FINMASS	Final translational mass of the part ID or part set ID. The total mass of the PID or SID is computed and subtracted from the final mass of the part or part set to obtain the added translational mass, which must exceed zero. Set FINMASS to zero if ADDMASS is nonzero. FINMASS is available in the R3 release of version 971.
LCID	Optional load curve ID to scale the added mass at time = 0. This curve defines the scale factor as a function versus time. The curve must start at unity at $t = 0$. This option applies to deformable

VARIABLE	DESCRIPTION
	bodies only.
MWD	Optional flag for mass-weighted distribution, valid for SET option only: EQ.0: non-structural mass is distributed by area (shell) / volume (solid) weighted distribution, EQ.1: non-structural mass is distributed by mass weighted, area × density × thickness (shell) / volume × density (solid), distribution. Mixed uses with MWD for the same part should be avoided.

*ELEMENT_PLOTEL

Purpose: Define a null beam element for visualization.

Card	1	2	3	4	5	6	7	8	9	10
Variable	EID	N1	N2							
Type	I	I	I							
Default	none	none	none							

VARIABLE	DESCRIPTION
EID	Element ID. A unique number must be used.
N1	Nodal point (end) 1.
N2	Nodal point (end) 2.

Remarks:

1. **Part ID.** Part ID, 10000000, is assigned to PLOTEL elements.
2. **Beam Element IDs.** PLOTEL element IDs must be unique with respect to other beam elements.

*ELEMENT_SEATBELT

Purpose: Define a seat belt element.

Card	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	SBRID	SLEN		N3	N4	
Type	I	I	I	I	I	F		I	I	
Default	none	none	none	none	Rem 2	0.0		0	0	

VARIABLE	DESCRIPTION
EID	Element ID. A unique number is required. Since null beams are created for visualization, this element ID should not be identical to element IDs defined for *ELEMENT_BEAM and *ELEMENT_DISCRETE.
PID	Part ID
N1	Node 1 ID
N2	Node 2 ID
SBRID	Retractor ID, see *ELEMENT_SEATBELT_RETRACTOR.
SLEN	Initial slack length. See Remarks 1 and 3.
N3	Optional node 3 ID. When N3 > 0 and N4 > 0, the element becomes a shell seat belt element. The thickness of the shell seatbelt is defined in *SECTION_SHELL, not in *SECTION_SEATBELT. The shell-type seatbelt must be of a rectangular shape as shown in Figure 19-17 and contained in a logically regular mesh. See Remark 4.
N4	Node 4 ID, which is required if and only if N3 is defined.

Remarks:

1. **Unstretched Length.** The unstretched length, l, of the belt is given as

$$l = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} + \text{SLEN} \ ,$$

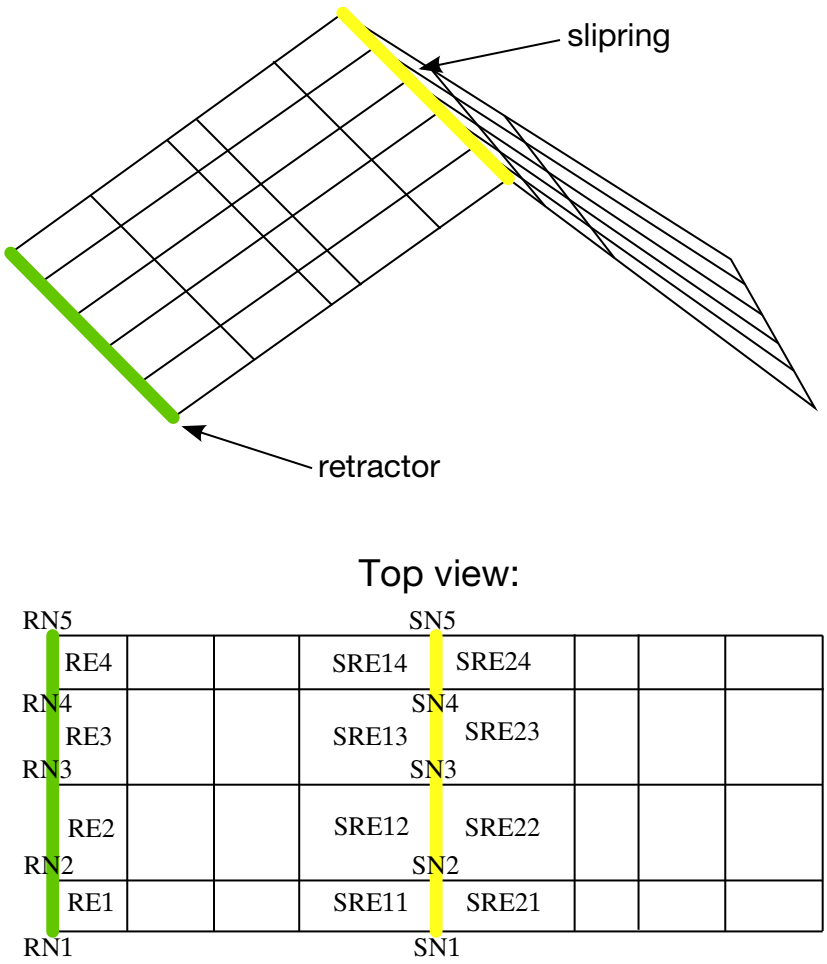


Figure 19-17. Definition of seatbelt shell elements. The ordering of the nodes and elements are important for seatbelt shells. See the input descriptions for SECTION_SHELL, ELEMENT_SEATBELT_RETRACTOR and ELEMENT_-SEATBELT_SLIPRING.

where (x_1, y_1, z_1) and (x_2, y_2, z_2) are the positions of the nodes defining the seatbelt element and SLEN is the slack length. All seatbelt elements *must* have an unstretched length *greater than* LMIN, where LMIN is a minimum length field defined on *MAT_SEATBELT. *However, when a seatbelt element is either initially connected to a slipring or is a mouth element for a retractor (see *ELEMENT_SEATBELT_SLIPRING and *ELEMENT_SEATBELT_RETRACTOR), l must be greater than 1.6 × LMIN. Also, if a seatbelt element has the potential of going through a slipring or retractor at some time during the simulation, it must have an unstretched length of at least 1.1 × LMIN.*

2. **Retractor.** The retractor ID should be defined only if the element is initially *inside* a retractor, see *ELEMENT_SEATBELT_RETRACTOR.

3. **Forces.** Belt elements are single degree of freedom elements connecting two nodes. When the strain in an element is positive (that is, the current length is

greater than the unstretched length), a tension force is calculated from the material characteristics and is applied along the current axis of the element to oppose further stretching.

4. **Shell Elements.** Seatbelt shell elements are included as a feature as of version 971 and must be used with caution. The seatbelt shells distribute the loading on the surface of the dummy more realistically than the two node belt elements. For the seatbelt shells to work with sliprings and retractors it is necessary to use a logically regular mesh of quadrilateral elements. *A seatbelt defined by a part ID must not be disjoint.*

5. **Material ID.** 1D and 2D seatbelt elements may not share the same material ID.

*ELEMENT_SEATBELT_ACCELEROMETER

Purpose: Define an accelerometer. Contrary to the keyword name, an accelerometer need not be associated with a seat belt. The accelerometer is fixed to a rigid body containing the three nodes defined below. An accelerometer will exhibit considerably less numerical noise than a deformable node, thereby reporting more meaningful data to the user. Whenever computed accelerations are compared to experimental data, or whenever computed accelerations are compared between different runs, this feature is essential.

Card	1	2	3	4	5	6	7	8
Variable	SBACID	NID1	NID2	NID3	IGRAV	INTOPT	MASS	
Type	I	I	I	I	I	I	F	
Default	none	none	none	none	0	0	0.	

VARIABLE	DESCRIPTION
SBACID	Accelerometer ID. A unique number must be used.
NID1	Node 1 ID
NID2	Node 2 ID
NID3	Node 3 ID
IGRAV	Gravitational accelerations due to body force loads: EQ.-6: z and x components removed from acceleration output EQ.-5: y and z components removed from acceleration output EQ.-4: x and y components removed from acceleration output EQ.-3: z component removed from acceleration output EQ.-2: y component removed from acceleration output EQ.-1: x component removed from acceleration output EQ.0: all components included in acceleration output EQ.1: all components removed from acceleration output GT.1: IGRAV is a curve ID defining the gravitation-flag versus time. The ordinate values, representing the gravitation-flag, can be -6, -5, -4, -3, -2, -1, 0 or 1, as described above.

VARIABLE	DESCRIPTION
	For example, a curve with 4 data points of (0.,1), (10.,1), (10.000001,0), (200.,0) sets the gravitation flag to be 1 when time ≤ 10, and 0 when time > 10. In other words, all components of gravitational accelerations are removed when time ≤ 10., and then included when time > 10.0.
INTOPT	Integration option. If the accelerometer undergoes rigid body translation without rotation, this option has no effect; however, if rotation occurs, INTOPT affects how translational velocities (and displacements) are calculated. Note that the acceleration values written to the nodout file are unaffected by INTOPT. EQ.0: velocities are integrated from the global accelerations and transformed into the local system of the accelerometer. EQ.1: velocities are integrated directly from the local accelerations of the accelerometer.
MASS	Optional added mass for accelerometer. This mass is equally distributed to nodal points NID1, NID2, and NID3. This option avoids the need to use the *ELEMENT_MASS keyword input if additional mass is required.

Remarks:

The presence of the accelerometer means that the accelerations and velocities of node 1 will be output to **all** output files in local instead of global coordinates.

The local coordinate system is defined by the three nodes as follows:

1. local **x** from node 1 to node 2,
2. local **z** perpendicular to the plane containing nodes 1, 2, and 3 ($\mathbf{z} = \mathbf{x} \times \mathbf{a}$), where **a** is from node 1 to node 3,
3. local $\mathbf{y} = \mathbf{z} \times \mathbf{x}$.

The three nodes should all be part of the same rigid body. The local axis then rotates with the body.

*ELEMENT_SEATBELT_PRETENSIONER

Purpose: Define seat belt pretensioner. A combination with sensors and retractors is also possible.

Card 1	1	2	3	4	5	6	7	8
Variable	SBPRID	SBPRTY	SBSID1	SBSID2	SBSID3	SBSID4		
Type	I	I	I	I	I	I		
Default	none	none	0	0	0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	SBRID	TIME	PTLCID	LMTFRC	LMTPIN			
Type	I	F	I	F	F			
Default	0	0.0	0	0	0			

VARIABLE	DESCRIPTION
SBPRID	Pretensioner ID. A unique number has to be used.
SBPRTY	Pretensioner type (see Activation): EQ.1: Pyrotechnic retractor with force limits (see Type 1), EQ.2: Pre-loaded spring becomes active (see Types 2 and 3), EQ.3: Lock spring removed (see Types 2 and 3), EQ.4: Force as a function of time retractor (see Type 4). EQ.5: Pyrotechnic retractor (older implementation of type 1) but with optional force limiter, LMTFRC (see Type 1 and Type 5). EQ.6: Combination of types 4 and 5 as described in Type 6 below. EQ.7: Independent pretensioner/retractor (see Type 7).

VARIABLE	DESCRIPTION
	EQ.8: Energy as a function of time retractor pretensioner with optional force limiter, LMTFRC (see Type 8). EQ.9: Energy as a function of time buckle or anchor pretensioner (see Type 9).
SBSID1	Sensor 1, see *ELEMENT_SEATBELT_SENSOR. See Activation.
SBSID2	Sensor 2, see *ELEMENT_SEATBELT_SENSOR.
SBSID3	Sensor 3, see *ELEMENT_SEATBELT_SENSOR.
SBSID4	Sensor 4, see *ELEMENT_SEATBELT_SENSOR.
SBRID	Retractor number (SBPRTY = 1, 4, 5, 6, 7 or 8) or spring element number (SBPRTY = 2, 3 or 9).
TIME	Time between sensor triggering and pretensioner acting.
PTLCID	Load curve for pretensioner (Time after activation, Pull-in) (SBPRTY = 1, 4, 5, 6, 7, 8 or 9).
LMTFRC	Optional limiting force for pretensioner types 5 and 8. If zero, this option is ignored. See Type 5 and Type 8.
LMTPIN	Optional limiting pull-in for pretensioner types 4, 6, and 7. LMTPIN must be ≥ 0.0. If zero, this option is ignored. See Type 4, Type 6, and Type 7.

Activation:

To activate the pretensioner, the following sequence of events must occur:

1. Any one of up to four sensors must be triggered.
2. Then a user-defined time delay occurs.
3. Then the pretensioner acts.

At least one sensor should be defined.

Pretensioners allow modeling of seven types of active devices that tighten the belt during the initial stages of a crash. Types 1 and 5 implement a pyrotechnic device that spins the spool of a retractor, reeling in the belt. You define pull-in as a function of time curve that

applies once the pretensioner activates. Types 2 and 3 implement preloaded springs or torsion bars that move the buckle when released.

Types 2 and 3:

The pretensioner is associated with any type of spring element, including rotational. Note that the preloaded spring, locking spring, and any restraints on the motion of the associated nodes are defined in the normal way; the action of the pretensioner is merely to cancel the force in one spring until (or after) it fires. With the second type, the force in the spring element is canceled out until the pretensioner is activated. In this case, the spring in question is normally a stiff, linear spring that acts as a locking mechanism, preventing motion of the seat belt buckle relative to the vehicle. A preloaded spring is defined in parallel with the locking spring. This type avoids the problem of the buckle being free to "drift" before the pretensioner is activated. Types 4, 6, and 7 (force types) are described below.

Type 1:

The older implementation of the type 1 pretensioner (essentially available as type 5, but type 5 restricts the force value) requires a load curve that tabulates the pull-in of the pretensioner as a function of time. This pretensioner type interacts with the retractor, forcing it to pull in by the indicated amount of belt. It works well and does exactly what it says it will, but it can be difficult to use because it has no regard for the forces applied to the belt. If a pull-in of 20 mm is specified at a particular time, then 20 mm of belt will be pulled in, even if this results in unrealistic forces in the seatbelt. Furthermore, there was no explicit way to turn this pretensioner off. Once defined, it completely overrode the retractor, and the amount of belt passing into or out of the retractor depended solely on the specified load curve.

The behavior of the type 1 pretensioner was modified in response to user feedback about these shortcomings. Each retractor has a loading (and optional unloading) curve that describes the force on the belt element as a function of the amount of belt pulled out of the retractor since the retractor locked. The current type 1 pretensioner acts as a shift on this retractor load curve. An example will make this clear. Suppose that at a particular time, 5 mm of belt material has left the retractor. The retractor will respond with a force corresponding to 5 mm pull-out on its loading curve. But suppose this retractor has a type 1 pretensioner defined, and at this instant of time, the pretensioner specifies a pull-in of 20 mm. The retractor will then respond with a force that corresponds to (5mm + 20mm) on its loading curve. This results in a much larger force. The effect can be that belt material is pulled in, but unlike with the older implementation, there is no guarantee. The benefit of this implementation is that the force as a function of pull-in load curve for the retractor is followed, and no unrealistic forces are generated. Still, it may be difficult to produce realistic models using this option, so other pretensioners have been added.

Type 4:

The type 4 pretensioner requires a force as a function of time curve (see Figure 19-18). At each time step, the retractor computes the desired force without regard to the pretensioner. If the resulting force is less than that specified by the pretensioner load curve, then the pretensioner value is used instead. As time goes on, the pretensioner load curve should drop below the forces generated by the retractor, and the pretensioner is then essentially inactive. If LMTPIN > 0, the pretensioner will be disabled if the pull-in is > LMTPIN. This provides for good control of the actual forces, so no unrealistic values are generated. The actual direction and amount of belt movement are unspecified and depend on the other forces being exerted on the belt. This is suitable when the force the pretensioner exerts over time is known.

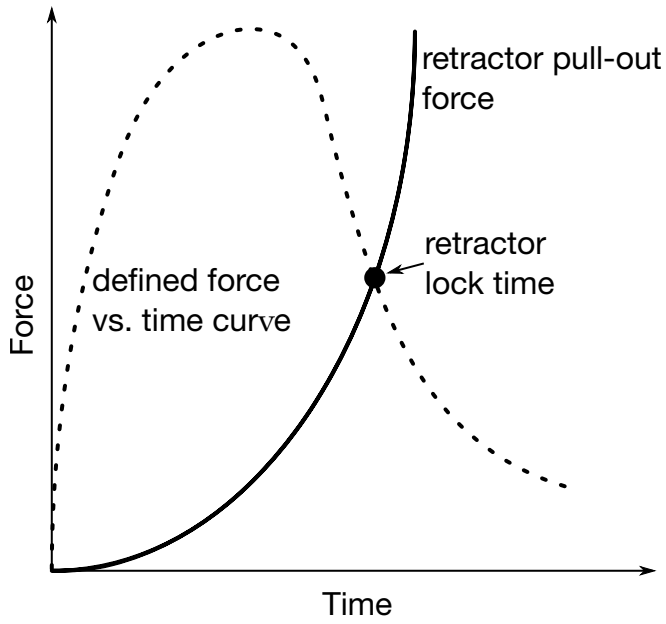


Figure 19-18. Force as a function of time for the pretensioner. At the intersection, the retractor locks.

Type 5:

The type 5 pretensioner is essentially the same as the old type 1 pretensioner, but with the addition of a force-limiting value. The pull-in is given as a function of time, and the belt is drawn into the retractor exactly as desired. However, if at any point the forces generated in the belt exceed the pretensioner force limit, then the pretensioner is deactivated, and the retractor takes over. In order to prevent a large discontinuity in the force at this point, the loading curve for the retractor is shifted (in the abscissa) by the amount required to put the current (pull-out,force) on the load curve. For example, suppose the current force is 1000, and the current pull-out is -10 (10 mm of belt has been *pulled in* by the pretensioner). If the retractor would normally generate a force of 1000 after 25 mm of

belt had been *pulled out*, then the load curve is shifted to the left by 35 and remains that way for the duration of the calculation. So that at the current pull-in of 10, it will generate the force normally associated with a pull-out of 25. If the belt reaches a pull-out of 5, the force will be as if it were pulled out 40 (5 + the shift of 35), and so on. This option is available for those who liked the general behavior of the old type 1 pretensioner, but it includes a force limit to prevent unrealistic behavior.

Note that the retractor will be locked if either of the following two conditions is met:

- The current time exceeds the maximum abscissa of PTLCID, or
- LMTFRC is reached.

Type 6:

The type 6 pretensioner is a variation of the type 4 pretensioner, with features of the type 5 pretensioner. A force as a function of time curve is input, and the pretensioner force is computed each cycle. The retractor linked to this pretensioner should specify a positive value for pull, which is the distance the belt pulls out before it locks. As the pretensioner pulls the belt into the retractor, the amount of pull-in is tracked. As the pretensioner force decreases and drops below the belt tension, belt begins to move back out of the retractor. If LMTPIN > 0, the pretensioner will be disabled if the pull-in is > LMTPIN. Once pull amount of belt has moved out of the retractor (relative to the maximum pull in encountered), the retractor will lock. At this time, the pretensioner is disabled, and the retractor force curve is shifted to match the current belt tension. This shifting is done just like the type 5 pretensioner. It is important to specify a positive pull value to prevent premature retractor locking, which could occur due to small outward belt movements caused by noise in the simulation.

Type 7:

The type 7 pretensioner is a simple combination of retractor and pretensioner. It is similar to the type 6 pretensioner except for the following changes:

1. When the retractor locks, the pretensioner is *not* disabled – it continues to exert force according to the force as a function of time curve until the end of the simulation. (The force as a function of time curve should probably drop to 0 at some time.)

2. The retractor load curve is not shifted – the retractor begins to exert force according to the force as a function of pull-out curve. These two forces are added together and applied to the belt. Thus, the pretensioner and retractor are essentially independent.

Type 8:

The type 8 pretensioner is a variation of the type 5 pretensioner. The pretension energy, instead of pull-in for type 5, is given as a function of time. This enables users to use a single pretensioner curve, PTLCID, for various sizes of dummies. The energy could be yielded from the baseline test or simulation by

$$E(t) = \int_0^t f \mathrm{d}p \ ,$$

where f is the force of the mouth element of the retractor and dp is the incremental pull-in.

Note that the retractor will be locked if either of the following two conditions are met:

- The current time exceeds the maximum abscissa of PTLCID, or
- LMTFRC is reached.

Type 9:

The type 9 pretensioner is designed for a pretension-energy-based buckle or anchor pretensioner. The pretensioner is modeled as a spring element, SBRID. One end of the spring element is attached to the vehicle. For a buckle pretensioner, the other end of SBRID is the slip ring node, SBRNID, of a slip ring representing the buckle. For an anchor pretensioner, SBRID shares the other end with a belt element; see Figure 19-19.

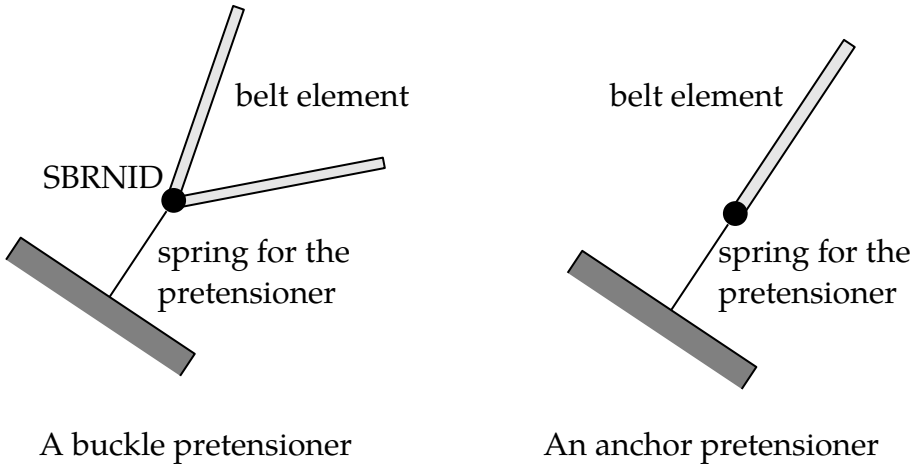


Figure 19-19. Schematics of different kinds of pretensioners.

*ELEMENT_SEATBELT_RETRACTOR

Purpose: Define seat belt retractor. See Remark 4 below for seat belt shell elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SBRID	SBRNID	SBID	SID1	SID2	SID3	SID4	DSID
Type	I	I	I	I	I	I	I	I
Default	none	none	none	0	0	0	0	0
Remarks		3, 4	4	5				

Card 2	1	2	3	4	5	6	7	8
Variable	TDEL	PULL	LLCID	ULCID	LFED	LCFL	FLOPT	
Type	F	F	I	I	F	I	I	
Default	0.0	0.0	0	0	0.0	0	0	
Remarks			6, 8	7, 8	2, 1	11		

VARIABLE	DESCRIPTION
SBRID	Retractor ID. A unique number has to be used.
SBRNID	Retractor node ID
SBID	Seat belt element ID
SID1	Sensor ID 1. This ID refers to a *ELEMENT_SEATBELT_SENSOR, not a *SENSOR.
SID2	Sensor ID 2. This ID refers to a *ELEMENT_SEATBELT_SENSOR, not a *SENSOR.
SID3	Sensor ID 3. This ID refers to a *ELEMENT_SEATBELT_SENSOR, not a *SENSOR.

VARIABLE	DESCRIPTION
SID4	Sensor ID 4. This ID refers to a *ELEMENT_SEATBELT_SENSOR, not a *SENSOR.
DSID	Retractor deactivation sensor. This ID refers to a *ELEMENT_SEATBELT_SENSOR, not a *SENSOR.
TDEL	Time delay after the sensor triggers
PULL	Amount of pull-out between the time delay ending and the retractor locking, a length value.
LLCID	Load curve for loading (Pull-out, Force); see Figure 19-20.
ULCID	Load curve for unloading (Pull-out, Force); see Figure 19-20.
LFED	Fed length
LCFL	Curve representing an adaptive multi-level load limiter (see Remark 10). The abscissa is the ID of a *SENSOR_SWITCH, and the ordinate is the corresponding force limit when the sensor switch meets the switch condition. For example, a curve of two data points (100, 4000.) and (200, 3000.) has a load limit of 4000 from switch 100 and a load limit of 3000 from switch 200. The setting of FLOPT determines how LS-DYNA uses these pairs to determine the load limit. Alternatively, a more general adaptive load limiter can be created using *DEFINE_CURVE_FUNCTION. With this method, LCFL refers to a *DEFINE_CURVE_FUNCTION with the SENSORD option used in the function. The SENSORD function must have the value of n set such that it refers to a *SENSOR_DEFINE_FUNCTION. Then, through the *SENSOR_DEFINE_FUNCTION, the flexible *DEFINE_FUNCTION determines the adaptive load limiter. See Remark 11 for an example. FLOPT is not used in this case and should be left as zero.
FLOPT	Flag giving the algorithm for determining the limiting force from the data points of curve LCFL. EQ.0: Check the status of the sensor switches based on the sequence of data points. For example, this algorithm checks the switch of the 1st data point first. If its switch condition is met, the active force limit is the force of the 1st data point. That force level stays active until the 2nd switch meets the switch condition.

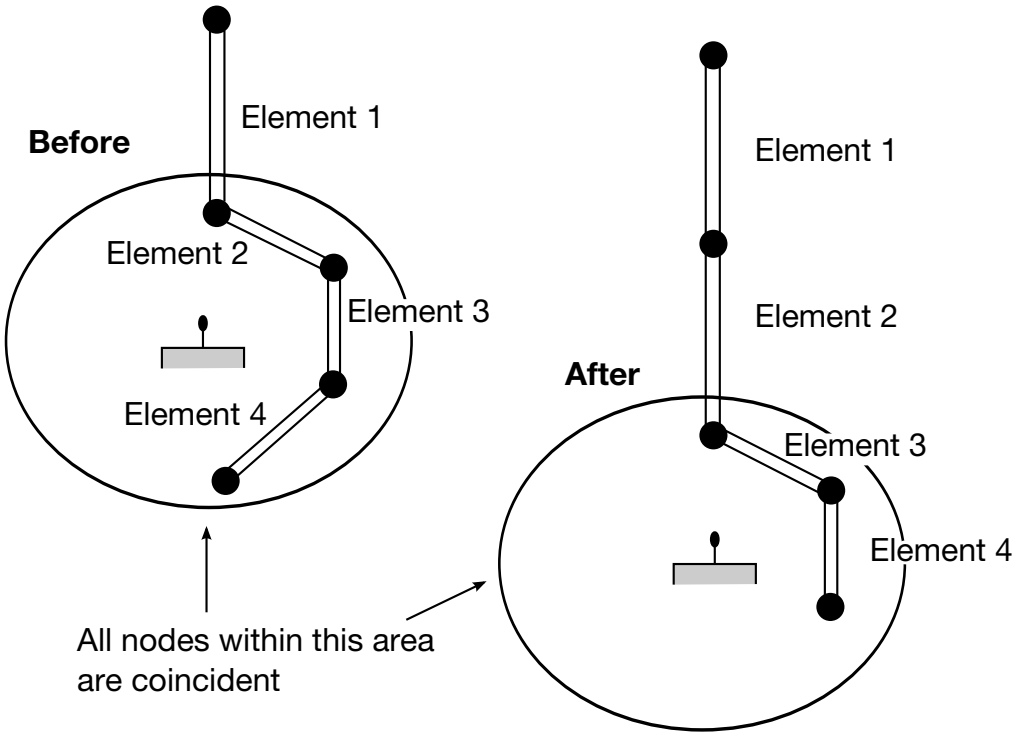


Figure 19-20. Elements in a retractor.

VARIABLE	DESCRIPTION
	EQ.1: Check the status of all unfired switches in curve LCFL. The force of the most recently fired switch gives the active force limit. The algorithm does not check that switch again in the future. That force level remains active until any other switch fires. The input order of the data points is irrelevant.
	EQ.2: Check the status of all switches of all data points in curve LCFL. The input order of the data points is irrelevant. The active force limit is the minimum of the forces of all fired switches.

Remarks:

1. **Defining a retractor.** To define a retractor, input the retractor node and the 'mouth' element (into which belt material will be fed), shown as Element 1 in Figure 19-20. Creating a retractor also requires specifying up to four sensors for triggering unlocking, a time delay, a pay-out delay (optional), load and unload curves, and the fed length. To optionally deactivated the retractor, provide a deactivation sensor, DSID. Deactivation stops any further pay-in or pay-out. Typically part of the vehicle contains the retractor node. Do not connect belt elements to this node directly. However, you can attach any other feature,

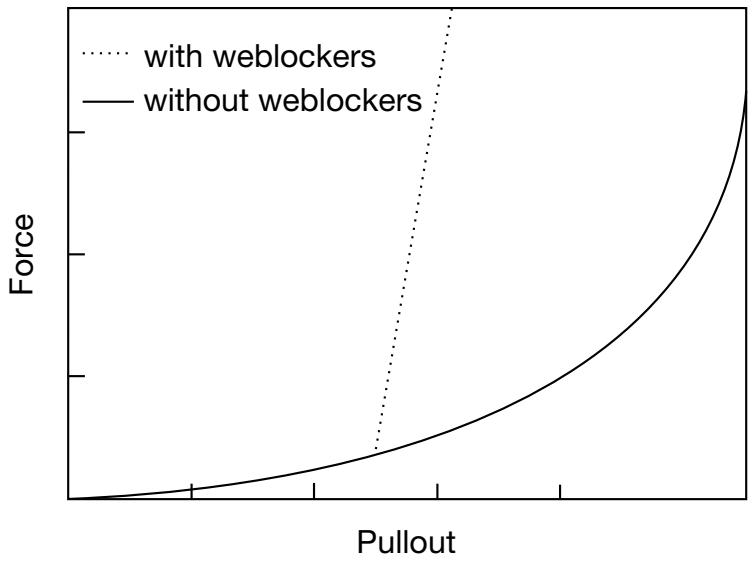


Figure 19-21. Retractor force pull characteristics.

including rigid bodies, to this node. The mouth element should have a node coincident with the retractor but should not be inside the retractor. The unstretched length of the mouth element during initialization *must be at least* 1.6 × LMIN, where LMIN is a field defined in *MAT_SEATBELT. Also, any other element that might become a mouth element *must have an unstretched length of at least* 1.1 × LMIN. The unstretched length, l, of a seat belt element is

$$l = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} + \text{SLEN} \ ,$$

where (x_1, y_1, z_1) and (x_2, y_2, z_2) are the locations of the two nodes defining the seat belt element, and SLEN is the slack length (see *ELEMENT_SEATBELT). The fed length, LFED, is usually set to either a typical element initial length or the distance between painted marks on a real belt for comparisons with high-speed film. *The fed length should be at least three times the minimum length, LMIN.*

If the retractor initially has elements inside it (Elements 2, 3, and 4 in Figure 19-20), the retractor input should not refer to them, but the element input for these elements should identify the retractor. Their nodes should all coincide with the retractor node and should not be restrained or constrained. LS-DYNA automatically sets the initial slack to 1.1 × minimum length for these elements; this overrides any user-defined value.

To include weblockers in the retractor representation, enter a 'locking up' characteristic in the force pull-out curve; see Figure 19-21. The final section can be very steep (but must have a finite slope).

2. **Multiple belt elements.** If desired, you can define belt elements that are initially inside the retractor. These emerge as belt material is paid out and return into the retractor if sufficient material is reeled in during unloading. For example,

the retractor in Figure 19-20 initially contains Elements 2, 3, and 4. It pays out material into Element 1. When the retractor has fed L_{crit} into Element 1, where

$$L_{crit} = \text{LFED} - 1.1 \times \text{LMIN} ,$$

Element 2 emerges with an unstretched length of 1.1 × minimum length; the algorithm reduces the unstretched length of Element 1 by the same amount. Here, LMIN is the minimum length defined on the belt material input (see *MAT_SEATBELT), and LFED is the fed length. The force and strain in Element 1 are unchanged; in Element 2, they are set equal to those in Element 1. The retractor now pays out material into Element 2.

If no elements are inside the retractor, Element 2 can continue to extend as more material is fed into it.

As the retractor pulls in the belt (for example, during initial tightening), if the unstretched length of the mouth element becomes less than the minimum length, the element goes into the retractor.

3. **Retractor node and belt elements.** No belt elements should include the retractor node. The element specified with SBID should have one node coincident with the retractor node but should not be inside the retractor.

4. **Shell-type seat belt.** When SBRNID < 0, this retractor is for shell-type seat belts. -SBRNID is the *SET_NODE containing RN1, RN2, …RN5. SBID is then a *SET_SHELL_LIST. Note that the numbering of –SBRNID and SBID must be consistent in the direction of numbering (see Figure 19-17). For example, if *SET_NODE for SBRNID has nodes of (RN1, RN2, RN3, RN4, RN5), then *SET_SHELL_LIST for SBID should have elements of (RE1, RE2, RE3, RE4). .

5. **Sensor (*ELEMENT_SEATBELT_SENSOR) definitions.** Define at least one sensor. These sensors refer to a *ELEMENT_SEATBELT_SENSOR.

6. **Loading curve.** The first point of the load curve should be $(0, T_{min})$. T_{min} is the minimum tension. All subsequent tension values must be greater than T_{min}.

7. **Unloading curve.** The unloading curve should start at zero tension and increase monotonically (i.e., no segments of negative or zero slope).

8. **Retractor characteristics and loading.** Retractors allow belt material to be paid out into a belt element. Retractors operate in one of two regimes: unlocked when the belt material is paid out, or reeled in under constant tension and locked when a user-defined force-pull-out relationship applies.

   The retractor is initially unlocked. The following sequence of events must occur for it to become locked:

a) Any one of up to four sensors must be triggered. (The sensors are described below.)

b) Then, a user-defined time delay occurs.

c) Next, a user-defined belt length must be paid out (optional).

d) Finally, the retractor locks, and once locked, it remains locked.

In the unlocked regime, the retractor attempts to apply constant tension to the belt. This feature allows an initial tightening of the belt and takes up any slack whenever it occurs. The tension value comes from the first point on the force-pull-out load curve. $0.01 \times$ fed length per time step gives the maximum rate of pull-out or pull-in. Because of this, the constant tension value is not always achieved.

In the locked regime, a user-defined curve describes the relationship between the force in the attached element and the amount of belt material paid out. If the tension in the belt subsequently relaxes, a different user-defined curve applies for unloading. The unloading curve is followed until the minimum tension is reached.

Define the curves in terms of the initial length of the belt. For example, if a belt is marked at 10 mm intervals and then wound onto a retractor, and the force required to make each mark emerge from the (locked) retractor is recorded, the curves used for input would be as follows:

0 mm Minimum tension (should be > zero)

10 mm Force to emergence of first mark

20 mm Force to emergence of second mark

⋮

You may define pyrotechnic pretensions, which cause the retractor to pull in the belt at a predetermined rate. This feature overrides the retractor force-pull-out relationship from the moment when the pretensioner activates.

9. **Locking and pretensioners.** When the model only contains retractors, be aware that we measure the pull-out from the point when the retractor locks. If the belt has pulled in since the retractor locked, then the minimum force occurs in the retractor until the system pays out enough belt to return to its locking point.

If the behavior described above is undesirable, we recommend the type 6 pre-tensioner model for the seat belt system. It defines a constant force as a function

of time load curve with a force equal to minimum tension, with a small PULL value on the retractor. This setup has an active pretensioner until the belt pulls all the way in. As soon as the belt starts to move back out, the algorithm disables the pretensioner, and the retractor takes over.

10. **Load limiters of pretensioners and retractors.** The load limiter of a retractor differs from the load limiter of pretensioner types 5 and 8. The algorithm activates the pretensioner's load limiter when the pretensioner is on and the retractor has not locked yet. The algorithm applies the retractor's load limiter when the pretensioner is off.

11. **Example of specifying LCFL with *DEFINE_CURVE_FUNCTION.** The following partial keyword deck input illustrates how to specify an adaptive load limiter using *DEFINE_CURVE_FUNCTION.

```
*ELEMENT_SEATBELT_RETRACTOR
$#    sbrid     sbrnid       sbid       sid1       sid2       sid3       sid4       sid5
     99001    5002165   10024916          2                                           6
$#     tdel       pull      llcid      ulcid      lfed       lcfl      flopt
     0.000      0.000       5002       2007 20.000000        999
*SENSOR_DEFINE_MISC
         1       TIME
*SENSOR_DEFINE_FUNCTION
         2         77          1
*DEFINE_CURVE_FUNCTION
999
SENSORD(2)
$
*DEFINE_FUNCTION
        77load limiter func
float flimit(float time1)
   {
    float flimit;
    flimit=1.e16;
    if (time1 >= 50.) flimit= 3.5;
    if (time1 >= 60.) flimit= 3.0;
    if (time1 >= 70.) flimit= 2.5;
    return flimit;
   }
```

*ELEMENT_SEATBELT_SENSOR

Purpose: Define a seat belt sensor. Five types are possible; see Remark 3.

Card Summary:

Card 1. This card is required.

SBSID	SBSTYP	SBSFL					

Card 2a. This card is included if and only if SBSTYP = 1.

NID	DOF	ACC	ATIME				

Card 2b. This card is included if and only if SBSTYP = 2.

SBRID	PULRAT	PULTIM					

Card 2c. This card is included if and only if SBSTYP = 3.

TIME							

Card 2d. This card is included if and only if SBSTYP = 4.

NID1	NID2	DMX	DMN	NID1			

Card 2e. This card is included if and only if SBSTYP = 5.

SBRID	PAYMX	PAYMN					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SBSID	SBSTYP	SBSFL					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
SBSID	Sensor ID. A unique number must be used.

VARIABLE	DESCRIPTION
SBSTYP	Sensor type: EQ.1: acceleration of node EQ.2: retractor pull-out rate EQ.3: time EQ.4: distance between nodes EQ.5: retractor pull-out
SBSFL	Sensor flag: EQ.0: sensor is not active during dynamic relaxation EQ.1: sensor can be triggered during dynamic relaxation

Additional card for SBSTYP = 1.

Card 2a	1	2	3	4	5	6	7	8
Variable	NID	DOF	ACC	ATIME				
Type	I	I	F	F				
Default	none	none	0.0	0.0				

VARIABLE	DESCRIPTION
NID	Node ID of sensor. See Remark 1.
DOF	Degree of freedom: EQ.1: x EQ.2: y EQ.3: z
ACC	Activating acceleration
ATIME	Time over which acceleration must be exceeded

Additional card for SBSTYP = 2.

Card 2b	1	2	3	4	5	6	7	8
Variable	SBRID	PULRAT	PULTIM					
Type	I	F	F					
Default	0	0.0	0.0					

VARIABLE	DESCRIPTION
SBRID	Retractor ID; see *ELEMENT_SEATBELT_RETRACTOR.
PULRAT	Rate of pull-out (length/time units)
PULTIM	Time over which rate of pull-out must be exceeded

Additional card for SBSTYP = 3.

Card 2c	1	2	3	4	5	6	7	8
Variable	TIME							
Type	F							
Default	0.0							

VARIABLE	DESCRIPTION
TIME	Time at which sensor triggers

Additional card for SBSTYP = 4.

Card 2d	1	2	3	4	5	6	7	8
Variable	NID1	NID2	DMX	DMN				
Type	I	I	F	F				
Default	0	0	0.0	0.0				
Remarks			2	2				

VARIABLE	DESCRIPTION
NID1	Node 1 ID
NID2	Node 2 ID
DMX	Maximum distance
DMN	Minimum distance

Additional card for SBSTYP = 5.

Card 2e	1	2	3	4	5	6	7	8
Variable	SBRID	PULMX	PULMN					
Type	I	F	F					
Default	0	10^{16}	-10^{16}					

VARIABLE	DESCRIPTION
SBRID	Retractor ID; see *ELEMENT_SEATBELT_RETRACTOR.
PULMX	Maximum pull-out
PULMN	Minimum pull-out

Remarks:

1. **Node ID for Acceleration of Node Sensor.** Node should not be on rigid body, velocity boundary condition, or other 'imposed motion' feature.

2. **Distance Sensor.** Sensor triggers when the distance between the two nodes is $d \geq d_{\max}$ or $d \leq d_{\min}$.

3. **Sensor Types.** Sensors are used to trigger locking of retractors and to activate pretensioners. Four types of sensors are available which trigger according to the following criteria:

   Type 1: When the magnitude of x-, y-, or z-acceleration of a given node has remained above a given level continuously for a given time, the sensor triggers. This does not work with nodes on rigid bodies.

   Type 2: When the rate of belt payout from a given retractor has remained above a given level continuously for a given time, the sensor triggers.

   Type 3: The sensor triggers at a given time.

   Type 4: The sensor triggers when the distance between two nodes exceeds a given maximum or becomes less than a given minimum. This type of sensor is intended for use with an explicit mass/spring representation of the sensor mechanism.

   Type 5: The sensor triggers when the retractor's payout is out of the range specified by PULMX and PULMN.

   By default, the sensors are inactive during dynamic relaxation. This allows initial tightening of the belt and positioning of the occupant on the seat without locking the retractor or firing any pretensioners. However, a flag can be set in the sensor input to make the sensors active during the dynamic relaxation phase.

*ELEMENT_SEATBELT_SLIPRING

Purpose: Define seat belt slip ring.

Card 1	1	2	3	4	5	6	7	8
Variable	SBSRID	SBID1	SBID2	FC	SBRNID	LTIME	FCS	ONID
Type	I	I	I	F	I	F	F	I
Default	0	0	0	0.0	0	10^{20}	0.0	0

Optional Card.

Card 2	1	2	3	4	5	6	7	8
Variable	K	FUNCID	DIRECT	DC		LCNFFD	LCNFFS	
Type	F	I	I	F		I	I	
Default	0.0	0	0	0.0		0	0	

VARIABLE	DESCRIPTION
SBSRID	Slip ring ID. A unique number has to be used. See Remark 4 below for the treatment of slip rings for shell belt elements.
SBID1	Seat belt element 1 ID
SBID2	Seat belt element 2 ID
FC	Coulomb dynamic friction coefficient. If less than zero, |FC| refers to a curve which defines the dynamic friction coefficient as a function of time.
SBRNID	Slip ring node, NID
LTIME	Slip ring lockup time. After this time no material is moved from one side of the slip ring to the other. This option is not active during dynamic relaxation.

VARIABLE	DESCRIPTION
FCS	Optional Coulomb static friction coefficient. If less than zero, |FCS| refers to a curve which defines the static friction coefficient as a function of time.
ONID	Optional orientation node ID used to define the skew angle, α (see Figures 19-22 and 19-25). If ONID is undefined, the skew angle is assumed to be 0.0.
K	Optional coefficient for determining the Coulomb friction coefficient related to the skew angle, α (see Figure 19-25). See Remark 5.
FUNCID	Function ID to determine friction coefficient.
DIRECT	Direction of belt movement. To disable the belt slip behavior imposed by DIRECT, see the Remark "Seatbelt Slip Ring" under *SENSOR_CONTROL. EQ.0: If the belt can move along both directions. EQ.12: If the belt is only allowed to slip along the direction from SBID1 to SBID2. EQ.21: If the belt is only allowed to slip along the direction from SBID2 to SBID1.
DC	Optional decay constant to allow a smooth transition between the static and dynamic friction coefficients, that is, $\mu_c = \text{FC} + (\text{FCS} - \text{FC})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}$
LCNFFD	Optional curve for normal-force-dependent Coulomb dynamic friction coefficient. When defined, the dynamic friction coefficient becomes $\text{FC} + f_{\text{LCNFFD}}(F_n)$, where $f_{\text{LCNFFD}}(F_n)$ is the function value of LCNFFD at contact force, F_n. The normal direction is defined as the average of the directions along the length of the seatbelt elements SBID1 and SBID2. The normal (or contact) force, F_n, is the sum of the projection of the tension forces on these elements, T_1 and T_2, onto the normal direction.
LCNFFS	Optional curve for normal-force-dependent Coulomb static friction coefficient. When defined, the static friction coefficient becomes $\text{FCS} + f_{\text{LCNFFS}}(F_n)$, where $f_{\text{LCNFFS}}(F_n)$ is the function value of LCNFFS at contact force, F_n.

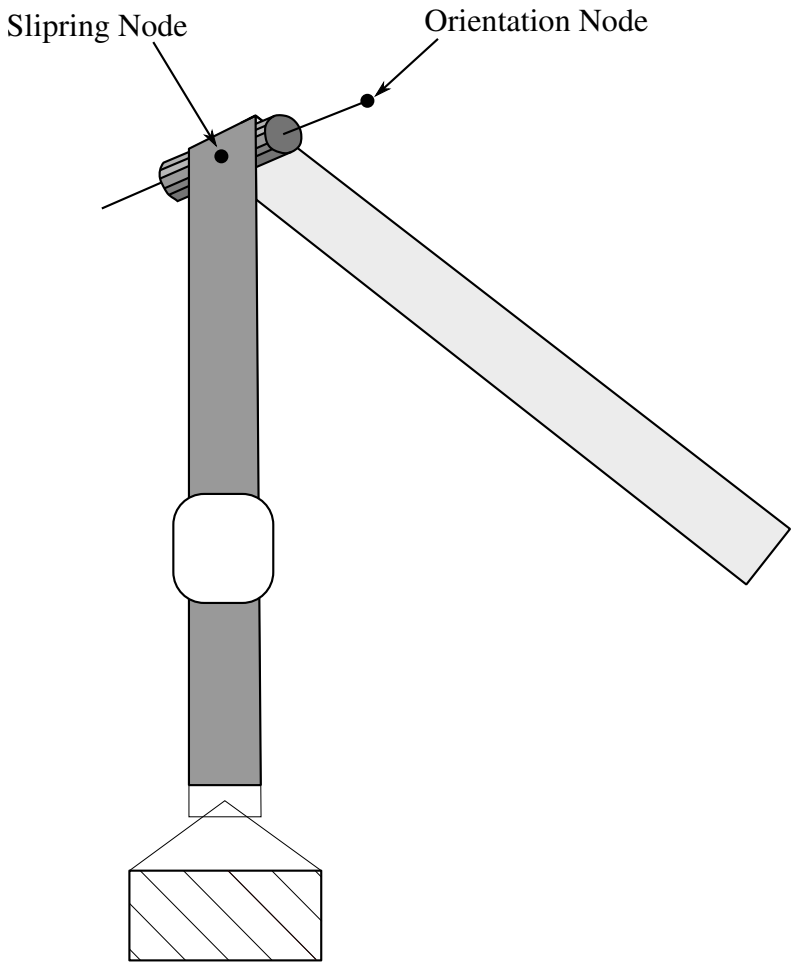


Figure 19-22. Orientation node.

Remarks:

1. **Slip Ring Model.** Slip rings allow continuous sliding of a belt through a sharp change of angle. To define a slip ring, the user identifies the two belt elements which meet at the slip ring (such as Elements 1 and 2 shown in Figure 19-23), the friction coefficient, and the slip ring node. The two elements must have a common node coincident with the slip ring node and should not be referenced by any other slip ring definition. No attempt should be made to restrain or constrain the common node because its motion will automatically be constrained to follow the slip ring node. Typically, the slip ring node is part of the vehicle body structure; therefore, belt elements should not be connected to this node directly, but any other feature can be attached, including rigid bodies.

   As shown in Figure 19-23, during slip Node B in the belt material remains attached to the slip ring node, but belt material (in the form of unstretched length) is passed from Element 1 to Element 2 to achieve slip. The amount of slip at each time step is calculated from the ratio of forces in Elements 1 and 2. The ratio of

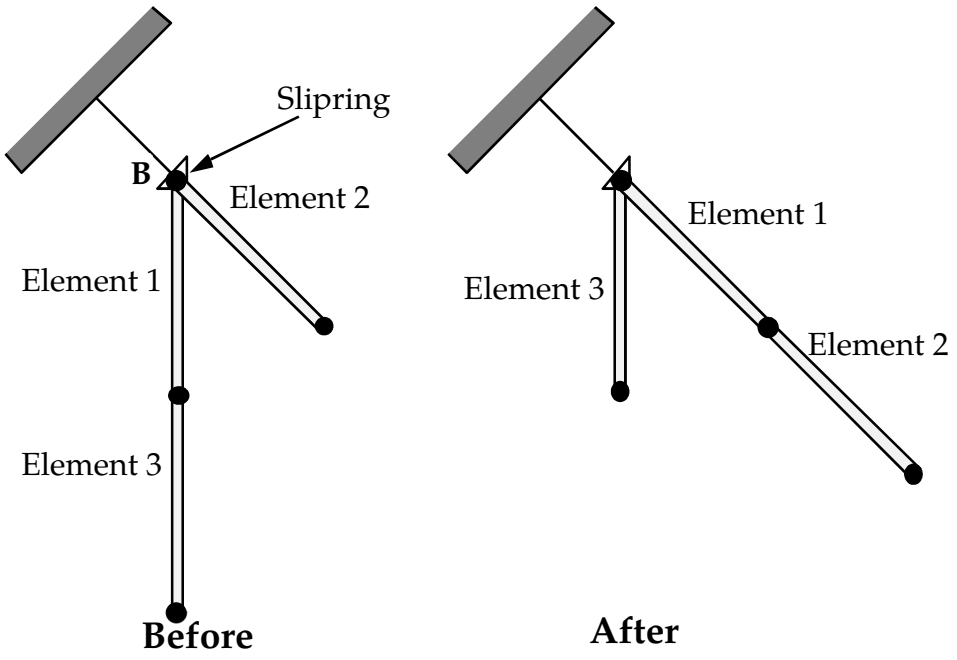


Figure 19-23. Elements passing through slipring.

forces is determined by the relative angle between the elements and the coefficient of friction, FC. The tension in the belts are T_1 and T_2, where T_2 is on the high tension side and T_1 is the force on the low tension side. Thus, if T_2 is sufficiently close to T_1, no slip occurs; otherwise, slip is just sufficient to reduce the ratio T_2/T_1 to $e^{\mathrm{FC}\times\theta}$, where θ is the wrap angle; see Figures 19-22 and 19-24. No slip occurs if both elements are slack. The out-of-balance force at Node B is reacted on the slip ring node; the motion of node B follows that of slip ring node.

2. **Elements Connected to Sliprings.** The unstretched length of a seatbelt element connected to a slipring during initialization *must be at least* 1.6 × LMIN where LMIN is a field defined in *MAT_SEATBELT. The unstretched length, l, of a seatbelt element is

$$l = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} + \mathrm{SLEN} \ ,$$

where (x_1, y_1, z_1) and (x_2, y_2, z_2) are the locations of the two nodes defining the seatbelt element and SLEN is the slack length (see *ELEMENT_SEATBELT). Any other element that may interact with the slipring *must have an unstretched length of at least* 1.1 × LMIN.

3. **Local Remeshing.** If, due to slip through the slip ring, the unstretched length of an element becomes less than the minimum length (as entered on the belt material card), the belt is remeshed locally: the short element passes through the slip ring and reappears on the other side (see Figure 19-23). The new unstretched length of element 1 is 1.1 × minimum length. Force and strain in elements 2 and 3 are unchanged; force and strain in element 1 are now equal to

those in element 2. Subsequent slip will pass material from element 3 to element 1. This process can continue with several elements passing in turn through the slip ring.

4. **Shell Elements.** When SBRNID < 0, this slip ring is for shell-type seatbelt; -SBRNID is the *SET_NODE containing SN1, SN2, …SN5. SBID1 and SBID2 are then *SET_SHELL_LIST. Note that the numbering of -SBRNID, SBID1 and SBID2 has to be consistent in the direction of numbering. For example, if *SET_-NODE for SBRNID has nodes (SN1, SN2, SN3, SN4, SN5) then *SET_SHELL_-LIST for SBID1 should have elements (SRE11, SRE12, SRE13, SRE14) and *SET_-SHELL_LIST for SBID2 should have elements (SRE21, SRE22, SRE23, SRE24). See Figure 19-22.

5. **Coulomb Friction.** If K is undefined, the limiting force ratio is taken as $e^{\mathrm{FC}\times\theta}$. If K is defined, the maximum force ratio is computed as

$$e^{\mathrm{FC}\times\theta(1+\mathrm{K}\times\alpha^2)},$$

where α is the angle shown in Figure 19-25. The function is defined using the *DEFINE_FUNCTION keyword input. This function is a function of three variables, and the ratio is given by evaluating

$$\frac{T_2}{T_1} = \mathrm{FUNC}(\mathrm{FCT}, \theta, \alpha),$$

where FCT is the instantaneous friction coefficient at time t, that is, it has the value of FC if the belt has moved in the last time-step and the value of FCS if the belt has been stationary. For example, the default behavior can be obtained using the function definition (assuming FCT has a value of 0.025 and the function ID is 1):

```
*DEFINE_FUNCTION
    1,
f(fct,theta,alpha) = exp(0.025*theta)
```

Behavior like the default option can be obtained with (K = 0.1):

```
*DEFINE_FUNCTION
    1,
f(fct,theta,alpha) = exp(0.025*theta*(1.+0.1*alpha*alpha))
```

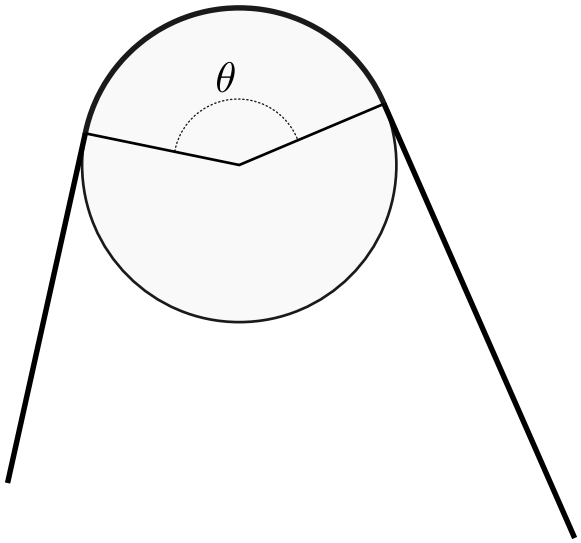


Figure 19-24. Front view showing wrap angle.

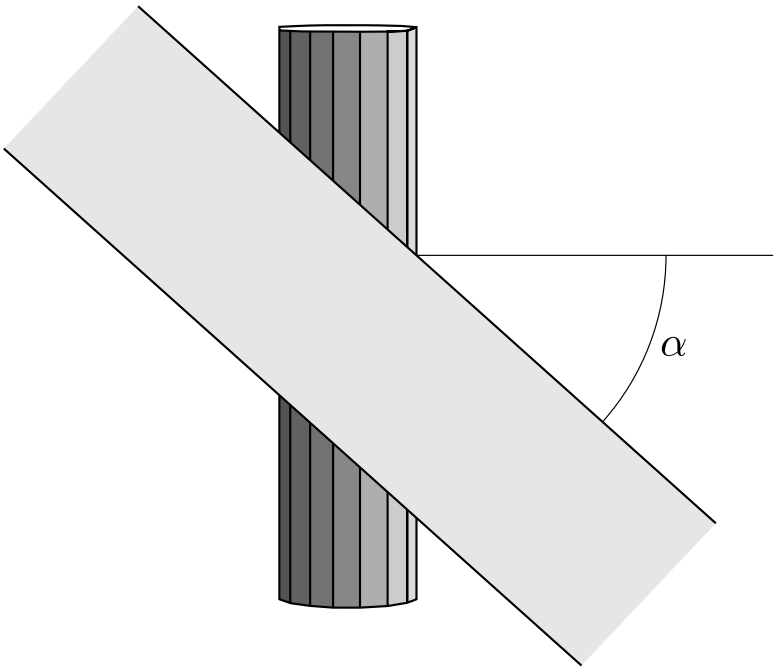


Figure 19-25. Top view shows orientation of belt relative to axis.

***ELEMENT_SHELL**_{OPTION}

Available options include:

<BLANK>

THICKNESS

BETA or MCID

OFFSET

DOF

COMPOSITE

COMPOSITE_LONG

SHL4_TO_SHL8

Stacking of options, e.g., THICKNESS_OFFSET, is allowed in some cases. When combining options in this manner, check d3hsp to confirm that all the options are acknowledged.

Purpose: Define three, four, six, and eight node elements including 3D shells, membranes, 2D plane stress, plane strain, and axisymmetric solids. The type of the element and its formulation is specified through the part ID (see *PART) and the section ID (see *SECTION_SHELL). Also, the thickness of each element can be specified when applicable on the element cards or else a default thickness value is used from the section definition.

For orthotropic and anisotropic materials, a local material angle (variable BETA) can be defined which is cumulative with the integration point angles specified in *SECTION_SHELL, *PART_COMPOSITE, *ELEMENT_SHELL_COMPOSITE, or *ELEMENT_SHELL_COMPOSITE_LONG. Alternatively, the material coordinate system can be defined as the projection of a local coordinate system, MCID, onto the shell.

An offset option, OFFSET, is available for moving the shell reference surface from the nodal points that define the shell.

The COMPOSITE or COMPOSITE_LONG option allows an arbitrary number of integration points across the thickness of shells sharing the same part ID. This is independent of thickness defined in *SECTION_SHELL. To maintain a direct association of through-thickness integration point numbers with physical plies in the case where the number of plies varies from element to element, see Remark 12.

The option, SHL4_TO_SHL8, converts 3 node triangular and 4 node quadrilateral shell elements to 6 node triangular and 8 node quadrilateral quadratic shell elements, respectively, by the addition of mid-side nodal points. See Remark 9 below.

For the shell formulation that uses additional nodal degrees-of-freedom, the option DOF is available to connect the nodes of the shell to corresponding scalar nodes. Four scalar nodes are required for element type 25 to model the thickness changes that require 2 additional degrees-of-freedom per shell node. Defining these nodes is optional; if left undefined, they will be automatically created.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	0	0	0	0
Remarks			3	3	3	3				

Thickness Card. Additional card for THICKNESS, BETA, and MCID keyword options.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	THIC1		THIC2		THIC3		THIC4		BETA or MCID	
Type	F		F		F		F		F	
Default	0.		0.		0.		0.		0.	
Remarks	1								2	

Thickness Card. Additional card for THICKNESS, BETA, and MCID keyword options, is only required if mid-side nodes are defined (N5-N8).

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	THIC5		THIC6		THIC7		THIC8			
Type	F		F		F		F			
Default	0.		0.		0.		0.		.	
Remarks	6									

Offset Card. Additional card for OFFSET keyword options.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	OFFSET									
Type	F									
Default	0.									
Remarks	7									

Scalar Node Card. Additional card for DOF keyword option.

Card 5	1	2	3	4	5	6	7	8	9	10
Variable			NS1	NS2	NS3	NS4				
Type			I	I	I	I				
Default										
Remarks			8	8	8	8				

COMPOSITE Cards. Additional set of cards for the COMPOSITE keyword option. Values of material ID, thickness, and material angle for each through-thickness integration point of a composite shell are defined using these cards (up to two integration points per card), and these values overrule values specified elsewhere. The integration point data should be given sequentially starting with the bottommost integration point. The total number of integration points is determined by the number of entries on these cards. The thickness of each shell is the summation of the integration point thicknesses. Define as many cards as needed to define all the through-thickness integration points. The fourth field must be zero or blank to be interpreted as a Card 6.

Card 6	1	2	3	4	5	6	7	8
Variable	MID1	THICK1	B1		MID2	THICK2	B2	
Type	I	F	F		I	F	F	

COMPOSITE_LONG Cards. Additional set of cards for the COMPOSITE_LONG keyword option. Values of material ID, thickness, and material angle for each through-thickness integration point of a composite shell are defined using these cards (one integration point per card), and these values overrule values specified elsewhere. The integration point data should be given sequentially starting with the bottommost integration point. The total number of integration points is determined by the number of entries on these cards. The thickness of each shell is the summation of the integration point thicknesses. Define as many cards as needed to define all the through-thickness integration points. The fourth field must be zero or blank to be interpreted as a Card 7.

Card 7	1	2	3	4	5	6	7	8
Variable	MID1	THICK1	B1		PLYID1			
Type	I	F	F		I			

VARIABLE	DESCRIPTION
EID	Element ID. Chose a unique number with respect to other elements.
PID	Part ID, see *PART.
N1	Nodal point 1
N2	Nodal point 2
N3	Nodal point 3

VARIABLE	DESCRIPTION
N4	Nodal point 4
N5 - N8	Mid-side nodes for eight node shell
THIC1	Shell thickness at node 1
THIC2	Shell thickness at node 2
THIC3	Shell thickness at node 3
THIC4	Shell thickness at node 4
BETA	Orthotropic material base offset angle (see Remarks 5 and 6 below). The angle is given in degrees. If blank, the default is set to zero.
MCID	Material coordinate system ID. The **a**-axis of the base (or element-level) material coordinate system is the projection of the x-axis of coordinate system MCID onto the surface of the shell element. The **c**-axis of the material coordinate system aligns with the shell normal. The **b**-axis is taken as $\mathbf{b} = \mathbf{c} \times \mathbf{a}$. Each layer in the element can have a unique material orientation by defining a rotation angle for each layer as described in Remark 5.
THIC5	Shell thickness at node 5
THIC6	Shell thickness at node 6
THIC7	Shell thickness at node 7
THIC8	Shell thickness at node 8
OFFSET	The offset distance from the plane of the nodal points to the reference surface of the shell in the direction of the normal vector to the shell.
NS1	Scalar node 1, parameter NDOF on the *NODE_SCALAR is normally set to 2. If the thickness is constrained, set NDOF = 0.
NS2	Scalar node 2
NS3	Scalar node 3
NS4	Scalar node 4
MID*i*	Material ID of integration point *i*, see *MAT_… Section.

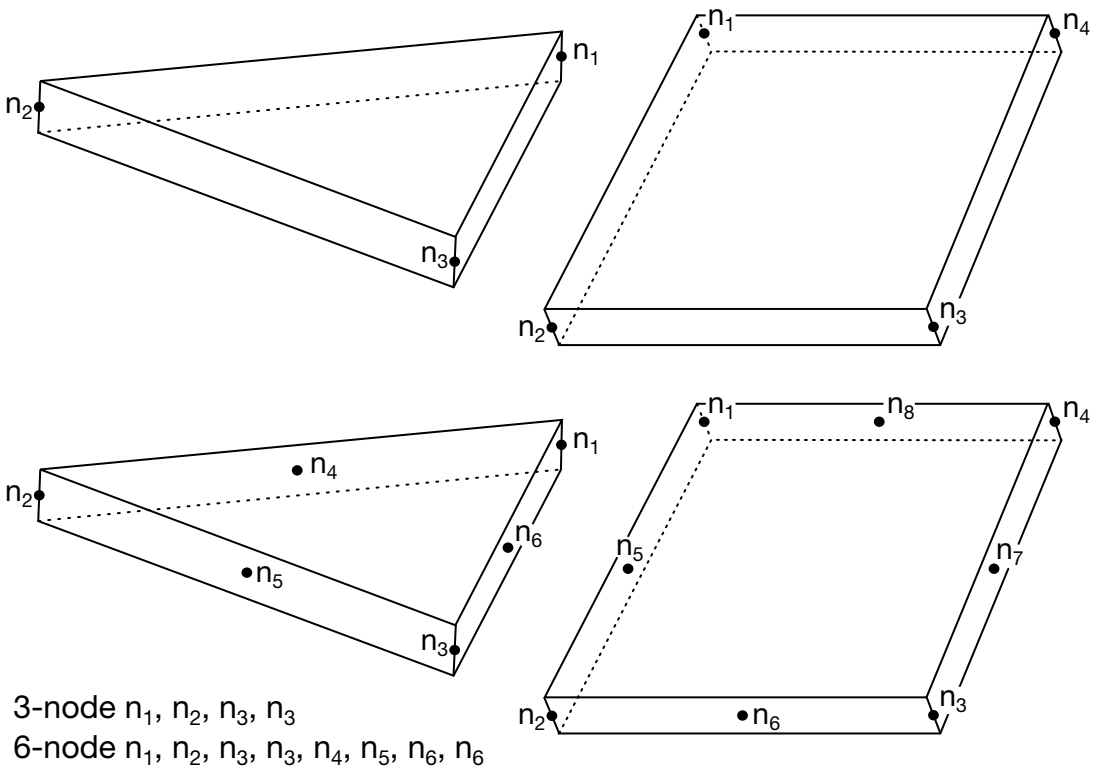


Figure 19-26. LS-DYNA shell elements. Counterclockwise node numbering determines the top surface.

VARIABLE	DESCRIPTION
THICK*i*	Thickness of integration point *i*.
B*i*	Material angle of integration point *i*.
PLYID*i*	Ply ID for integration point *i* (for post-processing purposes).

Remarks:

1. **Default thickness.** Default values in place of zero shell thicknesses are taken from the cross-section property definition of the PID; see *SECTION_SHELL.

2. **Ordering.** Counterclockwise node numbering determines the top surface, see Figure 19-26

3. **Coordinate systems.** Stresses and strain output in the binary databases are given in the global coordinate system, whereas stress resultants are output in the local coordinate system for the shell element.

4. **Convexity.** Interior angles must be less than 180 degrees.

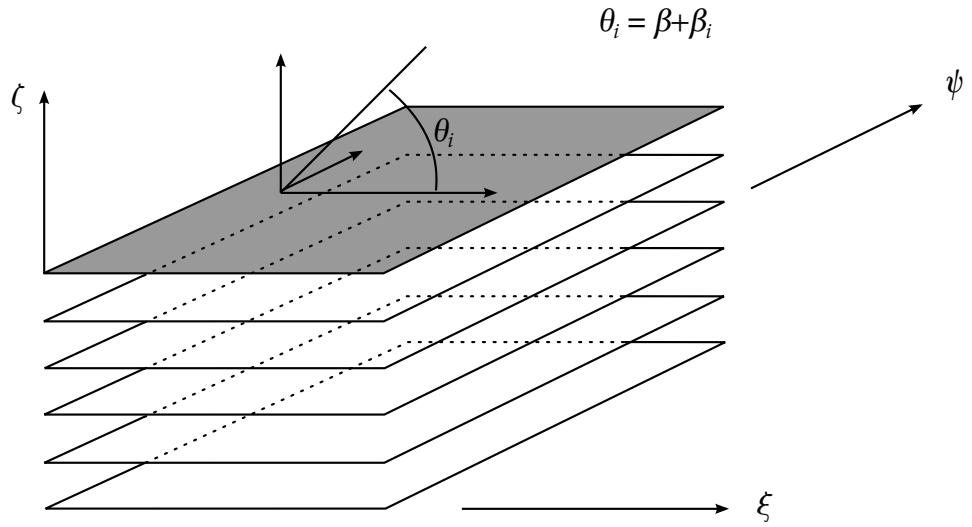


Figure 19-27. A multi-layer laminate can be defined. The angle β_i is defined for the i^{th} lamina (integration point), see *SECTION_SHELL.

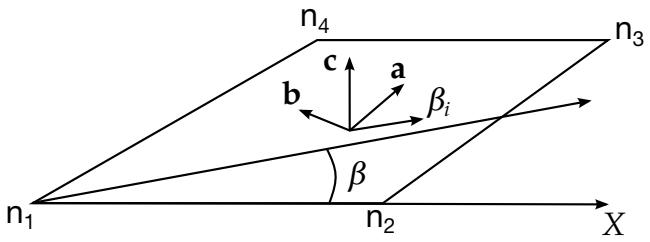


Figure 19-28. Orientation of material directions (shown relative to the 1-2 side as when AOPT = 0 in *MAT).

5. **Material orientation.** To allow the orientation of orthotropic and anisotropic materials to be defined for each shell element, a BETA angle can be defined. This BETA angle is used with the AOPT parameter and associated data on the *MAT card to determine an element reference direction for the element. The AOPT data defines a coordinate system and the BETA angle defines a subsequent rotation about the element normal to determine the element reference system. For composite modeling, each layer in the element can have a unique material direction by defining an additional rotation angle for the layer, using either the ICOMP and B*i* parameters on *SECTION_SHELL or the B*i* parameter on *PART_COMPOSITE. The material direction for layer *i* is then determined by a rotation angle, θ_i as shown in Figures 19-27 and 19-28.

6. **Activation of the BETA field and its interpretation.** To activate the BETA field, either the BETA or THICKNESS keyword option must be used. There is a difference in how a zero value or empty field is interpreted. When the BETA keyword option is used, a zero value or empty BETA field will override the BETA on *MAT. However, when the THICKNESS keyword option is used, a zero value or empty BETA field will not override the BETA value on *MAT.

Therefore, to input BETA = 0, the BETA keyword option is recommended. If a THIC value is omitted or input as zero, the thickness will default to the value on the *SECTION_SHELL card. If mid-side nodes are defined (N5 - N8), then a second line of thickness values will be read. This line may be left blank, but cannot be omitted.

7. **Offset for the reference surface.** The parameter OFFSET gives the offset from the nodal points of the shell to the reference surface. This option applies to most shell formulations excluding two-dimensional elements, membrane elements, and quadratic shell elements. Except for Mortar contacts, the reference surface offset given by OFFSET is not taken into account in the contact subroutines unless CNTCO is set to 1 in *CONTROL_SHELL. For Mortar contacts the OFFSET determines the location of the contact surface.

8. **Scalar nodes.** The scalar nodes specified on the optional card refer to the scalar nodes defined by the user to hold additional degrees of freedom for shells with this capability. Scalar nodes are used with shell element type 25 and 26.

9. **Automatic order increase.** The option, SHL4_TO_SHL8, converts 3 node triangular and 4 node quadrilateral shell elements to 6 node triangular and 8 node quadrilateral quadratic shell elements, respectively, by the addition of mid-side nodal points. The user node ID's for these generated nodes are offset after the largest user node ID defined in the input file. When defining the *SECTION_SHELL keyword, the element type must be specified as either 23 or 24 corresponding to quadratic quadrilateral and triangular shells, respectively.

10. **Cohesive elements.** Cohesive elements (ELFORM = 29, 46 or 47 on *SECTION_SHELL) may be defined with zero depth to connect surfaces with no gap between them, but must have nodes 1 and 2 on one surface and nodes 3 and 4 on the other.

11. **Contact thickness when using *ELEMENT_SHELL_THICKNESS in MPP.** For MPP, THEORY = 1 in *CONTROL_SHELL has a special meaning when dealing with non-uniform-thickness shells: it serves to set the nodal contact thickness equal to the average of the nodal thicknesses from the shells sharing that node. Thus, when non-uniform-thickness shells comprise a contact surface, we recommend THEORY = 1 and setting the actual shell formulation with ELFORM in *SECTION_SHELL. (This remark does not apply to segment based (SOFT=2) contact wherein each contact segment is considered to be of uniform thickness.)

12. **Assignment of zero thickness to integration points.** The ability to assign zero- thickness integration points in the stacking sequence allows the number of integration points to remain constant even as the number of physical plies varies from element to element and eases post-processing since a particular integration point corresponds to a physical ply. Such a capability is important when one or more of the physical plies are not continuous across a part. To represent a

missing ply in *ELEMENT_SHELL_COMPOSITE, set THICK*i* to 0.0 for the corresponding integration point and additionally, either set MID = -1 or, if the LONG option is used, set PLYID to any nonzero value

When postprocessing the results using LS-PrePost version 4.5, read both the keyword deck and d3plot database into the code and then select Option → N/A gray fringe. Then, when viewing fringe plots for a particular integration point (FriComp → IPt → intpt#), the element will be grayed out if the selected integration point is missing (or has zero thickness) in that element.

13. **PID with COMPOSITE keyword option.** When using the COMPOSITE keyword option, the PID on Card 1 must reference a *PART card that does not also use the COMPOSITE keyword option.

***ELEMENT_SHELL_NURBS_PATCH**_{OPTION1}_{OPTION2}

Available options for *OPTION1* include:

<BLANK>

TRIMMED

Available options for *OPTION2* include:

<BLANK>

TITLE

Purpose: Define a NURBS surface patch using two univariate knot vectors, a rectangular grid of control points, and optionally a set of control weights. The two knot vectors define the necessary shape functions and parameterize the surface by virtue of the tensor product scheme.

The *r*- and *s*-direction knot vectors have lengths of NPR+PR+1 and NPS+PS+1, respectively. NPR and NPS are the number of control points in each direction while PR and PS define the polynomial degree in each direction. The control grid consists of NPR × NPS control points. A control weight may be defined for each control point. There is no limit on the size of a NURBS surface patch.

The total number of necessary data cards depends on the parameters given in Cards 1 and 2, that is,

$$\#\ \text{of cards} = 2 + \left\lceil \frac{\text{NPR} + \text{PR} + 1}{8} \right\rceil + \left\lceil \frac{\text{NPS} + \text{PS} + 1}{8} \right\rceil + (\text{WFL} + 1) \times \text{NPS} \times \left\lceil \frac{\text{NPR}}{8} \right\rceil ,$$

where $\lceil x \rceil = \text{ceil}(x)$. The flag WFL (see Card 2 below) indicates whether the control weights are user-defined. An example NURBS surface and the partial keyword deck using this keyword is given in Figures 19-29 and 19-30, respectively.

A NURBS surface patch may be trimmed using the TRIMMED option. This requires the definition of additional trimming loop(s) which can be defined using Cards E and F. Note that optional cards used to define trimming loops are not counted in the above expression. See Remark 6.

A description of the NURBS surface patch may be provided using the option TITLE which is also written to the d3hsp file.

Card Summary:

Card Title. This card is included if and only if the TITLE keyword option is used.

TITLE

Card 1. This card is required.

NPID	PID	NPR	PR	NPS	PS		

Card 2. This card is required.

WFL	ELFORM	INT	NISR	NISS	IMASS		IDFNE

Cards A. Include ceil[(NPR + PR + 1)/8] of this card.

RK1	RK2	RK3	RK4	RK5	RK6	RK7	RK8

Cards B. Include ceil[(NPS + PS + 1)/8] of this card.

SK1	SK2	SK3	SK4	SK5	SK6	SK7	SK8

Cards C. Include NPS × ceil(NPR/8) of this card.

N1	N2	N3	N4	N5	N6	N7	N8

Cards D. Include NPS × ceil(NPR/8) of this card if WFL = 1.

W1	W2	W3	W4	W5	W6	W7	W8

Cards E. This card is included if the keyword option TRIMMED is used.

TITLE

Cards F. This card is included if the keyword option TRIMMED is used.

C1	C2	C3	C4	C5	C6	C7	C8

Data Cards:

Title Card. Additional card for the TITLE keyword option.

Card Title	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							

VARIABLE	DESCRIPTION
TITLE	Description of the NURBS patch surface

Card 1	1	2	3	4	5	6	7	8
Variable	NPID	PID	NPR	PR	NPS	PS		
Type	I	I	I	I	I	I		
Default	none	none	None	none	none	none		

VARIABLE	DESCRIPTION
NPID	NURBS surface element / patch ID. A unique number must be chosen.
PID	PART ID. See *PART.
NPR	Number of control points in the local *r*-direction.
PR	Polynomial degree of the basis function in the local *r*-direction.
NPS	Number of control points in the local *s*-direction.
PS	Polynomial degree of the basis function in the local *s*-direction.

Card 2	1	2	3	4	5	6	7	8
Variable	WFL	ELFORM	INT	NISR	NISS	IMASS		IDFNE
Type	I	I	I	F	F	I		I
Default	0	0	0	PR	PS	0		0

VARIABLE **DESCRIPTION**

WFL Flag for user defined control weights:

EQ.0: Control weights are assumed to be uniform and positive; that is, the surface is a B-spline surface. No optional Cards D is allowed.

EQ.1: Control weights are defined using optional Cards D.

ELFORM Shell formulation to be used (see Remark 3):

EQ.0: Reissner-Mindlin with fibers at the control points

EQ.1: Kirchhoff-Love with fibers at the control points (not recommended)

EQ.2: Kirchhoff-Love with fibers at the integration points

EQ.3: Reissner-Mindlin with fibers at the integration points

EQ.-4/4: Combination of FORM = 0 and FORM = 1. See Remark 4.

INT In-plane numerical integration rule:

EQ.0: Uniformly reduced Gauss integration. NIP = PR × PS.

EQ.1: Full Gauss integration. NIP = (PR + 1) × (PS + 1).

EQ.2: Reduced, patch-wise integration rule for C^1-continuous quadratic NURBS surfaces.

NISR Number or average edge length of automatically created interpolation shell elements per each knot span in the *r*-direction. See Remark 5 and Figure 19-30.

GT.0: NINT(NISR) is the number of interpolation elements in the *r*-direction

LT.0: |NISR| is the average edge length of the interpolation

VARIABLE	DESCRIPTION
	elements in the *r*-direction.
NISS	Number or average edge length of automatically created interpolation shell elements per each knot span in the *s*-direction. See Remark 5 and Figure 19-30. GT.0: NINT(NISS) is the number of interpolation elements in the *s*-direction LT.0: |NISS| is the average edge length of the interpolation elements in the *s*-direction
IMASS	Mass matrix lumping scheme: EQ.0: Row sum. EQ.1: Diagonal weighting.
IDFNE	Element ID of first NURBS-Element within this NURBS-Patch definition.

Knot Vector Cards (for *r*-direction). The knot vector in *r*-direction with length NPR + PR + 1 is given below requiring a total of ceil[(NPR + PR + 1)/8] cards.

Cards A	1	2	3	4	5	6	7	8
Variable	RK1	RK2	RK3	RK4	RK5	RK6	RK7	RK8
Type	F	F	F	F	F	F	F	F
Default	none	none	None	none	none	none	none	none

VARIABLE	DESCRIPTION
RK*m*	Values of the univariate knot vector in *r*-direction

Knot Vector Cards (for *s*-direction). The knot vector in *s*-direction with length NPS + PS + 1 is given below requiring a total of ceil[(NPS + PS + 1)/8] cards.

Cards B	1	2	3	4	5	6	7	8
Variable	SK1	SK2	SK3	SK4	SK5	SK6	SK7	SK8
Type	F	F	F	F	F	F	F	F
Default	none	none	None	none	none	none	none	none

VARIABLE	DESCRIPTION
SK*n*	Values of the univariate knot vector in *s*-direction

Connectivity Cards. The connectivity of the control grid is a two dimensional table of NPS rows and NPR columns. This data fills the NPS sets (one set for each row) of NPR points tightly packed into ceil(NPR/8) connectivity cards, for a total of NPS × ceil(NPR/8) cards.

Cards C	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
N*k*	Control point IDs, defined via *NODE, to define the control grid
	LT.0: Control point with rotational DOFs for FORM = 4 /-4; see Remark 4.

Control Weight Cards (Optional). Additional cards are used to set a weight for each control point if WFL = 1 on Card 2. These cards have an ordering identical to the connectivity cards (Cards C).

Cards D	1	2	3	4	5	6	7	8
Variable	W1	W2	W3	W4	W5	W6	W7	W8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
W*k*	Control weights of the surface patch

Trimming Loop Title Card.

Cards E	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							
Default	trimming loop							

Trimming Loop Connectivity Cards. Define a trimming loop. See Remark 6.

Cards F	1	2	3	4	5	6	7	8
Variable	C1	C2	C3	C4	C5	C6	C7	C8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
TITLE	Title of the trimming loop
C*l*	Trimming curve ID pointing to a curve defined using *DEFINE_-NURBS_CURVE. A unique number has to be chosen.

Remarks:

1. **Shell Thickness.** The thickness of the shell is defined in *SECTION_SHELL and referenced via *PART.

2. **Element Formulation.** ELFORM = 201 must be used in *SECTION_SHELL.

3. **Thin Shell Formulation.** For thin IGA shells we do not recommend using FORM = 1. Instead we recommend using FORM = 2.

4. **Rotational Degrees of Freedom.** FORM = 4 allows the mixture of control points with and without rotational DOFs. This might be useful at the boundaries of NURBS patches where the continuity usually drops to C^0 and rotational DOFs are necessary. To indicate control points with rotational DOFs (6 DOFs/control point), the node number of the corresponding control point has to be set as the negative node ID in the connectivity Cards C. Positive node IDs indicate control points without rotational DOFs (3 DOFs/control point).

   If FORM = -4 is used, the control points at the patch boundary are automatically treated with rotational DOFs without the need to specify them explicitly in the connectivity cards (Cards C). This might be sufficient in many cases.

5. **Interpolation Elements.** The post-processing and the treatment of contact boundary conditions are presently dealt with interpolation elements, defined via interpolation nodes. These nodes and elements are automatically created, see Figure 19-30, where NISR and NISS indicate the number or average length of interpolation elements to be created per NURBS-element in the r- and s-directions, respectively.

6. **Trimming Loops.** Trimmed NURBS surface elements / patches can be analyzed by defining trimming loops. A trimming loop is formed by a set of NURBS curves defined in the surface parametric coordinate system. Each trimming curve is defined using the *DEFINE_NURBS_CURVE keyword.

   Trimming loops may be given a distinct title on Cards E and the connectivity of trimming curves defining a loop is stored on Cards F. The end and starting points of two consecutive curves *must* coincide. If the loop is defined by a single curve, the starting and end points of the curve *must* match. Furthermore, the orientation of the trimming loop is essential to define the trimmed surface. Travelling along the trimming loop, the surface on the right-hand side of the loop will be trimmed. There is no limitation on either the number of trimming curves forming a loop or the number of loops used to trim a NURBS surface element/patch.

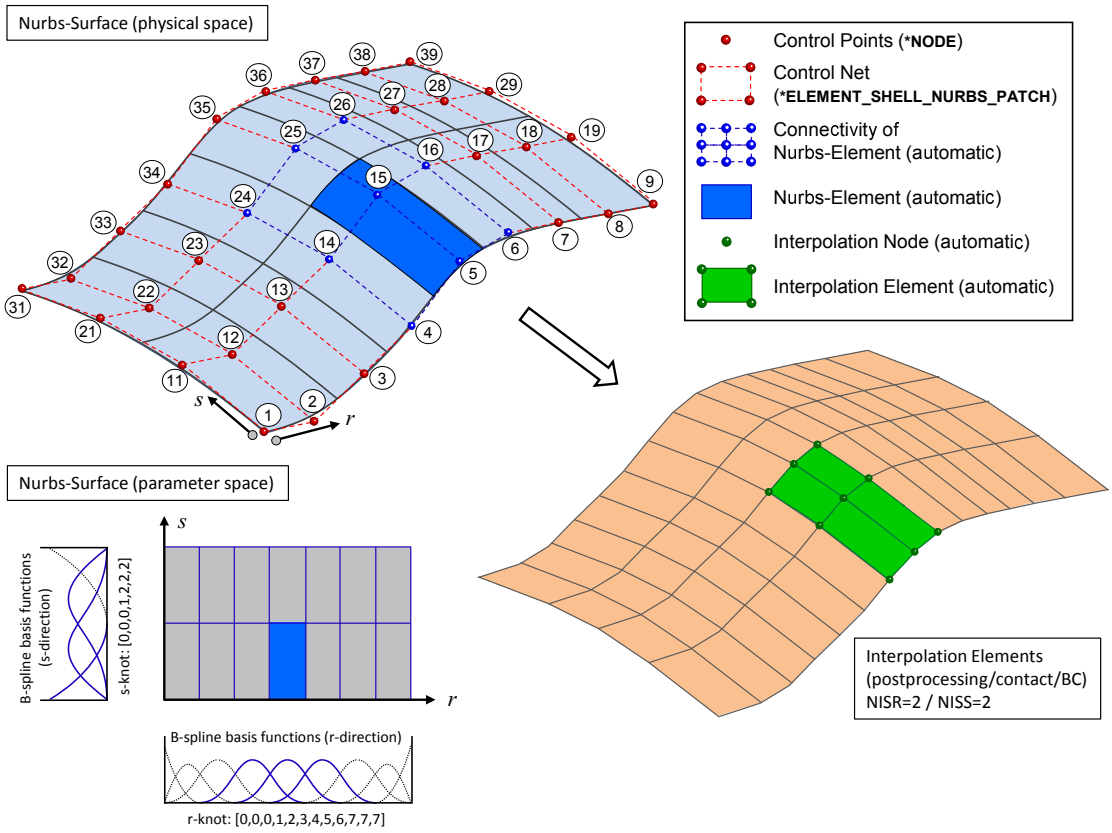


Figure 19-29. An example biquadratic NURBS surface patch and its interpolation mesh.

```
*ELEMENT_SHELL_NURBS_PATCH
$ Card 1
$---+-NPID----+--PID----+--NPR----+---PR----+--NPS----+---PS----+----7----+----8
        11         12          9          2          4          2
$ Card 2
$---+--WFL----+-FORM----+--INT----+-NISR----+-NISS----+IMASS----+----7----+----8
          1          0          1          2          2          0
$ Cards A
$rk-+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
       0.0        0.0        0.0        1.0        2.0        3.0        4.0        5.0
       6.0        7.0        7.0        7.0
$ Cards B
$sk-+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
       0.0        0.0        0.0        1.0        2.0        2.0        2.0
$ Cards C
$net+---N1----+---N2----+---N3----+---N4----+---N5----+---N6----+---N7----+---N8
         1          2          3          4          5          6          7          8
         9
        11         12         13         14         15         16         17         18
        19
        21         22         23         24         25         26         27         28
        29
        31         32         33         34         35         36         37         38
        39
$ Cards D (optional if WFL.ne.0)
$wgt+---W1----+---W2----+---W3----+---W4----+---W5----+---W6----+---W7----+---W8
       1.0        0.9        0.8        0.7        0.8        0.9        0.7        0.8
       1.0
       0.8        0.7        0.6        0.5        0.6        0.7        0.6        0.7
       0.8
       0.7        0.6        0.5        0.4        0.5        0.6        0.5        0.6
       0.7
       1.0        0.9        0.8        0.7        0.8        0.9        0.7        0.8
       1.0
```

Figure 19-30. Excerpt of the *ELEMENT_SHELL_NURBS_PATCH keyword defining the biquadratic NURBS surface patch in Figure 19-29.

*ELEMENT_SHELL_SOURCE_SINK

Purpose: Define a strip of shell elements of a single part ID to simulate a continuous forming operation. This option requires logical regular meshing of rectangular elements, which implies that the number of nodal points across the strip is constant along the length. Elements are created at the source and disappear at the sink. The advantage of this approach is that it is not necessary to define an enormous number of elements to simulate a continuous forming operation. Currently, only one source-sink definition is allowed. The boundary conditions at the source are discrete nodal point forces to keep the work piece in tension. At the sink, displacement boundary conditions are applied.

Card	1	2	3	4	5	6	7	8
Variable	NSSR	NSSK	PID					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
NSSR	Node set at source. Provide an ordered set of nodes between corner nodes, which include the corner nodes.
NSSK	Node set at sink. Provide an ordered set of nodes between corner nodes, which include the corner nodes.
PID	Part ID of work piece.

*ELEMENT_SOLID_{OPTION}

Available options include:

<BLANK>

ORTHO

DOF

TET4TOTET10

H20

H8TOH20

H27

H64

H8TOH27

H8TOH64

P21

P40

T15

T20

Stacking of options, such as ORTHO_DOF, is allowed in some cases. When combining options in this manner, check d3hsp to confirm that all options are acknowledged.

Purpose: Define three-dimensional solid elements. The type of solid element and its formulation is specified through the part ID (see *PART) and the section ID (see *SECTION_SOLID_*OPTION*). Also, a local coordinate system for orthotropic and anisotropic materials can be defined by using the ORTHO option. If extra degrees of freedom are needed, the DOF option should be used. The option TET4TOTET10 converts 4 node tetrahedrons to 10 node tetrahedrons, and H8TOH20/H8TOH27 converts 8-node hexahedrons to 20-node/27-node hexahedrons. The option H8TOH64 converts elements with linear basis functions to elements with cubic basis functions. See Remark 1.

Card Summary:

Card Sets. Include as many of the following sets of cards matching the keyword options as desired. This input ends with the next keyword ("*") card.

Card 1. This card is required.

EID	PID								

Card 2. This card is required.

N1	N2	N3	N4	N5	N6	N7	N8	N9	N10

Card 3. Include as many of this card as needed to specify the nodes of the 15-node tetrahedron, 20-node tetrahedron, 20-node hexahedron, 21-node pentahedron, 40-node pentahedron, 27-node hexahedron, and 64-node hexahedron.

N11	N12	N13	N14	N15	N16	N17	N18	N19	N20

Card 4. Include this card for the ORTHO keyword option.

A1/BETA	A2	A3		

Card 5. Include this card for the ORTHO keyword option.

D1	D2	D3		

Card 6. Include this card for the DOF keyword option.

		NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID								
Type	I	I								
Default	none	none								
Remarks	2									

VARIABLE	DESCRIPTION
EID	Element ID. A unique number must be chosen.
PID	Part ID, see *PART.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none	none	none

Additional Cards for Nodes. Additional cards as needed for the 15-node tetrahedron, 20-node tetrahedron and hexahedron, 21-node and 40 node pentahedron, and the 27-node and 64-node hexahedron.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	N11	N12	N13	N14	N15	N16	N17	N18	N19	N20
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
N1	Nodal point 1
N2	Nodal point 2
N3	Nodal point 3
⋮	⋮
N64	Nodal point 64

Orthotropic Card 1. Additional card for ORTHO keyword option.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	A1 or BETA		A2		A3					
Type	F		F		F					
Default	0.		0.		0.					
Remarks	5									

Orthotropic Card 2. Second additional card for ORTHO keyword option.

Card 5	1	2	3	4	5	6	7	8	9	10
Variable	D1		D2		D3					
Type	F		F		F					
Default	0.		0.		0.					
Remarks	5									

VARIABLE	DESCRIPTION
A1 or BETA	*x*-component of local material direction **a**, or else rotation angle BETA in degrees (see Remark 5)
A2	*y*-component of local material direction **a**
A3	*z*-component of local material direction **a**
D1	*x*-component of vector in the plane of the material vectors **a** and **b**
D2	*y*-component of vector in the plane of the material vectors **a** and **b**
D3	*z*-component of vector in the plane of the material vectors **a** and **b**

Scalar Node Card. Additional card for DOF keyword option.

Card 6	1	2	3	4	5	6	7	8	9	10
Variable			NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8
Type			I	I	I	I	I	I	I	I
Default			none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
NS1	Scalar node 1. See Remark 7.
NS2	Scalar node 2
⋮	⋮
NS8	Scalar node 8

Remarks:

1. **Automatic node generation**. The option TET4TOTET10 automatically converts 4 node tetrahedral solids to 10 node quadratic tetrahedral solids. Additional mid-side nodes are created which are shared by all tetrahedral elements that contain the edge. The user node ID's for these generated nodes are offset after the largest user node ID defined in the input file. When defining the *SECTION_SOLID keyword, the element type must be specified as either 16 or 17 which are the 10-noded tetrahedrons in LS-DYNA. Mid-side nodes created as a result of TET4TOTET10 will not be automatically added to node sets that include the nodes of the original tetrahedron. So, for example, if the tetrahedrons are to have an initial velocity, velocity initialization by part ID or part set ID using *INITIAL_VELOCITY_GENERATION is necessary as opposed to velocity initialization by node set ID using *INITIAL_VELOCITY. The option H8TOH20/H8TOH27 provides the same functionality for converting 8-node to 20-node/27-node elements. The option H8TOH64 converts 8-node to 64-node hexahedra, 4-nodel to 20-node tetraheda, and 6-node to 40-node pentahedra. Mid-side nodes and face nodes created as a result of H8TOH20/H8TO27/H8TO64 will automatically be added to node sets that include the nodes of the original hexahedron.

2. **Node numbering.** Four, six, and eight node elements are shown in Figure 19-31 where the ordering of the nodal points is shown. This ordering must be followed

or code termination with occur during the initialization phase with a negative volume message. The input of nodes on the element cards for the tetrahedral and pentahedral elements is given by:

4-noded tetrahedron	N1, N2, N3, N4, N4, N4, N4, N4, 0, 0
6-noded pentahedron	N1, N2, N3, N4, N5, N5, N6, N6, 0, 0

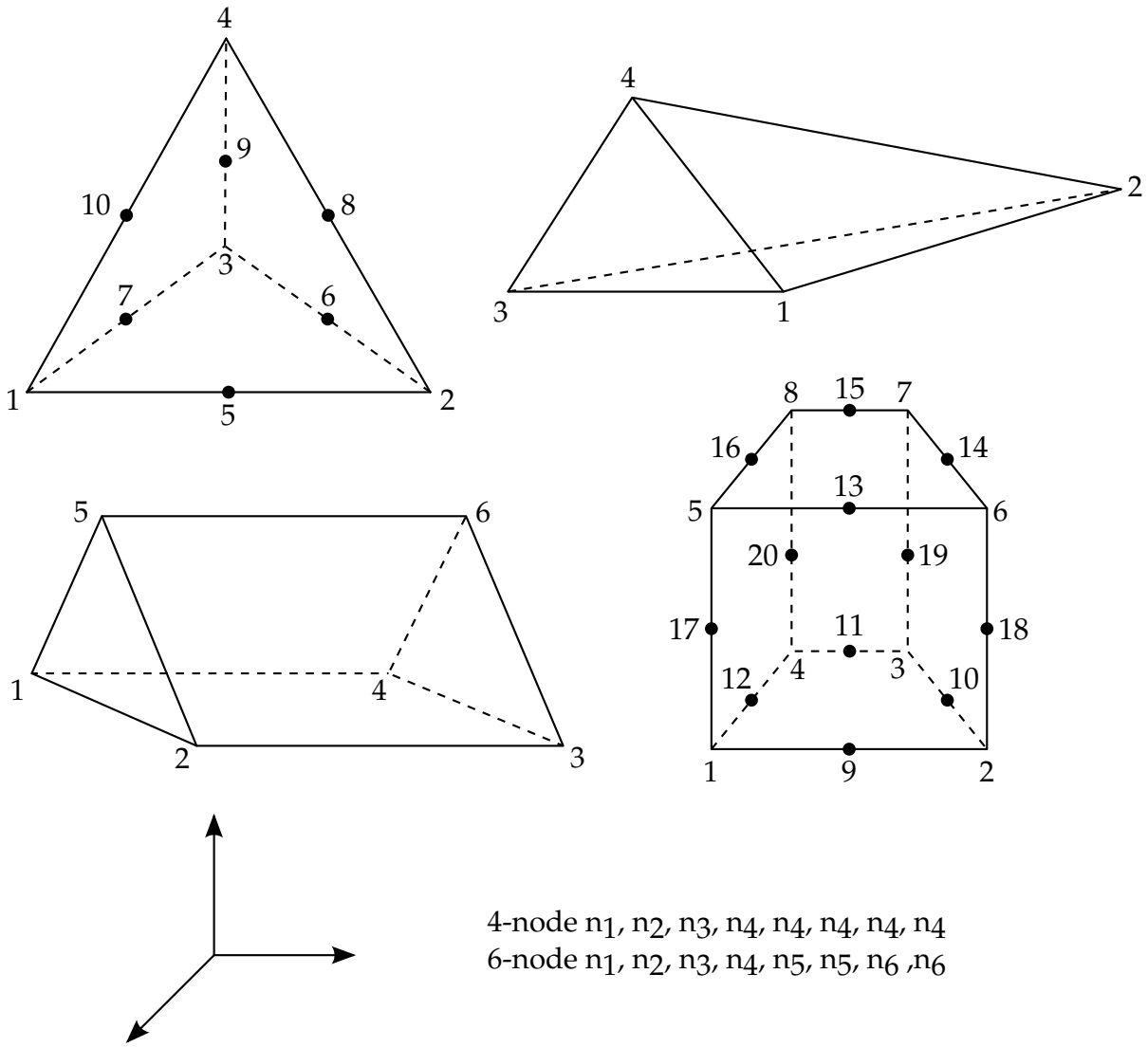


Figure 19-31. Four, six, eight, ten, and twenty node solid elements. For the hexahedral and pentahedral shapes, nodes 1-4 are on the bottom surface.

To visualize the node numbering for higher order elements, that is, options H27, H64, P21, P40, T15, and T20, see hexahedron 27, hexahedron 64, pentahedron 21, pentahedron 40, tetrahedron 15, and tetrahedron 20.

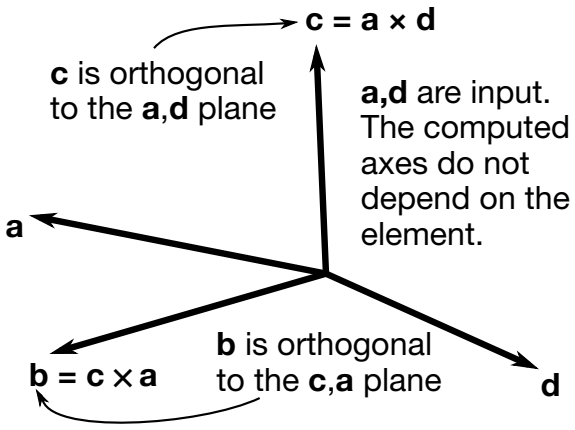


Figure 19-32. Two vectors **a** and **d** are defined and the triad is computed and stored.

3. **Degenerate solids.** If hexahedrons are mixed with tetrahedrons and pentahedrons in the input under the same part ID, *degenerate* tetrahedrons and pentahedrons are used. One problem with degenerate elements is related to an uneven mass distribution (node 4 of the tetrahedron has five times the mass of nodes 1-3) which can make these elements somewhat unstable with the default time step size. By using the control flag under the ord, *CONTROL_SOLID, automatic sorting can be invoked to treat the degenerate elements as type 10 and type 15 tetrahedral and pentahedral elements, respectively. The elements with cubic basis functions do not support degeneration.

4. **Obsolete card format.** For elements with 4-8 nodes the cards in the format of LS-DYNA versions 940-970 are still supported. The older format does not include Card 2.

Obsolete Element Solid Card.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I	I	I

5. **Local directions.** For the orthotropic and anisotropic material models the local directions may be defined on the second card following the element connectivity definition. The local directions are then computed from the two vectors such that (see Figure 19-32):

$$\mathbf{c} = \mathbf{a} \times \mathbf{d} \quad \text{and} \quad \mathbf{b} = \mathbf{c} \times \mathbf{a}.$$

These vectors are internally normalized within LS-DYNA. In this case, $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ are not calculated from AOPT and associated parameters for this element. If the

material model uses AOPT = 3, the *a* and *b*-axes will be rotated about the *c*-axis by the angle BETA specified on the material card.

If vector **d** is input as a zero length vector, then A1 is interpreted as an offset rotation angle BETA in degrees. BETA describes a rotation about the **c**-axis of the **a**-**b**-**c** coordinate system that is found through AOPT and associated parameters on the *MAT input. Note that the axes may be switched before or after the application of BETA depending on the value of MACF in the material keyword input.. This BETA angle applies to all values of AOPT, and it overrides the BETA angle on the *MAT card in the case of AOPT = 3.

6. **Stress output coordinates.** Stress output for solid elements is in the global coordinate system by default.

7. **Optional "scalar" nodes.** The scalar nodes specified on Card 6 refer to extra nodes used by certain features (usually user defined) to store additional degrees of freedom.

*ELEMENT_SOLID_NURBS_PATCH

Purpose: Define a NURBS-block element (patch) based on a cuboid grid of control points. This grid consists of NPR*NPS*NPT control points, where NPR, NPS, and NPT are the number of control points in local *r*-, *s*- and *t*-directions, respectively. The necessary shape functions are defined through three knot-vectors:

1. Knot-Vector in *r*-direction with length NPR + PR + 1
2. Knot-Vector in *s*-direction with length NPS + PS + 1
3. Knot-Vector in *t*-direction with length NPT + PT + 1

There is no limit on the size of the underlying grid to define a NURBS-block element, so the total number of necessary cards depends on the parameters given in the first two cards and is given by

$$\#\ \text{of cards} = 2 + \left\lceil \frac{\text{NPR} + \text{PR} + 1}{8} \right\rceil + \left\lceil \frac{\text{NPS} + \text{PS} + 1}{8} \right\rceil + \left\lceil \frac{\text{NPT} + \text{PT} + 1}{8} \right\rceil + \text{NPT} \times \text{NPS} \times \left\lceil \frac{\text{NPR}}{8} \right\rceil$$

where $\lceil x \rceil = \text{ceil}(x)$. (NOTE: the last term in the sum is doubled if WFL = 1, indicating that the weights are user-specified).

Card Summary:

Card 1. This card is required.

NPID	PID	NPR	PR	NPS	PS	NPT	PT

Card 2. This card is required.

WFL	NISR	NISS	NIST	IMASS	IINT		IDFNE

Card 3. Include ceil[(NPR +PR + 1)/8] of this card to define the knot vector in the local *r*-direction.

RK1	RK2	RK3	RK4	RK5	RK6	RK7	RK8

Card 4. Include ceil[(NPS +PS + 1)/8] of this card to define the knot vector in the local *s*-direction.

SK1	SK2	SK3	SK4	SK5	SK6	SK7	SK8

Card 5. Include ceil[(NPT+PT + 1)/8] of this card to define the knot vector in the local *t*-direction.

TK1	TK2	TK3	TK4	TK5	TK6	TK7	TK8

Card 6. The connectivity of the control grid is a two-dimensional table of NPT × NPS rows and NPR columns. This data fills NPT × NPS sets (one set for each row) of NPR points tightly packed into ceil(NPR/8) Connectivity Cards, for a total of NPT × NPS × ceil(NPR/8) cards.

N1	N2	N3	N4	N5	N6	N7	N8

Card 7. Additional card for WFL ≠ 0. Set a weight for each control point. These cards have an ordering identical to the Connectivity Cards (Card 6).

W1	W2	W3	W4	W5	W6	W7	W8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NPID	PID	NPR	PR	NPS	PS	NPT	PT
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
NPID	NURBS-Patch Element ID. A unique number has to be chosen
PID	Part ID, see *PART.
NPR	Number of control points in local *r*-direction.
PR	Order of polynomial of univariate NURBS basis functions in local *r*-direction.
NPS	Number of control points in local *s*-direction.
PS	Order of polynomial of univariate NURBS basis functions in local *s*-direction.

VARIABLE	DESCRIPTION
NPT	Number of control points in local *t*-direction.
PT	Order of polynomial of univariate NURBS basis functions in local *t*-direction.

Card 2	1	2	3	4	5	6	7	8
Variable	WFL	NISR	NISS	NIST	IMASS	IINT		IDFNE
Type	I	I	I	I	I	I		I
Default	0	PR	PS	PT	0	0		0

VARIABLE	DESCRIPTION
WFL	Flag for weighting factors of the control points EQ.0: all weights at the control points are set to 1.0 (B-spline basis) and no optional Card 7 sets are allowed NE.0: the weights at the control points are defined with optional Card 7 which must be defined after sets of Card 6.
NISR	Number of (automatically created) Interpolation Solid elements in local *r*-direction per created NURBS-element for visualization (post-processing) and contact.
NISS	Number of (automatically created) Interpolation Solid elements in local *s*-direction per created NURBS-element for visualization (post-processing) and contact.
NIST	Number of (automatically created) Interpolation Solid elements in local *t*-direction per created NURBS-element for visualization (postprocessing) and contact.
IMASS	Option for lumping of mass matrix: EQ.0: row sum EQ.1: diagonal weighting.
IINT	Numerical integration rule:

VARIABLE	DESCRIPTION
	EQ.0: uniformly reduced Gauss integration. NIP = PR × PS × PT.
	EQ.1: full Gauss integration. NIP = (PR+1) × (PS+1) × (PT+1).
	EQ.2: reduced, patch-wise integration rule for C1-continuous quadratic NURBS surfaces.
	EQ.3: non-uniformly reduced Gauss integration. Full integration on the boundary elements and reduced integration on the interior elements.
IDFNE	Element ID of first NURBS-Element within this NURBS-Patch definition.

Knot Vector Cards (for *r*-direction). The knot-vector in local *r*-direction with length NPR +PR + 1 is given below (up to eight values per card) requiring a total of ceil[(NPR +PR + 1)/8] cards.

Card 3	1	2	3	4	5	6	7	8
Variable	RK1	RK2	RK3	RK4	RK5	RK6	RK7	RK8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

Knot Vector Cards (for *s*-direction). The knot-vector in local *s*-direction with length NPS +PS + 1 is given below (up to eight values per card) requiring a total of ceil[(NPS +PS + 1)/8] cards.

Card 4	1	2	3	4	5	6	7	8
Variable	SK1	SK2	SK3	SK4	SK5	SK6	SK7	SK8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

Knot Vector Cards (for *t*-direction). The knot-vector in local *t*-direction with length NPT +PT + 1 is given below (up to eight values per card) requiring a total of ceil[(NPT+PT + 1)/8] cards.

Card 5	1	2	3	4	5	6	7	8
Variable	TK1	TK2	TK3	TK4	TK5	TK6	TK7	TK8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
RK*i*	Values of the univariate knot vector in local *r*-direction
SK*i*	Values of the univariate knot vector in local *s*-direction
TK*i*	Values of the univariate knot vector in local *t*-direction

Connectivity Cards. The connectivity of the control grid is a two-dimensional table of NPT × NPS rows and NPR columns. This data fills NPT × NPS sets (one *set* for each row) of NPR points tightly packed into ceil(NPR/8) Connectivity Cards, for a total of NPT × NPS × ceil(NPR/8) cards.

Card 6	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
N*i*	Control points *i* (defined via *NODE) to define the control grid

Weight Cards. Additional card for WFL ≠ 0. Set a weight for each control point. These cards have an ordering identical to the Connectivity Cards (Card 6).

Card 7	1	2	3	4	5	6	7	8
Variable	W1	W2	W3	W4	W5	W6	W7	W8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
W*i*	Weighting factors of control point *i*

Remarks:

1. **ELFORM.** ELFORM = 201 has to be used in *SECTION_SOLID.

2. **Interpolation Elements.** The post-processing and the treatment of contact boundary conditions are presently dealt with using interpolation elements, defined via interpolation nodes. These nodes and elements are automatically created, where NISR, NISS and NIST indicate the number of interpolation elements to be created per NURBS-element in the local *r*-, *s*- and *t*-direction, respectively.

Example:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ An Isogeometric Solid NURBS Example :
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*SECTION_SOLID
$#   secid    elform
         1       201
*ELEMENT_SOLID_NURBS_PATCH
$CARD 1
$#   npeid       pid       npr        pr       nps        ps       npt        pt
         1         1         3         2         6         2         3         2
$CARD 2
$#     wfl      nisr      niss      nist     imass
         0         2         2         2         0
$CARD A
$#     rk1       rk2       rk3       rk4       rk5       rk6       rk7       rk8
       0.0       0.0       0.0       1.0       1.0       1.0
$CARD B
       0.0       0.0       0.0      0.25       0.5      0.75       1.0       1.0
```

```
       1.0
$CARD C
       0.0       0.0       0.0       1.0       1.0       1.0
$CARD D
      1001      1002      1003
       …
      1052      1053      1054
$CARD E (Optional if wfl .eq.  0)
```

*ELEMENT_SOLID_PERI

Purpose: Define the connectivity of four node surface elements for Peridynamics laminate parts.

Include this card for each element. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	PID	N1	N2	N3	N4		
Type	I	I	I	I	I	I		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
EID	Element ID. A unique number must be chosen.
PID	Part ID
N*i*	Nodal point *i*

Remarks:

1. **Discretizing the laminate.** To discretize a laminate, first, construct a layer of 3D surface mesh with 4 node elements, such as shell elements, according to the geometric shape of the laminate. Then, copy this layer of mesh and move it to the middle surface of a lamina. Continue copying and moving the layer of mesh until each lamina has one layer of mesh. See Figure 19-33.

2. **Coincident nodes.** To represent in-plane material split, Peridynamics laminate elements must not share nodes. As a result, shared corners will have coincident nodes, one for each element. In other words, the total number of nodes is 4 × the total number of Peridynamics laminate elements.

3. **MPP.** We recommend for computational efficiency decomposing the model for MPP such that corresponding elements in each layer are on the same processor. To ensure this decomposition, we suggest specifying box regions that have the lumped grouping in the pfile (see Appendix O) around the corresponding elements. Figure 19-33 illustrates this decomposition for four processors.

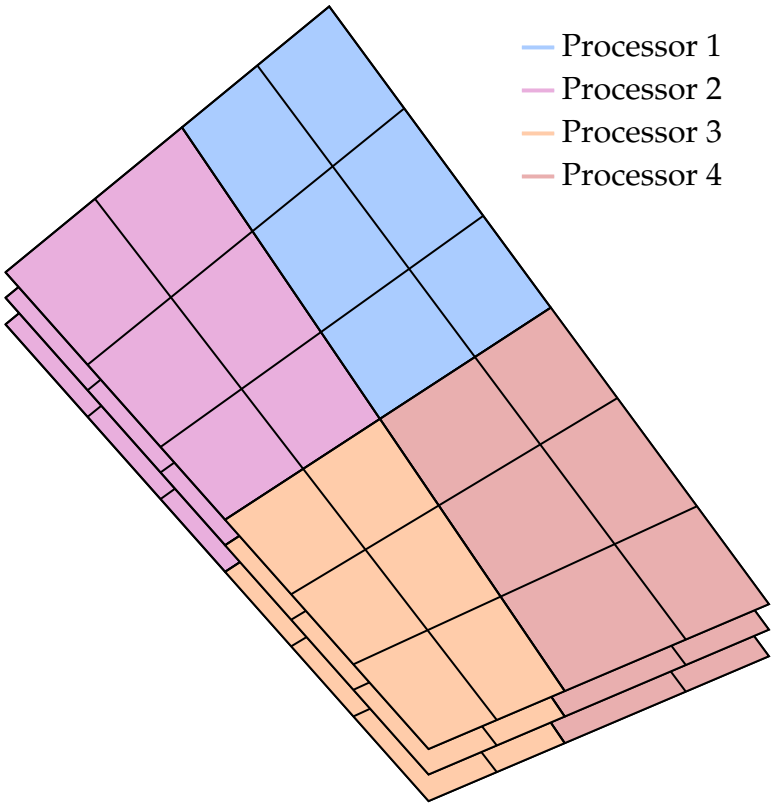


Figure 19-33. Simple example of the mesh for a three layer laminate. Each color represents an example decomposition of the mesh for MPP.

*ELEMENT_SOLID_REFERENCE

Purpose: Define a lower-order (hexahedral, tetrahedral, or pentahedral) or higher-order (10-noded tetrahedral) reference element. This keyword is a generalization of *INITIAL_FOAM_REFERENCE_GEOMETRY in the sense that the connectivity of the reference geometry is independent of the one for the current geometry. *NODE_REFERENCE specifies the corresponding reference coordinates for *ELEMENT_SOLID_REFERENCE. Furthermore, IREF on the *MAT cards does not need to be specified for this keyword. If a reference element is present, LS-DYNA automatically uses it for computing the required quantities, in particular deformation gradients. This keyword is primarily used in the processing of dynain and dynain.lsda files (see *INTERFACE_SPRINGBACK_LSDYNA) and not explicitly introduced by the user. *INTERFACE_SPRINGBACK_LSDYNA causes LS-DYNA to automatically write it for the materials that require reference information.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID									
Type	I									
Default	none									

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EID	Element ID. This ID must correspond to the same ID that is used on the associated *ELEMENT_SOLID card.
N*i*	Node number in the connectivity, referring to the corresponding ID in *NODE_REFERENCE.

*ELEMENT_SPH_{OPTION}

Available options include:

<BLANK>

VOLUME

Purpose: Define a lumped mass element assigned to a nodal point.

If the VOLUME option is used, the field for MASS is treated as particle volume. It has the same effect as giving a negative number in the MASS field.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	NID	PID	MASS		NEND					
Type	I	I	F		I					
Default	none	none	0.		0					

VARIABLE	DESCRIPTION
NID	Node ID and Element ID are the same for the SPH option.
PID	Part ID to which this node (element) belongs
MASS	Mass/volume (see Remark 1): GT.0.0: Mass value LT.0.0: Volume. The absolute value will be used as volume. The density, ρ, will be retrieved from the material card defined in PID. SPH element mass is calculated by $\lvert\text{MASS}\rvert \times \rho$.
NEND	Optional input: GT.0: *ELEMENT_SPH cards are generated between NID to NEND using current PID and MASS data.

Remarks:

1. **Axisymmetric SPH.** Axisymmetric SPH (IDIM = -2 in *CONTROL_SPH) is defined on the global XY-plane with the Y-axis as the axis of rotation. An

axisymmetric SPH element has a mass of ρA, where ρ is its density and A is the area of the SPH element. A can be approximated by the area of its corresponding axisymmetric shell element (see Figure 19-34). The mass printout in the d3hsp file is the mass per radian, that is, $\rho A x_i$. See Figure 19-35.

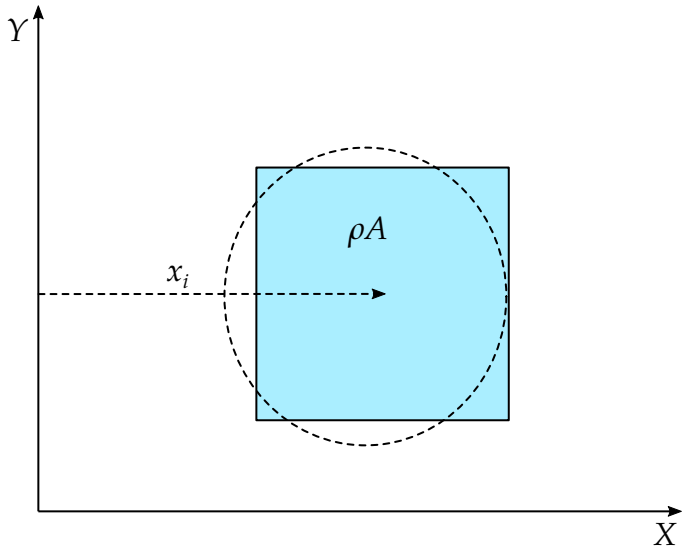


Figure 19-34. Schematic of axisymmetric SPH cross section

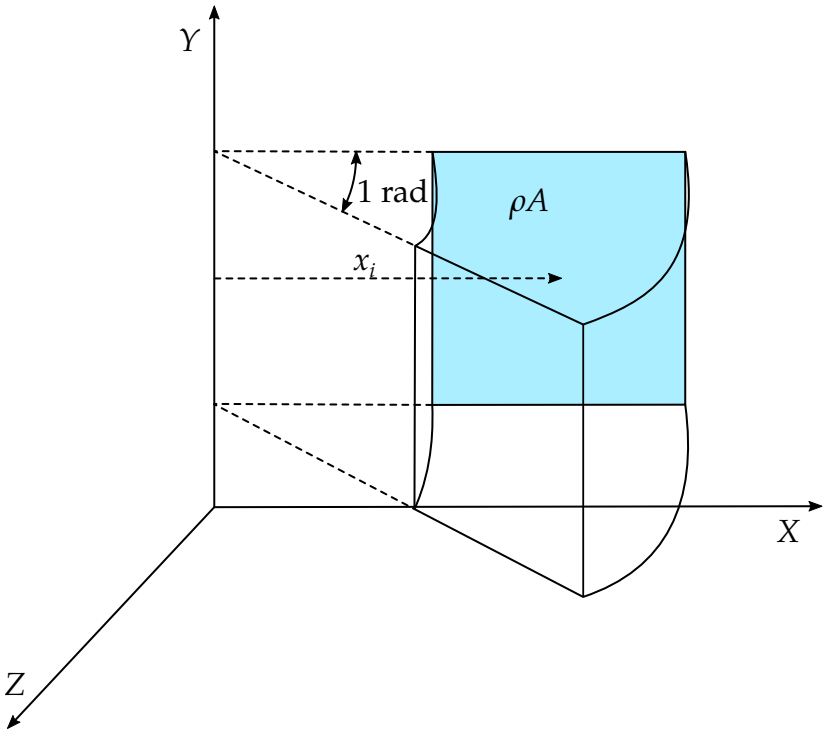


Figure 19-35. Mass printout in d3hsp

*ELEMENT_TRIM

NOTE: This keyword was replaced by *CONTROL_FORMING_TRIMMING starting in Revision 87566.

*ELEMENT_TSHELL_{OPTION}

Available options include:

<BLANK>

BETA

COMPOSITE

Purpose: Define an eight node thick shell element which is available with either fully reduced or selectively reduced integration rules. Thick shell formulations 1, 2, and 6 are plane stress elements that can be used as an alternative to the 4 node shell elements in cases where an 8-node element is desired. Thick shell formulations 3, 5 and 7 are layered solids with 3D stress updates. Formulation 5, 6, and 7 are based on an enhanced strain. The number of through-thickness integration points is specified by the user.

For orthotropic and anisotropic materials, a local material angle (variable BETA) can be defined which is cumulative with the integration point angles specified in *SECTION_-TSHELL or *PART_COMPOSITE_TSHELL.

The COMPOSITE option for *ELEMENT_TSHELL allows a unique stack of integration points for each element sharing the same part ID, and is available only when combined with *PART. The COMPOSITE option is not available in combination with *PART_COMPOSITE_TSHELL. To maintain a direct association of through-thickness integration point numbers with physical plies in the case where the number of plies varies from element to element, see Remark 5.

Card Summary:

Card 1. This card is required.

EID	PID	N1	N2	N3	N4	N5	N6	N7	N8

Card 2a. This card is included if the BETA keyword option is used.

			BETA

Card 2b. This card is included if the COMPOSITE keyword option is used.

MID1	THICK1	B1		MID2	THICK2	B2	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	EID	PID	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none	none	none
Remarks			1							

VARIABLE	DESCRIPTION
EID	Element ID. Unique numbers have to be used.
PID	Part ID, see *PART.
N*i*	Node *i*

Beta Card. Additional card for the BETA keyword option.

Card 2a	1	2	3	4	5	6	7	8	9	10
Variable									BETA	
Type									F	
Default									0.	
Remarks									4	

VARIABLE	DESCRIPTION
BETA	Orthotropic material base offset angle (see Remark 4). The angle is given in degrees. If blank, the default is set to zero.

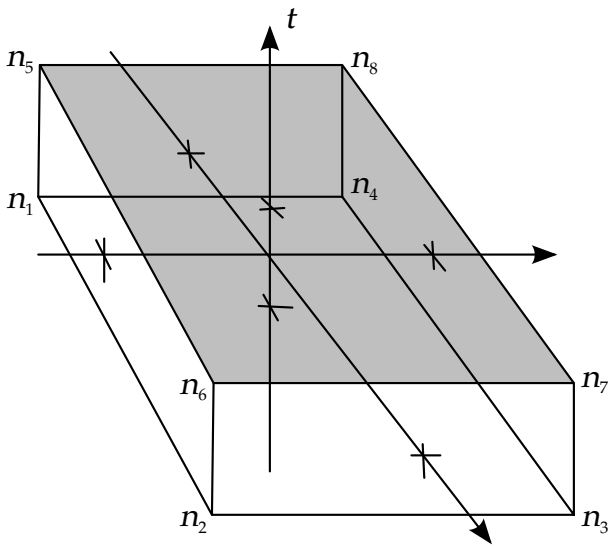


Figure 19-36. 8-node Thick Shell Element.

Composite Cards. Additional card for the COMPOSITE keyword option. Values of material ID, thickness, and material angle for each through-thickness integration point of a composite shell are defined using these cards, and these values override values specified elsewhere. The integration point data should be given sequentially starting with the bottommost integration point. The total number of integration points is determined by the number of entries on these cards. The total thickness is the distance between the top and bottom surface as determined by the element connectivity, so the THICK*i* values are scaled to fit the element. Define as many cards as needed to define all the through-thickness integration points. The fourth field must be zero or blank to be interpreted as a Card 2b.

Card 2b	1	2	3	4	5	6	7	8
Variable	MID1	THICK1	B1		MID2	THICK2	B2	
Type	I	F	F		I	F	F	

VARIABLE	DESCRIPTION
MID*i*	Material ID of integration point *i*, see the *MAT_… cards.
THICK*i*	Thickness of integration point *i*
B*i*	Material angle of integration point *i*

Remarks:

1. **Orientation.** Extreme care must be used in defining the connectivity to ensure proper orientation of the through-thickness direction. For a hexahedron, nodes n_1 to n_4 define the lower surface, and nodes n_5 to n_8 define the upper surface. For a pentahedron, nodes n_1, n_2, n_3 form the lower triangular surface and the eight variables N1 to N8 should be defined using nodes n_1, n_2, n_3, n_3, n_4, n_5, n_6, n_6, respectively. Note that node n_3 and node n_6 are each repeated.

2. **Integration.** Element formulations 1 and 5 (see *SECTION_TSHELL), use one point integration and the integration points then lie along the t-axis as shown in Figure 19-36. Element formulations 2 and 3 use two by two selective reduced integration in each layer.

3. **Stress Output.** The stresses for thick shell elements are output in the global coordinate system.

4. **Local Coordinate System.** To allow the orientation of orthotropic and anisotropic materials to be defined for each thick shell element, a beta angle can be defined. This beta angle is used with the AOPT parameter and associated data on the *MAT card to determine an element reference direction for the element.

   The AOPT data defines a coordinate system and the BETA angle defines a subsequent rotation about the element normal to determine the element reference system. For composite modeling, each layer in the element can have a unique material direction by defining an additional rotation angle for the layer, using either the ICOMP and Bi fields on *SECTION_TSHELL or the Bi parameter on *PART_COMPOSITE_TSHELL. The material direction for layer i is then determined by a rotation angle, θ_i.

5. **Assignment of Zero Thickness to Integration Points.** The ability to assign zero- thickness integration points in the stacking sequence allows the number of integration points to remain constant even as the number of physical plies varies from element to element and eases post-processing since a particular integration point corresponds to a physical ply. Such a capability is important when one or more of the physical plies are not continuous across a part. To represent a missing ply in *ELEMENT_TSHELL_COMPOSITE, set THICKi to 0.0 for the corresponding integration point and additionally, set MID to -1.

   When post-processing the results using LS-PrePost version 4.5, read both the keyword deck and d3plot database into the code and then select Option→N/A gray fringe. Then, when viewing fringe plots for a particular integration point (FriComp→IPt →intpt#), the element will be grayed out if the selected integration point is missing (or has zero thickness) in that element.

6. **PID with COMPOSITE keyword option.** When using the COMPOSITE keyword option, the PID on Card 1 must reference a *PART card that does not also use the COMPOSITE keyword option.

*END

The *END command is optional and signals the conclusion of a keyword input file. Data in a keyword file beyond a *END command are not read by LS-DYNA.

*EOS

Please see LS-DYNA Keyword User's Manual, Volume II (Material Models).

*FATIGUE

The keyword *FATIGUE provides a way of defining and solving fatigue and durability analysis problems. The keyword cards in this section are defined in alphabetical order:

FATIGUE_*{OPTION}

*FATIGUE_FAILURE

*FATIGUE_LOADSTEP

*FATIGUE_MEAN_STRESS_CORRECTION

*FATIGUE_MULTIAXIAL

*FATIGUE_SUMMATION

***FATIGUE**_*{OPTION}*

Available options include:

<BLANK> (see Remark 5)

D3PART

D3PLOT

ELOUT

STATIC

Purpose: Perform fatigue analysis on parts or structures to obtain the cumulative damage ratio and expected fatigue life. See Remark 5.

Card Summary:

Card 1. Include this card when the keyword option is unused (<BLANK>) or is STATIC. See Remark 5.

PID	PTYPE						

Card 2. Include this card when the keyword option is unused (<BLANK>) or is STATIC. See Remark 5.

DT							

Card 3. This card is required.

STRSN	INDEX	RESTRT	TEXPOS	DMGMIN			

Card 4. Include this card when the keyword option STATIC is used or RESTRT = 1. When the option STATIC is used, this card is required. For RESTRT = 1, this card is optional.

FILENAME

Card 5. Include this card when the keyword option STATIC is used.

LCID	NSTATE						

Data Card Definitions:

Include this card when the keyword option is unused (<BLANK>) or is STATIC.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	PTYPE						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
PID	Part ID, part set ID, or element (solid, shell, beam, thick shell) set ID. EQ.0: Fatigue analysis is performed on the whole structure.
PTYPE	Type of PID: EQ.0: Part (default) EQ.1: Part set EQ.2: Solid set EQ.3: Beam set EQ.4: Shell set EQ.5: Thick shell set

Include this card when the keyword option is unused (<BLANK>) or is STATIC.

Card 2	1	2	3	4	5	6	7	8
Variable	DT							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
DT	Time step for saving the stress/strain data in transient analysis

Card 3	1	2	3	4	5	6	7	8
Variable	STRSN	INDEX	RESTRT	TEXPOS	DMGMIN			
Type	I	I	I	F	F			
Default	0	0	0	0.0	0.0			

VARIABLE	DESCRIPTION
STRSN	Type of fatigue analysis variable: EQ.0: Stress (default) EQ.1: Strain
INDEX	Stress/strain index for performing fatigue analysis: EQ.0: von Mises stress/strain EQ.1: Absolute maximum principal stress/strain (see Remark 6) EQ.2: Maximum shear stress/strain EQ.3: Maximum principal stress/strain (see Remark 6) EQ.-1: *xx*- stress/strain component EQ.-2: *yy*- stress/strain component EQ.-3: *zz*- stress/strain component EQ.-4: *xy*- stress/strain component EQ.-5: *yz*- stress/strain component EQ.-6: *zx*- stress/strain component
RESTRT	Restart options. This flag determines whether to save the results of an LS-DYNA transient analysis or use the stress/strain time history data created in previous runs. The fatigue analysis uses the history data from the transient analysis. See Remark 7. EQ.0: Initial run. A binary file of the type specified with keyword option is EQ.1: Restart with existing stress/strain binary database

VARIABLE	DESCRIPTION
TEXPOS	Exposure time. If this is 0, the exposure time is the same as ENDTIM in *CONTROL_TERMINATION.
DMGMIN	Minimum fatigue damage ratio for parts undergoing fatigue analysis: EQ.0.0: No change on computed fatigue damage ratio LT.0.0: For each part, the minimum fatigue damage ratio dumped to D3FTG is |DMGMIN| × the computed nonzero minimum fatigue damage ratio computed on the current part. GT.0.0: For each part, the minimum fatigue damage ratio dumped to D3FTG is DMGMIN.

Stress/Strain Binary Database Card. Card 4 is required when the keyword option STATIC is used. Card 4 is optional for RESTRT = 1.

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A							

VARIABLE	DESCRIPTION
FILENAME	Path and name of existing stress/strain binary database (such as d3part, d3plot or binout). For the STATIC keyword option, the database file must be a d3plot file.

STATIC Parameters Card. Include this card when the keyword option STATIC is used.

Card 5	1	2	3	4	5	6	7	8
Variable	LCID	NSTATE						
Type	I	I						

VARIABLE	DESCRIPTION
LCID	Load curve ID to describe how the real load magnitude changes with time

VARIABLE	DESCRIPTION
NSTATE	State number in d3plot for the state that provides the reference static stress/strain for *FATIGUE_STATIC

Remarks:

1. **D3PART.** When D3PART is used, the stress/strain data are provided by the D3PART database.

2. **D3PLOT.** When D3PLOT is used, the stress/strain data are provided by the D3PLOT database.

3. **ELOUT.** When ELOUT is used, the stress/strain data are provided by the ELOUT database. The keyword *DATABASE_ELOUT must be included to define time step for writing the stress/strain data. In addition, BINARY must be set to 2 in *DATABASE_ELOUT so that the data is written to a binary database binout. The keyword(s) *DATABASE_HISTORY_*OPTION* with the element type specified in the option must be included to define the elements where the stress/strain data are needed, and the fatigue analysis is requested.

4. **STATIC.** With the STATIC keyword option, the solver generates stress/strain time history based on the reference static stress/strain from a binary plot file defined in Card 4 (e.g., d3plot) and a load curve which serves as scale factors for the reference static stress/strain. This approach is limited to proportional loading case where the location and direction of the loading is unchanged. In addition, it is limited to low stress, high cycle cases where the stress response is in the linear range.

5. **Current restrictions.** Currently, the fatigue analysis is performed based on the D3PART, D3PLOT, or ELOUT databases or based on using the STATIC approach. Thus, a keyword option is always needed.

6. **Absolute maximum versus maximum principal stress/strain.** Let σ_1, σ_2, and σ_3 be the three principal stresses with $\sigma_1 \geq \sigma_2 \geq \sigma_3$. For INDEX = 1, the absolute maximum principal stress is used, meaning

$$\sigma = \begin{cases} \sigma_3 & \text{if } |\sigma_3| > |\sigma_1| \\ \sigma_1 & \text{otherwise} \end{cases}$$

   For INDEX = 3, the maximum principal stress is used, meaning $\sigma = \sigma_1$. Similarly, this same definition applies to strain with the principal strains replacing the principal stresses.

7. **Restarts.** When RESTRT = 1, LS-DYNA reads the existing binary database (by default, d3plot if the option D3PLOT is used, d3part if the option D3PART is

used, or binout if the option ELOUT is used) to obtain the stress/strain time history data for the fatigue analysis. This binary database should be in the same directory as the current run. Otherwise, the path and name of the binary database should be defined by Card 4. When running a restart with the ELOUT keyword option, *DATABASE_ELOUT must be removed from the input deck to avoid overwriting the original ELOUT database (which is saved in binout).

Note that for STATIC, if RESTRT = 0, a transient analysis is performed but the file indicated in Card 4 is used as the reference static stress/strain distribution for the fatigue analysis.

8. **Exposure time.** The exposure time is ENDTIM in *CONTROL_TERMINATION, unless it is defined by a nonzero TEXPOS in Card 3. The exposure time is needed for computing the expected fatigue life, which is given as "exposure time" / "cumulative damage ratio". The expected fatigue life is only computed for the case without initial fatigue damage ratio. This is because for the case with initial fatigue damage ratio (see *INITIAL_FATIGUE_DAMAGE_RATIO), the cumulative damage ratio includes the damage ratio from preceding loads. For structures subjected to long term periodic fatigue loading, the computation can be run for this small fraction of the exposure time using a small ENDTIM in *CONTROL_TERMINATION (if the stress / strain response in this period is a good representation for the whole exposure time). The computed cumulative damage ratio is scaled accordingly to provide the cumulative damage ratio for the whole exposure time.

9. **Minimum damage ratio.** The minimum damage ratio dumped to d3ftg can be defined and adjusted by DMGMIN. For some parts, some elements may have 0 damage ratio from computation if they are rigid bodies or they have minimum stress (e.g. the stress is lower than the fatigue limit on the SN curve). The 0 damage ratio may cause a mathematical problem if you want to calculate the fatigue life of the element, or take log scale on the damage ratio for better visualization of the fatigue damage distribution. DMGMIN can define a nonzero minimum damage ratio for the parts undergoing fatigue analysis to avoid this problem.

10. **Results.** The results (fatigue life, cumulative damage ratio, etc.) are saved in binary plot database d3ftg.

*FATIGUE_FAILURE

Purpose: Determine the treatment of elements that failed due to fatigue in subsequent analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	IFAILURE	DRATIO						
Type	I	F						
Default	0	1.0						

VARIABLE	DESCRIPTION
IFAILURE	Treatment of elements failed due to fatigue: EQ.0: keep the elements in the model. EQ.1: delete the elements from the model.
DRATIO	Threshold value of cumulative damage ratio for an element to be considered failed

Remarks:

1. **Cumulative Damage Ratio.** This keyword controls the treatment of elements which fail during fatigue loading. Usually an element fails if the cumulative damage ratio is ≥ 1. For shell elements and thick shell elements, it is assumed that the elements fail if the cumulative damage ratio from one of the integration points through thickness is ≥ 1. For added safety, a smaller DRATIO may be used, such as DRATIO = 0.5.

2. **Deleted Elements.** If IFAILURE = 1, all the elements with cumulative damage ratio ≥ DRATIO, will be deleted from the model. They will not participate in subsequent computations.

3. **Applicability.** This keyword is only applicable to time domain fatigue analysis.

*FATIGUE_LOADSTEP

Purpose: Define load steps for fatigue analysis.

Load Step Card. This card may be repeated if multiple load steps are present.

Card 1	1	2	3	4	5	6	7	8
Variable	TSTART	TEND	TEXPOS					
Type	F	F	F					
Default	none	none	0.0					

VARIABLE	DESCRIPTION
TSTART	Start time of current load step
TEND	End time of current load step
TEXPOS	Exposure time of current load step EQ.0.0: set to TEND – TSTART (default).

Remarks:

1. **Fatigue Calculation and Exposure Time.** This keyword is applicable to time domain fatigue analysis. For each fatigue load step, LS-DYNA performs a fatigue calculation only for the stress / strain history extracted between TSTART and TEND. The calculated cumulative damage ratio is multiplied by a scale factor (= TEXPOS / (TEND – TSTART)) to provide a cumulative damage ratio for the real load step (for exposure time of TEXPOS). It is assumed that the stress / strain cycles between TSTART and TEND are a good representative of the stress / strain response for the whole exposure time TEXPOS. TEXPOS defined in this Card overrides the same variable defined in Card 3 in the keyword *FATIGUE_*{OPTION}*.

2. **Multiple Load Steps.** Card 1 can be repeated if there are multiple fatigue load steps. The cumulative damage ratios from each load step are summed up to provide the final cumulative damage ratio to be saved in the binary plot database d3ftg. If initial fatigue damage ratios are defined (by the keyword *INITIAL_FATIGUE_DAMAGE_RATIO), they are also added to the final cumulative damage ratio.

*FATIGUE_MEAN_STRESS_CORRECTION

Purpose: Define mean stress correction method for fatigue analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	METHOD							
Type	I							
Default	0							

Card 2	1	2	3	4	5	6	7	8
Variable	MID	SIGMA						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
METHOD	Mean stress correction method: EQ.-1: Interpolation on multiple SN curves EQ.0: Goodman equation EQ.1: Soderberg equation EQ.2: Gerber equation EQ.3: Goodman tension only EQ.4: Gerber tension only EQ.5: Morrow equation (for correction on SN curve) EQ.11: Morrow equation (for correction on EN curve) EQ.12: Smith-Watson-Topper equation
MID	Material ID for which the current mean stress correction method is applied.
SIGMA	Ultimate tensile strength, σ_u, to be used in the Goodman equation

VARIABLE	DESCRIPTION
	(METHOD = 0, 3) or the Gerber equation (METHOD = 2, 4), yield strength, σ_y, to be used in the Soderberg equation (METHOD = 1), or true fracture strength, σ_f, to be used in the Morrow equation (METHOD = 5)

Remarks:

1. **Fatigue analysis types.** This keyword can be applied in both time domain and frequency domain fatigue analysis. The method options 0, 1, 2, 3, and 4 are used for stress-based fatigue analysis; the method options 11 and 12 are used for strain-based fatigue analysis.

2. **Multiple material models.** Card 2 can be repeated if there are multiple material models in the input deck and they have different SIGMA values.

*FATIGUE_MULTIAXIAL

Purpose: Define criterion for multiaxial fatigue analysis. Many mechanical components experience multiaxial cyclic loadings during their service life. Compared with the uniaxial fatigue problem, the multiaxial fatigue problem is more complex due to the complex stress / strain states and loading histories. Fatigue cracks initiate and grow on certain critical planes in the material.

Card 1	1	2	3	4	5	6	7	8
Variable	MAXIAL	NPLANE						
Type	I	I						
Default	0	18						

VARIABLE	DESCRIPTION
MAXIAL	Multiaxial fatigue analysis criterion: EQ.0: Fatigue analysis using equivalent stress or strain index (defined by INDEX in *FATIGUE) EQ.1: Fatigue analysis on multiple planes EQ.2: Fatigue analysis on critical plane which is determined by the highest 1st principal stress or strain
NPLANE	Number of planes for fatigue analysis (for MAXIAL = 1 only)

Remarks:

1. **Multiaxial Fatigue Models.** Several criteria have been implemented in LS-DYNA to run multiaxial fatigue problems.

   a) With MAXIAL = 0, the multiaxial fatigue problem can be simplified to uniaxial fatigue problem using an equivalent scalar stress or strain index, such as Von-Mises stress / strain and maximum principal stress / strain. The equivalent scalar stress or strain index is defined by INDEX in *FATIGUE.

   b) With MAXIAL = 1, fatigue analysis can be performed on multiple predefined planes. The highest fatigue damage value from the planes is taken as the fatigue damage value on that element. The number of planes is defined by NPLANE, with the default value of 18. The inclination angles of the

planes range from 0 to NPLANE × $\delta\theta$ where $\delta\theta = 180°/\text{NPLANE}$. This criterion is only valid for shell elements with plane stress condition (and this is true for shell elements on the surface of the structure components).

c) With MAXIAL = 2, fatigue analysis can be performed on a critical plane. LS-DYNA goes through whole time history of stress / strain states and computes the principal stress / strain at each time point. At the time point which gives the highest 1st principal stress / strain, orientation of the principal plane is computed, and this defines the critical plane for fatigue damage computation for the whole time history.

*FATIGUE_SUMMATION

Purpose: Cause LS-DYNA to read in existing fatigue databases defined by *INITIAL_-FATIGUE_DAMAGE_RATIO and sum up the damage ratio results from them to obtain the final cumulative damage ratio. The final cumulative damage ratio results are dumped to a new d3ftg database if BINARY in *DATABASE_FREQUENCY_BINARY_D3FTG is set to 1.

*FREQUENCY_DOMAIN

Purpose: The keyword *FREQUENCY_DOMAIN provides a way of defining and solving frequency domain vibration and acoustic problems. The keyword cards in this section are defined in alphabetical order:

*FREQUENCY_DOMAIN_ACCELERATION_UNIT

FREQUENCY_DOMAIN_ACOUSTIC_BEM_*{OPTION}

*FREQUENCY_DOMAIN_ACOUSTIC_DIRECTIVITY

*FREQUENCY_DOMAIN_ACOUSTIC_FEM

*FREQUENCY_DOMAIN_ACOUSTIC_FREQUENCY

FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT_*{OPTION}

*FREQUENCY_DOMAIN_ACOUSTIC_INCIDENT_WAVE

*FREQUENCY_DOMAIN_ACOUSTIC_SOUND_SPEED

*FREQUENCY_DOMAIN_FRF

FREQUENCY_DOMAIN_LOCAL_*{OPTION}

FREQUENCY_DOMAIN_MODE_*{OPTION}

*FREQUENCY_DOMAIN_PATH

FREQUENCY_DOMAIN_RANDOM_VIBRATION_*{OPTION}

*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM

*FREQUENCY_DOMAIN_SEA_CONNECTION

*FREQUENCY_DOMAIN_SEA_INPUT

*FREQUENCY_DOMAIN_SEA_SUBSYSTEM

*FREQUENCY_DOMAIN_SSD

*FREQUENCY_DOMAIN_ACCELERATION_UNIT

Purpose: LS-DYNA's default behavior is to *assume* that accelerations are derived:

$$[\text{acceleration unit}] = \frac{[\text{length unit}]}{[\text{time unit}]^2}.$$

This card extends LS-DYNA to support *other* units for acceleration.

Card 1	1	2						
Variable	UNIT	UMLT						
Type	I	F						

VARIABLE	DESCRIPTION
UNIT	Flag for acceleration unit conversion: EQ.0: use [length unit]/[time unit]^2 as unit of acceleration. EQ.1: use g as unit for acceleration, and SI units (Newton, kg, meter, second, etc.) elsewhere. EQ.2: use g as unit for acceleration, and Engineering units (lbf, lbf × second^2/inch, inch, second, etc.) elsewhere. EQ.3: use g as unit for acceleration, and units (kN, kg, mm, ms, GPa, etc.) elsewhere. EQ.4: use g as unit for acceleration, and units (Newton, ton, mm, second, MPa, etc.) elsewhere. EQ.-1: use g as unit for acceleration and provide the multiplier for converting g to [length unit]/[time unit]^2.
UMLT	Multiplier for converting g to [length unit]/[time unit]^2 (used only for UNIT = -1).

Remarks:

LS-DYNA uses consistent units. With *consistent units* acceleration is defined using:

$$[\text{acceleration unit}] = \frac{[\text{length unit}]}{[\text{time unit}]^2}.$$

However, it is the convention of many industries to use g (gravitational acceleration on the Earth's surface) as the base unit for acceleration. Usually, data from vibration tests,

both random and sine sweep, are expressed in systems for which *g* is the unit of acceleration. With this keyword, LS-DYNA supports such conventions. Internally, LS-DYNA implements this keyword by converting the input deck into consistent units, and then proceeding with the calculation as usual. However, results are output in the unit system specified with this keyword.

*FREQUENCY_DOMAIN_ACOUSTIC_BEM_{OPTION1}_{OPTION2}

Available options include:

<BLANK>

ATV (see Remark 15)

MATV (see Remark 16)

HALF_SPACE

PANEL_CONTRIBUTION

POWER (see Remark 17)

Purpose: Activate the boundary element method in the frequency domain for acoustic problems. Unless the BEM=**filename** option is included in the LS-DYNA command line, this keyword is ignored.

Card Summary:

Card 1. This card is required.

RO	C	FMIN	FMAX	NFREQ	DTOUT	TSTART	PREF

Card 2. This card is required.

NSIDEXT	TYPEXT	NSIDINT	TYPINT	FFTWIN	TRSLT	IPFILE	IUNITS

Card 2.1a. Include this card if FFTWIN = 5 or 8.

T_HOLD	DECAY						

Card 2.1b. Include this card if FFTWIN = 7.

ALPHA							

Card 3. This card is required.

METHOD	MAXIT	TOLITR	NDD	TOLLR	TOLFCT	IBDIM	NPG

Card 4. This card is required.

	NBC	RESTRT	IEDGE	NOEL	NFRUP	VELOUT	DBA

Card 4.1. Include NBC of this card.

SSID	SSTYPE	NORM	BEMTYP	LC1	LC2		

Card 5. Include this card if using the PANEL_CONTRIBUTION keyword option.

NSIDPC	PCTYPE						

Card 6. Include this card if using the HALF_SPACE keyword option.

PID	RC						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	RO	C	FMIN	FMAX	NFREQ	DTOUT	TSTART	PREF
Type	F	F	F	F	I	F	F	F
Default	none	none	none	none	0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
RO	Fluid density
C	Sound speed of the fluid: GT.0: Real constant sound speed LT.0: |C| is the load curve ID, which defines the frequency-dependent complex sound speed. See *FREQUENCY_DOMAIN_ACOUSTIC_SOUND_SPEED.
FMIN	Minimum value of output frequencies
FMAX	Maximum value of output frequencies
NFREQ	Number of output frequencies
DTOUT	Time interval between writing velocity or acceleration, and pressure at boundary elements in the binary file, to be processed at the end of a transient simulation.

VARIABLE	DESCRIPTION
TSTART	Start time for recording velocity or acceleration in a simulation. See Remark 1.
PREF	Reference pressure used to output pressure in dB to the file Press_dB. If PREF = 0.0, the Press_dB file is not generated. A file called Press_Pa is generated and contains the pressure at the output nodes (see Card 2). See Remark 2.

Card 2	1	2	3	4	5	6	7	8
Variable	NSIDEXT	TYPEXT	NSIDINT	TYPINT	FFTWIN	TRSLT	IPFILE	IUNITS
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
NSIDEXT	Node ID, node set ID, or segment set ID of output exterior field points.
TYPEXT	Output exterior field point type: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID
NSIDINT	Node ID, node set ID, or segment set ID of output interior field points.
TYPINT	Output interior field point type: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID
FFTWIN	FFT windows (Default = 0). See Remark 3. EQ.0: Rectangular window EQ.1: Hanning window

VARIABLE	DESCRIPTION
	EQ.2: Hamming window
	EQ.3: Blackman window
	EQ.4: Raised cosine window
	EQ.5: Exponential window
	EQ.6: Triangular window
	EQ.7: Kaiser window
	EQ.8: Auto exponential window
TRSLT	Flag to output time domain results (see Remark 4):
	EQ.0: Do not output time domain results.
	EQ.1: Request time domain results (Press_Pa_t gives absolute value pressure as a function of time).
	EQ.2: Requiest time domain results (Press_Pa_t gives real value pressure as a function of time).
IPFILE	Flag for output files (default = 0):
	EQ.0: Press_Pa (magnitude of pressure as a function of frequency), Press_dB (sound pressure level or SPL as a function of frequency), and bepres (ASCII database file for LS-Prepost) are provided.
	EQ.1: Press_Pa_real (the real part of the pressure as a function of frequency) and Press_Pa_imag (the imaginary part of the pressure as a function of frequency) are provided, in addition to Press_Pa, Press_dB, and bepres.
IUNITS	Flag for unit changes (see Remark 5):
	EQ.0: Do not apply a unit change.
	EQ.1: Use MKS units; thus, no change is needed.
	EQ.2: Use units: lbf × s^2/in, inch, s, lbf, psi, etc., and change to MKS in the BEM acoustic computation.
	EQ.3: Use units: kg, mm, ms, kN, GPa, etc., and change to MKS in the BEM acoustic computation.
	EQ.4: Use units: ton, mm, s, N, MPa, etc., and change to MKS in the BEM acoustic computation.

Exponential FFT Window. Additional card for FFTWIN = 5 or 8.

Card 2.1a	1	2	3	4	5	6	7	8
Variable	T_HOLD	DECAY						
Type	F	F						
Default	0.0	0.02						

VARIABLE	DESCRIPTION
T_HOLD	Hold-off period before the exponential window. The length of the hold-off period should coincide with the pre-trigger time to reduce the effects of noise in the captured time domain data. It is only used when FFTWIN = 5. For FFTWIN = 8 (auto exponential window), T_HOLD is ignored. LT.0.0: |T_HOLD| is a curve ID defining the specific hold-off period for each panel. The abscissa value is the panel ID, and the ordinate value gives the hold-off period for that panel. This is useful when the system includes multiple panels, and each panel experiences impact at a different time (thus, the hold-off period differs for each panel).
DECAY	Decay ratio at the end of capture duration. For example, DECAY = 0.02 means the vibration must decay to 2% of its amplitude within the capture duration.

Kaiser FFT Window. Additional card for FFTWIN = 7.

Card 2.1b	1	2	3	4	5	6	7	8
Variable	ALPHA							
Type	F							
Default	0.0							

VARIABLE	DESCRIPTION
ALPHA	A non-negative real number that determines the shape of the

VARIABLE	DESCRIPTION
	Kaiser window and, therefore, controls the tradeoff between main-lobe width and side lobe labels of the spectral leakage pattern.

Card 3	1	2	3	4	5	6	7	8
Variable	METHOD	MAXIT	TOLITR	NDD	TOLLR	TOLFCT	IBDIM	NPG
Type	I	I	F	I	F	F	I	I
Default	0	100	10^{-4}	1	10^{-6}	10^{-6}	200	2

VARIABLE	DESCRIPTION
METHOD	Method used in the acoustic analysis: EQ.0: Rayleigh method (very fast). See Remark 6. EQ.1: Kirchhoff method coupled to FEM for acoustics (*MAT_-ACOUSTIC). See Remark 6. EQ.2: Variational indirect BEM EQ.3: Collocation BEM EQ.4: Collocation BEM with Burton-Miller formulation for exterior problems (no irregular frequency phenomenon)
MAXIT	Maximum number of iterations for the iterative solver (default = 100) if METHOD ≥ 2.
TOLITR	Tolerance for the iterative solver. The default value is 10^{-4}.
NDD	Number of domain decompositions. It is used for saving memory. For large problems, the boundary mesh is decomposed into NDD domains for less memory allocation. NDD = 0 invokes using an internally-determined, recommended domain decomposition. NDD is only used if METHOD ≥ 2. See Remark 7.
TOLLR	Tolerance for low-rank approximation of a dense matrix (default = 10^{-6}).
TOLFCT	Tolerance in the factorization of the low-rank matrix (default = 10^{-6}).

VARIABLE	DESCRIPTION
IBDIM	Inner iteration limit in GMRES iterative solver (default = 1000).
NPG	Number of Gauss integration points (default = 2).

Card 4	1	2	3	4	5	6	7	8
Variable		NBC	RESTRT	IEDGE	NOEL	NFRUP	VELOUT	DBA
Type		I	I	I	I	I	I	I
Default		1	0	0	0	0	0	0
Remark			8	9	10	11		

VARIABLE	DESCRIPTION
NBC	Number of boundary condition cards (default = 1). See Card 4.1.
RESTRT	This flag can save an LS-DYNA transient analysis if the binary output file in the (bem=**filename**) option has not been changed (default = 0).

EQ.-30: Read in the user-provided velocity spectrum saved in a binary file, bevelf.lsda. This option is supported for METHOD = 0, 2, 3, and 4.

EQ.-3: Read in the user-provided velocity spectrum saved in an ASCII file, bevelf. This option is supported for METHOD = 0, 2, 3, and 4.

EQ.0: Process the LS-DYNA time domain analysis, generating a new binary file.

EQ.1: Do not process the LS-DYNA time domain analysis. Use the binary files from the previous run. The files include the binary output file filename and the binary file bin_velfreq, which saves the boundary velocity from FFT.

EQ.2: Restart from the d3dump file by using the "R=" command line parameter. This is useful when the last run was interrupted by sense switches such as "sw1".

EQ.3: Read in user-provided velocity history saved in an

VARIABLE	DESCRIPTION
	ASCII file, bevel.
	EQ.4: Run acoustic computation on a boundary element mesh with velocity information given with a denser finite element mesh in the last run. This option requires both "bem = filename" and "lbem = filename2" in the command line, where filename2 is the name of the binary file generated in the last run with denser mesh.
	EQ.5: Do not process the LS-DYNA time domain analysis. Use the binary file filename from the previous run. An FFT computes the new frequency domain boundary velocity with results saved in bin_velfreq.
	EQ.30: Read in user-provided velocity history saved in a binary file, bevel.lsda.
IEDGE	Free edge and multi-connection constraints option (default = 0): EQ.0: Free edge and multi-connection constraints not considered. EQ.1: Free edge and multi-connection constraints considered. EQ.2: Only free edge constraints are considered. EQ.3: Only multi-connection constraints are considered.
NOEL	Location where normal velocity or acceleration is taken (default = 0): EQ.0: Elements or segments EQ.1: Nodes
NFRUP	Preconditioner update option: EQ.0: Updated at every frequency. GE.1: Updated every NFRUP frequencies.
VELOUT	Flag for writing out nodal or elemental velocity data: EQ.0: Do not write out velocity data. EQ.1: Write out time-domain velocity data (in the x, y, and z directions). EQ.2: Write out frequency domain velocity data (in the normal direction).

VARIABLE	DESCRIPTION
DBA	Flag for writing out weighted SPL files with different weighting options: EQ.0: Do not write out weighted SPL files. EQ.1: Write out Press_dB(A) by using A-weighting. EQ.2: Write out Press_dB(B) by using B-weighting. EQ.3: Write out Press_dB(C) by using C-weighting. EQ.4: Write out Press_dB(D) by using D-weighting.

Boundary Condition Cards. The deck must include NBC cards in this format: one for each boundary condition.

Card 4.1	1	2	3	4	5	6	7	8
Variable	SSID	SSTYPE	NORM	BEMTYP	LC1	LC2		
Type	I	I	I	I	I	I		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
SSID	Part, part set ID, or segment set ID of boundary elements (see Remark 12)
SSTYPE	Boundary element type: EQ.0: Part set ID EQ.1: Part ID EQ.2: Segment set ID
NORM	NORM should be set such that the normal vectors point away from the fluid. EQ.0: Normal vectors are not inverted (default). EQ.1: Normal vectors are inverted.
BEMTYP	Type of input boundary values in BEM analysis. EQ.0: The BEM analysis processes the boundary velocity.

VARIABLE	DESCRIPTION
	EQ.1: The BEM analysis processes the boundary acceleration. EQ.2: LC1 and LC2 prescribe the real and imaginary parts of the pressure. EQ.3: LC1 and LC2 prescribed the real and imaginary parts of the normal velocity. EQ.4: LC1 and LC2 prescribe the real and imaginary parts of the impedance. EQ.5: LC1 gives the acoustic absorption coefficient. EQ.6: Specify a symmetry condition (or rigid wall). LT.0: Set the normal velocity (only the real part) through an amplitude as a function of frequency load curve with curve ID |BEMTYP|.
LC1	Load curve ID for defining the real part of the pressure, the real part of the normal velocity, the real part of the impedance, or the acoustic absorption coefficient. See BEMTYP = 2, 3, 4, and 5, respectively.
LC2	Load curve ID for defining the imaginary part of the pressure, normal velocity, or impedance. See BEMTYP = 2, 3, and 4, respectively.

Panel Contribution Card. Additional for PANEL_CONTRIBUTION keyword option.

Card 5	1	2	3	4	5	6	7	8
Variable	NSIDPC	PCTYPE						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
NSIDPC	Node set ID for the field points where panel contributions to SPL (Sound Pressure Level) are requested. See Remark 13.
PCTYPE	Type of acoustic panel contribution analysis: EQ.0: All the loads are applied. Then, integration is performed

VARIABLE	DESCRIPTION
	panel by panel. EQ.1: The loads are applied panel by panel, and the integration is performed for the whole surface.

Half Space Card. Additional card for HALF_SPACE keyword option.

Card 6	1	2	3	4	5	6	7	8
Variable	PID	RC						
Type	I	F						
Default	0	1.0						

VARIABLE	DESCRIPTION
PID	Plane ID for defining the half-space problem; see keyword *DEFINE_PLANE.
RC	Reflection coefficient for the plane (see Remark 18). RC is supported for METHOD = 0, 2, and 3.

Remarks:

1. **TSTART field.** TSTART indicates the time at which velocity or acceleration and pressure are stored in the binary file.

2. **PREF field.** This reference pressure is required to compute the pressure in dB. Usually, in the International Unit System, the reference pressure is 20 μPa.

3. **FFT windowing.** The LS-DYNA analysis provides velocity or acceleration (pressure) written to a binary file (bem=**filename**). The boundary element method is processed after the LS-DYNA transient analysis. An FFT algorithm transforms the time domain data into the frequency domain to use the boundary element method for acoustics. We propose several windows to overcome the FFT leakage problem due to the truncation of the temporal response. Windowing creates a periodic velocity, acceleration, and pressure needed for the FFT.

4. **TRSLT field.** If requesting time domain results, FMIN automatically changes to 0.0

5. **IUNITS field.** Units are automatically converted into kg, m, s, N, and Pa so that the reference pressure is not too small. For example, it may be as low as 20.0×10^{-15} GPa if using the units kg, mm, ms, kN, and GPa, possibly resulting in a truncation error in the computation, especially for single precision.

6. **Rayleigh and Kirchhoff methods.** The Rayleigh method (METHOD = 0) is an approximation suitable only for external radiation problems. It is very fast since there is no linear system to solve. The Kirchhoff method (METHOD = 1) involves coupling the BEM and FEM for acoustics (*MAT_ACOUSTIC) with a non-reflecting boundary condition; see *BOUNDARY_NON_REFLECTING. In this case, at least one fluid layer with a non-reflecting boundary condition is merged with the vibrating structure. This additional fluid is given in *MAT_ACOUSTIC with the same density and sound speed used in this keyword. Both methods provide a good approximation to a full BEM calculation for external problems when used appropriately.

7. **NDD field.** The BEM formulation for large and medium size problems (more than 2000 boundary elements) is memory and time consuming. In this case, LS-DYNA may be run using the memory option. To save memory, domain decomposition can be used.

8. **RESTRT field.** The binary file generated by a previous run can be used for the next run by using the restart option. With the restart option, the BEM calculation uses the binary file generated from a previous calculation. In this case, the frequency range can be changed. However, the time parameters should not be modified between calculations.

9. **IEDGE field.** This field only applies to METHOD = 2, the Variational Indirect BEM.

10. **NOEL field.** This field specifies whether the element or nodal velocity (or acceleration) is taken from FEM computation. NOEL should be 0 if Kirchhoff method (METHOD = 1) is used since elemental pressure is processed in FEM. NOEL should be 0 if Burton-Miller collocation method (METHOD = 4) is used since a constant strength element formulation is adopted. In other cases, it is strongly recommended to use element velocity or acceleration (NOEL = 0) if a T-intersection appears in the boundary element mesh. See Figure 23-1.

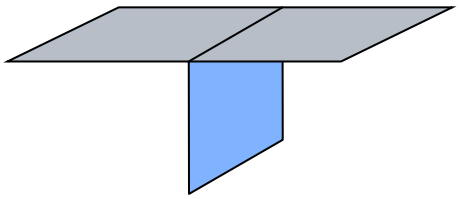


Figure 23-1. T-intersection in the Boundary Element Mesh

11. **NFRUP field.** The preconditioner is obtained with the factorization of the influence coefficient matrix. To conserve CPU time, it can be retained for several frequencies. By default (NFRUP = 0), the preconditioner is updated for every frequency. Note that in the MPP version, the preconditioner is updated every NFRUP frequencies on each processor.

12. **Boundary condition cards.** Card 4.1 can be defined if the boundary elements are composed of several panels. It can be defined multiple times if more than 2 panels are used. Each Card 4.1 defines one panel.

13. **NSIDPC field.** The field points where the panel contribution analysis is requested must be one of the field points for acoustic computation (it must be included in the nodes specified by the NSIDEXT or NSIDINT). The panels are defined by Cards 4 and 4.1.

14. **Element sizing.** To obtain accurate results, the element size should not exceed 1/6 of the wavelength λ ($\lambda = c/f$ where c is the wave speed and f is the frequency).

15. **Acoustic transfer vector.** Including the ATV keyword option enables obtaining the Acoustic Transfer Vector (ATV). Only METHOD = 2 and 3 are supported for this option. The solver calculates acoustic pressure (and sound pressure level) at field points from a unit normal velocity of each surface node. ATV depends on the structure model, properties of the acoustic fluid, and the location of field points. When the ATV option is included, the structure does not need external excitation, and the curve IDs LC1 and LC2 are ignored. Setting BINARY = 1 in *DATABASE_FREQUENCY_BINARY_D3ATV causes the output of binary plot database d3atv.

16. **Modal acoustic transfer vector.** Including the MATV keyword option enables calculating the Modal Acoustic Transfer Vector (MATV). The MATV option requires using the implicit eigenvalue solver. *CONTROL_IMPLICIT_GENERAL and *CONTROL_IMPLICIT_EIGENVALUE activate this solver. It calculates acoustic pressure (and sound pressure level) at field points due to vibration in the form of eigenmodes. For each excitation frequency f, LS-DYNA generates the pseudo-velocity boundary condition $2\pi i f\{\phi\}_j$. Here $i = \sqrt{-1}$ is the imaginary unit. LS-DYNA then runs the acoustic computation for each field point, based on the pseudo-velocity boundary conditions, to obtain the MATV matrices. LS-DYNA saves the MATV matrices to binary file bin_bepressure for future use. Like ATV, MATV depends on the structure model, properties of the acoustic fluids, and the locations of the field points.

17. **Power option.** With the POWER keyword option, acoustic radiated power can be computed with the Rayleigh method, Kirchhoff method, or variational indirect method.

18. **Reflection coefficient.** The reflection coefficient is 1.0 for a rigid reflection plane (zero velocity) and -1.0 for a soft reflection plane (zero sound pressure, such as a water-air interface in underwater acoustics).

19. **Output files.** The result files: Press_Pa, Press_dB, Press_Pa_real, Press_Pa_imag, Press_Pa_t, and Press_dB_t have an *xy*-plot format that LS-PrePost can read and plot.

*FREQUENCY_DOMAIN_ACOUSTIC_DIRECTIVITY

Purpose: Define field point circles for acoustic directivity plot.

Include as many of this card as needed (see Remark 1). This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	CENTER	R	NP	NORM	ANGLE	X	Y	Z
Type	I	F	I	I	F	F	F	F
Default	none	none	none	none	0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
CENTER	Flag for defining the center point for the circle: EQ.1: Mass center of the original structure EQ.2: Geometry center of the original structure EQ.3: Defined by input variables X, Y, and Z
R	Radius of the circle
NP	Number of points on the circle
NORM	Norm direction of the circle. EQ.1: x-direction EQ.2: y-direction EQ.3: z-direction
ANGLE	Angle for the first point on the circle
X	x-coordinate of the center
Y	y-coordinate of the center
Z	z-coordinate of the center

Remarks:

1. **Multiple directivity plot.** Card 1 can be repeated if multiple acoustic directivity plot is desired.

2. **Results.** ASCII files Press_Pa_directivity_n, Press_dB_directivity_n, and AC_intensity_directivity_n are output. Press_Pa_directivity_n provides a list of acoustic pressures (magnitude) for each angle on the n^{th} circle for a series of frequencies. Press_dB_directivity_n provides a list of SPLs (or decibels) for each angle on the n^{th} circle for a series of frequencies. AC_intensity_directivity_n provides a list of acoustic intensities for each angle on the n^{th} circle for a series of frequencies. The frequencies are defined by FMIN, FMAX and NFREQ in the keyword *FREQUENCY_DOMAIN_ACOUSTIC_BEM.

*FREQUENCY_DOMAIN_ACOUSTIC_FEM_{OPTION}

Available options include:

<BLANK>

EIGENVALUE

MODAL

Purpose: Define an interior acoustic problem and solve the problem with a frequency domain finite element method. When the EIGENVALUE option is used, LS-DYNA computes eigenvalues and eigenvectors of the acoustic system. When the MODAL option is used, LS-DYNA uses a modal superposition method to solve the acoustic problem (see Remark 9).

Card Summary:

Card MDL. LS-DYNA only reads this card for the MODAL keyword option. It is optional and may be omitted. LS-DYNA assumes that the input contains this card if no data is in the fifth column.

MDMIN	MDMAX	FNMIN	FNMAX				

Card 1. This card is required.

RO	C	FMIN	FMAX	NFREQ	DTOUT	TSTART	PREF

Card 2. This card is required.

	FFTWIN	MTXDMP	RESTRT				

Card 3. This card is required.

PID	PTYP						

Card 4. This card can be repeated if multiple boundary conditions are present. For the EIGENVALUE keyword option, this card is optional.

SID	STYP	VAD	DOF	LCID1	LCID2	SF	VID

Card 5. Include this card only if the keyword option is *not* EIGENVALUE.

NID	NTYP	IPFILE	DBA				

Data Card Definitions:

MODAL Card. LS-DYNA only reads this optional card for the MODAL keyword option. LS-DYNA assumes that the input contains this card if the fifth column has no data. See Remark 9.

Card MDL	1	2	3	4	5	6	7	8
Variable	MDMIN	MDMAX	FNMIN	FNMAX				
Type	I	I	F	F				
Default	1	optional	0.0	optional				

VARIABLE	DESCRIPTION
MDMIN	First mode in modal superposition method (optional)
MDMAX	Last mode in modal superposition method (optional)
FNMIN	Minimum natural frequency in modal superposition method (optional)
FNMAX	Maximum natural frequency in modal superposition method (optional)

Card 1	1	2	3	4	5	6	7	8
Variable	RO	C	FMIN	FMAX	NFREQ	DTOUT	TSTART	PREF
Type	F	F	F	F	I	F	F	F
Default	Rem 2	none	none	none	0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
RO	Fluid density
C	Sound speed of the fluid. GT.0: Real constant sound speed. LT.0: |C| is the load curve ID, which defines the frequency-

VARIABLE	DESCRIPTION
	dependent complex sound speed. See *FREQUENCY_-DOMAIN_ACOUSTIC_SOUND_SPEED.
FMIN	Minimum value of output frequencies.
FMAX	Maximum value of output frequencies.
NFREQ	Number of output frequencies.
DTOUT	Time step for writing velocity or acceleration in the binary file.
TSTART	Start time for recording velocity or acceleration in transient analysis.
PREF	Reference pressure for converting the acoustic pressure to dB. See Remark 3.

Card 2	1	2	3	4	5	6	7	8
Variable		FFTWIN	MTXDMP	RESTRT				
Type		I	I	I				
Default		0	0	0				

VARIABLE	DESCRIPTION
FFTWIN	FFT windows (Default = 0): EQ.0: Rectangular window. EQ.1: Hanning window. EQ.2: Hamming window. EQ.3: Blackman window. EQ.4: Raised cosine window.
MTXDMP	Acoustic stiffness and mass matrices dumping (when using the option EIGENVALUE): EQ.0: No dumping. EQ.1: Dumping globally assembled acoustic stiffness and mass

VARIABLE	DESCRIPTION
	matrices in Harwell-Boeing sparse matrix format.
RESTRT	This flag is used to save an LS-DYNA time domain analysis if the binary output file (bin_femac) has not been changed (Default = 0): EQ.0: LS-DYNA time domain analysis is processed and generates a new binary file (bin_femac) which saves the boundary velocity history data. EQ.1: LS-DYNA time domain analysis is not processed. The binary file bin_femac from the last run is used.

Card 3	1	2	3	4	5	6	7	8
Variable	PID	PTYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
PID	Part ID or part set ID to define the acoustic domain.
PTYP	Set type: EQ.0: Part; see *PART. EQ.1: Part set; see *SET_PART.

Boundary Condition Definition Card. It can be repeated if multiple boundary conditions are present. This card is optional when option EIGENVALUE is present.

Card 4	1	2	3	4	5	6	7	8
Variable	SID	STYP	VAD	DOF	LCID1	LCID2	SF	VID
Type	I	I	I	I	I	I	F	I
Default	none	0	0	none	0	0	1.0	0

VARIABLE	DESCRIPTION
SID	Part ID, part set ID, segment set ID, or node set ID to define where the vibration boundary condition is.
STYP	Set type: EQ.0: Part; see *PART. EQ.1: Part set; see *SET_PART. EQ.2: Segment set; see *SET_SEGMENT. EQ.3: Node set; see *SET_NODE.
VAD	Boundary condition flag: EQ.0: Velocity by steady state dynamics (SSD). EQ.1: Velocity by transient analysis. EQ.2: Opening (zero pressure). EQ.11: Velocity by LCID1 (amplitude) and LCID2 (phase). EQ.12: Velocity by LCID1 (real) and LCID2 (imaginary). EQ.21: Acceleration by LCID1 (amplitude) and LCID2 (phase). EQ.22: Acceleration by LCID1 (real) and LCID2 (imaginary). EQ.31: Displacement by LCID1 (amplitude) and LCID2 (phase). EQ.32: Displacement by LCID1 (real) and LCID2 (imaginary). EQ.41: Impedance by LCID1 (amplitude) and LCID2 (phase). EQ.42: Impedance by LCID1 (real) and LCID2 (imaginary). EQ.51: Pressure by LCID1 (amplitude) and LCID2 (phase). EQ.52: Pressure by LCID1 (real) and LCID2 (imaginary).
DOF	Applicable degrees-of-freedom: EQ.0: Determined by steady state dynamics. EQ.1: x-translational degree-of-freedom, EQ.2: y-translational degree-of-freedom, EQ.3: z-translational degree-of-freedom, EQ.4: Translational motion in direction given by VID, EQ.5: Normal direction of the element or segment.

VARIABLE	DESCRIPTION
LCID1	Load curve ID to describe the amplitude (or real part) of velocity; see *DEFINE_CURVE.
LCID2	Load curve ID to describe the phase (or imaginary part) of velocity; see *DEFINE_CURVE.
SF	Load curve scale factor.
VID	Vector ID for DOF values of 4.

Field Points Definition Card. Not used when option EIGENVALUE is present.

Card 5	1	2	3	4	5	6	7	8
Variable	NID	NTYP	IPFILE	DBA				
Type	I	I	I	I				
Default	none	0	0	0				

VARIABLE	DESCRIPTION
NID	Node ID, node set ID, or segment set ID for acoustic result output
NTYP	Set type: EQ.0: Node; see *NODE. EQ.1: Node set; see *SET_NODE.
IPFILE	Flag for output files (default = 0): EQ.0: Press_Pa (magnitude of pressure as function of frequency), Press_dB (sound pressure level or SPL as a function of frequency) are provided. EQ.1: Press_Pa_real (real part of pressure as a function of frequency) and Press_Pa_imag (imaginary part of pressure as a function of frequency) are provided, in addition to Press_Pa, Press_dB.
DBA	Flag for writing out weighted SPL files with different weighting options. EQ.0: No writing out weighted SPL files.

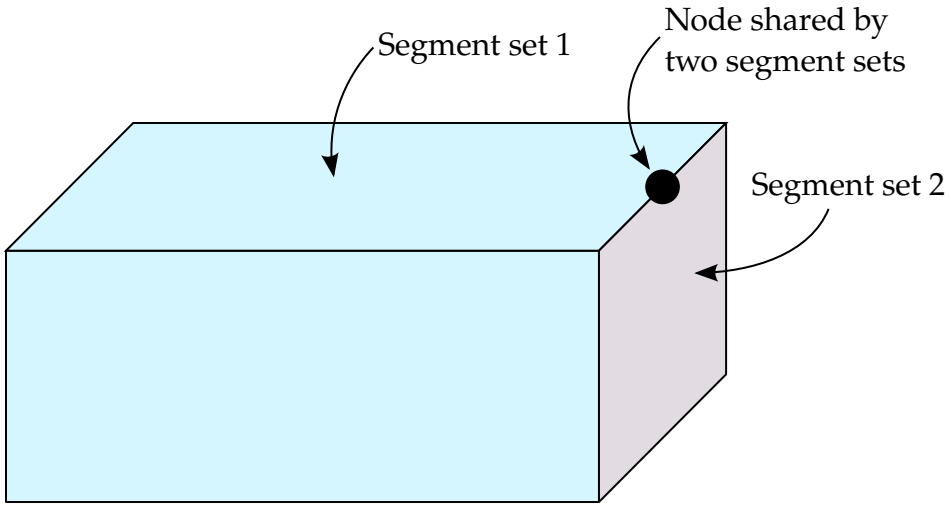


Figure 23-2. Nodes with Multiple Boundary Conditions.

VARIABLE	DESCRIPTION
	EQ.1: Write out Press_dB(A) by using A-weighting.
	EQ.2: Write out Press_dB(B) by using B-weighting.
	EQ.3: Write out Press_dB(C) by using C-weighting.
	EQ.4: Write out Press_dB(D) by using D-weighting.

Remarks:

1. **Acoustic problem.** This command solves the interior acoustic problems which is governed by Helmholtz equation $\nabla^2 p + k^2 p = 0$ with the boundary condition $\partial p/\partial n = -i\omega\rho v_n$, where p is the acoustic pressure; $k = \omega/c$ is the wave number; ω is the round frequency; c is the acoustic wave speed (sound speed); $i = \sqrt{-1}$ is the imaginary unit; ρ is the mass density; and v_n is the normal velocity. This command solves the acoustic problem in the frequency domain.

2. **Mass density.** If mass density RO is not given, the mass density of PID (the part which defines the acoustic domain), will be used

3. **Reference pressure.** PREF is the reference pressure to convert the acoustic pressure to dB $L_p = 10 \log_{10}(p^2/p_{\text{ref}}^2)$. Note that generally $p_{\text{ref}} = 20\ \mu$Pa for air.

4. **Material models for acoustic domain.** If the boundary velocity is obtained from steady state dynamics (VAD = 0) using the keyword *FREQUENCY_DOMAIN_SSD, the part (PID) which defines the acoustic domain should use material model *MAT_ELASTIC_FLUID which enables implicit eigenvalue analysis. If the boundary excitation is given by load curves LCID1 and LCID2 (VAD > 0), the part (PID) which defines the acoustic domain can use any material model

that is compatible with 8-node hexahedron, 6-node pentahedron, or 4-node tetrahedron elements since only the mesh of the PID will be used in the computation. For example, *MAT_ACOUSTIC or *MAT_ELASTIC_FLUID can be used.

5. **Steady state dynamics.** If VAD = 0, the boundary excitation is given as velocity obtained from steady state dynamics. The other parameters in Card 3 (DOF, LCID1, LCID2, SF and VID) are ignored.

6. **Multiple boundary condition definitions.** If a node's vibration boundary condition is defined multiple times, only the last definition is considered. This happens usually when a node is on an edge, is shared by two or more parts, part sets, node sets, or segment sets, and a different vibration condition is defined on each of the node or segment sets. See Figure 23-2.

7. **Output Files.** Results including acoustic pressure and SPL are given in d3acs binary files, which can be accessed by LS-PrePost. Nodal pressure and SPL values for nodes specified by NID and NTYP are given in ASCII file Press_Pa and Press_dB, which can be accessed by LS-PrePost. Press_Pa gives magnitude of the pressure. Press_dB gives the SPL in terms of dB.

8. **Frequencies and Steady State Dynamics.** If the boundary velocity condition is given by Steady State Dynamics (VAD = 0), the range and number of frequencies (FMIN, FMAX and NFREQ) should be compatible with the corresponding parameters in Card 1 of the keyword *FREQUENCY_DOMAIN_SSD.

9. **Modal Acoustics.** When keyword option MODAL is used, LS-DYNA computes the acoustic pressure response by a modal superposition method. An acoustic eigenvalue analysis problem (*FREQUENCY_DOMAIN_ACOUSTIC_FEM_EIGENVALUE) must be solved first and the binary database d3eigv_ac must be present in the same directory to run modal acoustics. Card MDL is optional, and it defines the eigenmodes to be used in modal superposition if present. Otherwise, all the acoustic eigenmodes saved in d3eigv_ac will be used.

*FREQUENCY_DOMAIN_ACOUSTIC_FREQUENCY

Purpose: Specify output frequencies for a frequency domain acoustic computation, especially with different spacing options. The frequencies defined here override those defined in *FREQUENCY_DOMAIN_ACOUSTIC_BEM or *FREQUENCY_DOMAIN_ACOUSTIC_FEM.

Card 1	1	2	3	4	5	6	7	8
Variable	FMIN	FMAX	NFREQ	FSPACE	LCFREQ	BIAS	SPREADF	FRACTN
Type	F	F	I	I	I	F	F	I
Default	none	none	none	0	0	3	0.1	3

VARIABLE	DESCRIPTION
FMIN	Minimum frequency for output, f_{min}
FMAX	Maximum frequency for output, f_{max}. Ignored if FSPACE = 6.
NFREQ	Total number of frequencies for output: GT.0: NFREQ is the number of frequencies for the whole range. LT.0: |NFREQ| is the number of frequencies for each interval.
FSPACE	Frequency spacing option for output (see Figure 23-3 and Remark 1): EQ.0: Linear EQ.1: Logarithmic EQ.2: Biased spacing (range) EQ.3: Eigenfrequencies only EQ.4: Biased spacing (eigenfrequency) EQ.5: Biased spacing (eigenfrequency spread) EQ.6: Octave frequencies starting with FMIN
LCFREQ	Load curve ID defining the frequencies for output. If defined, all other fields are ignored.
BIAS	Bias parameter, p (FSPACE = 2, 4, and 5 only)

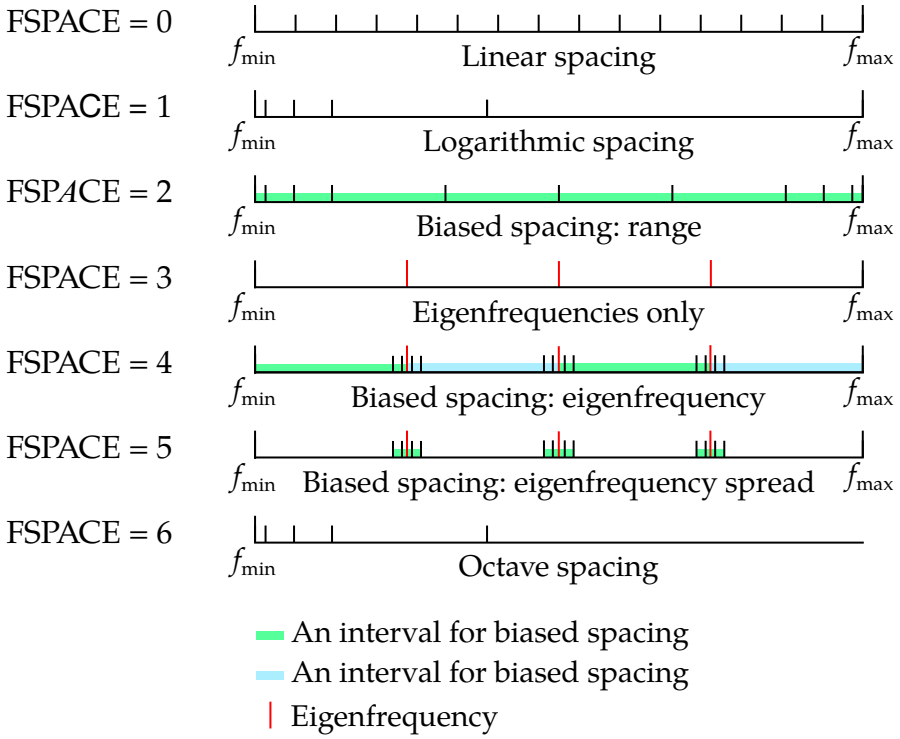


Figure 23-3. Frequency spacing depending on the choice of FSPACE

VARIABLE	DESCRIPTION
SPREADF	Spread ratio, σ (FSPACE = 5 only)
FRACTN	Octave fraction, α (FSPACE = 6 only). For example, FRACTN = 3 means 1/3 octave spacing. FMAX is ignored.

Remarks:

1. **Output frequencies.** Unless LCID is defined, the output frequencies depend on the choice of FSPACE as shown in Figure 23-3. Here we will give equations for determining the output frequencies.

   a) *Linear spacing (FSPACE = 0).* For linear spacing, the output frequencies are given by:

   $$f_k = f_{\min} + \frac{k-1}{N-1}(f_{\max} - f_{\min}) \ .$$

   Here, k is the k^{th} frequency with $k = 1, \ldots, N$ where N is the number of frequencies.

   b) *Logarithmic spacing (FSPACE = 1).* For logarithmic spacing, the output frequencies are given by:

$$f_k = f_{\min}\left(\frac{f_{\max}}{f_{\min}}\right)^{\frac{k-1}{N-1}} \quad .$$

Here, k is the k^{th} frequency with $k = 1, \ldots, N$ where N is the number of frequencies.

c) *Biased spacing (FSPACE = 2, 4, and 5).* For biased spacing, the output frequencies in each frequency interval $[\bar{f}_1, \bar{f}_2]$ are given by:

$$f_k = \frac{1}{2}(\bar{f}_1 + \bar{f}_2) + \frac{1}{2}(\bar{f}_2 - \bar{f}_1)|x|^{1/p}\text{sign}(x) \quad ,$$

where

$$x = -1 + \frac{2(k-1)}{n-1}$$

In the above, k is the k^{th} frequency is the k^{th} frequency in the interval with $k = 1, \ldots, n$ where n is the number of frequency points in the frequency interval, and p is the bias parameter.

The lower and upper limits of the frequency interval, $\bar{f}_1$ and $\bar{f}_2$, depend on the setting of FSPACE. For FSPACE = 2:

$$\bar{f}_1 = f_{\min} \quad , \qquad \bar{f}_2 = f_{\max} \quad .$$

For FSPACE = 4:

$$\bar{f}_1 = F_i \quad , \qquad \bar{f}_2 = F_{i+1}$$

with $\bar{f}_1 = f_{\min}$ in the first interval and $\bar{f}_2 = f_{\max}$ in the last interval. Other than than the endpoints, F_i is the i^{th} eigenfrequency. For FSPACE = 5:

$$\bar{f}_1 = F_i(1-\sigma) \, , \qquad \bar{f}_2 = F_i(1+\sigma) \quad .$$

Here, F_i is the i^{th} eigenfrequency, and σ is the spread ratio.

d) *Octave spacing (FSPACE = 6).* For octave spacing, the output frequencies are given by:

$$f_k = f_{\min}\left(2^{(1/\alpha)}\right)^{k-1}$$

Here, k is the k^{th} frequency with $k = 1, \ldots, N$ where N is the number of frequencies, and α is the octave fraction.

FREQUENCY_DOMAIN_ACOUSTIC_FRINGE_PLOT_*OPTION

Available options include:

CUBE

NODE_SET

PART

PART_SET

PLATE

SPHERE

Purpose: Define field points for the acoustic pressure computation by the BEM acoustic solver and save the results to D3ACP binary database. The field points can be defined using existing structure components if option NODE_SET, PART, or PART_SET is used. The field points can be created by LS-DYNA if option SPHERE, PLATE, or CUBE is used. SPHERE, PLATE, and CUBE are only available for SMP while PART, PART_SET, and NODE_SET are available for both MPP and SMP.

Card Summary:

Card 1a. This card is included if the keyword option is NODE_SET, PART, or PART_-SET.

PID/SID							

Card 1b. This card is included if the keyword option is SPHERE.

CENTER	R	DENSITY	X	Y	Z	HALF1	HALF2

Card 1c. This card is included if the keyword option is PLATE.

NORM	LEN_X	LEN_Y	X	Y	Z	NELM_X	NELM_Y

Card 1d.1. This card is included if the keyword option is CUBE.

LEN_X	LEN_Y	LEN_Z	X	Y	Z		

Card 1d.2. This card is included if the keyword option is CUBE.

NELM_X	NELM_Y	NELM_Z					

Data Card Definitions:

PART, PART_SET, and NODE_SET Card.

Card 1a	1	2	3	4	5	6	7	8
Variable	PID/SID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PID/SID	Part ID, part set ID, or node set ID

SPHERE Card.

Card 1b	1	2	3	4	5	6	7	8
Variable	CENTER	R	DENSITY	X	Y	Z	HALF1	HALF2
Type	I	F	I	F	F	F	I	I
Default	1	none	none	none	none	none	0	0

VARIABLE	DESCRIPTION
CENTER	Flag for defining the center point for the sphere: EQ.1: Mass center of the original structure EQ.2: Geometry center of the original structure EQ.3: Defined by (x,y,z)
R	Radius of the sphere.
DENSITY	Parameter to define how coarse or dense the created sphere mesh is. It is a number between 3 and 39, where "3" gives you 24 elements while "39" gives you 8664 elements.
X	x-coordinate of the center

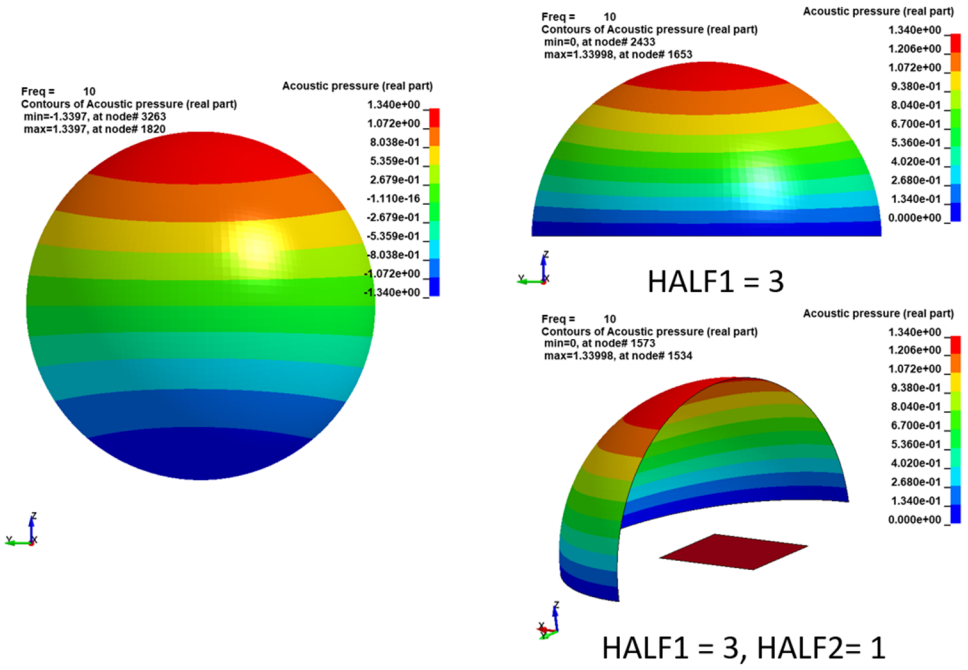


Figure 23-4. Defining a half or quarter sphere

VARIABLE	DESCRIPTION
Y	y-coordinate of the center
Z	z-coordinate of the center
HALF1	Create a half sphere by trimming the defined sphere (see Remark 3 and Figure 23-4). Note that (x_0, y_0, z_0) below is the center of the sphere. EQ.0: A full sphere is created, no trimming EQ.1: Keep $x \geq x_0$ EQ.-1: Keep $x \leq x_0$ EQ.2: Keep $y \geq y_0$ EQ.-2: Keep $y \leq y_0$ EQ.3: Keep $z \geq z_0$ EQ.-3: Keep $z \leq z_0$
HALF2	Create a quarter sphere by trimming the half sphere defined with HALF1 (see Remark 3 and Figure 23-4): EQ.0: No second trimming

VARIABLE	DESCRIPTION
	EQ.1: Keep $x \geq x_0$
	EQ.-1: Keep $x \leq x_0$
	EQ.2: Keep $y \geq y_0$
	EQ.-2: Keep $y \leq y_0$
	EQ.3: Keep $z \geq z_0$
	EQ.-3: Keep $z \leq z_0$

Plate Card.

Card 1c	1	2	3	4	5	6	7	8
Variable	NORM	LEN_X	LEN_Y	X	Y	Z	NELM_X	NELM_Y
Type	I	F	F	F	F	F	I	I
Default	1	none	none	none	none	none	10	10

VARIABLE	DESCRIPTION
NORM	Norm direction of the plate. EQ.1: x-direction EQ.2: y-direction EQ.3: z-direction
LEN_X	Length of longer side of the plate
LEN_Y	Length of shorter side of the plate
X	x-coordinate of the center
Y	y-coordinate of the center
Z	z-coordinate of the center
NELM_X	Number of elements on longer side of the plate
NELM_Y	Number of elements on shorter side of the plate

Cube Card 1.

Card 1d.1	1	2	3	4	5	6	7	8
Variable	LEN_X	LEN_Y	LEN_Z	X	Y	Z		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
LEN_X	Length of *x*-side of the cube
LEN_Y	Length of *y*-side of the cube
LEN_Z	Length of *z*-side of the cube
X	*x*-coordinate of the corner of the cube with smallest nodal coordinates
Y	*y*-coordinate of the corner of the cube with smallest nodal coordinates
Z	*z*-coordinate of the corner of the cube with smallest nodal coordinates

Cube Card 2.

Card 1d.2	1	2	3	4	5	6	7	8
Variable	NELM_X	NELM_Y	NELM_Z					
Type	I	I	I					
Default	10	10	10					

VARIABLE	DESCRIPTION
NELM_X	Number of elements on *x*-side of the cube
NELM_Y	Number of elements on *y*-side of the cube

VARIABLE	DESCRIPTION
NELM_Z	Number of elements on *z*-side of the cube

Remarks:

1. **LS-PrePost.** The acoustic pressure results at those field points are saved in D3ACP binary database and are accessible by LS-PrePost. With the FCOMP tool in LS-PrePost, the fringe plot of the results (real part acoustic pressure, imaginary part acoustic pressure, magnitude of acoustic pressure and Sound Pressure Level) can be generated.

2. **Field Points.** The field points defined by this keyword are separate from the field points defined in Card 2 of *FREQUENCY_DOMAIN_ACOUSTIC_BEM. The acoustic pressure results for the latter are only saved in ASCII database Press_Pa, Press_dB, etc. (in a tabular format that can be plotted in LS-PrePost by using the XYPlot tool).

3. **Half or Quarter Sphere.** The parameters HALF1 and HALF2 change a full sphere to a half sphere or a quarter sphere. See Figure 23-4 for illustration.

*FREQUENCY_DOMAIN_ACOUSTIC_INCIDENT_WAVE

Purpose: Define incident sound wave for acoustic scattering problems.

Wave Definition Cards. This card may be repeated to define multiple incident waves. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPE	MAG	XC	YC	ZC			
Type	I	F	F	F	F			
Default	1	none	none	none	none			

VARIABLE	DESCRIPTION
TYPE	Type of incident sound wave: EQ.1: plane wave. EQ.2: spherical wave.
MAG	Magnitude of the incident sound wave: GT.0: constant magnitude. LT.0: |MAG| is a curve ID, which defines the frequency dependent magnitude. See *DEFINE_CURVE.
XC, YC, ZC	Direction cosines for the plane wave (TYPE = 1) or coordinates of the point source for the spherical wave (TYPE = 2).

Remarks:

1. **Plane Waves.** For plane waves, the incident wave is defined as

$$p(x,y,z) = Ae^{-ik(\alpha x+\beta y+\gamma z)} \quad ,$$

where A is the magnitude of the incident wave and α, β, and γ are the direction cosines along the incident direction. $i = \sqrt{-1}$ is the imaginary unit and $k = \omega/c$ is the wave number. ω is the round frequency and c is the sound speed.

2. **Spherical Waves.** For spherical waves, the incident wave is defined as

$$p(r) = A\frac{e^{-ikr}}{r} \quad ,$$

where A is the magnitude of the incident wave and r is the distance measured from the position of the point source.

*FREQUENCY_DOMAIN_ACOUSTIC_SOUND_SPEED

Purpose: Define frequency dependent complex sound speed to be used in frequency domain finite element method or boundary element method acoustic analysis.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	LCID1	LCID2						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ID	Complex sound speed ID.
LCID1	Curve ID for real part of frequency dependent complex sound speed.
LCID2	Curve ID for imaginary part of frequency dependent complex sound speed.

Remarks:

1. **Acoustic Mediums and Damping.** The sound speed in an acoustic medium is usually defined as a constant real value. But it can also be defined as a complex value which is dependent on frequency, to introduce damping in the system.

2. **Using Complex Sound Speed.** To use the frequency dependent complex sound speed defined here, set the sound speed C = -ID in *FREQUENCY_DO-

MAIN_ACOUSTIC_FEM or *FREQUENCY_DOMAIN_ACOUSTIC_BEM keywords.

*FREQUENCY_DOMAIN_FRF_{OPTION}

Available options include:

<BLANK>

REDUCED

SUBCASE

Purpose: Compute frequency response functions due to nodal excitations. To run an FRF analysis based on reduced eigenvector output, use the REDUCED keyword option (see Remark 9). When the SUBCASE option is applied, multiple loading cases can be included in one run (see Remark 10).

> **NOTE:** Natural frequencies and mode shapes are needed for computing the frequency response functions. Thus, *CONTROL_IMPLICIT_EIGENVALUE *must* be included in the input. See Remark 1.

Card Summary:

Card 1a. Include this card when the keyword option is unused (<BLANK>).

N1	N1TYP	DOF1	VAD1	VID1	FNMAX	MDMIN	MDMAX

Card 1a.1. Include this card when the keyword option is unused (<BLANK>) and VAD1 = 12 on Card 1a.

N11	N11TYP						

Card 1b. Include this card when the keyword option is SUBCASE.

NCASES				RELATV	FNMAX	MDMIN	MDMAX

Card 2. This card is required.

DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF			

Card 3. Include this card when the keyword option is unused (<BLANK>).

N2	N2TYP	DOF2	VAD2	VID2	RELATV		

Card 4. This card is required.

FMIN	FMAX	NFREQ	FSPACE	LCFREQ	RESTRT	OUTPUT	

Card 5. Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE. Each repeat defines one subcase for FRF computation, including the title, excitation input, and response output for the subcase.

TITLE

Card 6. Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE.

N1	N1TYP	DOF1	VAD1	VID1			

Card 6.1. Include this card after Card 6 if VAD1 = 12 on Card 6 (SUBCASE keyword option only).

N11	N11TYP						

Card 7. Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE.

N2	N2TYP	DOF2	VAD2	VID2			

Data Card Definitions:

Include this card when the keyword option is unused (<BLANK>).

Card 1a	1	2	3	4	5	6	7	8
Variable	N1	N1TYP	DOF1	VAD1	VID1	FNMAX	MDMIN	MDMAX
Type	I	I	I	I	I	F	I	I
Default	none	0	none	3	0	0.0	0	0

VARIABLE	DESCRIPTION
N1	ID of the entity specified by N1TYP for excitation input. This field is ignored when VAD1, the excitation type, is set to 0, 1, or 2 (base velocity, base acceleration, or base displacement, respectively). In this case, the excitation is applied through the node(s) under constraints, that is, the node(s) defined in *BOUNDARY_SPC_NODE or *BOUNDARY_SPC_SET.
N1TYP	Type of N1:

VARIABLE	DESCRIPTION
	EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW) EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW) EQ.5: Parametric edge ID (see *IGA_EDGE_UVW) EQ.6: Parametric edge set ID (see *SET_IGA_EDGE_UVW) EQ.7: Physical face ID (see *IGA_FACE_XYZ) EQ.8: Physical face set ID (see *SET_IGA_FACE_XYZ) EQ.9: Parametric face ID (see *IGA_FACE_UVW). This option is only supported for VAD1 = 4. EQ.10: Parametric face set ID (see *SET_IGA_FACE_UVW). This option is only supported for VAD1 = 4. When VAD1, the excitation type, is set to 0, 1, or 2, which is base velocity, base acceleration, or base displacement, this field is ignored.
DOF1	Applicable degrees of freedom for excitation input (ignored if VAD1 = 4): EQ.0: Translational movement in the direction given by vector VID1 EQ.1: x-translational degree of freedom, or x-rotational degree of freedom (for torque excitation, VAD1 = 8) EQ.2: y-translational degree of freedom, or y-rotational degree of freedom (for torque excitation, VAD1 = 8) EQ.3: z-translational degree of freedom, or z-rotational degree of freedom (for torque excitation, VAD1 = 8)
VAD1	Excitation input type: EQ.0: Base velocity EQ.1: Base acceleration EQ.2: Base displacement EQ.3: Nodal force EQ.4: Pressure

VARIABLE	DESCRIPTION
	EQ.5: Enforced velocity by large mass method EQ.6: Enforced acceleration by large mass method EQ.7: Enforced displacement by large mass method EQ.8: Torque EQ.9: Base angular velocity EQ.10: Base angular acceleration EQ.11: Base angular displacement EQ.12: Torsion
VID1	Vector ID for DOF1 = 0 for excitation input. See *DEFINE_VECTOR.
FNMAX	Optional maximum natural frequency employed in FRF computation. See Remark 3.
MDMIN	The first mode employed in FRF computation (optional). See Remarks 3 and 4.
MDMAX	The last mode employed in FRF computation (optional). It should be set as a positive integer in a restart run (RESTRT = 1 or 3) based on the number of eigenmodes available in the existing d3eigv database. See Remarks 3 and 4.

Include this card when the keyword option is unused (<BLANK>) and VAD1 = 12 on Card 1a.

Card 1a.1	1	2						
Variable	N11	N11TYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
N11	ID of the entity specified by N11TYP for the excitation input for the torque load (VAD1 = 12). This node or set combines with N1 to form a torque load.

VARIABLE	DESCRIPTION
N11TYP	Type of N11: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW) EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW) EQ.5: Parametric edge ID (see *IGA_EDGE_UVW) EQ.6: Parametric edge set ID (see *SET_IGA_EDGE_UVW) EQ.7: Physical face ID (see *IGA_FACE_XYZ) EQ.8: Physical face set ID (see *SET_IGA_FACE_XYZ)

Include this card when the keyword option is SUBCASE.

Card 1b	1	2	3	4	5	6	7	8
Variable	NCASES				RELATV	FNMAX	MDMIN	MDMAX
Type	I				I	F	I	I
Default	none				0	0.0	0	0

VARIABLE	DESCRIPTION
NCASES	Number of subcases
RELATV	Flag for displacement, velocity, and acceleration results: EQ.0: Absolute values are requested. EQ.1: Relative values are requested (for VAD1 = 0, 1, 2 only).
FNMAX	Optional maximum natural frequency employed in FRF computation. See Remark 3.
MDMIN	The first mode employed in FRF computation (optional). See Remarks 3 and 4.
MDMAX	The last mode employed in FRF computation (optional). It should be set as a positive integer in a restart run (RESTRT = 1 or 3) based on the number of eigenmodes available in the existing d3eigv

VARIABLE	DESCRIPTION
	database. See Remarks 3 and 4.

Card 2	1	2	3	4	5	6	7	8
Variable	DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF			
Type	F	I	I	F	F			
Default	0.0	0	0	0.0	0.0			

VARIABLE	DESCRIPTION
DAMPF	Modal damping coefficient, ζ. See Remark 5.
LCDAM	Load Curve ID defining mode dependent modal damping coefficient, ζ. See Remark 5.
LCTYP	Type of load curve defining modal damping coefficient: EQ.0: Abscissa value represents frequency, EQ.1: Abscissa value gives mode number. See Remark 5.
DMPMAS	Mass proportional damping constant, α, in Rayleigh damping. See Remark 5.
DMPSTF	Stiffness proportional damping constant, β, in Rayleigh damping. See Remark 5.

Include this card when the keyword option is unused (<BLANK>).

Card 3	1	2	3	4	5	6	7	8
Variable	N2	N2TYP	DOF2	VAD2	VID2	RELATV		
Type	I	I	I	I	I	I		
Default	none	0	none	2	0	0		

VARIABLE	DESCRIPTION
N2	ID of the entity specified by N2TYP for response output
N2TYP	Type of N2: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW) EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW)
DOF2	Applicable degrees of freedom for response output: EQ.0: Direction given by vector VID2 EQ.1: x-translational degree of freedom EQ.2: y-translational degree of freedom EQ.3: z-translational degree of freedom EQ.4: x-rotational degree of freedom EQ.5: y-rotational degree of freedom EQ.6: z-rotational degree of freedom EQ.7: x, y, and z-translational degrees of freedom EQ.8: x, y, and z-rotational degrees of freedom
VAD2	Response output type: EQ.0: Velocity EQ.1: Acceleration EQ.2: Displacement EQ.3: Nodal force (see Remark 8)
VID2	Vector ID for DOF2 = 0 for response direction. See *DEFINE_VECTOR.
RELATV	Flag for displacement, velocity, and acceleration results: EQ.0: Absolute values are requested. EQ.1: Relative values are requested (for VAD1 = 0, 1, 2 only).

Card 4	1	2	3	4	5	6	7	8
Variable	FMIN	FMAX	NFREQ	FSPACE	LCFREQ	RESTRT	OUTPUT	
Type	F	F	I	I	I	I	I	
Default	none	none	2	0	none	0	0	

VARIABLE	DESCRIPTION
FMIN	Minimum frequency for FRF output (cycles/time). See Remark 6.
FMAX	Maximum frequency for FRF output (cycles/time). See Remark 6.
NFREQ	Number of frequencies for FRF output. See Remark 6.
FSPACE	Frequency spacing option for FRF output: EQ.0: Linear EQ.1: Logarithmic EQ.2: Biased See Remark 6.
LCFREQ	Load curve ID defining the frequencies for FRF output. See Remark 6.
RESTRT	Restart option: EQ.0: Initial run EQ.1: Restart with d3eigv family files EQ.2: Restart with dumpfrf EQ.3: Restart with d3eigv family files and dumpfrf See Remark 7.
OUTPUT	Output option: EQ.0: Write amplitude and phase angle pairs. EQ.1: Write real and imaginary pairs.

Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE. Each repeat defines one subcase for FRF computation, including the title, excitation input, and response output for the subcase.

Card 5	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
TITLE	Title for the subcase

Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE.

Card 6	1	2	3	4	5	6	7	8
Variable	N1	N1TYP	DOF1	VAD1	VID1			
Type	I	I	I	I	I			
Default	none	0	none	3	0			

VARIABLE	DESCRIPTION
N1	ID of the entity specified by N1TYP for excitation input. This field is ignored when VAD1, the excitation type, is set to 0, 1, or 2 (base velocity, base acceleration, or base displacement, respectively). In this case, the excitation is applied through the node(s) under constraints, that is, the node(s) defined in *BOUNDARY_SPC_NODE or *BOUNDARY_SPC_SET.
N1TYP	Type of N1: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW)

VARIABLE	DESCRIPTION
	EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW) EQ.5: Parametric edge ID (see *IGA_EDGE_UVW) EQ.6: Parametric edge set ID (see *SET_IGA_EDGE_UVW) EQ.7: Physical face ID (see *IGA_FACE_XYZ) EQ.8: Physical face set ID (see *SET_IGA_FACE_XYZ) EQ.9: Parametric face ID (see *IGA_FACE_UVW). This option is only supported for VAD1 = 4. EQ.10: Parametric face set ID (see *SET_IGA_FACE_UVW). This option is only supported for VAD1 = 4. When VAD1, the excitation type, is set to 0, 1, or 2, which is base velocity, base acceleration, or base displacement, this field is ignored.
DOF1	Applicable degrees of freedom for excitation input (ignored if VAD1 = 4): EQ.0: Translational movement in the direction given by vector VID1 EQ.1: x-translational degree of freedom, or x-rotational degree of freedom (for torque excitation, VAD1 = 8) EQ.2: y-translational degree of freedom, or y-rotational degree of freedom (for torque excitation, VAD1 = 8) EQ.3: z-translational degree of freedom, or z-rotational degree of freedom (for torque excitation, VAD1 = 8)
VAD1	Excitation input type: EQ.0: Base velocity EQ.1: Base acceleration EQ.2: Base displacement EQ.3: Nodal force EQ.4: Pressure EQ.5: Enforced velocity by large mass method EQ.6: Enforced acceleration by large mass method EQ.7: Enforced displacement by large mass method EQ.8: Torque

VARIABLE	DESCRIPTION
	EQ.9: Base angular velocity EQ.10: Base angular acceleration EQ.11: Base angular displacement EQ.12: Torsion
VID1	Vector ID for DOF1 = 0 for excitation input. See *DEFINE_VECTOR.

Include this card after Card 6 if VAD1 = 12 on Card 6 (SUBCASE keyword option only).

Card 6.1	1	2	3	4	5	6	7	8
Variable	N11	N11TYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
N11	ID of the entity specified by N11TYP for the excitation input for the torque load (VAD1 = 12). This node or set combines with N1 to form a torque load.
N11TYP	Type of N11: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW) EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW) EQ.5: Parametric edge ID (see *IGA_EDGE_UVW) EQ.6: Parametric edge set ID (see *SET_IGA_EDGE_UVW) EQ.7: Physical face ID (see *IGA_FACE_XYZ) EQ.8: Physical face set ID (see *SET_IGA_FACE_XYZ)

Include NCASE sets of Cards 5 through 7 when the keyword option is SUBCASE.

Card 7	1	2	3	4	5	6	7	8
Variable	N2	N2TYP	DOF2	VAD2	VID2			
Type	I	I	I	I	I			
Default	none	0	none	2	0			

VARIABLE	DESCRIPTION
N2	ID of the entity specified by N2TYP for response output
N2TYP	Type of N2: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: Parametric point ID (see *IGA_POINT_UVW) EQ.4: Parametric point set ID (see *SET_IGA_POINT_UVW)
DOF2	Applicable degrees of freedom for response output: EQ.0: Direction given by vector VID2 EQ.1: x-translational degree of freedom EQ.2: y-translational degree of freedom EQ.3: z-translational degree of freedom EQ.4: x-rotational degree of freedom EQ.5: y-rotational degree of freedom EQ.6: z-rotational degree of freedom EQ.7: x, y, and z-translational degrees of freedom EQ.8: x, y, and z-rotational degrees of freedom
VAD2	Response output type: EQ.0: Velocity EQ.1: Acceleration EQ.2: Displacement

VARIABLE	DESCRIPTION
	EQ.3: Nodal force (see Remark 8)
VID2	Vector ID for DOF2 = 0 for response direction. See *DEFINE_VECTOR.

Remarks:

1. **Frequency response functions.** The FRF (frequency response functions) can be given as Displacement/Force (called Admittance, Compliance, or Receptance), Velocity/Force (called Mobility), Acceleration/Force (called Accelerance, Inertance), etc.

2. **Enforced motion.** The excitation input can be given as enforced motion (VAD1 = 5, 6, 7). Large mass method is used for this type of excitation input. The user needs to attach a large mass to the nodes where the enforced motion is applied by using the keyword *ELEMENT_MASS_{OPTION} and report the large mass per node (MPN) in the keyword *CONTROL_FREQUENCY_DOMAIN. For more details, please refer to *CONTROL_FREQUENCY_DOMAIN.

3. **Maximum frequency.** FNMAX decides how many natural vibration modes are adopted in FRF computation. LS-DYNA uses only modes with lower or equal frequency than FNMAX in FRF computation. If FNMAX is not given, the number of modes for the FRF computation is the same as the number of modes, NEIG, from the *CONTROL_IMPLICIT_EIGENVALUE keyword card unless MDMIN and MDMAX are prescribed (see Remark 5).

4. **Maximum/minimum mode.** MDMIN and MDMAX decide which mode(s) are adopted in FRF computation. This option helps calculate the contribution from a single mode (MDMIN = MDMAX) or several modes (MDMIN < MDMAX). If only MDMIN is given, LS-DYNA uses the single mode (MDMIN) to compute FRF. In a restart run based on an existing eigenmode database d3eigv (RESTRT = 1 or 3), MDMAX should be a positive integer equal to or less than the number of eigenmodes available in d3eigv.

5. **Damping.** Damping can be prescribed in several ways:

   a) To use a constant modal damping coefficient ζ for all the modes, define DAMPF only. LCDMP, LCTYP, DMPMAS, and DMPSTF are ignored.

   b) To use mode-dependent modal damping, define a load curve (*DEFINE_CURVE) and specify if the abscissa value represents the frequency or mode number by LCTYP. DMPMAS and DMPSTF are ignored.

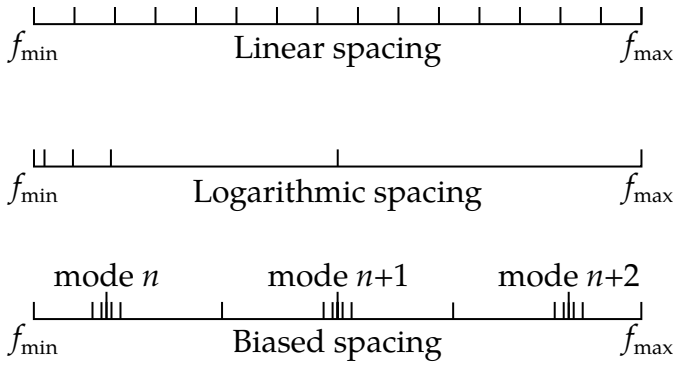


Figure 23-5. Spacing options of the frequency points.

c) To use Rayleigh damping, define DMPMAS (α) and DMPSTF (β), keep DAMPF as 0.0, and keep LCDMP and LCTYP as 0. The damping matrix in Rayleigh damping is defined as $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$, where, **C**, **M** and **K** are the damping, mass, and stiffness matrices, respectively.

6. **Frequency points.** There are two methods to define the frequencies.

   a) The first method is to define FMIN, FMAX, NFREQ, and FSPACE. FMIN and FMAX specify the frequency range of interest, and NFREQ specifies the number of frequencies at which results are required. FSPACE specifies the type of frequency spacing (linear, logarithmic, or biased) to be used.

      These frequency points for which results are required can be spaced equally along the frequency axis (on a linear or logarithmic scale). Or they can be biased toward the eigenfrequencies (the frequency points are placed closer together at eigenfrequencies in the frequency range) to obtain a detailed definition of the response close to resonance frequencies. See Figure 23-5.

   b) The second method uses a load curve (LCFREQ) to define the frequencies of interest.

7. **RESTRT field.** To save time in subsequent runs, the modal analysis stored in the d3eigv file during the first run can be reused by setting RESTRT = 1.

   RESTRT = 2 or 3 is used when the user wants to add extra vibration modes to the FRF computation. After the initial FRF computation, the number of vibration modes may not be enough. For example, in the initial computation, the only vibration modes used may be up to 500 Hz. However, vibration modes at higher frequencies are needed, so just computing the extra modes (frequencies above 500 Hz) and adding the contribution from these extra modes to the previous FRF results is more efficient.

   For RESTRT = 2, LS-DYNA runs a new modal analysis, reads in the previous FRF results (stored in the binary dump file dumpfrf), and adds the contribution

from the new modes. For RESTRT = 3, LS-DYNA reads in d3eigv family files generated elsewhere, reads in dumpfrf, and adds the contribution from the new modes.

8. **Nodal force response output.** For nodal force response (VAD2 = 3), the same nodes (or node set) must be defined in *DATABASE_NODAL_FORCE_GROUP. In addition, the MSTRES field for the *CONTROL_IMPLICIT_EIGENVALUE keyword must be set to 1.

9. **REDUCED option.** The REDUCED option enables running FRF analysis based on reduced eigenvector output (LanczosEigenVectors for EIGMTH = 2, 3, 5, and 6, LOBPCGEigenVectors for EIGMTH = 102, and FastLnzEigenVectors for EIGMTH = 103). See Remark 6 of the manual page for *CONTROL_IMPLICIT_-EIGENVALUE.

10. **SUBCASE option.** To execute multiple FRF subcase computations in one run under one keyword input, append SUBCASE to the keyword name. Each subcase has its own excitation input (Cards 6 and 6.1) and response output (Card 7). They share the same damping setting and the same set of eigenmodes for modal superposition. The result files have the prefix subcase*_ before the original file name. For example, subcase2_frf_amplitude is the frf_amplitude result for subcase 2 (in the same order as the input deck).

***FREQUENCY_DOMAIN_LOCAL**_*OPTION1_{OPTION2}*

Available options for *OPTION1* include:

NODE_SET

PART

PART_SET

For *OPTION2* the available option is:

<BLANK>

EXCLUDE

Purpose: Define local nodes and parts to be included in frequency domain analysis (random vibration, response spectrum analysis and SSD). When the keyword OPTION2 EXCLUDE is used, the nodes and parts defined in this keyword are excluded from participating in the frequency domain analysis. Without this keyword, the frequency domain analysis is always performed on the whole model.

Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
ID*n*	Node set ID or Part ID or Part set ID *n*.

Remarks:

This keyword defines the nodes/parts to be included in frequency domain analysis (random vibration analysis, response spectrum analysis and SSD). The nodes/parts not defined here will be skipped in the analysis (in result databases, the response for those nodes/parts is marked as 0). With this keyword, significant CPU cost saving can be obtained if the user wants only response on a fraction of the nodes/parts of the model. With the option EXCLUDE, this keyword can be used to remove some nodes/parts from the analysis. Without using this keyword, the frequency domain analysis is always performed on the whole model.

***FREQUENCY_DOMAIN_MODE**_*OPTION1*_*{OPTION2}*

Available options for *OPTION1* include:

LIST

GENERATE

SET

LOAD_PROJECTION

MODAL_COEFFICIENT

For *OPTION2* the available option is:

<BLANK>

EXCLUDE

Purpose: Define vibration modes to be used in modal superposition, modal acceleration or modal combination procedures for mode-based frequency domain analysis (such as frequency response functions, steady state dynamics, random vibration analysis and response spectrum analysis). When the keyword OPTION2 EXCLUDE is used, the modes defined in this keyword are excluded from participating in the modal superposition, modal acceleration, or modal combination procedures.

Card Summary:

Card 1a. This card is included if and only if OPTION1 is LIST. Include as many cards as necessary. This input ends at the next keyword ("*") card.

MID1	MID2	MID3	MID4	MID5	MID6	MID7	MID8

Card 1b. This card is included if and only if OPTION1 is GENERATE. Include as many cards as necessary. This input ends at the next keyword ("*") card.

M1BEG	M1END	M2BEG	M2END	M3BEG	M3END	M4BEG	M4END

Card 1c. This card is included if and only if OPTION1 is SET.

SID							

Card 1d. This card is included if and only if OPTION1 is LOAD_PROJECTION.

NMSORT	DSKIP						

Card 1e. This card is included if and only if OPTION1 is MODAL_COEFFICIENT.

DSKIP							

Data Cards:

Mode ID Cards. For LIST keyword option, list the mode IDs. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	MID1	MID2	MID3	MID4	MID5	MID6	MID7	MID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
MID*n*	Mode ID *n*.

Mode Block Cards. For GENERATE keyword option specify ranges of modes. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	M1BEG	M1END	M2BEG	M2END	M3BEG	M3END	M4BEG	M4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
M*n*BEG	First mode ID in block *n*.
M*n*END	Last mode ID in block *n*. All mode ID's between and including M*n*BEG and M*n*END are added to the list.

Mode Set Card. For SET keyword option specify a mode set. Include only one card.

Card 1c	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							

VARIABLE	DESCRIPTION
SID	Mode set identification (see *SET_MODE).

Load Projection Card. For LOAD_PROJECTION keyword option specify the number of modes to be retained, which have the largest load projection ratios, or specify a threshold load projection ratio for mode truncation.

Card 1d	1	2	3	4	5	6	7	8
Variable	NMSORT	DSKIP						
Type	I	F						

VARIABLE	DESCRIPTION
NMSORT	Number of modes to be retained which have the largest load projection ratios.
DSKIP	The threshold load projection ratio. All modes with load projection ratio less than this value will be skipped.

Modal Coefficient Ratio Card. For MODAL_COEFFICIENT keyword option, specify a threshold modal coefficient ratio (over the largest modal coefficient) for mode truncation.

Card 1e	1	2	3	4	5	6	7	8
Variable	DSKIP							
Type	F							

VARIABLE	DESCRIPTION
DSKIP	The threshold modal coefficient ratio. All modes with the ratio of its modal coefficient over the largest modal coefficient less than this value will be skipped.

Remarks:

1. **Excluding Modes.** This keyword can be used to remove the modes that add a much smaller contribution to the total structural response than the other modes from the modal superposition, modal acceleration or modal combination procedures of the mode-based frequency domain analysis.

2. **Overriding Mode Lists.** The mode list defined by this keyword overrides the modes specified by MDMIN, MDMAX (or FNMIX, FNMAX) in the keywords *FREQUENCY_DOMAIN_FRF, *FREQUENCY_DOMAIN_SSD, etc.

*FREQUENCY_DOMAIN_PATH_{OPTION}

Available options include:

<BLANK>

NOJOBID

PARTITION

RESIDUAL_VECTOR

Purpose: Specify the path and file name of binary databases, such as d3eigv, containing mode information for restarting frequency domain analyses, such as FRF, SSD, random vibration, and response spectrum analysis. When the RESIDUAL_VECTOR option is used, this keyword defines the path for locating the binary database d3resvec, which saves the residual vectors.

The PARTITION option supports assigning different binary databases to different frequency ranges. Specifically, each frequency range can be associated with different eigenmodes and modal shape vectors provided by the binary database. This option provides a model for materials that have frequency-dependent properties.

Card Summary:

Card 1a. Include this card if using the PARTITION keyword option.

FBEG	FEND	FILENAME

Card 1b. Include this card if you are *not* using a keyword option (<BLANK>) or if you are *not* using the PARTITION keyword option.

FILENAME

Data Card Definitions:

Partition Cards. Card 1 for the PARTITION keyword option. Include one card for each frequency range. This input ends at the next keyword ("*") card. See Remark 3.

Card 1a	1	2	3	4	5	6	7	8
Variable	FBEG	FEND	FILENAME					
Type	F	F	A60					
Default	none	none	none					

Filename Card. Card 1 format used when the keyword option is unused (<BLANK>) or is *not* PARTITION.

Card 1b	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FBEG	Beginning frequency for using this database
FEND	Ending frequency for using this database
FILENAME	Path to the database which contains modal information.

Remarks:

1. **Binary file location.** If the binary database files are located in the runtime directory, this card is not required for cases *without* partitioning.

2. **NOJOBID option.** When the option NOJOBID is active, LS-DYNA will not add a prefix to the FILENAME on card 1. Even if the user is running LS-DYNA with *CASE or has JOBID in the command line, LS-DYNA will not add a prefix when NOJOBID is active.

To understand the intent of this feature, recall that job ID numbers exist to ensure that filenames are unique so that output files are not overwritten for each calculation in some sequence. When modal data is read in as part of a restart calculation from some database already on disk, the default behavior is to replace FILENAME with FILENAME prefixed by the job ID. Consequently, without the NOJOBID option, reusing a single database across multiple modal analyses would involve either renaming or symbolically linking the existing file to the prefixed filename. The NOJOBID option circumvents this difficulty.

This is useful when the user wants to run multiple load cases for a single frequency domain analysis (FRF, SSD, random vibration, response spectrum analysis) and would like to run each load case with the restart option, RESTRT = 1, in *FREQUENCY_DOMAIN_FRF, *FREQUENCY_DOMAIN_RANDOM_VIBRATION, *FREQUENCY_DOMAIN_RESPONSE_SPECTRUM, etc., or RESTMD = 1 in *FREQUENCY_DOMAIN_SSD. In this case, with the option NOJOBID, a single eigenvector database, such as d3eigv, can be used for all the load cases.

3. **Frequency partition.** When the keyword option PARTITION is active, the binary database designated by FILENAME is used for the frequency range starting from (and including) FBEG and ending at (not including) FEND.

*FREQUENCY_DOMAIN_RANDOM_VIBRATION_{OPTION}

Available options include:

<BLANK>

FATIGUE

Purpose: Set random vibration control options. When the FATIGUE keyword option is used, compute fatigue life of structures or parts under random vibration (see Remark 2).

Card Summary:

Card 1. This card is required.

MDMIN	MDMAX	FNMIN	FNMAX	RESTRT			

Card 2. This card is required.

DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF	DMPTYP		

Card 3. This card is required.

VAFLAG	METHOD	UNIT	UMLT	VAPSD	VARMS	NAPSD	NCPSD

Card 3.1. This card is read for VAFLAG = 5, 6 and 7. This card is optional.

PREF							

Card 4. This card is required.

LDTYP	IPANELU	IPANELV	TEMPER		LDFLAG	ICOARSE	TCOARSE

Card 5. Include NAPSD of this card.

SID	STYPE	DOF	LDPSD	LDVEL	LDFLW	LDSPN	CID

Card 6. Include NCPSD of this card.

LOAD_I	LOAD_J	LCTYP2	LDPSD1	LDPSD2			

Card 7. This card is included if the FATIGUE keyword option is used.

MFTG	NFTG	STRTYP	TEXPOS	STRSF	INFTG	SRANGE	

Card 7.1a. Include NFTG cards that are any combination of this card and Card 7.1b. This card is included when LCID > 0.

PID	LCID	PTYPE	LTYPE				SNLIMT

Card 7.1b. Include NFTG cards that are any combination of this card and Card 7.1a. This card is included when LCID < 0.

PID	LCID	PTYPE		A	B	STHRES	SNLIMT

Card 7.2. Include INFTG of this card when INFTG > 0.

FILENAME

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MDMIN	MDMAX	FNMIN	FNMAX	RESTRT			
Type	I	I	F	F	I			
Default	1	optional	0.0	optional	0			

VARIABLE	DESCRIPTION
MDMIN	First mode in modal superposition method (optional)
MDMAX	Last mode in modal superposition method (optional)
FNMIN	Minimum natural frequency in modal superposition method (optional)
FNMAX	Maximum natural frequency in modal superposition method (optional)
RESTRT	Restart option (see Remark 4): EQ.0: A new modal analysis is performed. EQ.1: Restart with d3eigv. EQ.2: Sum up precomputed random vibration PSD and RMS results only.

Card 2	1	2	3	4	5	6	7	8
Variable	DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF	DMPTYP		
Type	F	I	I	F	F	I		
Default	0.0	0	0	0.0	0.0	0		

VARIABLE	DESCRIPTION
DAMPF	Modal damping coefficient, ζ.
LCDAM	Load Curve ID defining mode dependent modal damping coefficient ζ.
LCTYP	Type of load curve defining modal damping coefficient: EQ.0: Abscissa value defines frequency. EQ.1: Abscissa value defines mode number.
DMPMAS	Mass proportional damping constant α, in Rayleigh damping.
DMPSTF	Stiffness proportional damping constant β, in Rayleigh damping.
DMPTYP	Type of damping: EQ.0: Modal damping EQ.1: Broadband damping

Card 3	1	2	3	4	5	6	7	8
Variable	VAFLAG	METHOD	UNIT	UMLT	VAPSD	VARMS	NAPSD	NCPSD
Type	I	I	I	F	I	I	I	I
Default	none	0	none	↓	none	none	1	0

VARIABLE	DESCRIPTION
VAFLAG	Loading type: EQ.0: No random vibration analysis EQ.1: Base acceleration EQ.2: Random pressure EQ.3: Plane wave EQ.4: Shock wave EQ.5: Progressive wave EQ.6: Reverberant wave EQ.7: Turbulent boundary layer wave EQ.8: Nodal force EQ.9: Base velocity EQ.10: Base displacement EQ.11: Enforced acceleration by large mass method (see Remark 15) EQ.12: Enforced velocity by large mass method (see Remark 15) EQ.13: Enforced displacement by large mass method (see Remark 15)
METHOD	Method for modal response analysis (see Remark 6): EQ.0: Method set automatically by LS-DYNA (recommended) EQ.1: Modal superposition method EQ.2: Modal acceleration method EQ.3: Modal truncation augmentation method
UNIT	Flag for acceleration unit conversion (see Remark 7): EQ.0: Use [length unit]/[time unit]^2 as unit of acceleration. EQ.1: Use g as unit for acceleration and SI units (Newton, kg, meter, second, etc.) elsewhere. EQ.2: Use g as unit for acceleration and engineering units (lbf, lbf $\times$ second^2/inch, inch, second, etc.) elsewhere. EQ.3: Use g as unit for acceleration and units (kN, kg, mm, ms, GPa, etc.) elsewhere. EQ.4: Use g as unit for acceleration and units (Newton, ton,

VARIABLE	DESCRIPTION
	mm, second, MPa, etc.) elsewhere. EQ.-1: Use g as unit for acceleration and provide the multiplier for converting g to [length unit]/[time unit]^2.
UMLT	Multiplier for converting g to [length unit]/[time unit]^2 (used only for UNIT = -1).
VAPSD	Flag for PSD output: EQ.0: Absolute PSD output is requested. EQ.1: Relative PSD output is requested (used only for VAFLAG = 1).
VARMS	Flag for RMS output: EQ.0: Absolute RMS output is requested. EQ.1: Relative RMS output is requested (used only for VAFLAG = 1).
NAPSD	Number of auto PSD load definition. Card 5 is repeated NAPSD times; one for each auto PSD load definition.
NCPSD	Number of cross PSD load definition. Card 6 is repeated NCPSD times; one for each cross PSD load definition.

This card is read when VAFLAG = 5, 6, or 7. It is optional.

Card 3.1	1	2	3	4	5	6	7	8
Variable	PREF							
Type	I							
Default	2.e-5							

VARIABLE	DESCRIPTION
PREF	Reference pressure used to convert acoustic pressure to SPL (dB). This reference pressure is only needed if VAFLAG = 5, 6, or 7. For VAFLAG = 5, 6, or 7, if this Card 3.1 is not provided, LS-DYNA will use a default value 2.0×10^{-5} as the reference pressure.

Card 4	1	2	3	4	5	6	7	8
Variable	LDTYP	IPANELU	IPANELV	TEMPER		LDFLAG	ICOARSE	TCOARSE
Type	I	I	I	F		I	I	F
Default	0	0	0	0.0		0	0	0.1

VARIABLE	DESCRIPTION
LDTYP	Excitation load (LDPSD in Card 5) type: EQ.0: PSD EQ.1: SPL (for plane wave only) EQ.2: Time history load
IPANELU	Number of strips in *U*-direction (used only for VAFLAG = 5, 6, 7). See Remark 3.
IPANELV	Number of strips in *V*-direction (used only for VAFLAG = 5, 6, 7). See Remark 3.
TEMPER	Temperature
LDFLAG	Type of loading curves: EQ.0: Log-log interpolation (default) EQ.1: Semi-log interpolation EQ.2: Linear-linear interpolation
ICOARSE	Option for PSD curve coarsening: EQ.0: No coarsening; use original data (default). EQ.1: Coarsening by keeping only peaks and troughs EQ.2: Coarsening by removing intermediate points whose slope change percentage is less than prescribed tolerance (TCOARSE)
TCOARSE	Tolerance for slope change percentage for removing intermediate points from PSD curve (default is 0.1) for ICOARSE = 2

Auto PSD Cards. Include NAPSD cards of this format, one per excitation.

Card 5	1	2	3	4	5	6	7	8
Variable	SID	STYPE	DOF	LDPSD	LDVEL	LDFLW	LDSPN	CID/VID
Type	I	I	I	I	I	I	I	I
Default	↓	none	none	none	0	0	0	0

VARIABLE	DESCRIPTION
SID	GE.0: Set ID for the panel exposed to acoustic environment, or the nodes subjected to nodal force excitation, or nodal acceleration excitation. For VAFLAG = 1, base acceleration, leave this as blank LT.0: Used to define the cross-PSD. |SID| is the ID of the load cases.
STYPE	Flag specifying meaning of SID: EQ.0: Node EQ.1: Node set EQ.2: Segment set EQ.3: Part EQ.4: Part set LT.0: Used to define the cross-PSD. |STYPE| is the ID of the load cases.
DOF	Applicable degrees-of-freedom for nodal force excitation or base acceleration (DOF = 1, 2, and 3), or wave direction: EQ.0: Translational movement in direction given by vector VID EQ.1: x-translational degree-of-freedom EQ.2: y-translational degree-of-freedom EQ.3: z-translational degree-of-freedom
LDPSD	Load curve for PSD, SPL, or time history excitation

VARIABLE	DESCRIPTION
LDVEL	Load curve for phase velocity
LDFLW	Load curve for exponential decay for TBL in flow-wise direction
LDSPN	Load curve for exponential decay for TBL in span-wise direction
CID/VID	Coordinate system ID for defining wave direction, see *DEFINE_-COORDINATE_SYSTEM; or vector ID for defining load direction for nodal force or base excitation, see *DEFINE_VECTOR.

Cross PSD Cards. Include NCPSD cards of this format, one per excitation.

Card 6	1	2	3	4	5	6	7	8
Variable	LOAD_I	LOAD_J	LCTYP2	LDPSD1	LDPSD2			
Type	I	I	I	I	I			
Default	none	none	0	none	none			

VARIABLE	DESCRIPTION
LOAD_I	ID of load *i* for cross PSD
LOAD_J	ID of load *j* for cross PSD
LCTYP2	Type of load curves (LDPSD1 and LDPSD2) for defining cross PSD: EQ.0: LDPSD1 defines real part and LDPSD2 defines imaginary part. EQ.1: LDPSD1 defines magnitude and LDPSD2 defines phase angle (in degrees). EQ.2: LDPSD1 defines magnitude and LDPSD2 defines phase angle (in radians).
LDPSD1	Load curve for real part or magnitude of cross PSD
LDPSD2	Load curve for imaginary part or phase angle of cross PSD

Fatigue Card. Additional card for FATIGUE keyword option.

Card 7	1	2	3	4	5	6	7	8
Variable	MFTG	NFTG	STRTYP	TEXPOS	STRSF	INFTG	SRANGE	
Type	I	I	I	F	F	I	F	
Default	0	1	0	0.0	1.0	0	3.0	

VARIABLE	DESCRIPTION
MFTG	Method for random fatigue analysis: EQ.-1: Steinberg's three-band method using peak-crossing frequency to compute the number of stress cycles EQ.0: No fatigue analysis (default) EQ.1: Steinberg's three-band method using zero-crossing frequency (with positive slope) to compute the number of stress cycles EQ.2: Dirlik method EQ.3: Narrow band method EQ.4: Wirsching method EQ.5: Chaudhury and Dover method EQ.6: Tunna method EQ.7: Hancock method EQ.8: Lalanne method.
NFTG	Field specifying the number of S - N curves to be defined: GE.0: Number of S - N curves defined by Cards 7.1a and 7.1b which are included in the required combination to get to "NFTG" number of cards, one for each S - N fatigue curve definition. The default value is 1. EQ.-999: S - N curves are defined through *MAT_ADD_FATIGUE.
STRTYP	Stress type of S - N curve in fatigue analysis: EQ.0: von-Mises stress

VARIABLE	DESCRIPTION
	EQ.1: Maximum principal stress (not implemented) EQ.2: Maximum shear stress (not implemented) EQ.-n: The n^{th} stress component
TEXPOS	Exposure time
STRSF	Stress scale factor to accommodate different ordinates in S - N curve (not used if NFTG = -999): EQ.1: Used if the ordinate in S - N curve is stress range (default) EQ.2: Used if the ordinate in S - N curve is stress amplitude
INFTG	Flag for including initial damage ratio (see Remark 5): EQ.0: No initial damage ratio, GT.0: Read existing d3ftg files to obtain the initial damage ratio. When INFTG > 1, the initial damage ratio comes from multiple loading cases (correspondingly, multiple binary databases, defined by Card 7.2). The value of INFTG should be ≤ 10.
SRANGE	Stress range for fatigue analysis. The default is 3.0, which means 3.0 times the RMS stress)

S-N Curve Cards. NFTG additional cards for FATIGUE keyword option of a combination of Cards 7.1a and 7.1b. Each Card 7.1a or Card 7.1b defines one zone for fatigue analysis and the corresponding S - N fatigue curve for that zone. Card 7.1a is included for LCID > 0.

Card 7.1a	1	2	3	4	5	6	7	8
Variable	PID	LCID	PTYPE	LTYPE				SNLIMT
Type	I	I	I	I				I
Default	none	none	0	0				0

VARIABLE	DESCRIPTION
PID	Part ID, part set ID, or element (solid, shell, beam, thick shell) set ID

VARIABLE	DESCRIPTION
LCID	S - N fatigue curve ID for the current part or part set: GT.0: S - N fatigue curve ID EQ.-1: S - N fatigue curve uses equation $NS^b = a$ EQ.-2: S - N fatigue curve uses equation $\log(S) = a - b\log(N)$ EQ.-3: S - N fatigue curve uses equation $S = a\,N^b$ EQ.-4: S - N fatigue curve uses equation $S = a - b\log(N)$
PTYPE	Type of PID. EQ.0: Part (default) EQ.1: Part set EQ.2: SET_SOLID EQ.3: SET_BEAM EQ.4: SET_SHELL EQ.5: SET_TSHELL
LTYPE	Type of LCID. EQ.0: Semi-log interpolation (default) EQ.1: Log-log interpolation EQ.2: Linear-linear interpolation
SNLIMT	Flag setting algorithm used when stress is lower than the lowest stress on S - N curve (if LCID > 0): EQ.0: Use the life at the last point on S - N curve EQ.1: Extrapolation from the last two points on S - N curve EQ.2: Infinity

S-N Curve Cards. NFTG additional cards for FATIGUE keyword option of a combination of Cards 7.1a and 7.1b. Each Card 7.1a or Card 7.1b defines one zone for fatigue analysis and the corresponding S - N fatigue curve for that zone. Card 7.1b is included for LCID < 0.

Card 7.1b	1	2	3	4	5	6	7	8
Variable	PID	LCID	PTYPE		A	B	STHRES	SNLIMT
Type	I	I	I		F	F	F	I
Default	none	none	0		none	none	0.	0

VARIABLE	DESCRIPTION
PID	Part ID, part set ID, or element (solid, shell, beam, thick shell) set ID
LCID	S - N fatigue curve ID for the current part or part set: GT.0: S - N fatigue curve ID EQ.-1: S - N fatigue curve uses equation $NS^b = a$ EQ.-2: S - N fatigue curve uses equation $\log(S) = a - b\log(N)$ EQ.-3: S - N fatigue curve uses equation $S = a\,N^b$ EQ.-4: S - N fatigue curve uses equation $S = a - b\log(N)$
PTYPE	Type of PID. EQ.0: Part (default) EQ.1: Part set EQ.2: SET_SOLID EQ.3: SET_BEAM EQ.4: SET_SHELL EQ.5: SET_TSHELL
A	Material parameter a in S - N fatigue equation.
B	Material parameter b in S - N fatigue equation.
STHRES	Fatigue threshold

VARIABLE	DESCRIPTION
SNLIMT	Flag setting algorithm used when stress is lower than STHRES: EQ.0: Use the life at STHRES EQ.1: Ignored EQ.2: Infinity

Initial Damage Card. INFTG additional cards for FATIGUE keyword option when INFTG > 0.

Card 7.2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	d3ftg							

VARIABLE	DESCRIPTION
FILENAME	Path and name of existing binary database for fatigue information.

Remarks:

1. **Historical Background.** This command evaluates the structural random vibration response due to aero acoustic loads, base excitation, or nodal force. This capability originated in Boeing's in-house code N-FEARA, which is a NIKE3D-based Finite Element tool for performing structural analysis with vibro-acoustic loads. The main developer of N-FEARA is Mostafa Rassaian from the Boeing Company.

2. **Fatigue.** MSTRES must be set to 1 in the keyword *CONTROL_IMPLICIT_EIGENVALUE for the FATIGUE keyword option because the fatigue analysis is depending on stresses.

3. **IPANELU and IPANELV.** The number of strips the in U and V-directions are used to group the elements and thereby reduce the number integration domains reducing computational expense. This option is only available for VAFLAG = 5, 6, and 7.

4. **Restarting.** Restart option RESTRT = 1 is used when mode analysis has been done previously. In this case, LS-DYNA skips modal analysis and reads in the

d3eigv files from the prior execution. For RESTRM = 1, always use MDMIN = 1 and set MDMAX to the number of modes in the previous run (this can be found in the ASCII file eigout, or it can be extracted from the d3eigv files using LS-PrePost). Restart option RESTRT = 2 is used to sum up PSD and RMS results from multiple runs to provide total PSD and RMS response of the same model, subjected to multiple loading resources. The file name and path of the PSD and RMS binary databases (by default, d3psd and d3rms) to be summed, are defined by the keywords *DATABASE_FREQUENCY_BINARY_D3PSD_SUMMATION, and *DATABASE_FREQUENCY_BINARY_D3RMS_SUMMATION. The multiple loading resources can be of different loading type (defined by VAFLAG), and they are applied to the model simultaneously.

5. **Accumulated Fatigue.** The fatigue damage ratio can be accumulated over multiple load cases by setting INFTG = 1. This is useful when a structure is subjected to multiple independent random vibrations. LS-DYNA calculates the total damage ratio by adding the damage ratio from the current calculation to the damage ratio of the previous calculation which are stored in the previous calculation's fatigue database (d3ftg by default). The previous d3ftg file will be overwritten by the new one if it is in the same directory.

6. **Automatic Method Selection.** If METHOD = 0, LS-DYNA uses modal superposition method for cases (VAFLAG) 4, 5, 6, 7. For cases 1, 2, 3 and 8, LS-DYNA uses modal superposition method when preload condition is present and uses modal acceleration method when preload condition is not present.

7. **Units.** In a set of consistent units, the unit for acceleration is defined as

$$1\ (\text{acceleration unit}) = \frac{1(\text{length unit})}{[1(\text{time unit})]^2}$$

Some users in industry prefer to use g (acceleration due to gravity) as the unit for acceleration. For example,

$$1\text{g} = 9.81\frac{\text{m}}{\text{s}^2} = 386.089\frac{\text{inch}}{\text{s}^2}$$

If the input and output use g as the unit for acceleration, select UNIT = 1, 2, or 3.

If UNIT = -1, a multiplier (UMLT) for converting g to [length unit]/[time unit]^2 is needed and it is defined by

$$1\text{g} = \text{UMLT} \times \frac{[\text{length unit}]}{[\text{time unit}]^2}$$

For more information about the consistent units, see GS.21 (GETTING STARTED).

8. **Restrictions on Load Curves.** The load curves LDPSD, LDVEL, LDFLW, and LDSPN must all be defined using the same number of points. The number of

points in the load curve LDDAMP can be different from those for LDPSD, LDVEL, LDFLW, and LDSPN.

9. **Wave Direction.** Wave direction is determined by DOF and CID/VID. CID/VID represents a local *U-V-W* coordinate system for defining acoustic wave direction; only partially correlated waves (VAFLAG=5, 6, 7) need this local coordinate system. For nodal force, base excitation, plane wave or random pressure, CID represents a vector ID defining the load direction (DOF = ±4).

10. **Stress / Strain Computation.** To get stress results (PSD and RMS) from random vibration analysis, the MSTRES field of the *CONTROL_IMPLICIT_EIGENVALUE keyword should be set to 1. To get strain results (PSD and RMS) the STRFLG field of *DATABASE_EXTENT_BINARY should be set to 1. To get stress component results for beam elements (which are not based on resultant formulation), the BEAMIP field of *DATABASE_EXTENT_BINARY should be set greater than 0.

11. **Binary Plot Databases.** PSD and RMS results are given for all nodes and elements. PSD results are written in binary to a file named d3psd. Similarly, RMS results are written in binary to a file named d3rms. See the keyword *DATABASE_FREQUENCY_BINARY_*{OPTION}* for more details.

12. **ASCII Output for Displacement.** Displacement, velocity, and acceleration PSD results are output into ASCII file nodout_psd. The set nodes for which data is written to nodout_psd is specified with the *DATABASE_HISTORY_NODE keyword.

13. **ASCII Output for Stress.** Stress PSD results are output into ASCII file elout_psd. The set of solid, beam, shell, and thick shell elements to be written to the elout_psd file are specified with the following keywords:

    *DATABASE_HISTORY_SOLID

    *DATABASE_HISTORY_BEAM

    *DATABASE_HISTORY_SHELL

    *DATABASE_HISTORY_TSHELL

14. **Cross PSD.** The cross PSD can be defined as complex variables to consider phase difference. In that case, two curves are needed to define the cross PSD (LCPSD1 and LCPSD2). Two load IDs are needed to define the cross PSD (LOAD_I and LOAD_J). They are simply the ordering numbers by which the auto PSDs are defined. For example, the first Card 5 defines load 1 and the second Card 5 defines the load 2. No cross PSD is required if two loads are uncorrelated. Cross PSD for any pair of two correlated loads is defined only once –

from lower load ID to higher load ID (e.g. 1->2, 1->3, 2->3, …). The cross PSD from higher load ID to lower load ID (e.g. 2->1, 3->1, 3->2, …) is added by LS-DYNA automatically by using the relationship

$$G_{ji} = \overline{G_{ij}}$$

where G_{ij} is the cross PSD from load i to load j, and $\overline{\ }$ represents the complex conjugate.

15. **Large Mass Method.** For the cases with enforced motion excitation, such as nodal velocity, acceleration, or displacement, the large mass method can be used to compute the random vibration response. The excitation input can be given as enforced motion curves (VAFLAG = 11, 12, 13). To use the large mass method, a large mass must be attached to the nodes where the enforced motion is applied by using the keyword *ELEMENT_MASS_*{OPTION}*; MPN must also be set in the keyword *CONTROL_FREQUENCY_DOMAIN. For more details, please refer to *CONTROL_FREQUENCY_DOMAIN.

16. **Cross Correlation.** Cross correlation can be defined only for same type of excitations (e.g. nodal force, random pressure). Correlation between different types of excitations is not allowed.

17. **Output for Fatigue Data.** When the FATIGUE option is used, a binary plot file, d3ftg, is written. Five results are included in d3ftg:

    Result 1. Cumulative damage ratio

    Result 2. Expected fatigue life

    Result 3. Zero-crossing frequency

    Result 4. Peak-crossing frequency

    Result 5. Irregularity factor

    These results are given as element variables. Irregularity factor is a real number from 0 to 1. A sine wave has an irregularity factor of 1, while white noise has an irregularity factor of 0. The lower the irregularity factor, the closer the process is to the broad band case.

18. **Stress Threshold for Fatigue.** In some materials, the S – N curve flattens out eventually, so that below a certain threshold stress STHRES failure does not occur no matter how long the loads are cycled. SNLIMT can be set to 2 in this case. For other materials, such as aluminum, no threshold stress exists and SNLIMT should be set to 0 or 1 for added level of safety.

19. **Restriction on Fatigue Cards.** When the FATIGUE option is used, all fatigue cards (Card 7) *must* be of the same PTYPE (PART or SET of ELEMENTS).

20. **Format for S - N Curves.** S - N curves can be defined by *DEFINE_CURVE or by using a predefined equation by setting LCID < 0. When specified with curves using *DEFINE_CURVE, the abscissa values (first column) represent N (number of cycles to failure) and the ordinate values represent S (stress). The predefined equations are listed as follows:

a) *LCID = -1.*

$$NS^b = a$$

b) *LCID = -2.*

$$\log(S) = a - b\log(N)$$

c) *LCID = -3.*

$$S = a\,N^b$$

d) *LCID = -4.*

$$S = a - b\log(N)$$

Here N is the number of cycles for fatigue failure and S is the stress amplitude. Please note that the first three equations can be converted to each other with some minor manipulations on the constants a and b.

References:

Mostafa Rassaian, Jung-Chuan Lee, N-FEARA – NIKE3D-based FE tool for structural analysis of vibro-acoustic loads, Boeing report, 9350N-GKY-02-036, December 5, 2003.

*FREQUENCY_DOMAIN_RESPONSE_SPECTRUM_{OPTION1}_{OPTION2}

Available options for *OPTION1* include:

<BLANK>

DDAM

Available options for *OPTION2* include:

<BLANK>

MISSING_MASS_CORRECTION (see Remark 13)

Purpose: Obtain the peak response of a structure by performing a response spectrum computation. The structure is subjected to an input response spectrum load, such as the acceleration spectrum load in earthquake engineering. The maximum response is, then, evaluated using the modal superposition method. With the DDAM option, LS-DYNA performs a Dynamic Design Analysis Method (DDAM) shock spectrum analysis. DDAM is a U.S. Navy methodology for qualifying shipboard equipment and supporting structures subject to dynamic loading caused by underwater explosions (UNDEX).

Card Summary:

Card 1. This card is required.

MDMIN	MDMAX	FNMIN	FNMAX	RESTRT	MCOMB	RELATV	MPRS

Card 2.1. Include this card if MCOMB = 99 in Card 1.

MCOMB1	MCOMB2						

Card 2.2. Include this card if MCOMB = 99 in Card 1.

W1	W2						

Card 3. Include this card if MPRS = 2 in Card 1.

R40							

Card 4. Include this card if *OPTION2* is MISSING_MASS_CORRECTION.

ZPA	FILENAME

Card 5. This card is required.

DAMPF	LCDAMP	LDTYP	DMPMAS	DMPSTF			

Card 6a. Include this card if *OPTION1* is unset (<BLANK>). This card can be repeated if 2 or more input spectra exist (multiple-point response spectrum).

LCTYP	DOF	LC/TBID	SF	VID	LNID	LNTYP	INFLAG

Card 6b. Include this card if using the DDAM keyword option.

STD	UNIT	AMIN	VID	XC	YC	ZC	EFFMAS

Card 7a. Include this card if using the DDAM keyword option when STD > 0 in Card 6b.

SHPTYP	MOUNT	MOVEMT	MATTYP				

Card 7b.1. Include this card if using the DDAM keyword option when STD = -1 in Card 6b.

AF	AA	AB	AC	AD			

Card 7b.2. Include this card if using the DDAM keyword option when STD = -1 in Card 6b.

VF	VA	VB	VC				

Card 8. Include this card if using the DDAM keyword option with MCOMB = -14.

SID							

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MDMIN	MDMAX	FNMIN	FNMAX	RESTRT	MCOMB	RELATV	MPRS
Type	I	I	F	F	I	I	I	I
Default	1	optional	0.0	optional	0	0	0	0

VARIABLE	DESCRIPTION
MDMIN	First mode in modal superposition method (optional)
MDMAX	Last mode in modal superposition method (optional)

VARIABLE	DESCRIPTION
FNMIN	Minimum natural frequency in modal superposition method (optional)
FNMAX	Maximum natural frequency in modal superposition method (optional)
RESTRT	Restart option: EQ.0: A new run including modal analysis, EQ.1: Restart with d3eigv family files created elsewhere.
MCOMB	Method for combination of modes: EQ.-14: NRL-SUM method with CSM (Closely Spaced Modes) treatment, where the CSM pairs are defined by SID (Mode set ID, see *SET_MODE) in Card 5 EQ.-4: NRL-SUM method with CSM (Closely Spaced Modes) treatment. The CSM pairs are automatically identified. EQ.0: SRSS method EQ.1: NRC Grouping method EQ.2: Complete Quadratic Combination method (CQC) EQ.3: Double Sum method based on Rosenblueth-Elorduy coefficient EQ.4: NRL-SUM method EQ.5: Double Sum method based on Gupta-Cordero coefficient EQ.6: Double Sum method based on modified Gupta-Cordero coefficient EQ.7: Rosenblueth method EQ.8: Absolute value method (ABS) EQ.99: Combining results provided by two mode combination methods defined in Card 2.1 with corresponding weights defined in Card 2.2
RELATV	Type of nodal displacement, velocity and acceleration results: EQ.0: Relative values (with respect to the ground) EQ.1: Absolute values

VARIABLE	DESCRIPTION
MPRS	Multi-point or multidirectional response combination method: EQ.0: SRSS EQ.1: 100-40-40 rule (Newmark method) (see Remark 7) EQ.2: 100-40-40 rule (Newmark method) with coefficient 0.4 replaced by R40 in Card 3.

Additional card included when MCOMB = 99 on Card 1.

Card 2.1	1	2	3	4	5	6	7	8
Variable	MCOMB1	MCOMB2						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
MCOMB*i*	i^{th} mode combination method for which results will be combined to the other combination method. It can have any value from the MCOMB description other than 99.

Additional card included when MCOMB = 99 on Card 1.

Card 2.2	1	2	3	4	5	6	7	8
Variable	W1	W2						
Type	F	F						
Default	0.5	0.5						

VARIABLE	DESCRIPTION
W*i*	Weight for the results given by the MCOMB*i* combination

Included when MPRS = 2 on Card 1.

Card 3	1	2	3	4	5	6	7	8
Variable	R40							
Type	F							
Default	0.4							

VARIABLE	DESCRIPTION
R40	Coefficient to replace 0.4 in 100-40-40 rule (see Remark 7)

Included when OPTION2 = MISSING_MASS_CORRECTION.

Card 4	1	2	3	4	5	6	7	8
Variable	ZPA	FILENAME						
Type	F	C						
Default	none	none						

VARIABLE	DESCRIPTION
ZPA	Zero period acceleration
FILENAME	Path and name of a file that contains the static response due to a unit gravity loading

Card 5	1	2	3	4	5	6	7	8
Variable	DAMPF	LCDAMP	LDTYP	DMPMAS	DMPSTF			
Type	F	I	I	F	F			
Default	none	0	0	0.0	0.0			

VARIABLE	DESCRIPTION
DAMPF	Modal damping ratio, ζ. Ignore if LCDAMP is defined.
LCDAMP	Load Curve ID for defining frequency-dependent modal damping ratio ζ
LDTYP	Type of load curve for LCDAMP: EQ.0: Abscissa value defines frequency, EQ.1: Abscissa value defines mode number.
DMPMAS	Mass proportional damping constant α in Rayleigh damping
DMPSTF	Stiffness proportional damping constant β in Rayleigh damping

Standard Spectrum Analysis Card. Include if *OPTION1* is unset. This card can be repeated if 2 or more input spectra exist (multiple-point response spectrum).

Card 6a	1	2	3	4	5	6	7	8
Variable	LCTYP	DOF	LC/TBID	SF	VID	LNID	LNTYP	INFLAG
Type	I	I	I	F	I	I	I	I
Default	none	none	0	1.0	0	none	none	0

VARIABLE	DESCRIPTION
LCTYP	Load curve type for defining the input spectrum: EQ.0: Base velocity (as a function of natural frequency), EQ.1: Base acceleration (as a function of natural frequency), EQ.2: Base displacement (as a function of natural frequency), EQ.3: Nodal force (as a function of natural frequency), EQ.4: Pressure (as a function of natural frequency), EQ.5: Base velocity (as a function of natural period), EQ.6: Base acceleration (as a function of natural period), EQ.7: Base displacement (as a function of natural period), EQ.8: Nodal force (as a function of natural period), EQ.9: Pressure (as a function of natural period),

VARIABLE	DESCRIPTION
	EQ.10: Base velocity time history, EQ.11: Base acceleration time history, EQ.12: Base displacement time history.
DOF	Applicable degrees-of-freedom for excitation input: EQ.1: *x*-translational degree-of-freedom, EQ.2: *y*-translational degree-of-freedom, EQ.3: *z*-translational degree-of-freedom, EQ.4: Translational movement in direction given by VID.
LC/TBID	Load curve or table ID (see *DEFINE_TABLE) defining the response spectrum for frequencies. If the table definition is used, a family of curves are defined for discrete critical damping ratios.
SF	Scale factor for the input load spectrum
VID	Vector ID for DOF values of 4
LNID	Node ID, node set ID, or segment set ID where the excitation is applied. If the input load is given as a base excitation spectrum, LNID = 0.
LNTYP	Set type for LNID: EQ.1: Node, see *NODE, EQ.2: Node set, see *SET_NODE, EQ.3: Segment set, see *SET_SEGMENT, EQ.4: Part, see *PART, EQ.5: Part set, see *SET_PART.
INFLAG	Frequency interpolation option: EQ.0: Logarithmic interpolation, EQ.1: Semi-logarithmic interpolation, EQ.2: Linear interpolation.

DDAM Spectrum Analysis Card. Include this card for the DDAM keyword option.

Card 6b	1	2	3	4	5	6	7	8
Variable	STD	UNIT	AMIN	VID	XC	YC	ZC	EFFMAS
Type	I	I	F	I	F	F	F	F
Default	1	1	6	0	none	none	none	80

VARIABLE	DESCRIPTION
STD	Design spectrum standard for shock load: EQ.1: NRL-1396, EQ.-1: Spectrum constants defined through Cards 4b.1 and 4b.2.
UNIT	Unit system: EQ.1: MKS (kg, m, s, N, Pa) EQ.2: GPA (kg, mm, ms, kN, GPa) EQ.3: MPA (ton, mm, s, N, MPa) EQ.4: BIN (lbf-s^2/in, in, s, lbf, psi) EQ.5 miu_MKS (gm, mm, ms, N, N/mm2) EQ.6: CGS (gm, cm, s, dyne, dyne/cm2)
AMIN	Minimum acceleration (in g - gravity acceleration)
VID	Direction of shock load EQ.1: x-direction EQ.2: y-direction EQ.3: z-direction LT.0: Direction is given by vector |VID|
XC	X-directional cosine of shock load (if VID is undefined)
YC	Y-directional cosine of shock load (if VID is undefined)
ZC	Z-directional cosine of shock load (if VID is undefined)
EFFMAS	Minimum percentage requirement of total modal mass

NRL-1396 Design Spectrum Standard Card. Include when STD > 0 for the DDAM option.

Card 7a	1	2	3	4	5	6	7	8
Variable	SHPTYP	MOUNT	MOVEMT	MATTYP				
Type	I	I	I	I				
Default	1	1	1	1				

VARIABLE	DESCRIPTION
SHPTYP	Ship type: EQ.1: Submarine EQ.2: Surface ship
MOUNT	Mount type: EQ.1: Hull Mounted System EQ.2: Deck Mounted System EQ.3: Shell Plating Mounted System
MOVEMT	Movement type: EQ.1: Vertical EQ.2: Athwartship EQ.3: Fore and Aft
MATTYP	Material type: EQ.1: Elastic EQ.2: Elasto-plastic

Acceleration Spectrum Coefficients Card. Include this card for the DDAM keyword option if STD = -1.

Card 7b.1	1	2	3	4	5	6	7	8
Variable	AF	AA	AB	AC	AD			
Type	F	F	F	F	F			
Default	0.0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
AF, AA-AD	Coefficients to define the acceleration spectrum (see Remark 11)

Velocity Spectrum Coefficients Card. Include this card for the DDAM keyword option if STD = -1.

Card 7b.2	1	2	3	4	5	6	7	8
Variable	VF	VA	VB	VC				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
VF, VA-VC	Coefficients to define the velocity spectrum (see Remark 11)

Mode Set Card. Include this card for the DDAM option with MCOMB = -14.

Card 8	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SID	Mode set ID (see *SET_MODE)

Remarks:

1. **Modal analysis.** Modal analysis must be performed preceding the response spectrum analysis. Thus, the keywords *CONTROL_IMPLICIT_GENERAL and *CONTROL_IMPLICIT_EIGENVALUE are expected in the input file.

2. **Input spectrum.** MDMIN, MDMAX, FNMIN, and FNMAX should be set appropriately to cover all the natural modes inside the input spectrum.

3. **Stress results.** To include stress results, modal stress computation has to be requested in *CONTROL_IMPLICIT_EIGENVALUE (set MSTRES = 1).

4. **Base excitation output.** For base excitation cases, relative values or absolute values for displacement, velocity, and acceleration results can be output.

5. **Restarts.** RESTRT = 1 enables a fast restart run based on d3eigv family files generated in the last run or elsewhere. LS-DYNA reads d3eigv family files to get the natural vibration frequencies and mode shapes. If the d3eigv family files are located in a directory other than the working directory, the directory must be specified in *FREQUENCY_DOMAIN_PATH.

6. **Double Sum method and earthquake duration.** For the Double Sum method (MCOMB = 3), earthquake duration time is given by ENDTIM in the keyword *CONTROL_TERMINATION.

7. **100-40-40 rule.** The 100-40-40 gives a way of evaluating the maximum response for multi-directional / multi-point loading. As described by Nie et al, the 100-40-40 rule (ASCE 4-98) determines the maximum response, R, of a three-dimensional seismic input by taking all the possible permutations of the maxima directional responses:

$$R = \pm\left[R_x \pm 0.4R_y \pm 0.4R_z\right], \text{or } \pm\left[R_y \pm 0.4R_z \pm 0.4R_x\right], \text{or } \pm\left[R_z \pm 0.4R_x \pm 0.4R_y\right]$$

   R_x, R_y, and R_z are found by looking at the response from each of the loadings that make up the multi-directional / multi-point loading. For instance, R_x may be the response due to the acceleration in the x-direction of a node. This rule assumes that when one direction of loading occurs, the responses in the other directions are 40% of their maxima. The field R40 allows the user to adjust this 40% value.

8. **Frequency interpolation.** Three interpolation options are available for frequency interpolation when reading response spectrum values.

a) When INFLAG = 0 (default), logarithmic interpolation is used, that is,

$$\frac{\log y - \log y_1}{\log x - \log x_1} = \frac{\log y_2 - \log y_1}{\log x_2 - \log x_1} .$$

b) When INFLAG = 1, semi-logarithmic interpolation is used, that is,

$$\frac{\log y - \log y_1}{x - x_1} = \frac{\log y_2 - \log y_1}{x_2 - x_1} .$$

c) When INFLAG = 2, linear interpolation is used, that is,

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1} .$$

9. **Interpolation and damping ratios.** Linear interpolation is used for interpolation with respect to damping ratios.

10. **Base Velocity, acceleration, and displacement time history.** If the load curve is defined as base velocity, base acceleration, or base displacement time history (LCTYP = 10, 11, or 12), an ASCII xyplot database spectrum_curve_printID is generated. It shows the intermediate base acceleration spectrum converted from the time history data. ID is the ordering of the input spectra (defined in Card 3a). For single point response spectrum, ID is 1.

11. **User-defined velocity and acceleration spectra for DDAM.** The user-defined acceleration input spectrum is defined as:

$$A = \begin{cases} A_f \times A_a \dfrac{(A_b + W) \times (A_c + W)}{(A_d + W)^2} & \text{if } A_d \neq 0 \\ A_f \times A_a \dfrac{(A_b + W)}{A_c + W} & \text{if } A_d = 0 \end{cases}$$

where W is the modal weight in thousands of pounds (Kips).

The user-defined velocity input spectrum is defined as

$$V = V_f \times V_a \frac{(V_b + W)}{(V_c + W)} .$$

12. **NODFOR_SPCM.** The file nodfor_spcm contains nodal force and group force output for response spectrum analysis. To output this file, include *DATABASE_NODAL_FORCE_GROUP in the input deck.

13. **MISSING_MASS_CORRECTION.** The standard response spectrum method may miss some mass due to the truncation of eigenmodes in modal combination. The MISSING_MASS_CORRECTION option permits inclusion of the missing mass effect in response spectrum analysis. ZPA enables computing the inertia force due to ground acceleration, using the static response from a unit loading specified in FILENAME on Card 4. After that, the difference between the total inertia force and the sum of the modal inertia forces gives the missing inertia

force, and the static response due to the missing inertia force gives the missing mass response. Finally, the following equation gives the total response including the missing mass effect as:

$$R = \sqrt{\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\varepsilon_{ij}R_iR_j\right) + (R_M)^2}$$

where R_i and R_j are the modal responses at modes i and j, respectively, and R_M is the missing mass response.

*FREQUENCY_DOMAIN_SEA_CONNECTION

Purpose: Define connection of subsystems in a SEA model.

Card Summary:

Card 1. This card is required.

CONID	CTYPE	NSUB	IBEAM				

Card 2. This card is required.

SUB1	SUB2	SUB3	SUB4	SUB4	SUB6	SUB7	SUB8

Card 3a.1. This card is required for CTYPE = 1.

ANGLE1	ANGLE2	ANGLE3	ANGLE4	ANGLE5	ANGLE6	ANGLE7	ANGLE8

Card 3a.2. This card is required for CTYPE = 1.

LENGTH							

Card 3b. This card is required for CTYPE = 2.

ABSORB	THICK						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	CONID	CTYPE	NSUB	IBEAM				
Type	I	I	F	F				
Default	none	1	none	0				

VARIABLE	DESCRIPTION
CONID	Connection ID
CTYPE	Type of connection: EQ.1: Plate-plate

VARIABLE	DESCRIPTION
	EQ.2: Plate-cavity EQ.3: Plate-cavity-cavity EQ.4: Plate-beam
NSUB	Number of subsystems in this connection
IBEAM	Flag for plate connected to plate: EQ.0: Plate-plate connection EQ.1: Plate-plate-beam connection

Card 2	1	2	3	4	5	6	7	8
Variable	SUB1	SUB2	SUB3	SUB4	SUB5	SUB6	SUB7	SUB8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
SUB*i*	ID of the i^{th} subsystem

Card 3a.1	1	2	3	4	5	6	7	8
Variable	ANGLE1	ANGLE2	ANGLE3	ANGLE4	ANGLE5	ANGLE6	ANGLE7	ANGLE8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ANGLE*i*	Connection angle of the plate *i*

Card 3a.2	1	2	3	4	5	6	7	8
Variable	LENGTH							
Type	F							
Default	0							

VARIABLE	DESCRIPTION
LENGTH	Length of the edge in connection

Card 3b	1	2	3	4	5	6	7	8
Variable	ABSORB	THICK						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
ABSORB	Absorption coefficient
THICK	Thickness of the plate

Remarks:

1. **Number of subsystems in connection.** At this time, up to 8 subsystems can be included in one connection.

*FREQUENCY_DOMAIN_SEA_INPUT

Purpose: Define input power for a subsystem in SEA model.

Card Summary:

Card 1. This card is required.

SUBID	SUBTYP	LOADTYP					

Card 2. This card is required

BWAVE	LWAVE	SWAVE	TWAVE				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SUBID	SUBTYP	LOADTYP					
Type	I	I	I					
Default	none	1	1					

VARIABLE	DESCRIPTION
SUBID	ID of subsystem
SUBTYP	Type of subsystem: EQ.1: plate EQ.2: cavity EQ.3: beam
LOADTYP	Input power type: EQ.1: power EQ.2: force EQ.3: velocity EQ.4: bending wave power for plate EQ.5: shear wave power for plate.

Card 2	1	2	3	4	5	6	7	8
Variable	BWAVE	LWAVE	SWAVE	TWAVE				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
BWAVE	Bending wave
LWAVE	Longitudinal wave
SWAVE	Shear wave
TWAVE	Torsional wave (required only for beams, SUBTYP = 3)

Remarks:

1. **Input power in torsional wave.** TWAVE (input power in torsional wave) is not required for plate and cavity subsystems (SUBTYP = 1 or 2).

*FREQUENCY_DOMAIN_SEA_SUBSYSTEM

Purpose: Define input power for a subsystem in an SEA model.

Card Summary:

Card 1. This card is required.

FMIN	FMAX	NFREQ	NFSPACE	LCFREQ	IREAD	PREF	

Card 2. This card is required.

SUBID	SUBTYP	DENSITY	E	PR	OUTPUT		

Card 3a.1. This card is required for plate (SUBTYP = 1).

A	PERIM	THICK	WIDTH	LENGTH			

Card 3a.2. This card is required for plate (SUBTYP = 1).

DAMPB	DAMPL	DAMPS	LC1	LC2	LC3		

Card 3b.1. This card is required for cavity (SUBTYP = 2).

A	PERIM	VOLUME	WIDTH	LENGTH	HEIGHT		

Card 3b.2. This card is required for cavity (SUBTYP = 2).

DAMPB	LC1						

Card 3c.1. This card is required for beam (SUBTYP = 3).

A	ISS	ITT	J	LENGTH			

Card 3c.2. This card is required for beam (SUBTYP = 3).

DAMPB	DAMPL	DAMPS	DAMPT	LC1	LC2	LC3	LC4

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FMIN	FMAX	NFREQ	NFSPACE	LCFREQ	IREAD	PREF	
Type	I	I	I	F	I	I	F	
Default	none	1	1	0	0	0	2.0e-5	

VARIABLE	DESCRIPTION
FMIN	Minimum frequency for SEA output (cycles/time)
FMAX	Maximum frequency for SEA output (cycles/time)
NFREQ	Number of frequencies for SEA output (cycles/time)
FSPACE	Frequency spacing option for SEA output: EQ.0: Linear EQ.1: Logarithmic EQ.2: Biased
LCFREQ	Load curve ID defining the frequencies for SEA output
IREAD	Type of SEA run: EQ.0: Run the SEA analysis. EQ.1: Read the FEM keyword input deck and create the SEA model.
PREF	Reference pressure used to convert acoustic pressure to SPL (dB)

Card 2	1	2	3	4	5	6	7	8
Variable	SUBID	SUBTYP	DENSITY	E	PR	OUTPUT		
Type	I	I	F	F	F	I		
Default	none	none	none	none	none	0		

VARIABLE	DESCRIPTION
SUBID	ID of subsystem
SUBTYP	Type of subsystem: EQ.1: Plate EQ.2: Cavity EQ.3: Beam
DENSITY	Mass density of subsystem
E	Young's modulus of subsystem
PR	Poisson's ratio of subsystem
OUTPUT	Include this subsystem in output: EQ.0: No EQ.1: Yes

Card 3a.1	1	2	3	4	5	6	7	8
Variable	A	PERIM	THICK	WIDTH	LENGTH			
Type	F	F	F	F	F			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
A	Plate area

VARIABLE	DESCRIPTION
PERIM	Plate perimeter
THICK	Plate thickness
WIDTH	Plate width
LENGTH	Plate length

Card 3a.2	1	2	3	4	5	6	7	8
Variable	DAMPB	DAMPL	DAMPS	LC1	LC2	LC3		
Type	F	F	F	I	I	I		
Default	none	none	none	0	0	0		

VARIABLE	DESCRIPTION
DAMPB	Damping factor for bending wave
DAMPL	Damping factor for longitudinal wave
DAMPS	Damping factor for shear wave
LC1	Load curve for damping factor for bending wave
LC2	Load curve for damping factor for longitudinal wave
LC3	Load curve for damping factor for shear wave

Card 3b.1	1	2	3	4	5	6	7	8
Variable	A	PERIM	VOLUME	WIDTH	LENGTH	HEIGHT		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
A	Cavity area
PERIM	Cavity perimeter
VOLUME	Cavity volume
WIDTH	Cavity width
LENGTH	Cavity length
HEIGHT	Cavity height

Card 3b.2	1	2	3	4	5	6	7	8
Variable	DAMPB	LC1						
Type	F	I						
Default	none	0						

VARIABLE	DESCRIPTION
DAMPB	Damping factor for bending wave
LC1	Load curve for damping factor for bending wave

Card 3c.1	1	2	3	4	5	6	7	8
Variable	A	ISS	ITT	J	LENGTH			
Type	F	F	F	F	F			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
AREA	Beam area
ISS	I_{ss}, area moment of inertia about local s-axis

VARIABLE	DESCRIPTION
ITT	I_{tt}, area moment of inertia about local t-axis
J	J, torsional constant
LENGTH	Beam length

Card 3c.2	1	2	3	4	5	6	7	8
Variable	DAMPB	DAMPL	DAMPS	DAMPT	LC1	LC2	LC3	LC4
Type	F	F	F	F	I	I	I	I
Default	none	none	none	none	0	0	0	0

VARIABLE	DESCRIPTION
DAMPB	Damping factor for bending wave
DAMPL	Damping factor for longitudinal wave
DAMPS	Damping factor for shear wave
DAMPT	Damping factor for torsional wave
LC1	Load curve for damping factor for bending wave
LC2	Load curve for damping factor for longitudinal wave
LC3	Load curve for damping factor for shear wave
LC4	Load curve for damping factor for torsional wave

Remarks:

1. **Damping factor load curves.** When LC1, LC2, etc., are defined, the damping factors for the corresponding waves are frequency-dependent, and the values defined in these curves override the constant damping factor values defined by DAMPB, DAMPL, etc.

*FREQUENCY_DOMAIN_SSD_{OPTION}

Available options include:

DIRECT

DIRECT_FREQUENCY_DEPENDENT

FATIGUE

FRF

ERP

SUBCASE

Purpose: Compute the steady-state dynamic response due to a given spectrum of harmonic excitations.

The FATIGUE option invokes additionally calculating the cumulative fatigue damage ratio. Use the FRF option to compute the steady-state dynamic response due to a unit load, which is equivalent to the Frequency Response Function (FRF). The ERP option specifies additionally calculating the Equivalent Radiated Power (ERP) due to vibration.

The DIRECT and DIRECT_FREQUENCY_DEPENDENT keyword options specify performing the steady-state dynamic analysis using a direct method, instead of the mode-based method. This method is only available for SMP. See Remarks 1 and 14.

The SUBCASE option enables including multiple loading cases in one run (see Remark 16).

Card Summary:

Card 1. This card is required.

MDMIN	MDMAX	FNMIN	FNMAX	RESTMD	RESTDP	LCFLAG	RELATV

Card 2. This card is required.

DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF	DMPFLG		

Card 3. This card is required.

	ISTRESS	MEMORY	NERP	STRTYP	NOUT	NOTYP	NOVA

Card 4. Include this card if the ERP keyword option is used.

RO	C	ERPRLF	ERPREF	RADEFF			

Card 5. Include this card if the ERP keyword option is used and NERP > 0.

PID	PTYP						

Card 6. If the SUBCASE keyword option is used, include this card. For each case, a set of this keyword and NLOAD copies of Card 7 are included. See Remark 16.

CASEID	TITLE	NLOAD

Card 7. When the SUBCASE option is used, for each case include NLOAD of this card following the Card 6 for that case (see Remark 16). Otherwise, include as many definitions of this card as needed.

NID	NTYP	DOF	VAD	LC1	LC2	SF	VID

Card 7.1. Include this card if VAD = 15 on Card 7.

MASS	E						

Card 8. Include this card if the FATIGUE keyword option is used.

LCFTG							

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MDMIN	MDMAX	FNMIN	FNMAX	RESTMD	RESTDP	LCFLAG	RELATV
Type	I	I	F	F	I	I	I	I
Default	1	optional	0.0	optional	0	0	0	0

VARIABLE	DESCRIPTION
MDMIN	The first mode in modal superposition method (optional).
MDMAX	The last mode in modal superposition method (optional). See Remark 3. MDMAX should not be 0 for a restart run based on existing modal analysis results (RESTMD = 1 or 2); see Remark 4.
FNMIN	The minimum natural frequency in modal superposition method (optional).

VARIABLE	DESCRIPTION
FNMAX	The maximum natural frequency in modal superposition method (optional). See Remark 3.
RESTMD	Restart option: EQ.0: A new modal analysis is performed, EQ.1: Restart with d3eigv, EQ.2: Restart with modeshp binary scratch file.
RESTDP	Restart option: EQ.-1:A new run with writing dumpssd for future restart, EQ.0: A new run without writing dumpssd, EQ.1: Restart with dumpssd with writing new dumpssd for future restart, EQ.2: Restart with dumpssd without writing new dumpssd.
LCFLAG	Load curve definition flag: EQ.0: Load curves are given as amplitude / phase angle (defined in degrees in the range (-180°, 180°]). EQ.1: Load curves are given as real / imaginary components.
RELATV	Flag for displacement, velocity and acceleration results: EQ.0: Absolute values are requested. EQ.1: Relative values are requested (for VAD = 2, 3 and 4 only; see Card 7).

Damping Card. See Remark 5.

Card 2	1	2	3	4	5	6	7	8
Variable	DAMPF	LCDAM	LCTYP	DMPMAS	DMPSTF	DMPFLG		
Type	F	I	I	F	F	I		
Default	0.0	0	0	0.0	0.0	0		

VARIABLE	DESCRIPTION
DAMPF	Modal damping coefficient, ζ
LCDAM	Load Curve ID defining mode dependent modal damping coefficient ζ.
LCTYP	Type of load curve defining modal damping coefficient: EQ.0: Abscissa value defines frequency. EQ.1: Abscissa value defines mode number.
DMPMAS	Mass proportional damping constant α, in Rayleigh damping.
DMPSTF	Stiffness proportional damping constant β, in Rayleigh damping
DMPFLG	Damping flag: EQ.0: Use modal damping coefficient ζ defined by DAMPF or LCDAM or Rayleigh damping defined by DMPMAS and DMPSTF in this card. EQ.1: Use damping defined by *DAMPING_PART_MASS, *DAMPING_PART_STIFFNESS, or a material model like *MAT_LINEAR_ELASTIC_DISCRETE_BEAM.

Card 3	1	2	3	4	5	6	7	8
Variable		ISTRESS	MEMORY	NERP	STRTYP	NOUT	NOTYP	NOVA
Type		I	I	I	I	I	I	I
Default		0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
ISTRESS	Stress computation flag (for keyword option DIRECT only): EQ.0: Stress results are not requested. EQ.1: Stress results are requested.
MEMORY	Memory flag: EQ.0: Modal superposition will be performed in-core. This option runs faster.

VARIABLE	DESCRIPTION
	EQ.1: Modal superposition will be performed out-of-core. This is needed for some large scale problems that cannot fit in main memory. This method incurs a performance penalty associated with disk speed.
NERP	Number of ERP panels.
STRTYP	Stress used in fatigue analysis: EQ.0: Von Mises stress, EQ.1: Maximum principal stress, EQ.2: Maximum shear stress.
NOUT	Part, part set, segment set, or node set ID for response output (use with acoustic computation). See NOTYP below. See Remark 6.
NOTYP	Type of NOUT: EQ.0: Part set ID (not implemented) EQ.1: Part ID (not implemented) EQ.2: Segment set ID EQ.3: Node set ID EQ.-2: Segment set ID which mismatches with acoustic boundary nodes. Mapping of velocity or acceleration to the acoustic boundary nodes is performed.
NOVA	Response output type: EQ.0: Velocity EQ.1: Acceleration

ERP Card. This card is read only when the ERP keyword option is used.

Card 4	1	2	3	4	5	6	7	8
Variable	R0	C	ERPRLF	ERPREF	RADEFF			
Type	F	F	F	F	I			
Default	none	none	1.0	0.0	0			

VARIABLE	DESCRIPTION
RO	Fluid density
C	Sound speed of the fluid
ERPRLF	ERP radiation loss factor. LT.0: Curve ID = (-ERPRLF) specifies frequency-dependent radiation loss factor.
ERPREF	ERP reference value. This is used to convert the absolute ERP value to ERP in decibels (dB).
RADEFF	Radiation efficiency computation flag: EQ.0: Radiation efficiency computation is not requested, and a classic ERP computation is performed. EQ.1: Radiation efficiency computation is requested, and an enhanced ERP computation using Rayleigh integrals is performed. EQ.2: Radiation efficiency computation is not requested, and a corrected ERP computation is performed.

ERP Part Cards. This card is read NERP times if the ERP keyword option is used.

Card 5	1	2	3	4	5	6	7	8
Variable	PID	PTYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
PID	Part, part set, or segment set ID for ERP computation. See PTYP below. Note that the ERP computation works only on a part or part set of shell elements or a set of segments (2D surfaces).
PTYP	Type of PID: EQ.0: Part ID

VARIABLE	DESCRIPTION
	EQ.1: Part set ID
	EQ.2: Segment set ID

SUBCASE Cards. This card is read if the SUBCASE keyword option is used. A set of this card with NLOAD instantiations of Card 7 specifies a case. Include as many cases as needed.

Card 6	1	2	3	4	5	6	7	8
Variable	CASEID	TITLE						NLOAD
Type	C	C						I
Default	none	none						1

VARIABLE	DESCRIPTION
CASEID	Identification string to be used as the case ID (must include at least one letter).
TITLE	A description of the current loading case (can be blank)
NLOAD	Number of loads for this loading case

Excitation Loads. When the SUBCASE option is used, for each case include NLOAD of this card following the Card 6 for that case (see Remark 16). Otherwise repeat this card as many times as needed if multiple excitation loads are present.

Card 7	1	2	3	4	5	6	7	8
Variable	NID	NTYP	DOF	VAD	LC1	LC2	SF	VID
Type	I	I	I	I	I	I	F	I
Default	none	0	none	none	none	none	1.0	0

VARIABLE	DESCRIPTION
NID	ID of the entity to which the excitation input is applied. See NTYP below for the types. See Remark 7.

VARIABLE	DESCRIPTION
NTYP	Type of NID: EQ.0: Node ID EQ.1: Node set ID EQ.2: Segment set ID EQ.3: IGA parametric point ID (see *IGA_POINT_UVW) EQ.4: IGA parametric point set ID (see *SET_IGA_POINT_-UVW) EQ.7: IGA physical face ID (see *IGA_FACE_XYZ) EQ.8: IGA physical face set ID (see *SET_IGA_FACE_XYZ) EQ.9: IGA parametric face ID (see *IGA_FACE_UVW). This option is only supported for VAD = 2. EQ.10: IGA parametric face set ID (see *SET_IGA_FACE_-UVW). This option is only supported for VAD = 2.
DOF	Applicable degrees of freedom for excitation input (ignored if VAD = 1): EQ.1: x-translational degree of freedom or x-rotational degree of freedom (for torque excitation, VAD = 8) EQ.2: y-translational degree of freedom, or y-rotational degree of freedom (for torque excitation, VAD = 8) EQ.3: z-translational degree of freedom, or z-rotational degree of freedom (for torque excitation, VAD = 8) EQ.4: Translational movement in direction given by vector VID, or rotational movement with axis given by vector VID (for torque excitation, VAD = 8)
VAD	Excitation input type: EQ.0: Nodal force EQ.1: Pressure EQ.2: Base velocity EQ.3: Base acceleration EQ.4: Base displacement EQ.5: Enforced velocity by large mass method (see Remark 8)

VARIABLE	DESCRIPTION
	EQ.6: Enforced acceleration by large mass method (see Remark 8)
	EQ.7: Enforced displacement by large mass method (see Remark 8)
	EQ.8: Torque
	EQ.9: Base angular velocity
	EQ.10: Base angular acceleration
	EQ.11: Base angular displacement
	EQ.12: Enforced velocity (for DIRECT type keyword options only)
	EQ.13: Enforced acceleration (for DIRECT type keyword options only)
	EQ.14: Enforced displacement (for DIRECT type keyword options only)
	EQ.15: Rotating structure with mass imbalance (Card 7.1 is required)
LC1	Load Curve ID defining amplitude (LCFLAG = 0) or real (in-phase) part (LCFLAG = 1) of load as a function of frequency. It is not used if the keyword option FRF is active.
LC2	Load Curve ID defining phase angle (LCFLAG = 0) or imaginary (out-phase) part (LCFLAG = 1) of load as a function of frequency. It is not used if the keyword option FRF is active.
SF	Scale factor for the load. This scale factor applies to the amplitude of load (LCFLAG = 0) or real and imaginary parts of load (LCFLAG = 1).
VID	Vector ID for DOF = 4 for excitation input; see *DEFINE_VECTOR.

Mass Imbalance Card. Include this card when VAD = 15 on Card 7.

Card 7.1	1	2	3	4	5	6	7	8
Variable	MASS	E						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
MASS	Rotating mass
E	Eccentricity (distance between the rotating mass and the center of rotation)

Load Duration Card. This card is read only when the FATIGUE keyword option is used.

Card 8	1	2	3	4	5	6	7	8
Variable	LCFTG							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
LCFTG	Load curve ID defining the duration of excitation for each frequency

Remarks:

1. **Direct methods.** When the option DIRECT or DIRECT_FREQUENCY_DEPENDENT is not used, this feature computes the steady-state dynamic response due to a harmonic excitation spectrum by the modal superposition method. When the keyword option DIRECT or DIRECT_FREQUENCY_DEPENDENT is used, a direct method is used. For the direct method, the unknown primary variables are the nodal displacements. A system of dynamic equations is formed

in the frequency domain and solved directly without projecting the displacement variables into the modal space.

2. **Modal superposition method and required keywords.** The modal superposition method requires natural frequencies and mode shapes. Thus, the keyword *CONTROL_IMPLICIT_EIGENVALUE *must* be included in the input.

3. **Modes.** MDMIN/MDMAX and FNMIN/FNMAX together determine which modes are used in the modal superposition method. The first mode must have a mode number ≥ MDMIN and frequency ≥ FNMIN; the last mode must have a mode number ≤ MDMAX and frequency ≤ FNMAX. When MDMAX or FNMAX is not given, the last mode in modal superposition method is the last mode available in FILENM.

4. **Restarts.** Restart option RESTMD = 1 is used if mode analysis has been done previously. In this case, LS-DYNA skips modal analysis and reads in d3eigv family files generated previously. Restart option RESTMD = 2 is used if a SSD analysis has been done for the same model previously and the binary scratch file modeshp exists. The modeshp file saves the condensed modal shape (and modal stress if there is any) data. Therefore, the SSD analysis can be restarted with the existing modeshap file if no changes have been made to the modal structure, boundary conditions, and material models. Restarting SSD with RESTMD = 2 is faster than restarting SSD with RESTMD = 1 because of the saved modal shapes. For RESTMD = 1 or 2, always use MDMIN = 1 and MDMAX = number of modes given by modal analysis (can be found from ASCII file eigout or from d3eigv files using LS-PREPOST).

   To add the contribution of additional modes to previous SSD results, restart option RESTDP = 1 is used. In this case, LS-DYNA reads in the binary dump file dumpssd which contains previous SSD results and adds the contribution from new modes. For RESTDP = 1, the new modal analysis (RESTMD = 0) or the d3eigv family files created elsewhere (RESTMD = 1) should exclude the modes used in previous SSD computation. To do this, set LFLAG (and RFLAG, if necessary) and a nonzero LFTEND (and RHTEND) in *CONTROL_IMPLICIT_EIGENVALUE. The RESTDP option can also be used if the frequency range for modal analysis is divided into segments and modal analysis is performed for each frequency range separately.

   To add some acoustic field nodes and run BEM/FEM acoustic computation after SSD, the RESTMD and RESTDP options still work even if the number of nodes may be changed after a previous modal analysis, provided that the IDs of the old nodes are not changed.

5. **Damping.** Damping can be prescribed in several ways:

a) To use a constant modal damping coefficient ζ for all the modes, define DAMPF only. LCDMP, LCTYP, DMPMAS, and DMPSTF are ignored.

b) To use mode-dependent modal damping, define a load curve (*DEFINE_CURVE) and specify if the abscissa value defines the frequency or mode number with LCTYP. DMPMAS and DMPSTF are ignored.

c) To use Rayleigh damping, define DMPMAS (α) and DMPSTF (β). Keep DAMPF as its default (0.0) and keep LCDMP and LCTYP as 0. The damping matrix in Rayleigh damping is defined as $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$, where, **C**, **M** and **K** are the damping, mass, and stiffness matrices, respectively.

d) DAMFLG = 1 enables using nonclassical damping (meaning damping defined by *DAMPING_PART_MASS, *DAMPING_PART_STIFFNESS, or a material model like *MAT_LINEAR_ELASTIC_DISCRETE_BEAM) in SSD analysis. Due to this nonclassical damping, the damping matrix can no longer be diagonalized by multiplying it with the eigenvector matrix from the left side and its transpose from the right side. In other words, the dynamics equation system cannot be decoupled by projecting it to the modal space (something that classical damping can do). Thus, this coupled system must be solved directly.

6. **Nodal output and acoustic analysis.** NOUT and NOTYP are used to define the nodes where velocity or acceleration are requested to be written to a binary file bin_ssd or other filename defined by "bem=filename" (see keyword *FREQUENCY_DOMAIN_ACOUSTIC_BEM) on the command line. The velocity or acceleration data in this file can be used by the BEM or FEM acoustic solver to perform a vibro-acoustic analysis. If structure nodes and acoustic boundary nodes are mismatched, the option NOTYP = -2 can be used. The velocity or acceleration data given at a structure segment set NOUT is mapped to acoustic boundary nodes.

7. **Base velocity, acceleration, and displacement.** For base velocity, base acceleration, or base displacement (VAD = 2, 3, or 4) excitations, the parameters NID and NTYP are not used and can be blank. The loads are applied through the nodes constrained to the ground or to the shaker table (defined by the keyword *BOUNDARY_SPC or the variables TC or RC in *NODE). The base velocity, base acceleration, and base displacement cases are treated by applying an inertia force to the structure. In each direction, only one base excitation (velocity, acceleration, or displacement) is allowed. Duplicate definitions of base excitation in one direction cause an error termination. These comments apply to base angular cases (VAD = 9, 10, or 11) too.

8. **Large mass method.** For the cases with enforced motion excitation such as nodal velocity, acceleration, or displacement, the large mass method can be used to compute the SSD results. The excitation input can be given as enforced

motion curves (VAD = 5, 6, 7). To use the large mass method, a large mass must be attached to the nodes where the enforced motion is applied by using the keyword *ELEMENT_MASS_*{OPTION}*, and the large mass per node (MPN) must be reported in the keyword *CONTROL_FREQUENCY_DOMAIN. For more details, please refer to *CONTROL_FREQUENCY_DOMAIN.

9. **Displacement, velocity, and acceleration results.** Displacement, velocity, and acceleration results are output into ASCII file NODOUT_SSD. The nodes to be output to NODOUT_SSD are specified by card *DATABASE_HISTORY_NODE. The frequencies for the nodal results in NODOUT_SSD are specified by card *DATABASE_FREQUENCY_ASCII_NODOUT_SSD. If the card *DATABASE_FREQUENCY_ASCII_NODOUT_SSD is not defined, the frequencies for the nodal results in NODOUT_SSD are the same as those in D3SSD binary database, which is defined by card *DATABASE_FREQUENCY_BINARY_D3SSD.

10. **Stress and strain results.** Stress (and strain) results are output into ASCII file ELOUT_SSD. The solid, beam, shell, and thick shell elements to be output to ELOUT_SSD are specified by the following cards:

    - *DATABASE_HISTORY_SOLID_*{OPTION}*
    - *DATABASE_HISTORY_BEAM_*{OPTION}*
    - *DATABASE_HISTORY_SHELL_*{OPTION}*
    - *DATABASE_HISTORY_TSHELL_*{OPTION}*

    The frequencies for the element stress / strain results in ELOUT_SSD are specified by card *DATABASE_FREQUENCY_ASCII_ELOUT_SSD. If the card *DATABASE_FREQUENCY_ASCII_ELOUT_SSD is not defined, the frequencies for the element stress / strain results in ELOUT_SSD are same as those in D3SSD binary database, which is defined by card *DATABASE_FREQUENCY_BINARY_D3SSD.

11. **Phase angle units.** The phase angle is given in degrees in the range (-180°, 180°], for both the input decks and output databases (e.g. D3SSD, NODOUT_SSD, and ELOUT_SSD).

12. **Fatigue.** When the FATIGUE keyword option is used, the cumulative fatigue damage ratio due to the harmonic vibration is computed and saved in binary plot database d3ftg. The *MATERIAL_ADD_FATIGUE keyword is needed to define the S-N fatigue curve for each material.

13. **FRF.** The FRF keyword option invokes the calculation of the steady-state dynamic response due to a unit load spectrum. Numerically, it is equivalent to the Frequency Response Function (FRF). In this case, LC1 and LC2 are not used and can be blank. LS-DYNA generates a unit load spectrum according to the excitation location, the excitation degree of freedom, and the excitation type defined by NID, NTYP, DOF, and VAD in Card 7. This is useful if the user is interested

in obtaining FRF fringe plots at some frequencies or would like to get the stress (and strain) FRF results. The name of the binary database is changed to D3FRF.

14. **Frequency-dependent material properties.** For the option DIRECT_FREQUENCY_DEPENDENT, the damping and stiffness matrices can change with frequency, as can the material properties. The keyword *MAT_ADD_PROPERTY_DEPENDENCE can be used to define the properties of a material model that change with frequency.

15. **Radiation efficiency.** For the keyword option ERP and RADEFF = 1, radiated acoustic power can be computed using the Rayleigh integral. Radiation efficiency is computed as the ratio between the radiated acoustic power and the absolute ERP values. The radiated acoustic power results are saved in an ASCII xyplot database radiated_power. The radiation efficiency results are saved in an ASCII xyplot database radiation_efficiency. The computation is suitable for a baffled plate vibrating in out-of-plane motion and placed in light fluid, such as air.

16. **SUBCASE.** With the keyword option SUBCASE, multiple loading cases can be included with one keyword. Each loading case includes one set of Card 6 and Card 7. Card 7 can be repeated for each case if the current loading case has multiple excitation loads. CASEID defined in Card 6 is used as a prefix for the output databases (such as D3SSD and binout). An automatic case ID is assigned (meaning CASE1.D3SSD, CASE2.D3SSD, …) if CASEID is defined as "case" or "CASE". The TITLE field, when defined, is used as the title for the output databases (D3SSD, NODOUT_SSD, ELOUT_SSD, and NODFOR_SSD). Otherwise, the output databases use the global title, defined by the keyword *TITLE. The multiple loading cases may share the same output frequencies (defined by *DATABASE_FREQUENCY_BINARY_D3SSD, *DATABASE_FREQUENCY_ASCII_NODOUT_SSD, *DATABASE_FREQUENCY_ASCII_ELOUT_SSD, and *DATABASE_FREQUENCY_ASCII_NODFOR_SSD). They can also have different output frequencies by appending the option SUBCASE to the corresponding database keywords. With this option, you can set FMIN, FMAX, NFREQ, FSPACE, and LCFREQ for each loading case sequentially.

*HASH

The *HASH keywords are for generating a checksum of a portion of the input.

The available keyword cards in alphabetical order are:

*HASH

*HASH_END

*HASH

Purpose: Generate a checksum of a portion of the input. This checksum provides an easy confirmation that this block of input has not been modified.

This keyword denotes the beginning of a block of input, which is terminated by the next *HASH_END card. All the text between the two, including the data card for this keyword, is hashed using the SHA256 algorithm and included in a table at the bottom of d3hsp. However, these keywords only have an effect if the command line option hash is used (see Execution Syntax in Getting Started).

Card 1	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	A							
Default	none							

VARIABLE	DESCRIPTION
TITLE	A descriptive string that identifies the block of input to be hashed

*HASH_END

Purpose: Denote the end of a block that begins with *HASH. See *HASH.

*HOURGLASS

Purpose: Define hourglass and bulk viscosity properties which are referenced using HGID in the *PART command. Properties specified here, when invoked for a particular part, override those in *CONTROL_HOURGLASS and *CONTROL_BULK_VISCOSITY.

An additional option **TITLE** may be appended to *HOURGLASS keywords. If this option is used, then an additional line is read for each section in 80a format which can be used to describe the section. At present, the title serves no purpose other than to perhaps lend clarity to input decks.

Card 1	1	2	3	4	5	6	7	8
Variable	HGID	IHQ	QM	IBQ	Q1	Q2	QB, VDC	QW
Type	I/A	I	F	I	F	F	F	F
Default	0	Rem 9	.10	not used	1.5	0.06	QM, 0.	QM

VARIABLE	DESCRIPTION
HGID	Hourglass ID. A unique number or label must be specified. This ID is referenced by HGID in the *PART command.
IHQ	Hourglass control type. For solid elements six options are available. For quadrilateral shell and membrane elements the hourglass control is based on the formulation of Belytschko and Tsay, that is, options 1-3 are identical, and options 4-5 are identical. See Remark 1. EQ.0: See Remark 9. EQ.1: Standard LS-DYNA viscous form EQ.2: Flanagan-Belytschko viscous form EQ.3: Flanagan-Belytschko viscous form with exact volume integration for solid elements EQ.4: Flanagan-Belytschko stiffness form EQ.5: Flanagan-Belytschko stiffness form with exact volume integration for solid elements.

VARIABLE	DESCRIPTION
	EQ.6: Belytschko-Bindeman [1993] assumed strain co-rotational stiffness form for 2D and 3D solid elements only. See Remark 4.
	EQ.7: Linear total strain form of type 6 hourglass control. (See Remark 6 below).
	EQ.8: Activates the full projection warping stiffness for shell formulations 9, 16 and -16. A speed penalty of 25% is common for this option.
	EQ.9: Puso [2000] enhanced assumed strain stiffness form for 3D hexahedral elements. See Remark 8.
	EQ.10: Cosserat Point Element (CPE) developed by Jabareen and Rubin [2008] and Jabareen et.al. [2013]; see *CONTROL_HOURGLASS.
	A discussion of the viscous and stiffness hourglass control for shell elements follows at the end of this section.
QM	Hourglass coefficient. See Remark 7. Values of QM that exceed 0.15 may cause instabilities for brick elements used with forms IHQ = 0 to 5 and all the IHQ forms applicable to shell elements. The stiffness forms can stiffen the response especially if deformations are large and therefore should be used with care. For the shell and membrane elements QM is taken as the membrane hourglass coefficient, the bending as QB, and warping as QW. These coefficients can be specified independently, but generally, QM = QB = QW is adequate. For type 6 solid element hourglass control, see Remark 4 below. For hourglass type 9, see Remark 8.
IBQ	Not used. Bulk viscosity is always on for solids. Bulk viscosity for beams and shells can only be turned on using the variable TYPE in *CONTROL_BULK_VISCOSITY; however, the coefficients can be set using Q1 and Q2 below.
Q1	Quadratic bulk viscosity coefficient. See Remark 3.
Q2	Linear bulk viscosity coefficient. See Remark 3.
QB	Hourglass coefficient for shell bending. By default, QB = QM. See Remark 5.
VDC	Viscous damping coefficient to help suppress hourglass modes. VDC only applies to type 1 solids with type 6 or 7 hourglass control and to tshell formulations 5 and 6. VDC is a unitless coefficient in

VARIABLE	DESCRIPTION
	which VDC = 1.0 corresponds to critical damping.
QW	Hourglass coefficient for shell warping. By default, QB = QW. See Remark 5.

Remarks:

1. **Viscous versus stiffness control.** Viscous hourglass control is recommended for problems deforming with high velocities. Stiffness control is often preferable for lower velocities, especially if the number of time steps is large. For solid elements the exact volume integration provides some advantage for highly distorted elements.

2. **Automotive crash hourglass control.** For automotive crash, the stiffness form of the hourglass control with a coefficient (QM) of 0.05 is preferred by many users.

3. **Bulk viscosity.** Bulk viscosity is necessary to propagate shock waves in solid materials. Generally, the default values are okay except in problems where pressures are very high, larger values may be desirable. In low-density foams, it may be necessary to reduce the viscosity values since the viscous stress can be significant. It is not advisable to reduce it by more than an order of magnitude.

4. **Belytschko-Bindeman hourglass control.** Type 6 hourglass control is for 2D and 3D solid elements only. Based on elastic constants and an assumed strain field, it produces accurate coarse mesh bending results for elastic material when QM = 1.0. For plasticity models with a yield stress tangent modulus that is much smaller than the elastic modulus, a smaller value of QM (0.001 to 0.1) may produce better results. For foam or rubber models, larger values (0.5 to 1.0) may work better. For any material, keep in mind that the stiffness is based on the elastic constants, so if the material softens, a QM value smaller than 1.0 may work better. For anisotropic materials, an average of the elastic constants is used. For fluids modeled with null material, type 6 hourglass control is viscous and is scaled to the viscosity coefficient of the material (see *MAT_NULL).

5. **One-point quadrature and hourglass deformation modes.** In part, the computational efficiency of the Belytschko-Lin-Tsay and the under-integrated Hughes-Liu shell elements are derived from their use of one-point quadrature in the plane of the element. To suppress the hourglass deformation modes that accompany one-point quadrature, hourglass viscous or stiffness-based stresses are added to the physical stresses at the local element level. The discussion of the hourglass control that follows pertains to all one-point quadrilateral shell and membrane elements in LS-DYNA.

The hourglass shape vector τ_I is defined as

$$\tau_I = h_I - \left(h_J \hat{x}_{\alpha J}\right) B_{\alpha I} \ ,$$

where $\hat{x}_{\alpha J}$ are the element coordinates in the local system at the I^{th} element node and $B_{\alpha I}$ is the strain displacement matrix. The hourglass basis vector is:

$$h = \begin{bmatrix} +1 \\ -1 \\ +1 \\ -1 \end{bmatrix} \ ,$$

the basis vector that generates the deformation mode that is neglected by one-point quadrature. In the above equations and the remainder of this subsection, the Greek subscripts have a range of 2, that is, $\hat{x}_{\alpha I} = (\hat{x}_{1I}\ , \hat{x}_{2I}) = \left(\hat{x}_I\ , \hat{y}_I\right)$.

The hourglass shape vector then operates on the generalized displacements to produce the generalized hourglass strain rates

$$\begin{aligned} \dot{q}_\alpha^M &= \tau_I \hat{v}_{\alpha I} \\ \dot{q}_\alpha^B &= \tau_I \hat{\theta}_{\alpha I} \\ \dot{q}_3^W &= \tau_I \hat{v}_{zI} \end{aligned}$$

where the superscripts M, B, and W denote membrane, bending, and warping modes, respectively. The corresponding hourglass stress rates are then given by

$$\begin{aligned} \dot{Q}_\alpha^M &= \frac{\text{QM} \times EtA}{8} B_{\beta I} B_{\beta I} \dot{q}_\alpha^M \\ \dot{Q}_\alpha^B &= \frac{\text{QB} \times Et^3 A}{192} B_{\beta I} B_{\beta I} \dot{q}_\alpha^B \\ \dot{Q}_3^W &= \frac{\text{QW} \times \kappa G t^3 A}{12} B_{\beta I} B_{\beta I} \dot{q}_3^B \end{aligned}$$

where t is the shell thickness. The hourglass coefficients: QM, QB, and QW are generally assigned values between 0.05 and 0.10.

Finally, the hourglass stresses which are updated using the time step, Δt, from the stress rates in the usual way, that is,

$$\mathbf{Q}^{n+1} = \mathbf{Q}^n + \Delta t \dot{\mathbf{Q}} \ ,$$

and the hourglass resultant forces are then

$$\begin{aligned} \hat{f}_{\alpha I}^H &= \tau_I Q_\alpha^M \\ \hat{m}_{\alpha I}^H &= \tau_I Q_\alpha^B \\ \hat{f}_{3I}^H &= \tau_I Q_3^W \end{aligned}$$

where the superscript H emphasizes that these are internal force contributions from the hourglass deformations.

6. **Linear total strain formulation.** IHQ=7 is a linear total strain formulation of the Belytschko-Bindeman [1993] stiffness form for 2D and 3D solid elements.

This linear form was developed for visco-elastic material and guarantees that an element will spring back to its initial shape regardless of the severity of deformation.

7. **QM default value.** The default value for QM is 0.1 unless superseded by a nonzero value of QH in *CONTROL_HOURGLASS. A nonzero value of QM supersedes QH.

8. **Puso hourglass control.** Hourglass type 9 is available for hexahedral elements and is based on physical stabilization using an enhanced assumed strain method. In performance it is similar to the Belytschko-Bindeman hourglass formulation (type 6) but gives more accurate results for distorted meshes, such as for skewed elements. If QM = 1.0, it produces accurate coarse bending results for elastic materials. The hourglass stiffness is by default based on elastic properties, hence the QM parameter should be reduced to about 0.1 for plastic materials in order not to stiffen the structure during plastic deformation. For materials 3, 18 and 24 a negative value of QM may be used. With this option, the hourglass stiffness is based on the current material properties, that is, the plastic tangent modulus, and is scaled by |QM|.

9. **IHQ default values.** The default value for IHQ, if not defined on *CONTROL_-HOUGRGLASS is as follows:

   a) *Shells.* Viscous type (1 = 2 = 3) for explicit; stiffness type (4=5) for implicit.

   b) *Solids.* Type 2 for explicit; type 6 for implicit.

   c) *Tshell Formulation 1.* Type 2.

10. **IHQ for implict analysis.** For implicit analysis, hourglass forms 6, 7, 9, and 10 are available for solid elements, and the stiffness form (4 = 5) is available for shells.

11. **Tshells and hourglass control.** Hourglass control applicability for tshells depends on the formulation.

    a) Tshell formulations 2 and 3 have 2 × 2 in-plane integration and therefore do not use hourglass control.

    b) In the case of tshell formulation 1, there are two viscous hourglass types (IHQ = 1,2) and one stiffness type (IHQ > 2).

    c) The hourglass type IHQ has no bearing on tshell formulations 5 and 6 as these formulations are based on an assumed strain field, similar to formulation 1 solids with hourglass type 6. The hourglass coefficient QM does affect the behavior of tshell formulations 5 and 6.

*IGA

The *IGA keywords set up and control the isogeometric-related capabilities of LS-DYNA. The *IGA keyword cards may be classified into two sets. Keywords included in the first set are used to describe basic geometric objects. Keywords in the second set are used to create isogeometric elements for the numerical model. In addition, keywords referenced and used in combination with the *IGA keywords are also listed. Keywords in each group are listed in alphabetical order.

Keywords to define geometry:

*IGA_1D_BREP

*IGA_1D_NURBS_UVW

*IGA_1D_NURBS_XYZ

*IGA_2D_BASIS_TRANSFORM_XYZ

*IGA_2D_BEZIER_XYZ

*IGA_2D_BREP

*IGA_2D_NURBS_UVW

*IGA_2D_NURBS_XYZ

*IGA_3D_BASIS_TRANSFORM_XYZ

*IGA_3D_BEZIER_XYZ

*IGA_3D_NURBS_XYZ

*IGA_EDGE_UVW

*IGA_EDGE_XYZ

*IGA_FACE_UVW

*IGA_FACE_XYZ

*IGA_INCLUDE_BEZIER (obsolete as of R14)

*IGA_POINT_UVW

*IGA_VOLUME_XYZ

*IGA

Keywords to create isogeometric elements:

*IGA_REFINE_SHELL

*IGA_REFINE_SOLID

*IGA_SHELL

*IGA_SOLID

Additional IGA-related keywords:

*IGA_INTEGRATION_SHELL_REDUCE

*IGA_INTEGRATION_SOLID_REDUCE

IGA_MASS_*OPTION

*IGA_TIED_EDGE_TO_EDGE

*SECTION_IGA_SHELL

*SECTION_IGA_SOLID

*SET_IGA_EDGE

*SET_IGA_FACE

*SET_IGA_POINT_UVW

*IGA_1D_BREP

Purpose: Define a one-dimensional boundary representation, that is, a closed loop, composed of parametric edges(s).

Card 1	1	2	3	4	5	6	7	8
Variable	BRID							
Type	I							
Default	none							

Parametric Edge Cards. Include as many cards in the following format as desired, see Remark 1. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BRID	One-dimensional boundary representation ID. A unique number must be chosen.
EID*i*	Parametric edge IDs, see *IGA_EDGE_UVW.

Remarks:

1. **Parametric Edge Ordering.** Parametric edges listed on Card 2 need to be ordered such that they form a closed loop. That is, the start and end points of an arbitrary parametric edge should coincide with the end point of the previous and starting point of the next parametric edge, respectively. If the loop is comprised of a single parametric edge only, its start and end points should be identical.

*IGA_1D_NURBS_UVW

Purpose: Define a parametric univariate non-uniform rational B-spline (NURBS).

Card Summary:

Card 1. This card is required.

PATCHID	NR	PR					

Card 2. This card is required.

UNIR							

Card 3a. Include ceil[(NR + PR + 1)/4] of this card if UNIR = 0.

R1	R2	R3	R4

Card 3b. Include this card if UNIR ≠ 0.

RFIRST	RLAST		

Card 4. Include NR of this card.

U	V	W	WGT

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NR	PR					
Type	I	I	I					

VARIABLE	DESCRIPTION
PATCHID	Parametric univariate NURBS patch ID. A unique number must be chosen.
NR	Number of control points in the local *r*-direction
PR	Polynomial degree of the basis in the local *r*-direction

Card 2	1	2	3	4	5	6	7	8
Variable	UNIR							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
UNIR	Knot vector type in the local *r*-direction: EQ.0: Specify the entire knot vector in the local *r*-direction EQ.1: Uniform open knot vector in the local *r*-direction EQ.2: Uniform periodic vector in the local *r*-direction

Knot Vector Cards for the *r*-direction. Include ceil[(NR + PR + 1)/4] cards if UNIR = 0.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1		R2		R3		R4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
R*i*	Knot values in the local *r*-direction with *i* = 1, …, NR + PR + 1

Knot Vector Cards for the *r*-direction. Include this card if UNIR ≠ 0. See Remark 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	RFIRST		RLAST					
Type	F		F					

VARIABLE	DESCRIPTION
RFIRST	First knot value in the local *r*-direction

VARIABLE	DESCRIPTION
RLAST	Last knot value in the local r-direction

Control Point and Weight Cards. Include NR cards; see Remark 2.

Card 4	1	2	3	4	5	6	7	8
Variable	U		V		W		WGT	
Type	F		F		F		F	
Default	none		none		none		1.0	

VARIABLE	DESCRIPTION
Uj	Nonhomogeneous control point coordinates in the parametric u-direction with j = 1, …, NR
Vj	Nonhomogeneous control point coordinates in the parametric v-direction with j = 1, …, NR
Wj	Nonhomogeneous control point coordinates in the parametric w-direction with j = 1, …, NR
WGTj	Control weights with j = 1, …, NR, see Remark 3.

Remarks:

1. **Uniform Knot Vectors.** RFIRST and RLAST define the interval for uniform knot spans. As an example suppose that NR = 4, PR = 3, RFIRST = 1, and RLAST = 8 which yields the following uniform open and uniform periodic vectors [1 1 1 1 8 8 8 8] and [1 2 3 4 5 6 7 8] for UNIR = 1 and UNIR = 2, respectively.

2. **Parametric Curve.** A parametric curve may be defined in the parametric space of an n-variate physical NURBS, such as *IGA_nD_NURBS_XYZ with n = 2 or 3, which requires the definition of (U, V) and (U, V, W) coordinates, respectively. Note that the parametric u-, v-, w-directions and (U, V, W) coordinates are equivalent to the local r-, s-, t-directions and (R, S, T) coordinates in the parametric space of the physical NURBS. This notation is adopted here only to distinguish between the parametric space of the parametric univariate NURBS and the parametric space of the physical bi- or trivariate NURBS in which it is embedded.

3. **Control Weights.** Control weights must be positive and will otherwise be reset to the default unit value.

*IGA_1D_NURBS_XYZ

Purpose: Define a physical univariate non-uniform rational B-spline (NURBS).

Card Summary:

Card 1. This card is required.

PATCHID	NR	PR					

Card 2. This card is required.

UNIR							

Card 3a. Include ceil[(NR + PR + 1)/4] of this card if UNIR = 0.

R1	R2	R3	R4

Card 3b. Include this card if UNIR ≠ 0.

RFIRST	RLAST		

Card 4. Include NR of this card.

X	Y	Z	WGT

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NR	PR					
Type	I	I	I					

VARIABLE	DESCRIPTION
PATCHID	Physical univariate NURBS patch ID. A unique number must be chosen.
NR	Number of control points in the local *r*-direction
PR	Polynomial degree of the basis in the local *r*-direction

Card 2	1	2	3	4	5	6	7	8
Variable	UNIR							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
UNIR	Knot vector type in the local *r*-direction: EQ.0: Specify the entire knot vector in the local *r*-direction EQ.1: Uniform open knot vector in the local *r*-direction EQ.2: Uniform periodic knot vector in the local *r*-direction

Knot Vector Cards for the *r*-direction. Include ceil[(NR + PR + 1)/4] cards if UNIR = 0.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1		R2		R3		R4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
R*i*	Knot values in the local *r*-direction with $i = 1, \ldots, NR + PR + 1$

Knot Vector Cards for the *r*-direction. Include this card if UNIR ≠ 0, see Remark 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	RFIRST		RLAST					
Type	F		F					

VARIABLE	DESCRIPTION
RFIRST	First knot value in the local *r*-direction

VARIABLE	DESCRIPTION
RLAST	Last knot value in the local *r*-direction

Control Point and Weight Cards. Include NR cards.

Card 4	1	2	3	4	5	6	7	8
Variable	X		Y		Z		WGT	
Type	F		F		F		F	
Default	none		none		none		1.0	

VARIABLE	DESCRIPTION
X*j*	Non-homogeneous control point coordinates in the global *x*-direction with *j* = 1, …, NR
Y*j*	Non-homogeneous control point coordinates in the global *y*-direction with *j* = 1, …, NR.
Z*j*	Non-homogeneous control point coordinates in the global *z*-direction with *j* = 1, …, NR.
WGT*j*	Control weights with *j* = 1, …, NR, see Remark 2.

Remarks:

1. **Uniform Knot Vectors.** RFIRST and RLAST define the interval for uniform knot spans. As an example suppose that NR = 4, PR = 3, RFIRST = 1, and RLAST = 8 which yields the following uniform open and uniform periodic knot vectors [1 1 1 1 8 8 8 8] and [1 2 3 4 5 6 7 8] for UNIR = 1 and UNIR = 2, respectively.

2. **Control Weights.** Control weights have to be positive and will otherwise be reset to the default unit value.

*IGA_2D_BASIS_TRANSFORM_XYZ

Purpose: Define a physical bivariate (rational) spline expressed as a basis transform.

Data Card. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	FILENAME						
Type	I	A70						

VARIABLE	DESCRIPTION
PATCHID	Physical bivariate (rational) spline patch ID
FILENAME	Name of file containing patch data; see Remark 1.

Remarks:

1. **Filename.** The current description of the input data is available as a separate document.

*IGA_2D_BEZIER_XYZ

Purpose: Define a physical bivariate (rational) spline using Bézier extraction.

> **NOTE:** As of R15, this keyword is obsolete. Please use *IGA_-2D_BASIS_TRANSFORM_XYZ to define a bivariate (rational) spline expressed as a basis transform.

Data Card. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	FILENAME						
Type	I	A70						

VARIABLE	DESCRIPTION
PATCHID	Physical bivariate (rational) spline patch ID
FILENAME	Name of file containing patch data; see Remark 1.

Remarks:

1. **Filename.** The current description of the input data is available as a separate document.

*IGA_2D_BREP

Purpose: Define a two-dimensional boundary representation composed of parametric faces(s).

Card 1	1	2	3	4	5	6	7	8
Variable	BRID							
Type	I							
Default	none							

Parametric Face Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	FID1	FID2	FID3	FID4	FID5	FID6	FID7	FID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BRID	Two-dimensional boundary representation ID. A unique number must be chosen.
FID*i*	Parametric face IDs, see *IGA_FACE_UVW.

*IGA_2D_NURBS_UVW

Purpose: Define a parametric bivariate non-uniform rational B-spline (NURBS).

Card Summary:

Card 1. This card is required.

PATCHID	NR	NS	PR	PS			

Card 2. This card is required.

UNIR	UNIS						

Card 3a. Include ceil[(NR + PR + 1)/4] of this card if UNIR = 0.

R1	R2	R3	R4

Card 3b. Include this card if UNIR ≠ 0.

RFIRST	RLAST		

Card 4a. Include ceil[(NS + PS + 1)/4] of this card if UNIS = 0.

S1	S2	S3	S4

Card 4b. Include this card if UNIS ≠ 0.

SFIRST	SLAST		

Card 5. Include NR × NS of this card.

U	V	W	WGT

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NR	NS	PR	PS			
Type	I	I	I	I	I			

VARIABLE	DESCRIPTION
PATCHID	Parametric bivariate NURBS patch ID. A unique number must be chosen.
NR	Number of control points in the local *r*-direction
NS	Number of control points in the local *s*-direction
PR	Polynomial degree of the basis in the local *r*-direction
PS	Polynomial degree of the basis in the local *s*-direction

Card 2	1	2	3	4	5	6	7	8
Variable	UNIR	UNIS						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
UNIR	Knot vector type in the local *r*-direction: EQ.0: Specify the entire knot vector in the local *r*-direction. EQ.1: Uniform open knot vector in the local *r*-direction. EQ.2: Uniform periodic knot vector in the local *r*-direction.
UNIS	Knot vector type in the local *s*-direction: EQ.0: Specify the entire knot vector in the local *s*-direction. EQ.1: Uniform open knot vector in the local *s*-direction. EQ.2: Uniform periodic knot vector in the local *s*-direction.

Knot Vector Cards for the *r*-direction. If UNIR = 0, include ceil[(NR + PR + 1)/4] cards.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1		R2		R3		R4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
R*i*	Knot values in the local *r*-direction with *i* = 1, …, NR + PR + 1

Knot Vector Cards for the *r*-direction. Include this card if UNIR ≠ 0; see Remark 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	RFIRST		RLAST					
Type	F		F					

VARIABLE	DESCRIPTION
RFIRST	First knot value in the local *r*-direction
RLAST	Last knot value in the local *r*-direction

Knot Vector Cards for the *s*-direction. Include ceil[(NS + PS + 1)/4] cards if UNIS = 0.

Card 4a	1	2	3	4	5	6	7	8
Variable	S1		S2		S3		S4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
S*j*	Knot values in the local *s*-direction with *j* = 1, …, NS + PS + 1

Knot Vector Cards for the s-direction. Include this card if UNIS ≠ 0. See Remark 1.

Card 4b	1	2	3	4	5	6	7	8
Variable	SFIRST		SLAST					
Type	F		F					

VARIABLE	DESCRIPTION
SFIRST	First knot value in the local *s*-direction
SLAST	Last knot value in the local *s*-direction

Control Point and Weight Cards. Include NR × NS cards reflecting the connectivity. See Remark 2.

Card 5	1	2	3	4	5	6	7	8
Variable	U		V		W		WGT	
Type	F		F		F		F	
Default	none		none		none		1.0	

VARIABLE	DESCRIPTION
U*k*	Nonhomogeneous control point coordinates in the parametric *u*-direction with *k* = 1, …, NR × NS
V*k*	Nonhomogeneous control point coordinates in the parametric *v*-direction with *k* = 1, …, NR × NS
W*k*	Nonhomogeneous control point coordinates in the parametric *w*-direction with *k* = 1, …, NR × NS
WGT*k*	Control weights with *k* = 1, …, NR × NS; see Remark 3.

Remarks:

1. **Uniform Knot Vectors.** RFIRST and RLAST (SFIRST and SLAST) define the interval for uniform knot spans in the local *r*-direction (*s*-direction). As an

example suppose that NR = 4, PR = 3, RFIRST = 1, and RLAST = 8 which yields the following uniform open and uniform periodic knot vectors [1 1 1 1 8 8 8 8] and [1 2 3 4 5 6 7 8] for UNIR = 1 and UNIR = 2, respectively.

2. **Parametric Surface.** A parametric surface may be defined in the parametric space of a trivariate physical NURBS, meaning *IGA_3D_NURBS_XYZ, which requires the definition of (U, V, W) coordinates. Note that the parametric *u*-, *v*-, *w*-directions and (U, V, W) coordinates are equivalent to the local *r*-, *s*-, *t*-directions and (R,S,T) coordinates in the parametric space of the physical NURBS. The notation is adopted here only to distinguish between the parametric space of the parametric bivariate NURBS and the parametric space of the physical trivariate NURBS in which it is embedded in.

3. **Control Weights.** Control weights must be positive and will otherwise be reset to the default unit value.

*IGA_2D_NURBS_XYZ

Purpose: Define a physical bivariate non-uniform rational B-spline (NURBS).

Card Summary:

Card 1. This card is required.

PATCHID	NR	NS	PR	PS			

Card 2. This card is required.

UNIR	UNIS						

Card 3a. Include ceil[(NR + PR + 1)/4] of this card if UNIR = 0.

R1	R2	R3	R4

Card 3b. Include this card if UNIR ≠ 0.

RFIRST	RLAST		

Card 4a. Include ceil[(NS + PS + 1)/4] of this card if UNIS = 0.

S1	S2	S3	S4

Card 4b. Include this card if UNIS ≠ 0.

SFIRST	SLAST		

Card 5. Include NR × NS of this card.

X	Y	Z	WGT

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NR	NS	PR	PS			
Type	I	I	I	I	I			

VARIABLE	DESCRIPTION
PATCHID	Physical bivariate NURBS patch ID. A unique number must be chosen.
NR	Number of control points in the local *r*-direction
NS	Number of control points in the local *s*-direction
PR	Polynomial degree of the basis in the local *r*-direction
PS	Polynomial degree of the basis in the local *s*-direction

Card 2	1	2	3	4	5	6	7	8
Variable	UNIR	UNIS						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
UNIR	Knot vector type in the local *r*-direction: EQ.0: Specify the entire knot vector in the local *r*-direction EQ.1: Uniform open knot vector in the local *r*-direction EQ.2: Uniform periodic knot vector in the local *r*-direction
UNIS	Knot vector type in the local *s*-direction: EQ.0: Specify the entire knot vector in the local *s*-direction EQ.1: Uniform open knot vector in the local *s*-direction EQ.2: Uniform periodic knot vector in the local *s*-direction

Knot Vector Cards for the *r*-direction. Include ceil[(NR + PR + 1)/4] cards if UNIR = 0.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1		R2		R3		R4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
R*i*	Knot values in the local *r*-direction with *i* = 1, …, NR + PR + 1

Knot Vector Cards for the *r*-direction. Include this card if UNIR ≠ 0. See Remark 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	RFIRST		RLAST					
Type	F		F					

VARIABLE	DESCRIPTION
RFIRST	First knot value in the local *r*-direction
RLAST	Last knot value in the local *r*-direction

Knot Vector Cards for the *s*-direction. Include ceil[(NS + PS + 1)/4] cards if UNIS = 0.

Card 4a	1	2	3	4	5	6	7	8
Variable	S1		S2		S3		S4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
S*j*	Knot values in the local *s*-direction with *j* = 1, …, NS + PS + 1

Knot Vector Cards for the *s*-direction. Include this card if UNIS ≠ 0. See Remark 1.

Card 4b	1	2	3	4	5	6	7	8
Variable	SFIRST		SLAST					
Type	F		F					

VARIABLE	DESCRIPTION
SFIRST	First knot value in the local *s*-direction
SLAST	Last knot value in the local *s*-direction

Control Point and Weight Cards. Include NR × NS cards reflecting the connectivity.

Card 5	1	2	3	4	5	6	7	8
Variable	X		Y		Z		WGT	
Type	F		F		F		F	
Default	none		none		none		1.0	

VARIABLE	DESCRIPTION
X*k*	Nonhomogeneous control point coordinates in the global *x*-direction with *k* = 1, …, NR × NS.
Y*k*	Nonhomogeneous control point coordinates in the global *y*-direction with *k* = 1, …, NR × NS.
Z*k*	Nonhomogeneous control point coordinates in the global *z*-direction with *k* = 1, …, NR × NS.
WGT*k*	Control weights with *k* = 1, …, NR × NS. See Remark 2.

Remarks:

1. **Uniform Knot Vectors.** RFIRST and RLAST (SFIRST and SLAST) define the interval for uniform knot spans in the local *r*-direction (*s*-direction). As an example suppose that NR = 4, PR = 3, RFIRST = 1, and RLAST = 8 which yields the

following uniform open and uniform periodic knot vectors [1 1 1 1 8 8 8 8] and [1 2 3 4 5 6 7 8] for UNIR = 1 and UNIR = 2, respectively.

2. **Control Weights.** Control weights must be positive and will otherwise be reset to the default unit value.

*IGA_3D_BASIS_TRANSFORM_XYZ

Purpose: Define a physical trivariate (rational) spline expressed as a basis transform.

Data Card. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	FILENAME						
Type	I	A70						

VARIABLE	DESCRIPTION
PATCHID	Physical trivariate (rational) spline patch ID
FILENAME	Name of file containing patch data; see Remark 1.

Remarks:

1. **Filename.** The current description of the input data is available as a separate document.

*IGA_3D_BEZIER_XYZ

Purpose: Define a physical trivariate (rational) spline using Bézier extraction.

> **NOTE:** As of R15, this keyword is obsolete. Please use *IGA_3D_BASIS_TRANSFORM_XYZ to define a trivariate (rational) spline expressed as a basis transform.

Data Card. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	FILENAME						
Type	I	A70						

VARIABLE	DESCRIPTION
PATCHID	Physical trivariate (rational) spline patch ID
FILENAME	Name of file containing patch data; see Remark 1.

Remarks:

1. **Filename.** The current description of the input data is available as a separate document.

*IGA_3D_NURBS_XYZ

Purpose: Define a physical trivariate non-uniform rational B-spline (NURBS).

Card Summary:

Card 1. This card is required.

PATCHID	NR	NS	NT	PR	PS	PT	

Card 2. This card is required.

UNIR	UNIS	UNIT					

Card 3a. Include ceil[(NR + PR + 1)/4] of this card if UNIR = 0.

R1	R2	R3	R4

Card 3b. Include this card if UNIR ≠ 0.

RFIRST	RLAST		

Card 4a. Include ceil[(NS + PS + 1)/4] of this card if UNIS = 0.

S1	S2	S3	S4

Card 4b. Include this card if UNIS ≠ 0.

SFIRST	SLAST		

Card 5a. Include ceil[(NT + PT + 1)/4] of this card if UNIT = 0.

T1	T2	T3	T4

Card 5b. Include this card if UNIT ≠ 0.

TFIRST	TLAST		

Card 6. Include NR × NS × NT of this card.

X	Y	Z	WGT

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NR	NS	NT	PR	PS	PT	
Type	I	I	I	I	I	I	I	

VARIABLE	DESCRIPTION
PATCHID	Physical trivariate NURBS patch ID. A unique number must be chosen.
NR	Number of control points in the local *r*-direction
NS	Number of control points in the local *s*-direction
NT	Number of control points in the local *t*-direction
PR	Polynomial degree of the basis in the local *r*-direction
PS	Polynomial degree of the basis in the local *s*-direction
PT	Polynomial degree of the basis in the local *t*-direction

Card 2	1	2	3	4	5	6	7	8
Variable	UNIR	UNIS	UNIT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
UNIR	Knot vector type in the local *r*-direction: EQ.0: Specify the entire knot vector in the local *r*-direction EQ.1: Uniform open knot vector in the local *r*-direction EQ.2: Uniform periodic knot vector in the local *r*-direction

VARIABLE	DESCRIPTION
UNIS	Knot vector type in the local *s*-direction: EQ.0: Specify the entire knot vector in the local *s*-direction EQ.1: Uniform open knot vector in the local *s*-direction EQ.2: Uniform periodic knot vector in the local *s*-direction
UNIT	Knot vector type in the local *t*-direction: EQ.0: Specify the entire knot vector in the local *t*-direction EQ.1: Uniform open knot vector in the local *t*-direction EQ.2: Uniform periodic knot vector in the local *t*-direction

Knot Vector Cards for the *r*-direction. Include ceil[(NR + PR + 1)/4] cards if UNIR = 0.

Card 3a	1	2	3	4	5	6	7	8
Variable	R1		R2		R3		R4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
R*i*	Knot values in the local *r*-direction with *i* = 1, …, NR + PR + 1

Knot Vector Cards for the *r*-direction. Include this card if UNIR ≠ 0. See Remark 1.

Card 3b	1	2	3	4	5	6	7	8
Variable	RFIRST		RLAST					
Type	F		F					

VARIABLE	DESCRIPTION
RFIRST	First knot value in the local *r*-direction
RLAST	Last knot value in the local *r*-direction

Knot Vector Cards for the *s*-direction. Include ceil[(NS + PS + 1)/4] cards if UNIS = 0.

Card 4a	1	2	3	4	5	6	7	8
Variable	S1		S2		S3		S4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
S*j*	Knot values in the local *s*-direction with *j* = 1, …, NS + PS + 1

Knot Vector Cards for the *s*-direction. Include this card if UNIS ≠ 0. See Remark 1.

Card 4b	1	2	3	4	5	6	7	8
Variable	SFIRST		SLAST					
Type	F		F					

VARIABLE	DESCRIPTION
SFIRST	First knot value in the local *s*-direction
SLAST	Last knot value in the local *s*-direction

Knot Vector Cards for the *t*-direction. Include ceil[(NT + PT + 1)/4] cards if UNIT = 0.

Card 5a	1	2	3	4	5	6	7	8
Variable	T1		T2		T3		T4	
Type	F		F		F		F	

VARIABLE	DESCRIPTION
T*k*	Knot values in the local *t*-direction with *k* = 1, …, NT + PT + 1

Knot Vector Cards for the *t*-direction. Include this card if UNIT ≠ 0. See Remark 1.

Card 5b	1	2	3	4	5	6	7	8
Variable	TFIRST		TLAST					
Type	F		F					

VARIABLE	DESCRIPTION
TFIRST	First knot value in the local *t*-direction
TLAST	Last knot value in the local *t*-direction

Control Point and Weight Cards. Include NR × NS × NT cards reflecting the connectivity.

Card 6	1	2	3	4	5	6	7	8
Variable	X		Y		Z		WGT	
Type	F		F		F		F	
Default	none		none		none		1.0	

VARIABLE	DESCRIPTION
X*l*	Nonhomogeneous control point coordinates in the global *x*-direction with *l* = 1, …, NR × NS × NT
Y*l*	Nonhomogeneous control point coordinates in the global *y*-direction with *l* = 1, …, NR × NS × NT
Z*l*	Non-homogeneous control point coordinates in the global *z*-direction with *l* = 1, …, NR × NS × NT
WGT*l*	Control weights with *l* = 1, …, NR × NS × NT. See Remark 2.

Remarks:

1. **Uniform Knot Vectors.** RFIRST and RLAST (SFIRST and SLAST or TFIRST and TLAST) define the interval for uniform knot spans in the local *r*-direction (*s*-

direction or *t*-direction). As an example suppose that NR = 4, PR = 3, RFIRST = 1, and RLAST = 8 which yields the following uniform open and uniform periodic knot vectors [1 1 1 1 8 8 8 8] and [1 2 3 4 5 6 7 8] for UNIR = 1 and UNIR = 2, respectively.

2. **Control Weights.** Control weights must be positive and will otherwise be reset to the default unit value.

*IGA_EDGE_UVW_{OPTION}

Purpose: Define a parametric edge, meaning a trimmed and oriented parametric univariate non-uniform rational B-spline (NURBS). Parametric edges are used to compose one-dimensional boundary representations (see *IGA_1D_BREP) and trim parametric as well as physical faces (see *IGA_FACE_UVW and *IGA_FACE_XYZ). In addition, a parametric edge may play a role in defining (1) topological relations among physical faces and (2) boundary conditions and/or constraints. See Remarks 1 and 2, respectively.

Available options include:

<BLANK>

BASIS_TRANSFORM

Card Summary:

Card 1. This card is required.

EID	EXYZID	PATCHID	SENSE	RSTART	REND

Card 2. Included this card if the BASIS_TRANSFORM keyword option is used.

ELID	EDGEID						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	EID	EXYZID	PATCHID	SENSE	RSTART		REND	
Type	I	I	I	I	F		F	
Default	none	none	none	0	RFIRST		RLAST	

VARIABLE	DESCRIPTION
EID	Parametric edge ID. A unique number must be chosen.
EXYZID	Physical edge ID. See *IGA_EDGE_XYZ and Remarks 1 and 3.

VARIABLE	DESCRIPTION
PATCHID	Parametric univariate NURBS patch ID. See *IGA_1D_NURBS_UVW and Remarks 2, 3, and 5. PATCHID is ignored for the BASIS_TRANSFORM keyword option.
SENSE	Sense of orientation with respect to the physical edge (ignored for the BASIS_TRANSFORM keyword option). See Remarks 3 and 5. EQ.0: Same (default) EQ.1: Reversed
RSTART	Parametric coordinate defining the start of the trimmed parametric NURBS. See Remarks 4 and 5. RSTART is ignored for the BASIS_TRANSFORM keyword option.
REND	Parametric coordinate defining the end of the trimmed parametric NURBS. See Remarks 4 and 5. REND is ignored for the BASIS_TRANSFORM keyword option

Element Card. Include this card if using the BASIS_TRANSFORM keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	ELID	EDGEID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ELID	Element ID local to a basis transform. See *IGA_2D_BASIS_TRANSFORM_XYZ and Remark 6. A unique number must be chosen.
EDGEID	Edge ID. See Remark 7. The following convention is used to identify parametric edges on a parent element $[-1,+1]^2$: EQ.1: $u = -1, v \in [-1,+1]$ EQ.2: u $= +1, v \in [-1,+1]$ EQ.3: $v = -1, u \in [-1,+1]$ EQ.4: $v = +1, u \in [-1,+1]$

Remarks:

1. **Topology.** Parametric edges are the constituents of one-dimensional boundary representations (see *IGA_1D_BREP) which specify the exterior of a physical face (see *IGA_FACE_XYZ). Parametric edges referencing the same physical edge, meaning EXYZID, identify the topological relation between their parent physical faces.

2. **Boundary conditions and constraints.** A parametric edge may be invoked to define a curve segment in/on an isogeometric shell or solid; see *IGA_SHELL and *IGA_SOLID. In this case, the control points of a parametric univariate NURBS referenced by PATCHID are defined in the parametric space of its parent physical object. Consequently, a parametric edge may be defined in the parametric space of an n-variate physical NURBS, such as *IGA_nD_NURBS_XYZ with $n = 2$ or 3, which requires the definition of (U, V) or (U, V, W) control point coordinates, respectively. Alternatively, a parametric edge may be defined in the parametric space of an element in a basis transform (see *IGA_2D_BASIS_TRANSFORM_XYZ). In this case, the selection of a parametric edge is restricted to the edges of the parent element. See Remarks 6 and 7.

3. **Sense of orientation.** The orientation of the parametric edge and the parametric univariate NURBS always agree. The SENSE flag indicates if the orientation of the parametric and physical edges agrees. Consider, for instance, two patches connected along an edge as schematically shown in the figure below. The parametric edges f_2 and f_8 share the same physical edge e_2. The direction of f_2 and e_2 is the same, consequently SENSE = 0. In contrast, f_8 and e_2 point in opposite directions, hence SENSE = 1.

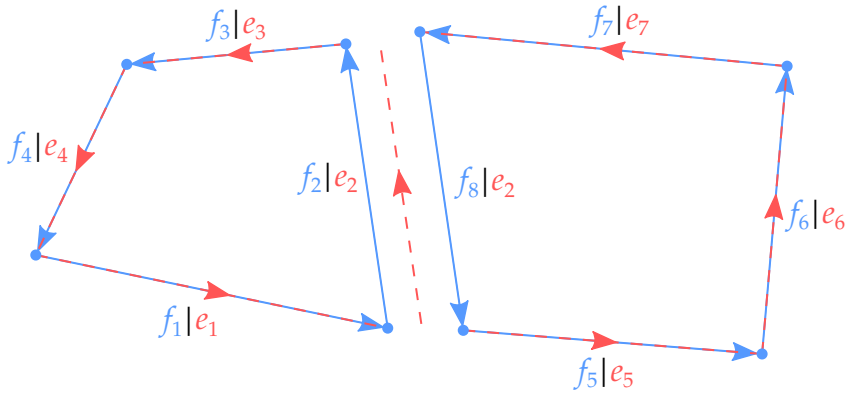


Figure 26-1. Schematic of two patches connected along an edge. Parametric edges are blue while physical edges are red.

4. **Trimming.** Parametric coordinates are used to trim the underlying parametric univariate NURBS. The parametric coordinates should have distinct values such that RSTART < REND. Out of bound parametric coordinates are disregarded and no trimming is performed at the respective ends.

5. **Fields ignored with BASIS_TRANSFORM.** The PATCHID, SENSE, RSTART, and REND fields on Card 1 are not used, and consequently do not need to be defined, if the BASIS_TRANSFORM keyword option is used.

6. **Local element ID.** Basis transforms (see *IGA_2D_BASIS_TRANSFORM_XYZ) by construction rely on elementwise local parameterizations. Consequently, unique definition of a parametric edge requires the local element ID in the patch containing the edge.

7. **Parametric feature edges/lines.** Relevant parametric feature edges/lines are captured by the underlying parameterization of a basis transform (see *IGA_-2D_BASIS_TRANSFORM_XYZ) which thereby facilitates simplified selection of a parametric edge in Card 2.

*IGA_EDGE_XYZ

Purpose: Define a physical edge, meaning a trimmed and oriented physical univariate non-uniform rational B-spline (NURBS). A physical edge is currently only used to define topological relations among different physical faces; see *IGA_EDGE_UVW and *IGA_-FACE_XYZ for more details.

Edge Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	PATCHID	ORI	PIDSTART	PIDEND	PSID		
Type	I	I	I	I	I	I		
Default	none	none	0	0	0	none		

VARIABLE	DESCRIPTION
EID	Physical edge ID. A unique number must be chosen.
PATCHID	Physical univariate NURBS patch ID; see *IGA_1D_NURBS_XYZ.
ORI	Orientation with respect to the physical univariate NURBS. See Remark 1. EQ.0: Same (default) EQ.1: Reversed
PIDSTART	Parametric point ID defining the start of the trimmed physical NURBS. If PIDSTART = 0, the physical univariate NURBS is not trimmed at its start; see Remarks 1, 2, and 3.
PIDEND	Parametric point ID defining the end of the trimmed physical NURBS. If PIDEND = 0, the physical univariate NURBS is not trimmed at its end; see Remarks 1, 2, and 3.
PSID	Parametric point set ID. See *IGA_POINT_UVW and *SET_-IGA_-POINT_UVW. See Remark 4.

Remarks:

1. **Orientation.** The orientation of a physical edge is defined by the parametric points referenced by PIDSTART and PIDEND or, if none of the parametric points are defined, by the orientation flag ORI. Simultaneous use of the parametric points and the orientation flag is only required if the underlying physical NURBS curve is closed. The orientation convention is illustrated in the figure below where the blue solid line is the physical univariate NURBS curve, the red dashed line is the physical edge, and the black circle and dot denote the parametric points specifying the start and end of the edge, respectively.

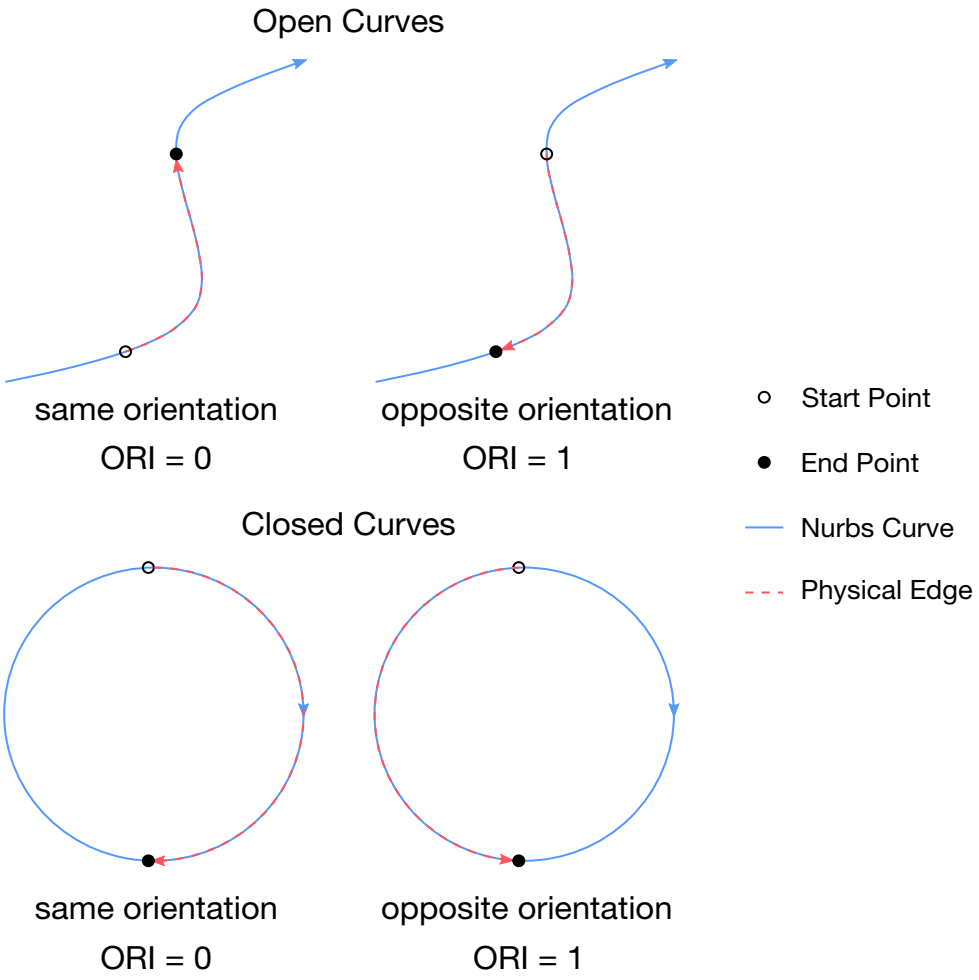


Figure 26-2. Orientation for different physical edges compared to the physical NURBS curve

2. **Trimming.** Parametric points determine where to trim the underlying physical univariate NURBS. Thus, the parametric points should be within the knot vector. An out of bound parametric point is disregarded with no trimming performed at that end.

3. **Topology.** Parametric points are also help establish topological relations among different physical edges.

4. **Dependent Parametric Point Set.** The collection of parametric points is defined within the parametric space of the underlying univariate NURBS and used in boundary condition and/or constraint definition(s).

*IGA_FACE_UVW_{OPTION}

Purpose: Define a parametric face, meaning a trimmed and oriented parametric bivariate non-uniform rational B-spline (NURBS). Parametric faces are used to compose two-dimensional boundary representations (see *IGA_2D_BREP) and trim physical volumes (see *IGA_VOLUME_XYZ). In addition, a parametric face may play a role in defining (1) topological relations among physical volumes and (2) boundary conditions and/or constraints; see Remarks 1 and 2, respectively.

Available options include:

<BLANK>

BASIS_TRANSFORM

Card Summary:

Card 1. This card is required.

FID	FXYZID	PATCHID	SENSE				

Card 2a. Include this card if no keyword option is used. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8

Card 2b. Include this card if using the BASIS_TRANSFORM keyword option.

ELID	FACEID						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FID	FXYZID	PATCHID	SENSE				
Type	I	I	I	I				
Default	none	none	none	0				

VARIABLE	DESCRIPTION
FID	Parametric face ID. A unique number must be chosen.

VARIABLE	DESCRIPTION
FXYZID	Physical face ID. See *IGA_FACE_XYZ and Remarks 1 and 3.
PATCHID	Parametric bivariate NURBS patch ID. See *IGA_2D_NURBS_UVW and Remarks 2 and 3. Ignored for the BASIS_TRANSFORM keyword option (see Remark 4).
SENSE	Sense of orientation with respect to the physical face. Ignored for the BASIS_TRANSFORM keyword option. See Remarks 3 and 4. EQ.0: Same (default) EQ.1: Reversed

Boundary Representation Cards. Include boundary representations defining the parametric face if no keyword option (<BLANK>) is used, see Remark 1. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BRID*i*	One-dimensional boundary representation IDs (see *IGA_1D_BREP) with $i = 1, \ldots, n$ and $n > 0$

Element Card. Include this card if the BASIS_TRANSFORM keyword option is used.

Card 2b	1	2	3	4	5	6	7	8
Variable	ELID	FACEID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ELID	Element ID local to a trivariate basis transform. See *IGA_3D_BASIS_TRANSFORM_XYZ and Remark 6. A unique number must be chosen.
FACEID	Face ID (see Remark 6). The following convention is used to identify parametric faces on a parent element $[-1,+1]^3$: EQ.1: $u = -1, \mathrm{v}, \mathrm{w} \in [-1,+1]$ EQ.2: $u = +1, \mathrm{v}, \mathrm{w} \in [-1,+1]$ EQ.3: $v = -1, \mathrm{u}, \mathrm{w} \in [-1,+1]$ EQ.4: $v = +1, \mathrm{u}, \mathrm{w} \in [-1,+1]$ EQ.5: $w = -1, \mathrm{u}, \mathrm{v} \in [-1,+1]$ EQ.6: $w = +1, \mathrm{u}, \mathrm{v} \in [-1,+1]$

Remarks:

1. **Topology.** Parametric faces are the constituents of two-dimensional boundary representations (see *IGA_2D_BREP) which specify the exterior of a physical volume (see *IGA_VOLUME_XYZ). Parametric faces referencing the same physical face, meaning FXYZID, identify the topological relation between their parent physical volumes.

2. **Boundary conditions and constraints.** A parametric face may be invoked to define a surface segment in/on an isogeometric solid; see *IGA_SOLID. In this case, the control points of a parametric bivariate NURBS referenced by PATCHID are defined in the parametric space of its parent physical object. Consequently, a parametric face may be defined in the parametric space of a trivariate physical NURBS (see *IGA_3D_NURBS_XYZ) which requires the definition of (U, V, W) control point coordinates. Alternatively, a parametric face may be defined in the parametric space of an element within a trivariate basis transform (see *IGA_3D_BASIS_TRANSFORM_XYZ). In the latter case, the selection of a parametric face is restricted to the faces of the parent element, see Remarks 5 and 6.

3. **Sense of orientation.** The orientation of the parametric face and the parametric bivariate NURBS always agree. The SENSE flag indicates if the orientation of the parametric and physical faces agrees.

4. **Fields ignored for BASIS_TRANSFORM keyword option.** The PATCHID and SENSE fields on Card 1 are not used, and consequently do not need to be defined, if the BASIS_TRANSFORM keyword option is used.

5. **Local element ID.** Trivariate basis transforms (see *IGA_3D_BASIS_TRANSFORM_XYZ) by construction rely on elementwise local parameterizations. Consequently, the unique definition of a parametric face requires the local element ID in the patch containing the face.

6. **Parametric feature surfaces.** Relevant parametric feature surfaces are captured by the underlying parameterization of a trivariate basis transform (see *IGA_3D_BASIS_TRANSFORM_XYZ) which thereby facilitates simplified selection of a parametric face in Card 2b.

*IGA_FACE_XYZ_{OPTION}

Purpose: Define a physical face, meaning a trimmed and oriented physical bivariate non-uniform rational B-spline (NURBS) or a physical bivariate basis transform. A physical face is used to define isogeometric shells (see *IGA_SHELL) as well as boundary conditions and/or constraints on its (sub)surface.

Available options include:

<BLANK>

BASIS_TRANSFORM

Card Summary:

Card 1. This card is required.

FID	PATCHID	ORI	PSID	ESID			

Card 2a. Include this card if no keyword option is used. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8

Card 2b. Include this card if using the BASIS_TRANSFORM keyword option.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FID	PATCHID	ORI	PSID	ESID			
Type	I	I	I	I	I			
Default	none	none	0	none	none			

VARIABLE	DESCRIPTION
FID	Physical face ID. A unique number must be chosen.
PATCHID	Depending on the keyword option either a physical bivariate NURBS patch ID for no keyword option (see *IGA_2D_NURBS_-

VARIABLE	DESCRIPTION
	XYZ) or a bivariate basis transform patch ID for BASIS_TRANSFORM keyword option (see *IGA_2D_BASIS_TRANSFORM_XYZ).
ORI	Orientation with respect to the physical bivariate NURBS. EQ.0: Same (default) EQ.1: Reversed
PSID	Parametric point set ID; see *IGA_POINT_UVW, *SET_IGA_POINT_UVW, and Remark 3.
ESID	Parametric edge set ID; see *IGA_EDGE_UVW, *SET_IGA_EDGE_UVW, and Remark 3.

Boundary Representation Cards. Include boundary representations defining the physical face if no keyword option (<BLANK>) is used, see Remarks 1, 2, and 4. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BRID*i*	One-dimensional boundary representation IDs, with $i = 1, \ldots, n$ and $n > 0$. See *IGA_1D_BREP and Remarks 1, 2, and 4.

Element Cards. Include element cards if the BASIS_TRANSFORM keyword option is used, see Remark 4. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for specifying the list of elements included in this *IGA_-FACE_XYZ. If this card is blank, all elements of the patch will be included. See the table below.
E1, …, E7	Specified entity. If this card is blank, all elements of the patch will be included. See the table below.

The BASIS_TRANSFORM Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the BASIS_TRANSFORM keyword option.

OPTION	DESCRIPTION
LIST	E1, …, E7 are element IDs, local to a bivariate basis transform. See *IGA_2D_BASIS_TRANSFORM_XYZ.
RANGE	E1, E3, and E5 are the starting element IDs, while E2, E4, and E6 are the ending element IDs. Element IDs between E1 and E2, E3 and E4, and E5 and E6 will be included.

Remarks:

1. **Trimming.** One-dimensional boundary representations, also called trimming loops, are used to trim the underlying bivariate NURBS. The orientation of the trimming loops uniquely define trimming, meaning travelling along the loop. The domain on the right-hand side of the loop is removed.

2. **Topology.** To establish topological relations among physical faces, at least a single boundary representation must be included.

3. **Dependent parametric object sets.** The collection of parametric points and edges is defined within the parametric space of the underlying bivariate NURBS and used in boundary condition and/or constraint definition(s).

4. **Partitioning.** Multiple isogeometric shells may be defined using the same underlying geometry. This may be done to assign different materials and/or sections, thereby subdividing the geometry into separate parts. To this end, the definition of multiple physical faces is required. Based on the keyword option, two cases can be distinguished:

   a) Without the keyword option (<BLANK>), the geometry is defined using a bivariate physical NURBS (see *IGA_2D_NURBS_XYZ). Multiple physical faces may be defined using different one-dimensional boundary representations (see *IGA_1D_BREP) on Card 2a.

   b) With the BASIS_TRANSFORM keyword option, the geometry is defined using a bivariate basis transform (see *IGA_2D_BASIS_TRANSFORM_XYZ). Multiple physical faces may be defined by including a different set of elements on Card 2b.

*IGA_INCLUDE_BEZIER

Purpose: Include novel computer-aided geometric definitions by virtue of Bézier extraction to perform isogeometric analysis. For more information on the include file format, please contact LS-DYNA support.

> **NOTE:** As of R14, this keyword is obsolete. Please use *IGA_2D_BEZIER_XYZ and *IGA_3D_BEZIER_XYZ to define bi- and trivariate (rational) splines using Bézier extraction, respectively.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	FILETYPE	PID	DIM					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
FILENAME	Name of the file to be included; see Remark 1.
FILETYPE	Type of the file to be included: EQ.1: ASCII
PID	Part ID
DIM	Parametric dimension: EQ.2: Surface EQ.3: Volume

Remarks:

1. **Multiple Patches.** The include file may contain multiple patches of the same parametric dimension which will also share the same part, and consequently, material and section IDs. Some properties may be set on a patch by patch basis (see *IGA_SHELL and *IGA_SOLID for DIM = 2 and DIM = 3, respectively)

*IGA_INTEGRATION_SHELL_REDUCE

Purpose: Apply a reduced Generalized Gaussian Quadrature rule to IGA shells. To activate this integration rule, set IRL = 3 in *SECTION_IGA_SHELL.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NRDR	NRDS					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
PATCHID	Patch ID defined in *IGA_SHELL: EQ.0: Apply to all IGA shell patches
NRDR	Reduced degree in the *r*-direction: EQ.0: Full integration EQ.1: Reduced by 1 EQ.2: Reduced by 2
NRDS	Reduced degree in the *s*-direction: EQ.0: Full integration EQ.1: Reduced by 1 EQ.2: Reduced by 2

*IGA_INTEGRATION_SOLID_REDUCE

Purpose: Apply a reduced Generalized Gaussian Quadrature rule to IGA solids. Activating this integration rule requires setting IR = 3 in *SECTION_IGA_SOLID.

Card 1	1	2	3	4	5	6	7	8
Variable	PATCHID	NRDR	NRDS	NRDT				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
PATCHID	PATCH ID defined in *IGA_SOLID: EQ.0: Apply to all IGA solid patches.
NRDR	Reduced degree in the *r*-direction: EQ.0: Full integration EQ.1: Reduced by 1 EQ.2: Reduced by 2
NRDS	Reduced degree in the *s*-direction: EQ.0: Full integration EQ.1: Reduced by 1 EQ.2: Reduced by 2
NRDT	Reduced degree in the *t*-direction: EQ.0: Full integration EQ.1: Reduced by 1 EQ.2: Reduced by 2

IGA_MASS_*OPTION

Available options include:

POINT_UVW

EDGE_UVW

FACE_XYZ

SET_POINT_UVW

SET_EDGE_UVW

SET_FACE_XYZ

Purpose: Define a uniformly distributed mass on IGA entities.

Input Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TMASS	DMASS					
Type	I	F	F					
Default	none	0.0	0.0					

VARIABLE	DESCRIPTION
ID	Parametric point, parametric point set, parametric edge, parametric edge set, physical face, or physical face set ID, depending on the choice of keyword option.
TMASS	Mass added to IGA entity. See Remark 1.
DMASS	Uniformly distributed mass added to IGA entity. See Remark 1.

Remarks:

1. **Added mass.** The final added mass is computed as the sum of TMASS divided by length (for parametric edges) or area (for physical faces) of the IGA entity and DMASS. LS-DYNA computes the length or area of the IGA entity. The unit of

TMASS is [MASS] while DMASS is [MASS]/[LENGTH or AREA]. Numerical integration determines the corresponding added mass at control points.

*IGA_POINT_UVW_{OPTION}

Purpose: Define a parametric point. A parametric point may play a role in defining (1) topological relations among physical edges and (2) boundary conditions and/or constraints; see Remarks 1 and 2, respectively.

Available options include:

<BLANK>

BASIS_TRANSFORM

Card Summary:

Card 1. This card is required.

PID	NID	U	V	W

Card 2. Include this card if using the BASIS_TRANSFORM keyword option.

ELID							

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NID	U		V		W	
Type	I	I	F		F		F	
Default	none	none	none		none		none	

VARIABLE	DESCRIPTION
PID	Parametric point ID. A unique number must be chosen.
NID	Node IDs; see *NODE and Remark 3.
U	Coordinates in the parametric u-direction
V	Coordinates in the parametric v-direction
W	Coordinates in the parametric w-direction

Element Card. Include this card if using the BASIS_TRANSFORM keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	ELID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
ELID	Element ID local to a basis transform. See *IGA_nD_BASIS_TRANSFORM_XYZ with n = 2 or 3 and Remark 4. A unique number must be chosen.

Remarks:

1. **Topology.** Parametric points may specify the starting and ending point of a physical edge; see *IGA_EDGE_XYZ. Parametric points referencing the same node (also interpreted as a physical point), meaning NID, identify the topological relation between their parent physical edges.

2. **Boundary conditions and constraints.** A parametric point may be invoked to define a discrete location in/on an isogeometric shell or solid; see *IGA_SHELL and *IGA_SOLID. In this case, the coordinates of the parametric point are defined in the parametric space of its parent physical object. Consequently, a parametric point may be defined in the parametric space of an n-variate physical NURBS, that is, *IGA_nD_NURBS_XYZ with n = 1, 2, or 3, which requires the definition of (U), (U,V), and (U,V,W) coordinates, respectively. Alternatively, a parametric point may be defined in the parametric space of an element within an n-variate basis transform, that is, *IGA_nD_BASIS_TRANSFORM_XYZ with n = 2, or 3, which requires the definition of (U,V), and (U,V,W) coordinates, respectively.

3. **The role of nodes.** Nodes can be interpreted as physical points.

4. **Local element ID.** Basis transforms (see *IGA_2D/3D_BASIS_TRANSFORM_XYZ) by construction rely on elementwise local parameterizations. Consequently, a unique definition of a parametric point requires the local element ID in the patch containing the point.

*IGA_REFINE_SHELL

Purpose: Define refinement for isogeometric shells (see *IGA_SHELL). This capability is limited to shells parameterized by bivariate structured splines (see *IGA_FACE_XYZ and *IGA_2D_NURBS_XYZ).

Card Summary:

Card 1. This card is required.

RID	RTYP						

Card 2. Include this card if RTYP = 1 or 2.

HRTYP	RR	RS					

Card 3. Include this card if RTYP = 2 or 3.

TR	TS						

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	RID	RTYP						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
RID	Isogeometric shell refinement ID. A unique number must be chosen.
RTYP	Refinement type: EQ.1: *h*-refinement (knot insertion) EQ.2: *k*-refinement, meaning *p*-refinement followed by *h*-refinement EQ.3: *p*-refinement (degree elevation)

Knot Insertion Card. Include this card if RTYP = 1 or 2.

Card 2	1	2	3	4	5	6	7	8
Variable	HRTYP	RR	RS					
Type	I	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
HRTYP	*h*-refinement type: EQ.1: Number of subdivisions-based. Each nonzero knot span is subdivided to NINT(RR) × NINT(RS) equal knot spans. Here, NINT(x) rounds x to the nearest integer.
RR	NINT(RR) is the number of equal parametric segments in the local *r*-direction. NINT(RR) means the input RR is rounded to the nearest integer.
RS	NINT(RS) is the number of equal parametric segments in the local *s*-direction. NINT(RS) means the input RS is rounded to the nearest integer.

Polynomial Degree Elevation Card. Include this card if RTYP =2 or 3.

Card 3	1	2	3	4	5	6	7	8
Variable	TR	TS						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
TR	Target polynomial degree in the local *r*-direction; see Remark 1.
TS	Target polynomial degree in the local *s*-direction; see Remark 1.

Remarks:

1. **Polynomial degree reduction.** Polynomial degree reduction is not facilitated, meaning $\mathrm{TR} = \max[0, (\mathrm{TR} - \mathrm{PR})]$ and $\mathrm{TS} = \max[0, (\mathrm{TS} - \mathrm{PS})]$, where PR and PS designate the polynomial degree in the local r- and s-directions, respectively. See *IGA_2D_NURBS_XYZ keyword for more details.

*IGA_REFINE_SOLID

Purpose: Define refinement for isogeometric solids (see *IGA_SOLID). This capability is limited to solids parameterized by trivariate structured splines (see *IGA_VOLUME_-XYZ and *IGA_3D_NURBS_XYZ).

Card Summary:

Card 1. This card is required.

RID	RTYP						

Card 2. Include this card if RTYP = 1 or 2.

HRTYP	RR	RS	RT				

Card 3. Include this card if RTYP = 2 or 3.

TR	TS	TT					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	RID	RTYP						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
RID	Isogeometric solid refinement ID. A unique number must be chosen.
RTYP	Refinement type: EQ.1: *h*-refinement (knot insertion) EQ.2: *k*-refinement, meaning *p*-refinement followed by *h*-refinement EQ.3: *p*-refinement (degree elevation)

Knot Insertion Card. Include this card if RTYP = 1 or 2.

Card 2	1	2	3	4	5	6	7	8
Variable	HRTYP	RR	RS	RT				
Type	I	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
HRTYP	*h*-refinement type: EQ.1: Number of subdivisions-based. Each nonzero knot span is subdivided to NINT(RR) × NINT(RS) × NINT(RT) equal knot spans. Here, NINT(x) rounds x to the nearest whole integer. EQ.2: Physical length-based. Each nonzero knot span is subdivided such that the physical dimension of the resulting knot spans is about RR × RS × RT. See Remark 1.
RR	Knot refinement parameter in the local *r*-direction: HRTYP.EQ.1: NINT(RR) is the number of equal parametric segments in the local *r*-direction. NINT(RR) means the input RR is rounded to the nearest integer. HRTYP.EQ.2: RR is the approximate physical length of the resulting knot spans in the local *r*-direction.
RS	Knot refinement parameter in the local *s*-direction: HRTYP.EQ.1: NINT(RS) is the number of equal parametric segments in the local *s*-direction. NINT(RS) means the input RS is rounded to the nearest integer. HRTYP.EQ.2: RS is the approximate physical length of the resulting knot spans in the local *s*-direction.
RT	Knot refinement parameter in the local *t*-direction: HRTYP.EQ.1: NINT(RT) is the number of equal parametric segments in the local *t*-direction. NINT(RT) means the input RT is rounded to the nearest integer. HRTYP.EQ.2: RT is the approximate physical length of the

VARIABLE	DESCRIPTION
	resulting knot spans in the local *t*-direction.

Polynomial Degree Elevation Card. Include this card if RTYP = 2 or 3.

Card 3	1	2	3	4	5	6	7	8
Variable	TR	TS	TT					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
TR	Target polynomial degree in the local *r*-direction; see Remark 2.
TS	Target polynomial degree in the local *s*-direction; see Remark 2.
TT	Target polynomial degree in the local *t*-direction; see Remark 2.

Remarks:

1. **Physical length-based refinement.** The approach is currently limited to the simplest trilinear parameterizations, meaning NR = NS = NT = 2 and PR = PS = PT = 1, where NR, NS, and NT denote the number of control points while PR, PS, and PT designate the polynomial degree in the local *r*-, *s*-, and *t*-directions, respectively. See *IGA_3D_NURBS_XYZ for more details.

2. **Polynomial degree reduction.** Polynomial degree reduction is not facilitated. Thus, $TR = \max[0, (TR - PR)]$, $TS = \max[0, (TS - PS)]$, and $TT = \max[0, (TT - PT)]$, where PR, PS, and PT designate the polynomial degree in the local *r*-, *s*-, and *t*-directions, respectively. See *IGA_3D_NURBS_XYZfor more details.

*IGA_SHELL

Purpose: Define a set of isogeometric shell elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	PID	NISR	NISS	RID		IDFNE	
Type	I	I	F	F	I		I	
Default	none	none	0.	0.	0		0	

VARIABLE	DESCRIPTION
SID	Isogeometric shell (patch) ID; see Remarks 1 and 2. A unique number must be chosen.
PID	Part ID
NISR	Interpolation elements in the local *r*-direction (see Remark 3): LT.0.0: |NISR| is the average edge length of the interpolation elements in the local *r*-direction. EQ.0.0: The number of interpolation elements per isogeometric element is equal to the polynomial degree in the local *r*-direction. GT.0.0: Number of interpolation elements per isogeometric element in the local *r*-direction. NISR must be an integer.
NISS	Interpolation elements in the local *s*-direction (see Remark 3): LT.0.0: |NISS| is the average edge length of the interpolation elements in the local *s*-direction. EQ.0.0: The number of interpolation elements per isogeometric element is equal to the polynomial degree in the local *s*-direction. GT.0.0: Number of interpolation elements per isogeometric element in the local *s*-direction. NISS must be an integer.
RID	Shell refinement ID. See *IGA_REFINE_SHELL.
IDFNE	Element ID of the first IGA element (knot span) within this isogeometric shell (patch) definition

Remarks:

1. **Discretized shells.** A shell ID is a physical face ID; see *IGA_FACE_XYZ.

2. **Mass matrix.** Like the standard convention in the structural solver, the row sum mass matrix formulation is invoked by default. A consistent mass matrix may be used at the model level by invoking the *CONTROL_IMPLICIT_CONSISTENT_MASS keyword.

3. **Interpolation elements.** Contact treatment and post-processing are presently dealt with interpolation elements defined by interpolation nodes. These nodes and elements are automatically created.

*IGA_SOLID

Purpose: Define a set of isogeometric solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	PID	NISR	NISS	NIST	RID		
Type	I	I	F	F	F	I		
Default	none	none	0.	0.	0.	0		

VARIABLE	DESCRIPTION
SID	Isogeometric solid (patch) ID; see Remarks 1 and 2. A unique number must be chosen.
PID	Part ID
NISR	Interpolation elements in the local r-direction (see Remark 3): LT.0.0: |NISR| is the average edge length of the interpolation elements in the local r-direction. EQ.0.0: The number of interpolation elements per isogeometric element is equal to the polynomial degree in the local r-direction. GT.0.0: Number of interpolation elements per isogeometric element in the local r-direction. NISR must be an integer.
NISS	Interpolation elements in the local s-direction (see Remark 3): LT.0.0: |NISS| is the average edge length of the interpolation elements in the local s-direction. EQ.0.0: The number of interpolation elements per isogeometric element is equal to the polynomial degree in the local s-direction. GT.0.0: Number of interpolation elements per isogeometric element in the local s-direction. NISS must be an integer.
NIST	Interpolation elements in the local t-direction (see Remark 3): LT.0.0: |NIST| is the average edge length of the interpolation elements in the local t-direction.

VARIABLE	DESCRIPTION
	EQ.0.0: The number of interpolation elements per isogeometric element is equal to the polynomial degree in the local t-direction.
	GT.0.0: Number of interpolation elements per isogeometric element in the local t-direction. NIST must be an integer.
RID	Solid refinement ID. See *IGA_REFINE_SOLID.

Remarks:

1. **Discretized solids.** A solid ID is a physical volume ID; see *IGA_VOLUME_XYZ.

2. **Mass matrix.** Similar to the standard convention in the structural solver, the row sum mass matrix formulation is invoked by default. A consistent mass matrix may be used at the model level by invoking the *CONTROL_IMPLICIT_CONSISTENT_MASS keyword.

3. **Interpolation elements.** Contact treatment and post-processing are presently dealt with interpolation elements defined by interpolation nodes. These nodes and elements are automatically created.

*IGA_TIED_EDGE_TO_EDGE

Purpose: Mechanically couple entities along topologically connected edges. The topology is given implicitly using the relation between parametric and physical edges; see *IGA_EDGE_UVW and *IGA_EDGE_XYZ.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYPE	FORM	SFD	SFR	SFT		
Type	I	I	I	F	F	F		
Default	0	0	0	1.0	1.0	1.0		

VARIABLE	DESCRIPTION
ID	Apply coupling to entities referenced by the ID field along topologically connected edges. The next field, TYPE, specifies the type of entity to which ID refers because entities of different kinds, such as parts and part sets, are not uniquely numbered. Currently (as of June 2020), Currently, no types requiring an ID are supported. This field is reserved for future enhancements.
TYPE	Type of ID: EQ.0: Include all topological connections in the model. No ID required.
FORM	Coupling formulation: EQ.0: Penalty-based tied contact
SFD	Scaling factor for displacement penalty stiffness
SFR	Scaling factor for rotational penalty stiffness
SFT	Scaling factor for thin constraint penalty stiffness (rotation free elements)

***IGA_VOLUME_XYZ**_*{OPTION}*

Purpose: Define a physical volume, that is, a physical trivariate non-uniform rational B-spline (NURBS) or a physical trivariate basis transform. A physical volume is used to define isogeometric solids (see *IGA_SOLID) as well as boundary conditions and/or constraints on its (sub)volume.

Available options include:

<BLANK>

BASIS_TRANSFORM

Card Summary:

Card 1. This card is required.

VID	PATCHID	PSID	ESID	FSID			

Card 2a. Include this card if no keyword option is used. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8

Card 2b. Include this card if using the BASIS_TRANSFORM keyword option.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	VID	PATCHID	PSID	ESID	FSID			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
VID	Parametric volume ID. A unique number must be chosen.
PATCHID	Depending on the keyword option either a physical trivariate NURBS patch ID for no keyword option (see *IGA_3D_NURBS_-

VARIABLE	DESCRIPTION
	XYZ) or a trivariate basis transform patch ID for the BASIS_TRANSFORM keyword option (see *IGA_3D_BASIS_TRANSFORM_XYZ).
PSID	Parametric point set ID; see *IGA_POINT_UVW and *SET_IGA_POINT_UVW. See Remark 2.
ESID	Parametric edge set ID; see *IGA_EDGE_UVW and *SET_IGA_EDGE_UVW. See Remark 2.
FSID	Parametric face set ID; see *IGA_FACE_UVW and *SET_IGA_FACE_UVW. See Remark 2.

Boundary Representation Cards. Include boundary representations defining the physical volume, see Remark 1. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	BRID1	BRID2	BRID3	BRID4	BRID5	BRID6	BRID7	BRID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
BRID*i*	Two-dimensional boundary representation IDs (see *IGA_2D_BREP) with $i = 1, \ldots, n$ and $n > 0$.

Element Cards. Include element cards if the BASIS_TRANSFORM keyword option is used; see Remark 3. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
OPTION	Option for specifying the list of elements included in this *IGA_-VOLUME_XYZ. If this card is blank, all elements of the patch will be included. See the table below.
E1, …, E7	Specified entity. If this card is blank, all elements of the patch will be included. See the table below.

The BASIS_TRANSFORM Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the BASIS_TRANSFORM keyword option.

OPTION	DESCRIPTION
LIST	E1, …, E7 are element IDs, local to a bivariate basis transform. See *IGA_3D_BASIS_TRANSFORM_XYZ.
RANGE	E1, E3, and E5 are the starting element IDs, while E2, E4, and E6 are the ending element IDs. Element IDs between E1 and E2, E3 and E4, and E5 and E6 will be included.

Remarks:

1. **Topology.** To establish topological relations among physical volumes, at least a single boundary representation must be included.

2. **Dependent parametric object sets.** The collection of parametric points, edges, and faces identifies, in the topological sense, reduced subsets of the underlying trivariate NURBS parameterization. These reduced subsets are used in boundary conditions and/or constraint definitions.

3. **Partitioning.** Multiple isogeometric solids may be defined using the same underlying geometry. This may be done to assign different materials and/or sections, thereby subdividing the geometry into separate parts. To this end, the definition of multiple physical volumes is required. With the BASIS_TRANSFORM keyword option, the geometry is defined using a bivariate basis transform (see *IGA_3D_BASIS_TRANSFORM_XYZ). Multiple physical volumes may be defined by including a different set of elements on Card 2b.

*INCLUDE

The keyword *INCLUDE provides a means of reading independent input files containing model data. The file contents are placed directly at the location of the *INCLUDE line.

INCLUDE_*{OPTION}

*INCLUDE_AUTO_OFFSET

*INCLUDE_COMPENSATION_BEFORE_SPRINGBACK

*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK

*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP

*INCLUDE_COMPENSATION_CURRENT_TOOLS

*INCLUDE_COMPENSATION_CURVE

*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE

*INCLUDE_COMPENSATION_NEW_RIGID_TOOL

*INCLUDE_COMPENSATION_NODE_SET

*INCLUDE_COMPENSATION_ORIGINAL_DYNAIN

*INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL

*INCLUDE_COMPENSATION_ORIGINAL_TOOL

*INCLUDE_COMPENSATION_SPRINGBACK_INPUT

*INCLUDE_COMPENSATION_SYMMETRIC_LINES

*INCLUDE_COMPENSATION_TANGENT_CONSTRAINT

*INCLUDE_COMPENSATION_TRIM_CURVE

*INCLUDE_COMPENSATION_TRIM_NODE

*INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE

*INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL

*INCLUDE_COSIM

*INCLUDE_ISPG

*INCLUDE_MULTISCALE

*INCLUDE_MULTISCALE_SPOTWELD

*INCLUDE_PATH

INCLUDE_STAMPED_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}

*INCLUDE_STAMPED_PART_SOLID_TO_SOLID

*INCLUDE_TRIM

*INCLUDE_UNITCELL

*INCLUDE_WD_FINAL_PART

*INLUDE_WD_INITIAL_BLANK

*INCLUDE_WD_WELDING_CURVE

*INCLUDE_{OPTION}

Purpose: Include independent input files containing model data.

Available options include:

<BLANK>

BINARY

NASTRAN

TRANSFORM

TRANSFORM_BINARY

The BINARY and TRANSFORM_BINARY options specify that the initial stress file, dynain, is written in a binary format. See the keyword *INTERFACE_SPRINGBACK.

The TRANSFORM and TRANSFORM_BINARY options allow for node, element, and set IDs to be offset and for coordinates and constitutive parameters to be transformed and scaled.

Card Summary:

Card 1a. This card is included if the keyword option is unset (<BLANK>). Include as many of this card as needed. The next keyword ("*") card terminates this input.

FILENAME

Card 1b. This card is included if the keyword option is set.

FILENAME

Card 2a. This card is included if the NASTRAN keyword option is used.

BEAMDF	SHELLDF	SOLIDDF					

Card 2b.1. This card is included if the TRANSFORM keyword option is used.

IDNOFF	IDEOFF	IDPOFF	IDMOFF	IDSOFF	IDFOFF	IDDOFF	

Card 2b.2. This card is included if the TRANSFORM keyword option is used.

IDROFF		PREFIX	SUFFIX				

Card 2b.3. This card is included if the TRANSFORM keyword option is used.

FCTMAS	FCTTIM	FCTLEN	FCTTEM	INCOUT1	FCTCHG		

Card 2b.4. This card is included if the TRANSFORM keyword option is used.

TRANID							

Data Card Definitions:

File name without Keyword Option Card. Include this card if the keyword option is unset (<BLANK>). Multiple file names can be specified (each on its own line(s)) which are processed sequentially. Note that each file name can be multiple lines (see Remark 2). File names are read until the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	File name of file to be included (see Remark 2)

File name with Keyword Option Card. Include this card if the keyword option is set. Only one file name can be specified.

Card 1a	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	File name of file to be included (see Remark 2)

Nastran Card. Additional card for the NASTRAN keyword option.

Card 2a	1	2	3	4	5	6	7	8
Variable	BEAMDF	SHELLDF	SOLIDDF					
Type	I	I	I					
Default	2	21	18					

VARIABLE	DESCRIPTION
BEAMDF	LS-DYNA beam element type. Defaults to type 2.
SHELLDF	LS-DYNA shell element type. Defaults to type 21.
SOLIDDF	LS-DYNA solid element type. Defaults to type 18.

Transform Card 1. Additional card for TRANSFORM keyword option.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	IDNOFF	IDEOFF	IDPOFF	IDMOFF	IDSOFF	IDFOFF	IDDOFF	
Type	I	I	I	I	I	I	I	

VARIABLE	DESCRIPTION
IDNOFF	Offset to node ID
IDEOFF	Offset to element ID
IDPOFF	Offset to part ID, nodal rigid body ID, constrained nodal set ID, rigidwall ID, and cross section ID (see *DATABASE_CROSS_SECTION)
IDMOFF	Offset to material ID and equation of state ID
IDSOFF	Offset to set ID
IDFOFF	Offset to function ID, table ID, and curve ID
IDDOFF	Offset to any ID defined through *DEFINE, except the FUNCTION, TABLE, and CURVE options (see IDFOFF)

Transform Card 2. Additional card for TRANSFORM keyword option.

Card 2b.2	1	2	3	4	5	6	7	8
Variable	IDROFF		PREFIX	SUFFIX				
Type	I		A	A				

VARIABLE	DESCRIPTION
IDROFF	Used for all offsets except for those listed above
PREFIX	Prefix added to the beginning of the titles/heads defined in the keywords (like *MAT, *PART, *SECTION, *DEFINE, for examples) of the included file. A dot, ".", is automatically added between the prefix and the existing title.
SUFFIX	Suffix added to the end of the titles/heads defined in the keywords of the included file. A dot, ".", is automatically added between the suffix and the existing title.

Transform Card 3. Additional card for TRANSFORM keyword option.

Card 2b.3	1	2	3	4	5	6	7	8
Variable	FCTMAS	FCTTIM	FCTLEN	FCTTEM	INCOUT1	FCTCHG		
Type	F	F	F	A	I	F		

VARIABLE	DESCRIPTION
FCTMAS	Mass transformation factor. For example, FCTMAS = 1000 when the original mass unit is in tons and the new unit is kg.
FCTTIM	Time transformation factor. For example, FCTTIM = .001 when the original time unit is in milliseconds and the new time unit is seconds.
FCTLEN	Length transformation factor
FCTTEM	Temperature transformation factor consisting of a four character flag: FtoC (Fahrenheit to Centigrade), CtoF, FtoK, KtoF, KtoC, and CtoK.

VARIABLE	DESCRIPTION
INCOUT1	Set to 1 for the creation of a file, DYNA.INC, which contains the transformed data. The data in this file can be used in future include files and should be checked to ensure that all the data was transformed correctly.
FCTCHG	Electric charge transformation factor. It currently only applies to piezoelectric material related cards, see *MAT_ADD_PZELECTRIC for details.

Transform Card 4. Additional card for TRANSFORM keyword option.

Card 2b.4	1	2	3	4	5	6	7	8
Variable	TRANID							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
TRANID	Transformation ID. If 0, no transformation will be applied. See *DEFINE_TRANSFORMATION.

Remarks:

1. **Scalability.** To make the input file easy to maintain, this keyword allows the input file to be split into subfiles. Each subfile can again be split into sub-subfiles and so on. This option is beneficial when the input data deck is very large. Consider the following example:

```
*TITLE
full car model
*INCLUDE
carfront.k
*INCLUDE
carback.k
*INCLUDE
occupantcompartment.k
*INCLUDE
dummy.k
*INCLUDE
bag.k
```

```
*CONTACT
⋮
*END
```

Note that the command *END terminates the include file.

The carfront.k file can again be subdivided into rightrail.k, leftrail.k, battery.k, wheel-house.k, shotgun.k, etc.. Each *.k file can include nodes, elements, boundary conditions, initial conditions, and so on.

```
*INCLUDE
rightrail.k
*INCLUDE
leftrail.k
*INCLUDE
battery.k
*INCLUDE
wheelhouse.k
*INCLUDE
shotgun.k
⋮
*END
```

2. **File Name Length Limitations.** File names are limited to 236 characters spread over up to three 80 character lines. When 2 or 3 lines are needed to specify the filename or pathname, end the preceding line with "␣+" (space followed by a plus sign) to signal that a continuation line follows. Note that the "␣+" combination is, itself, part of the 80 character line; hence the maximum number of allowed characters is 78 + 78 + 80 = 236.

3. **NASTRAN Option.** The transformed LS-DYNA deck for *INCLUDE_NASTRAN will be automatically written to file DYNA.INC.

INCLUDE_AUTO_OFFSET_*{OPTION}

Purpose: This particular *INCLUDE keyword offsets node and element IDs to avoid duplication during stamping simulations. In single process stamping simulations the rigid tools often undergo several iterations of modifications, so the node or element IDs comprising the new tools sometimes conflict with other parts of the model. In a multiple process simulation, the IDs of rigid tools from later processes may overlap the IDs of the tools or the formed sheet blank from the previous processes, which makes it difficult to automate the process simulation. This keyword automatically checks for and offsets the duplicate IDs. It works on shell, solid, beam and discrete (*ELEMENT_DISCRETE) elements. The *CONTROL_FORMING_MAXID keyword is related.

Available option includes:

<BLANK>

USER

The USER option allows for specifying offset IDs of the nodes and elements (shells, solids and beams) for the included file.

Note this keyword does not work on the dynain file with stress and strain information. Thus, in a multiple included files case, the dynain file for the blank should always be included first using simply *INCLUDE; see example below.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

User Set Offset Card. This card is included if and only if the USER option is used.

Card 2	1	2	3	4	5	6	7	8
Variable	NOFFSET	NEOFFSET						
Type	I	I						

VARIABLE	DESCRIPTION
FILENAME	Name of file to be included

VARIABLE	DESCRIPTION
NOFFSET	An offset value for the node IDs of the included file
NEOFFSET	An offset value for the element IDs of the included file

***INCLUDE_AUTO_OFFSET (without option):**

This keyword can be used to offset element and node IDs of the tooling. *This keyword will not offset IDs of meshes with initial stress and strain information*. As such, the sheet blank (including dynain file) should always be included first using the *INCLUDE keyword, followed by *INCLUDE_AUTO_OFFSET to offset tooling mesh IDs which do not have stress and strain information.

Incoming element and node IDs of the rigid tooling mesh files, such as the punch, die, and binder, could be overlapped with each other or overlapped with those on the sheet blank. Multiple *INCLUDE_AUTO_OFFSET keywords can be used to include the punch, die, and binder separately, if desired. For example, four different components of the tooling, the upper die, lower punch, binder and gage pins, can be included and their element and node IDs properly offset after those of a gravity-loaded sheet blank:

```
*INCLUDE
gravity.dynain
*INCLUDE_AUTO_OFFSET
upperdie.k
*INCLUDE_AUTO_OFFSET
lowerpunch.k
*INCLUDE_AUTO_OFFSET
binder.k
*INCLUDE_AUTO_OFFSET
pins.k
```

All of the included meshes can have conflicting node and element IDs starting from 1. Node and element IDs will be offset and reordered in the order of the included files. Included tool files whose node and element IDs do not overlap with those on either the blank or other tools will not be offset or reordered. In many circumstances this feature allows for bypassing a metal forming setup GUI when updating just one or two tooling pieces.

***INCLUDE_AUTO_OFFSET_USER:**

This option gives you more control on the starting node and element IDs of the included files. It applies to shell, solid and beam elements. The following example shows node and element IDs of multiple included files, upper.k, lower.k, binder.k, pins.k, and formed.dynain (formed sheet blank), are offset by 1000000, 2000000, 3000000, 4000000, and 5000000, respectively. As stated above, this option does not allow for the offset of the IDs of the sheet blank with stress and strain information, so formed.dynain file should be included first using just the keyword *INCLUDE. Care should be taken that the offset IDs

defined for all the rigid tools do not overlap with those in formed.dynain and also do not overlap each other.

```
*INCLUDE
./formed.dynain
*INCLUDE_AUTO_OFFSET_USER
./upper.k
1000000, 1000000
*INCLUDE_AUTO_OFFSET_USER
./lower.k
2000000, 2000000
*INCLUDE_AUTO_OFFSET_USER
./binder.k
3000000, 3000000
*INCLUDE_AUTO_OFFSET_USER
./pins.k
4000000, 4000000
```

Revision Information:

- This feature is available in SMP and MPP in LS-DYNA Revision 92417. Support for shell elements only.
- Revision 123467 extends to beams (used to model draw beads, for example) and solids.
- Revision 122115: extends to *ELEMENT_DISCRETE.
- Revision 123467: the option USER is available.

*INCLUDE_COMPENSATION

Purpose: This group of keywords allows for the inclusion of stamping tool and blank information for springback compensation. In addition, trim curves from the target geometry can be included for mapping onto the intermediate compensated tool geometry, which can be used for the next compensation iteration. Furthermore, compensation can be done for a localized tool region.

Basic options to perform springback compensation include the following:

*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK

*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK

*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE

*INCLUDE_COMPENSATION_CURRENT_TOOLS

*INCLUDE_COMPENSATION_ORIGINAL_TOOL

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP

If springback prediction is performed after trimming, the trimming curves need to be modified to account for the blank's boundary changes due to springback. The keyword to do that is:

*INCLUDE_COMPENSATION_TRIM_CURVE

Compensation can be performed locally with

*INCLUDE_COMPENSATION_CURVE

To reduce the number of iterations in springback compensation, the following keywords must be used with *INTERFACE_COMPENSATION_3D_ACCELERATOR:

*INCLUDE_COMPENSATION_ORIGINAL_DYNAIN

*INCLUDE_COMPENSATION_SPRINGBACK_INPUT

If the part has a symmetry condition, this keyword ensures that the compensated rigid tools are also symmetric:

*INCLUDE_COMPENSATION_SYMMETRIC_LINES

If there are regions with undesired mesh quality, the surface can be smoothed locally with the following keyword together with *INTERFACE_COMPENSATION_3D_LOCAL_SMOOTH:

*INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL

After compensation, the default file name for the new rigid tool is rigid.new. However, you can change its name using

*INCLUDE_COMPENSATION_NEW_RIGID_TOOL

After compensation has been completed, if some minor part change is needed, the compensated tool may be modified without redoing the iteration using the following keywords and *INTERFACE_COMPENSATION_3D_PART_CHANGE:

*INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE

*INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL

During hot forming, due to the blank's thickness change, the gap between the blank and the rigid tools is nonuniform, leading to inhomogeneous cooling. The following keywords are used with *INTERFACE_THICKNESS_CHANGE_COMPENSATION to modify the tool surface and ensure a homogenous gap:

*INCLUDE_COMPENSATION_BEFORE_SPRINGBACK

*INCLUDE_COMPENSATION_CURRENT_TOOLS

To ensure tangency continuity in the neighborhood of a trim curve, the following keyword can be used. These nodes can also be generated from *INTERFACE_COMPENSATION_3D_REFINE_RIGID.

*INCLUDE_COMPENSATION_TRIM_NODE

If you do not want to change the tangency along the punch boundary, the following keyword can be used:

*INCLUDE_COMPENSATION_TANGENT_CONSTRAINT

To specify node sets used during the compensation process, include:

*INCLUDE_COMPENSATION_NODE_SET

Full Example of a Springback Compensation:

The following example compensates a localized area, defined by the file curves.k. Trim lines are mapped onto the new, compensated rigid tool using trimcurves.k. Both files,

which were generated by LS-PrePost 4.0, are in the XYZ format. A detailed explanation of each keyword is given in the manual pages related to *INTERFACE_COMPENSATION_3D.

```
*KEYWORD
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*INTERFACE_COMPENSATION_3D
$   METHOD        SL        SF     ELREF      PSID    UNDRCT    ANGLE NLINEAR
         8    10.000     1.000         0         1         0      0.0       1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
blank0.k
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
spbk.k
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
reference0.k
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
reference1.k
*INCLUDE_COMPENSATION_CURRENT_TOOLS
tools.k
*INCLUDE_COMPENSATION_TRIM_CURVE
trimcurves.k
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ For compensation of a localize region only, add the following keyword:
*INCLUDE_COMPENSATION_CURVE
curves.k
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
*SET_PART_LIST
$      PSID
         1
$      PID
         3
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*END
```

*INCLUDE_COMPENSATION_BEFORE_SPRINGBACK

Purpose: In a hot forming process, the gap between the blank and rigid tools is not homogeneous due to the thickness change of the blank, which will cause inhomogeneity in cooling. This keyword, when used with *INTERFACE_THICKNESS_CHANGE_COMPENSATION, modifies the tool surface and ensures a homogeneous gap. The stamping tool set should be included with *INCLUDE_COMPENSATION_CURRENT_TOOLS. See *INTERFACE_THICKNESS_CHANGE_COMPENSATION for more details.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	A dynain file from a forming simulation; the varied, non-homogeneous thickness of the sheet blank will be used to change the uniform tool gap.

*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK

Purpose: Include the mesh information in keyword format for the last state (from d3plot) of the springback simulation or the dynain file after springback. This keyword is used with *INTERFACE_COMPENSATION_3D to compensate stamping tool shapes for springback with an iterative method.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	spbk.tmp							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information with adaptive constraints, if they exist, for the last state of the springback simulation.

Remarks:

Sheet blanks included using *INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK and *INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK must have the same number of nodes and elements for a compensation simulation.

*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK

Purpose: Include the mesh information in keyword format for the first state (from d3plot) of the springback simulation or the dynain file after trimming (before springback and with no mesh coarsening). This keyword is used with *INTERFACE_COMPENSATION_3D to compensate stamping tool shapes for springback with an iterative method.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	blank0.tmp							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information with adaptive constraints, if they exist, for the first state of the springback simulation

Remarks:

Sheet blanks included using *INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK and *INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK must have the same number of nodes and elements for a compensation simulation.

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE

Purpose: Include the mesh information in a dynain file for the compensated shape as a starting guess for a given compensation iteration. For the first compensation, use the target file. For the following compensation iterations, this file is obtained from disp.tmp, which is generated as an output file during the previous compensation iteration. This keyword is used with *INTERFACE_COMPENSATION_3D to compensate stamping tool shapes for springback with an iterative method.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	reference1.dat							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information with adaptive constraints, if they exist, for the first compensation iteration

Remarks:

Sheet blanks included using *INCLUDE_COMPENSATION_DESIRED_BLANK_-SHAPE and *INCLUDE_COMPENSATION_COMPENSATED_SHAPE must have the same number of nodes and elements for a compensation simulation.

*INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP

Purpose: Enable compensation of tools for the next die process. This keyword is used with *INTERFACE_COMPENSATION_3D_MULTI_STEPS.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information with adaptive constraints (if adaptive constraints exist for the next die process)

Remarks:

Sheet blanks included using *INCLUDE_COMPENSATION_DESIRED_BLANK_-SHAPE, *INCLUDE_COMPENSATION_COMPENSATED_SHAPE, and *INCLUDE_-COMPENSATION_COMPENSATED_NEXT_STEP must have the same number of nodes and elements for a compensation simulation:

*INCLUDE_COMPENSATION_CURRENT_TOOLS

Purpose: Include the tool mesh information in keyword format. This file is the tool mesh used for the current forming simulation. This keyword is used with *INTERFACE_COMPENSATION_3D to compensate stamping tool shapes for springback with an iterative method.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	rigid.tmp							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information for the last state of the springback simulation. If the file is named as rigid0.tmp, the elements of the tools get refined along the outline of the part. Draw bead nodes must be included in this file so that they will be modified together with the rigid tools.

*INCLUDE_COMPENSATION_CURVE

Purpose: Perform compensation on localized tooling areas by defining a compensation zone. This keyword allows for die face compensation of a local region in a stamping die.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of the keyword file containing x, y, z coordinates of two curves defining the compensation zone, using keywords: *DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN and *DEFINE_CURVE_COMPENSATION_CONSTRAINT_END.

*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE

Purpose: Include the dynain file for the blank after trimming or before springback in the first compensation iteration. This file is the target for compensation, so it never changes in all subsequent compensation iterations. This keyword is used with *INTERFACE_COMPENSATION_3D to compensate stamping tool shapes for springback with an iterative method.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	reference0.dat							

VARIABLE	DESCRIPTION
FILENAME	File that includes the nodes and element information with adaptive constraints, if they exist, for the blank after trimming in the first compensation iteration

Remarks:

Sheet blanks included using *INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE and *INCLUDE_COMPENSATION_COMPENSATED_SHAPE must have the same number of nodes and elements for a compensation simulation.

*INCLUDE_COMPENSATION_NEW_RIGID_TOOL

Purpose: Change the file name for the new rigid tool following compensation from rigid.new.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File name for new rigid tool

*INCLUDE_COMPENSATION_NODE_SET

Purpose: Specify node sets used during the compensation process.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of a keyword file that contains up to three node sets. One node set gives all the nodes that will be fixed during compensation. Another node set gives all the nodes that will be compensated. The third node set provides the fixed draft region where the nodes will mostly move in the *z*-direction and the slope of the geometry will be maintained. The order of these node sets is arbitrary and not all need to be included. The solver determines the kind of node set based on which elements the nodes belong to.

*INCLUDE_COMPENSATION_ORIGINAL_DYNAIN

Purpose: This keyword is to be used in conjunction with *INTERFACE_COMPENSATION_3D_ACCELERATOR and *INCLUDE_COMPENSATION_SPRINGBACK_INPUT for a springback compensation with a faster convergence rate and a simplified user interface. For detailed usage, please refer to the manual pages under *INTERFACE_COMPENSATION_3D_*{OPTION}*.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	The dynain file name that contains model information, adaptive constraints, and stress and strain tensor information.

*INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL

Purpose: Locally smooth mesh in regions with undesired mesh quality.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Keyword file that contains the meshes of the rigid tools

Remarks:

This keyword is used together with *INTERFACE_COMPENSATION_3D_LOCAL_-SMOOTH and *SET_NODE_LIST_SMOOTH to smooth local areas of distorted meshes of a tooling surface.

*INCLUDE_COMPENSATION_ORIGINAL_TOOL

Purpose: Obtain a smoother mesh for the addendum and binder region for the current compensation by using the original tool mesh (of better quality) instead of the last compensated tool mesh (maybe distorted). This method reduces the cumulative error in mesh extrapolation outside of the trim lines. Details can be found in the manual pages for *INTERFACE_COMPENSATION_3D.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Original tool (without any compensation) mesh file that contains node and element information in keyword format

*INCLUDE_COMPENSATION_SPRINGBACK_INPUT

Purpose: This keyword is to be used in conjunction with *INTERFACE_COMPENSATION_3D_ACCELERATOR and *INCLUDE_COMPENSATION_ORIGINAL_DYNAIN for a springback compensation with a faster convergence rate and a simplified user interface. For detailed usage, please refer to the manual pages under *INTERFACE_COMPENSATION_3D_*{OPTION}*.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File name of springback simulation input deck for the baseline iteration zero simulation.

*INCLUDE_COMPENSATION_SYMMETRIC_LINES

Purpose: Ensure the compensated rigid tools are symmetric if the part has a symmetry boundary condition. This keyword is used with *INTERFACE_COMPENSATION_3D for METHOD = 7 or 8.

Card 1	1	2	3	4	5	6	7	8
Variable	SYMID	SYMXY	X0	Y0				
Type	I	I	F	F				
Default	1	none	0.0	0.0				

VARIABLE	DESCRIPTION
SYMID	ID of the symmetry condition being defined
SYMXY	Symmetry boundary condition: EQ.1: Symmetrical about the y-axis EQ.2: Symmetrical about the x-axis
X0, Y0	Coordinates of a point on the symmetry plane

Example:

In a complete keyword input example below, part set ID 1 is being compensated with a symmetry boundary condition about the x-axis. The symmetry plane passes through a point with coordinates of $x = 101.5$ and $y = 0.0$.

```
*KEYWORD
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$*INTERFACE_COMPENSATION_3D
$ Method=8 changes the binder; Method=7 binder/P.O. no changes.
*INTERFACE_COMPENSATION_3D
$   METHOD        SL        SF     ELREF      PSID    UNDRCT     ANGLE   NLINEAR
         7    10.000     1.000         2         1         1       0.0         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
./state1.k
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
./state2.k
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
./state1.k
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
./state1.k
*INCLUDE_COMPENSATION_CURRENT_TOOLS
./currenttools.k
*INCLUDE_COMPENSATION_SYMMETRIC_LINES
```

```
$    SYMID     SYMXY        X0        Y0
         1         2     101.5       0.0
$ SYMXY=2: symmetric about X-axis
*SET_PART_LIST
$      PSID
         1
$      PID
         1
*END
```

*INCLUDE_COMPENSATION_TANGENT_CONSTRAINT

Purpose: Maintain tangency in the part boundary (along trim curves) between compensated and uncompensated areas of the tool. It also preserves the tangential transition of the rigid tool area at the trim curves to the tool's addendum. This keyword was replaced by the TANGENT variable under *INTERFACE_COMPENSATION_3D.

Card 1	1	2	3	4	5	6	7	8
Variable	UFLAG							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
UFLAG	Tangency flag: EQ.1: Maintain tangency between compensated and uncompensated parts of the tool and maintain tangential transition of the rigid tool area at the trim curves to the addendum part of the tool.

*INCLUDE_COMPENSATION_TRIM_CURVE

Purpose: Map trim curves of current tools onto the compensated tools. This keyword is used when springback predication is performed after trimming.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of the keyword file containing x, y, z coordinates as defined using keyword *DEFINE_CURVE_TRIM_3D (only TCTYPE = 0 or 1 is supported). This option is used to map the trim curve to the new, compensated tooling mesh for next iterative simulation.

Remarks:

If the trimming curve is in IGES format, a new file, geocur.trm, will be generated after trimming. The file contains XYZ data of the trim curves under the keyword *DEFINE_CURVE_TRIM_*{OPTIONS}*, which is used for the compensation run. Note that the variable TCTYPE in the keyword must be set to "0" (or "1") for the compensation. Length of lines everywhere in the compensated part are calculated according to springback amounts (including the die expansion factors, therefore no die expansion needs to be included in the NC machining of the compensated tooling). The manual pages for *INTERFACE_BLANKSIZE contain a procedure to convert IGES files to XYZ format in LS-PrePost.

*INCLUDE_COMPENSATION_TRIM_NODE

Purpose: Ensure tangency continuity in the neighborhood of a trim curve. These nodes can also be generated from *INTERFACE_COMPENSATION_3D_REFINE_RIGID.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	bndnd0.tmp							

VARIABLE	DESCRIPTION
FILENAME	File that contains a list of nodes (in type I10 per line) along part trim curves to ensure tangency during compensation. This file can be generated from *INTERFACE_COMPENSATION_3D_REFINE_RIGID.

*INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE

Purpose: After compensation has been done, modify the compensated tool without the need to re-do the iterations to compensate the tool. This keyword must be used with *INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL and *INTERFACE_COMPENSATION_3D_PART_CHANGE. Note that these options are intended only for small part changes that do not substantially affect springback. More details can be found in the manual pages for *INTERFACE_COMPENSATION_3D_PART_CHANGE.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	updatedpart.tmp							

VARIABLE	DESCRIPTION
FILENAME	Updated blank shape mesh that has been formed (or trimmed) to the shape based on the new tool geometry

*INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL

Purpose: After compensation has been done, modify the compensated tool without the need to re-do the iterations to compensate the tool. This keyword must be used with *INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE and *INTERFACE_COMPENSATION_3D_PART_CHANGE. Note that these options are intended only for small part changes that do not substantially affect springback. More details can be found in the manual pages for *INTERFACE_COMPENSATION_3D_PART_CHANGE.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	newrigid.tmp							

VARIABLE	DESCRIPTION
FILENAME	Updated rigid tool mesh

*INCLUDE_COSIM

Purpose: Specify an include file giving the coupling interface across two models in different scales (global and local) running on an MPI-based co-simulation. The input deck for each model should contain this keyword. The included file for each model gives the interface(s) for that model.

This feature requires MPP LS-DYNA version R13.1, R14.0, or newer.

The method invoked with this keyword differs from the method used with *INCLUDE_MULTISCALE in that it imposes strong coupling. In the weak coupling method, the large-scale model imposes kinematic constraints on the small-scale model at the coupling interface and drives its deformation. The failure of the large-scale representative beams is determined by the small-scale analysis. The strong coupling method invoked with this keyword, in contrast, is a fully concurrent simulation across two scales. The large-scale model imposes the kinematic constraints on the small-scale model and obtains the kinetic response in return. Therefore, this method does not need a simplified representation of the small-scale model in the large-scale model.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of a keyword input file that contains the coupling information, which includes sets of segments/solids in the global scale model and sets of nodes in the local scale one. See Remark 1 and the example below.

Remarks:

1. **Overview of multiscale coupling method.** This keyword is for specifying the coupling interface to exchange kinematic and kinetic information between global (large-scale) and local (small-scale) models running on two MPP jobs. The global model is usually a large-scale structure, such as a full car model or printed circuit board (PCB). The local model should be much smaller in dimension. For instance, it could be spot welds, rivets, or solders. The smaller

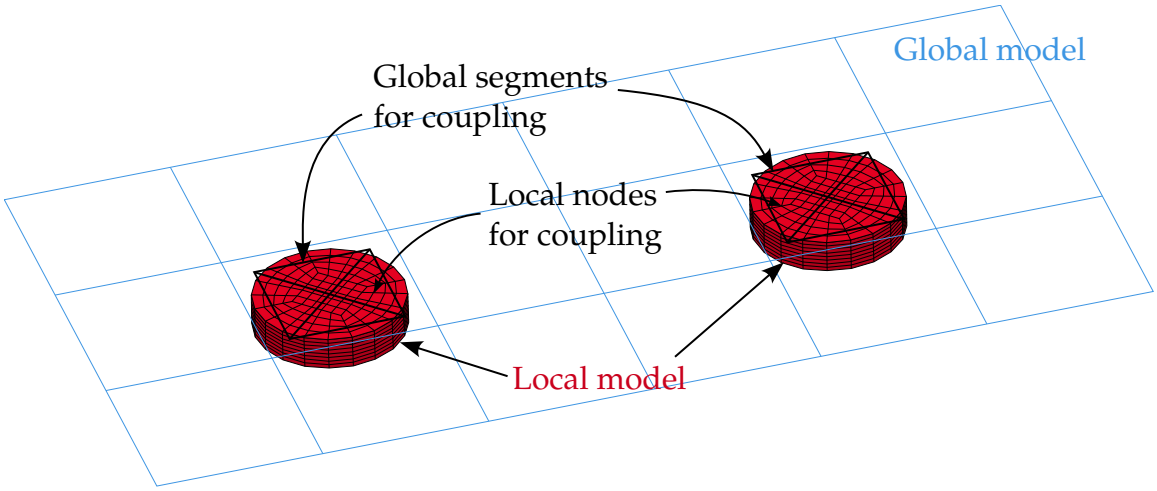


Figure 27-1. Example of tied contact coupling

dimension leads to a smaller mesh and time step size in explicit analysis. To accelerate the computation across global and local scales, you use two input files and run two MPP jobs simultaneously with different time step sizes. The synchronization of time-stepping information across the coupling interface is performed automatically at every time step of the global model. The time step size of the local model is adjusted accordingly to guarantee the numerical consistency and accuracy of state variables at the coupling interface in the spatial and temporal domains.

For the global model, a segment set (see *SET_SEGMENT) in the included file specifies the interface, except in the case of the solid-in-solid immersion coupling algorithm (see item c below). For solid-in-solid immersion, a solid set (see *SET_SOLID) defines the interface in the global model. For the local model, a node set (see *SET_NODE) determines the interface. These two sets must have the same set ID (SID) to indicate how to couple the global and local models. Note that the included files can contain multiple coupling interfaces (see the example below).

Three types of coupling algorithms are currently implemented:

a) *Tied contact*. With tied contact, the set of segments in the global model drives the motion of the set of nodes in the local model by imposing kinematic constraints on nodal translational DOFs. The local model returns the constraint forces interpolated onto the nodes of the set of global segments. To invoke this type of coupling, set the flag ITS to 1 for both the segment and node sets.

b) *Solid-in-shell immersion*. With solid-in-shell immersion, the local solid model occupies the same space as the global shell model at the coupling interface. The node set in the local model follows both the translational and rotational motion of the shell segment set in the global model, while the nodes of the

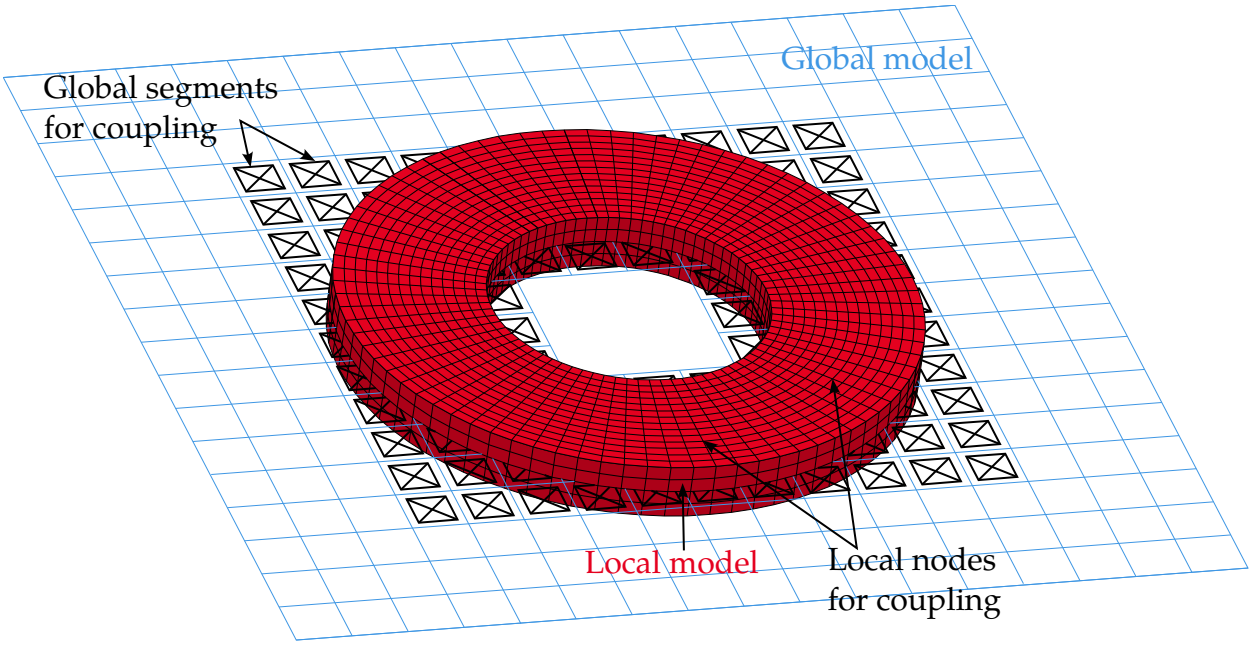


Figure 27-2. Example of solid-in-shell immersed coupling

global segments obtain the constraint forces and moments in return. To invoke this type of coupling, set the flag ITS to 2 for both the segment and node sets.

c) *Solid-in-solid immersion.* With solid-in-solid immersion, the local solid model occupies the same space as the global solid model at the coupling interface. The node set in the local model follows the translational motion of the solid set in the global model, while the nodes of the global solid elements obtain the constraint forces in return. In this case, set the flag ITS to 3 for both the solid and node sets.

2. **Running two MPP jobs.** The specifics of running two MPP jobs are installation-dependent. The following is an example of running two MPP jobs for two different scales:

```
mpirun -np 36 mppdyna i=input.k ncsp=24 jobid=jid
```

The "ncsp" flag informs the LS-DYNA software that the local model runs on a separate job using 24 out of the total 36 MPI processes. The main input file of the local model must have the same name as the global model with the addition of the prefix cs_. For example, the local model's main input file for the example above needs to be named cs_input.k. The main input files of both the global and local models must be in the same working folder. In practice, you can create an appfile file with the following lines:

```
-np 36 mppdyna i=input.k ncsp=24 jobid=jid
```

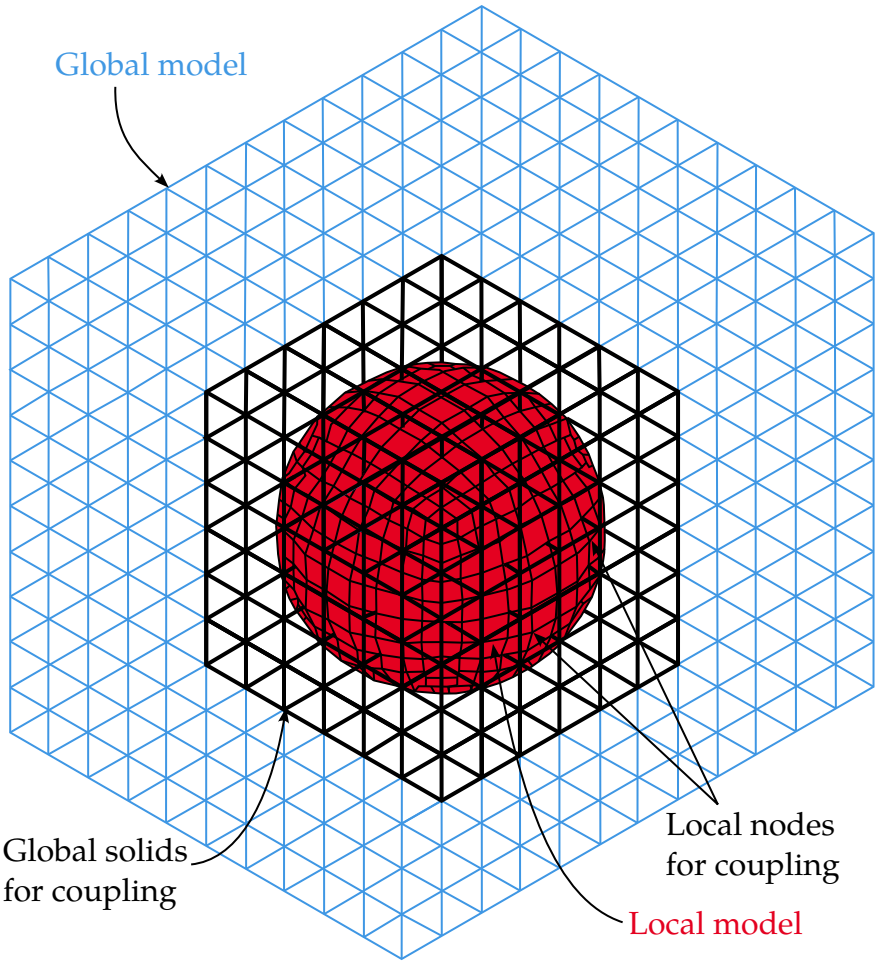


Figure 27-3. Example of solid-in-solid immersion coupling

Then, for Linux, run the problem with mpirun -f appfile. For Windows, use mpiexec -configfile appfile.

The message file contains a category called "Two-scale Cosim" at normal termination. This timing information under this category gives the CPU time spent on the data exchange between two jobs through MPI communication. For a good load balance between the two jobs, the number of MPI processes for the two jobs needs to be adjusted according to the time step size difference and other factors, such as the number of elements and contacts.

All the output and temporary files are separate for the two jobs. cs_ is prefixed to all the files of the local job run to avoid conflicts with the files from the global job.

3. **Termination time.** The global and local files must have the same termination time specified with *CONTROL_TERMINATION.

4. **d3plot output interval.** The time interval of d3plot output set in both input decks must be the same to have a time-consistent post-processing database.

Example:

```
input.k:

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$  * Main input file for the global model
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*KEYWORD
$
*CONTROL_TERMINATION
$   ENDTIM    ENDCYC     DTMIN    ENDENG    ENDMAS     NOSOL
      1e-4
$
*DATABASE_D3PLOT
      1e-6
$
*INCLUDE_COSIM
global_cosim.k
$
$$ Other keyword cards including control, mesh, contact, …
…
$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


global_cosim.k:

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$  * Define coupling interface in the global model
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*KEYWORD
$
*SET_SEGMENT
$      SID       DA1       DA2       DA3       DA4    SOLVER       ITS
        22       0.0       0.0       0.0       0.0MECH                 1
$       N1        N2        N3        N4        A1        A2        A3        A4
         8         9        12        11       0.0       0.0       0.0       0.0
        36        37        40        39       0.0       0.0       0.0       0.0
…
$
*SET_SEGMENT
$      SID       DA1       DA2       DA3       DA4    SOLVER       ITS
        11       0.0       0.0       0.0       0.0MECH                 1
$       N1        N2        N3        N4        A1        A2        A3        A4
        23        21        25        27       0.0       0.0       0.0       0.0
        47        45        49        51       0.0       0.0       0.0       0.0
…
$
*SET_SOLID
$#     SID    SOLVER
         3MECH                3
$#      K1        K2        K3        K4        K5        K6        K7        K8
       372       373       374       375       376       377       402       403
```

```
       404       405       406       407       432       433       434       435
…
$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

cs_input.k:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$  * Main input file for the local model
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*KEYWORD
$
*CONTROL_TERMINATION
$   ENDTIM    ENDCYC     DTMIN    ENDENG    ENDMAS     NOSOL
      1e-4
$
*DATABASE_D3PLOT
      1e-6
$
*INCLUDE_COSIM
local_cosim.k
$
$$ Other keyword cards including control, mesh, contact, …
…
$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

local_cosim.k:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$  * Define coupling interface in the local model
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
*KEYWORD
$
*SET_NODE_LIST
$      SID       DA1       DA2       DA3       DA4    SOLVER       ITS
        11       0.0       0.0       0.0       0.0MECH               1
$       N1        N2        N3        N4        N5        N6        N7        N8
         1         2         3         4         5         6         7         8
         9        10        11        12        13        14        15        16
…
$
*SET_NODE_LIST
$      SID       DA1       DA2       DA3       DA4    SOLVER       ITS
        22       0.0       0.0       0.0       0.0MECH               1
$       N1        N2        N3        N4        N5        N6        N7        N8
       929       930       931       932       933       934       935       936
       937       938       939       940       941       942       943       944
…
$
*SET_NODE_LIST
$      SID       DA1       DA2       DA3       DA4    SOLVER       ITS
         3       0.0       0.0       0.0       0.0MECH               3
$       N1        N2        N3        N4        N5        N6        N7        N8
      4250      4604      4508      4058      4106      4292      4171      4165
      4177      4183      4004      4381      4658      4322      4082      4164
```

...

```
$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*INCLUDE_ISPG

Purpose: Include a file that provides the model for the ISPG parts. These parts are used for simulating incompressible fluids and coupled with structural parts. For ease of defining ISPG models, these models use keywords for solids (*ELEMENT_SOLID, *NODE, *PART). However, the ISPG models are separate from the structural models (meaning different sets of numbering from structural models for the nodes and elements used by the ISPG models). Currently, only 4-noded tetrahedral elements can be used for the ISPG models.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Name of file containing the information about the ISPG models. Only one file can be included for each keyword ("*") card.

Remarks:

The following keywords can be defined in the ISPG model file included with this keyword:

*ISPG_ASCII_CPL

*ISPG_ASCII_SUMFORC

*ISPG_BOUNDARY_CONVECTION_SET

*ISPG_BOUNDARY_SYMMETRIC

*ISPG_CONTROL_ADAPTIVITY

*ISPG_CONTROL_D3ISPG

*ISPG_CONTROL_IMPLICIT

*ISPG_CONTROL_MPP

*ISPG_CONTROL_SOLUTION

*ISPG_DAMPING_SURF

*ISPG_DEFINE_MERGER_MPP

*ISPG_INITIAL_TEMPERATURE_SET

*ISPG_LOAD_GRAVITY

*MAT_ISPG_CARREAU

*MAT_ISPG_CROSS_CASTRO_MACOSKO

*MAT_ISPG_CROSSMODEL

*MAT_ISPG_ISO_NEWTONIAN

*MAT_THERMAL_ISPG

*SECTION_ISPG

*KEYWORD

*ELEMENT_SOLID

*NODE

*PART

*END

Note that *SECTION_FPD and *MAT_IFPD are not supported for ISPG models defined in this way.

*INCLUDE_MULTISCALE

Purpose: Provide a detailed local model for a coupled multiscale simulation. The local model is coupled to heuristic beams in the large-scale model. Failure of the heuristic beams is determined through this local model. This type of simulation allows including details necessary for failure without using an infeasible time step. See the Remarks section for how this type of simulation should be set up and performed. This keyword is an extension of *INCLUDE_MULTISCALE_SPOTWELD for a more general local model.

This capability is available *only* in the MPP version of LS-DYNA.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
ID	ID for this multiscale local model. This ID is used in the keyword *DEFINE_MULTISCALE. Any unique integer will do.
FILENAME	Name of file from which to read the local model definition.

Remarks:

With the multiscale feature, heuristic beams models are replaced with zoomed-in geometrically and constitutively correct continuum models, which, in turn are coupled to the large-scale calculation without reducing the time step. Because the detailed local models

are run in a separate process, they can run at a much smaller time step without slowing down the rest of the simulation. A brief outline of their use looks like this:

- Create one (or more) detailed local model(s) of their beams and include these definitions into the large-scale model using the keyword *INCLUDE_MULTISCALE. In the large-scale model, the beams must be on the microscale model side of a tied interface.
- Indicate which beams should be coupled to these models with *DEFINE_MULTISCALE.
- When MPP-DYNA is started, a special (MPI dependent) invocation is required to run in a "multiple program" mode. Effectively, two separate instances of MPP-DYNA are started together, one to run the full model and a separate instance to run the local models.
- As the macroscale process runs, each cycle it communicates to the microscale process deformation information for the area surrounding each coupled beam. The microscale process imposes this deformation on the detailed solid local models, computes a failure flag for each, and communicates this back to the macroscale process.
- The coupled beams in the macroscale process have their failure determined solely by these failure flags.

The file referred to on the *INCLUDE_MULTISCALE card should contain one generic instance of a detailed local model. For each coupled beam in the main model, a specific instance of this beam will be generated which is translated, rotated, and scaled to match the solid local model to which it is coupled. In this way, many beams can be coupled with only a single *INCLUDE_MULTISCALE. The included file should contain everything required to define a detailed local model, such as *MAT and *PART definitions, any required *DEFINE_CURVEs, etc., as well as *NODE and *ELEMENT definitions. For the translation and scaling to work properly, we make the following assumptions about the detailed local model:

- It consists entirely of solid elements.
- The z-axis is aligned with the coupled local model in the main model, with $z = 0$ and $z = 1$ at the two ends of the local model.
- The cross-sectional area of the local model in the xy-plane is equal to 1.
- That portion of the "top" and "bottom" of the local model that are coupled are identified using a single *SET_NODE_LIST card.
- One *BOUNDARY_COUPLED card referencing the *SET_NODE_LIST of the boundary nodes is required. It must specify a coupling type of 2 and a coupling program of 1.
- The detailed local model can include multiple parts.
- The detailed local model does not support *INCLUDE cards.

Failure of the local model is determined topologically. Any element of the detailed local model having all four nodes of one of its faces belonging to the *SET_NODE_LIST of tied nodes is classified as a "tied" element. The "tied" elements are partitioned into two disjoint sets: the "top," and "bottom". When there is no longer a complete path from any "top" to any "bottom" element (where a "path" passes through non-failed elements that share a common face), then this detailed model has failed. Note that this places some restrictions on the *SET_NODE_LIST and element geometry, namely that some "tied" elements exist, and the set of "tied" elements consists of exactly two disjoint subsets.

The specifics of launching a multi-program MPI program are installation-dependent. The idea behind running a coupled model is to run one set of MPI ranks as if running a normal MPP-DYNA job, such as:

```
mpirun -np 18 mppdyna i=input.k nmsp=12
```

The nmsp flag informs LS-DYNA that the local model runs in a separate job using 12 out of the total 18 MPI processors.

The main instance knows to look for the microscale model (because of the presence of the *INCLUDE_MULTISCALE card) and runs the main model. The "microscale" instance runs all the detailed local models. Due to the nature of the coupling, the main model cannot progress when the detailed local models are being processed, nor can the detailed local models run while the main model is being computed. From a processor efficiency standpoint, it therefore makes sense to run as many microscale processes as macroscale processes and run them on the same CPUs, so that each processing core has one microscale and one macroscale process running on it. But you don't have to – the processes are independent and you can have any number of either.

*INCLUDE_MULTISCALE_SPOTWELD

Purpose: Define a type of multiscale spot weld to be used for coupling and for modeling of spot weld failure.

Card 1	1	2	3	4	5	6	7	8
Variable	TYPE							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
TYPE	TYPE for this multiscale spot weld. This type is used in the keyword *DEFINE_SPOTWELD_MULTISCALE. Any unique integer will do.
FILENAME	Name of file from which to read the spot weld definition.

Remarks:

This capability is available *only* in the MPP version of LS-DYNA.

With the multiscale spot weld feature, heuristic spot weld models are replaced with zoomed-in geometrically and constitutively correct continuum models, which, in turn are coupled to the large-scale calculation without reducing the time step. In some respects, multiscale models are similar to the "hex spot weld assemblies," capability but more general in terms of their geometry. Because the spot weld models are run in a separate process, they can run at a much smaller time step without slowing down the rest of the simulation. A brief outline of their use looks like this:

- You create one (or more) detailed models of your spot welds and include these definitions into your model using the keyword *INCLUDE_MULTISCALE_-SPOTWELD. In the large-scale model, the spot weld must be on the tracked (SURFA) side of a tied interface.
- You indicate which beam (or hex assembly) spot welds should be coupled to these models with the keyword *DEFINE_SPOTWELD_MULTISCALE
- When MPP-DYNA is started, a special (MPI dependent) invocation is required in order to run in a "multiple program" mode. Effectively, two separate instances of MPP-DYNA are started together, one to run the full model (macroscale) and a separate instance to run the spot welds (microscale).
- As the macroscale process runs, each cycle it communicates deformation information for the area surrounding each coupled spot weld to the microscale process. The microscale process imposes this deformation on the detailed spot welds, computes a failure flag for each, and communicates this back to the macroscale process.
- The coupled spot welds in the macroscale process have their failure determined solely by these failure flags.

The file referred to on the *INCLUDE_MULTISCALE_SPOTWELD card should contain one generic instance of a detailed spot weld. For each coupled spot weld in the main model, a specific instance of this spot weld will be generated which is translated, rotated, and scaled to match the spot weld to which it is coupled. In this way, many spot welds can be coupled with only a single *INCLUDE_MULTISCALE_SPOTWELD. The included file should contain everything required to define the spot weld, such as *MAT and *PART definitions, any required *DEFINE_CURVEs, etc., as well as *NODE and *ELEMENT definitions. For the translation and scaling to work properly, we assume the following about the spot weld model:

- It consists entirely of solid elements.
- The z-axis is aligned with the coupled spot weld in the main model, with $z = 0$ and $z = 1$ at the two ends of the spot weld.
- The cross-sectional area of the spot weld in the xy-plane is equal to 1.
- The portion of the "top" and "bottom" of the spot weld that are coupled are identified using a single *SET_NODE_LIST card.
- One *BOUNDARY_COUPLED card referencing the *SET_NODE_LIST of the boundary nodes is required. It must specify a coupling type of 2 and a coupling program of 1.
- The spot weld model does not support *INCLUDE cards.

Failure of the microscale model is determined topologically. Any element of the spot weld having all four nodes of one of its faces belonging to the *SET_NODE_LIST of tied nodes is classified as a "tied" element. The "tied" elements are partitioned into two

disjoint sets: "top" and "bottom". When there is no longer a complete path from any "top" to any "bottom" element (where a "path" passes through non-failed elements that share a common face), then the spot weld has failed. Note that this places some restrictions on the *SET_NODE_LIST and element geometry, namely that some "tied" elements exist, and the set of "tied" elements consists of exactly two disjoint subsets.

The specifics of launching a multi-program MPI program are installation dependent. But the idea behind running a coupled model is that you want to run one set of MPI ranks as if you were running a normal MPP-DYNA job, such as:

mpirun –np 4 mppdyna i=**input.k** memory=**200m** p=**pfile**

and a second set with just the command line argument "microscale" (no input file):

mpirun –np 4 mppdyna microscale memory=**100m** p=**pfile**

The main instance knows to look for the microscale instance (because of the presence of the *INCLUDE_MULTISCALE_SPOTWELD card) and will run the main model. The "microscale" instance will run all the detailed spot weld models. Due to the nature of the coupling, the main model cannot progress when the detailed spot welds are being processed, nor can the detailed spot welds run while the main model is being computed. From a processor efficiency standpoint, it therefore makes sense to run as many microscale processes as macroscale processes, and run them on the same CPUs, so that each processing core has one microscale and one macroscale process running on it. But you don't have to: the processes are independent, and you can have any number of either.

***INCLUDE_PATH**_{OPTION}

Available options include:

<BLANK>

RELATIVE

Purpose: Define directories in which to look for include files. LS-DYNA's default behavior is to search for files in the local directory first. If an include file is not found and the filename has no path, LS-DYNA will search for it in all the directories defined by *INCLUDE_PATH.

When the RELATIVE option is used, all directories are relative to the location of the input file. For example, if "i=/home/test/problems/input.k" is given on the command line, and the input contains

```
*INCLUDE_PATH_RELATIVE
includes
../includes
```

then the two directories /home/test/problems/includes and /home/test/includes will be searched for include files.

Pathname Card. Directory paths are read until the next keyword ("*") card is encountered. A directory path can have up to 236 characters (see Remark 2).

Card 1	1	2	3	4	5	6	7	8
Variable	PATHNAME							
Type	C							

VARIABLE	DESCRIPTION
PATHNAME	Directory path. If the RELATIVE keyword option is used, this directory is relative to the input file.

Remarks:

1. **Pathname Length Limitations.** Pathnames are limited to 236 characters spread over up to three 80 character lines. When 2 or 3 lines are needed to specify the pathname, end the preceding line with "␣+" (space followed by a plus sign) to signal that a continuation line follows. Note that the "␣+" combination is, itself,

part of the 80 character line; hence the maximum number of allowed characters is 78 + 78 + 80 = 236.

INCLUDE_STAMPED_*OPTION1_{OPTION2}_{OPTION3}_{OPTION4}

Purpose: Map the plastic strain and thickness distribution of a shell part or part set from a stamping simulation (source part or part) to a shell part or part set in a crash model (target part or part set). This keyword only applies to shell elements. As of R15, it can also map stamping information from a standard finite element model to IGA shell (*IGA_SHELL) parts.

Available options for *OPTION1* include:

PART

SET

For PART a stamped part is mapped while for SET a stamped part set is mapped.

OPTION2 only applies when *OPTION1* is PART:

<BLANK>

SET

If SET is used, PID will be a part set ID. All the parts included in this set will be considered in this mapping. Note that *INCLUDE_STAMPED_PART_SET and *INCLUDE_STAMPED_SET are equivalent keyword names, meaning they call the same subroutines in LS-DYNA.

The available options for *OPTION3* are:

<BLANK>

MATRIX

If MATRIX is used, LS-DYNA directly reads the transformation matrix with a translation vector instead of using orientation nodes to create the transformation.

The available options for *OPTION4* are:

<BLANK>

INVERSE

INVERSE can only be included if using the MATRIX keyword option. If used, LS-DYNA inverts the matrix. This keyword option is useful for the case where you have the transformation from the crash part to the stamped part.

When using STAMPED_SET or STAMPED_PART_SET, the target is a part set. Between the stamped part and the crash part, note the following points:

1. The outer boundaries of the parts do not need to match because only the regions of the crash part that overlap the stamped part are initialized.
2. Arbitrary mesh patterns are assumed.
3. Element formulations can change.
4. Unless using the MATRIX keyword option, three nodes on each part are used to reorient the stamped part for the mapping of the strain and thickness distributions. After reorientation, the three nodes on each part should approximately coincide. Otherwise, directly input the transformation.
5. The number of in-plane integration points can change.
6. The number of through-thickness integration points can change. Full interpolation is used.
7. The node and element IDs between the stamped part and the crash part do not need to be unique.

Card Summary:

Card 1. This card is required.

FILENAME

Card 2. This card is required.

PID	THICK	PSTRN	STRAIN	STRESS	INCOUT		RMAX

Card 3a. Include this card if *not* using the MATRIX keyword option.

N1S	N2S	N3S	N1C	N2C	N3C	TENSOR	THKSCL

Card 3b.1. Include this card if using the MATRIX keyword option.

R11	R12	R13	XP	TENSOR	THKSCL		

Card 3b.2. Include this card if using the MATRIX keyword option.

R21	R22	R23	YP				

Card 3b.3. Include this card if using the MATRIX keyword option.

R31	R32	R33	ZP				

Card 4. This card is optional.

ISYM	IAFTER	PERCELE	IORTHO		ISRCOUT	MTYPE	

Card 5. This card is optional.

X01	Y01	Z01					

Card 6. This card is optional.

X02	Y02	Z02	X03	Y03	Z03		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	Name of file to be included; see Remark 2. This is the dynain file containing the metal stamping result. For *INCLUDE_STAMPED_PART, the file can be in either ASCII or LSDA format. For all other options, it must be ASCII.

Card 2	1	2	3	4	5	6	7	8
Variable	PID	THICK	PSTRN	STRAIN	STRESS	INCOUT		RMAX
Type	I	I	I	I	I	I		F
Default	none	0	0	0	0	0		20.0

VARIABLE	DESCRIPTION
PID	Part or part set ID of crash part/parts for remapping
THICK	Thickness remap: EQ.0: Map thickness EQ.1: Do not map thickness EQ.2: Average value inside a circle defined by RMAX
PSTRN	Plastic strain remap: EQ.0: Map plastic strain EQ.1: Do not map plastic strain EQ.2: Average value inside a circle defined by RMAX
STRAIN	Strain remap: EQ.0: Map strains EQ.1: Do not map strains
STRESS	Stress tensor remap: EQ.0: Map stress tensor and history variables EQ.1: Do not map stress tensor, only history variables EQ.2: Do not map stress tensor or history variables EQ.-1: Map stress tensor in an internal large format (binary files) EQ.-3: Do not map stress tensor in an internal large format, only history variables (binary files)
INCOUT	Save mapped data: EQ.1: Save the mapped data for the part/part set (PID) to a file called dyna.inc. This option is useful for when the mapped data may be required in a future simulation. EQ.2: Save the mapped data for the specified part or part set (PID) to a file called dynain_xx (xx is the part or part set ID). EQ.3: Save the mapped data for the specified part or part set (PID) to a file called nastran_xx (in nastran format). xx is the part or part set ID. EQ.4: Same as 2, but treated as an "only mapping" run, meaning

VARIABLE	DESCRIPTION
	the calculation stops right after the data is mapped. Note that INCOUT does *not* account for changes in the number of in-plane or through-thickness integration points.
RMAX	Search radius. LS-DYNA remaps history variables from the mesh of the stamped part to the mesh of the crash part with a spatial tolerance of RMAX. If an element in the crash part lies within RMAX of the stamped part, data will be mapped to that element. If set less than 0.001, RMAX automatically assumes the default value of 20.

Nodal Orientation Card. Include this card if the MATRIX option is *not* used.

Card 3a	1	2	3	4	5	6	7	8
Variable	N1S	N2S	N3S	N1C	N2C	N3C	TENSOR	THKSCL
Type	I	I	I	I	I	I	I	F
Default	0	0	0	0	0	0	0	1.0
Remarks	1	1	1	1	1	1	3	

VARIABLE	DESCRIPTION
N1S	First of 3 nodes needed to reorient the stamped part or part set
N2S	Second of 3 nodes needed to reorient the stamped part or part set
N3S	Third of 3 nodes needed to reorient the stamped part or part set
N1C	First of 3 nodes needed to reorient the crash model part or part set
N2C	Second of 3 nodes needed to reorient the crash model part or part set
N3C	Third of 3 nodes needed to reorient the crash model part or part set

VARIABLE	DESCRIPTION
TENSOR	Tensor remap: EQ.0: Map tensor data from history variables. EQ.1: Do not map tensor data from history variables
THKSCL	Thickness scale factor

Transformation Matrix Card 1. Include this card for the MATRIX keyword option.

Card 3b.1	1	2	3	4	5	6	7	8
Variable	R11	R12	R13	XP	TENSOR	THKSCL		
Type	F	F	F	F	I	F		
Default	0	0	0	0	0	1.0		

Transformation Matrix Card 2. Include this card for the MATRIX keyword option.

Card 3b.2	1	2	3	4	5	6	7	8
Variable	R21	R22	R23	YP				
Type	F	F	F	F				
Default	0	0	0	0				

Transformation Matrix Card 3. Include this card for the MATRIX keyword option.

Card 3b.3	1	2	3	4	5	6	7	8
Variable	R31	R32	R33	ZP				
Type	F	F	F	F				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
R*ij*	Components of the transformation matrix (see Remark 1)
XP, YP, ZP	Translational distance (see Remark 1)
TENSOR	Tensor remap: EQ.0: Map tensor data from history variables. EQ.1: Do not map tensor data from history variables
THKSCL	Thickness scale factor

This card is optional.

Card 4	1	2	3	4	5	6	7	8
Variable	ISYM	IAFTER	PERCELE	IORTHO		ISRCOUT	MTYPE	
Type	I	I	F	I		I	I	

VARIABLE	DESCRIPTION
ISYM	Symmetric switch EQ.0: No symmetric mapping EQ.1: *yz* plane symmetric mapping EQ.2: *zx* plane symmetric mapping EQ.3: *zx* and *yz* planes symmetric mapping EQ.4: User defined symmetric plane mapping
IAFTER	Mirroring sequence switch EQ.0: Generate a symmetric part before transformation EQ.1: Generate a symmetric part after transformation
PERCELE	Percentage of elements that should be mapped for the simulation to proceed (default = 0); otherwise an error termination occurs. See Remark 5.
IORTHO	Location of the material direction cosine in the array of history variables of an orthotropic material. See Remark 4.

VARIABLE	DESCRIPTION
ISRCOUT	Optional output of stamped part after transformation(s) EQ.0: No output is written. NE.0: Keyword output file "srcmsh_<ISRCOUT>" is created.
MTYPE	Expected material model. If given, the target part must have the material model given by MTYPE. Otherwise, an error termination occurs. Use the material number for MTYPE. This feature is helpful for detecting an unintended replacement of the material model when mapping history variables.

This card is optional.

Card 5	1	2	3	4	5	6	7	8
Variable	X01	Y01	Z01					
Type	F	F	F					

This card is optional.

Card 6	1	2	3	4	5	6	7	8
Variable	X02	Y02	Z02	X03	Y03	Z03		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
X01, Y01, Z01	First point in the symmetric plane (required if ISYM ≠ 0)
X02, Y02, Z02	Second point in the symmetric plane
X03, Y03, Z03	Third point in the symmetric plane

Remarks:

1. **Reorienting the result of a stamping simulation.** The target mesh must be read in before including the stamped part with this keyword.

If the MATRIX keyword is not used, N1S, N2S, N3S, N1C, N2C, and N3C are used for transforming the stamped part to the crash part, such that it is in the same position as the crash part. If the stamped part is in the same position as the crash part, then N1S, N2S, N3S, N1C, N2C, and N3C can all be set to 0. Note that if these 6 nodes are input as 0, LS-DYNA will not transform the stamped part. With the MATRIX keyword option, a transformation matrix with a translation vector reorients the part instead of nodes.

When symmetric mapping is used (ISYM is nonzero), the three points should not be in one line. If ISYM = 1, 2, or 3, only the first point, namely (X01, Y01, Z01), is needed. If ISYM = 4, all three points are needed.

2. **File name length limitations.** Filenames are limited to 236 characters spread over up to three 80-character lines. When 2 or 3 lines are needed to specify the filename or pathname, end the preceding line with "␣+" (space followed by a plus sign) to signal that a continuation line follows. Note that the "␣+" combination is, itself, part of the 80-character line; hence the maximum number of allowed characters is 78 + 78 + 80 = 236.

3. **Mapping material data for springback.** Certain material models (notably Material 190) have tensor data stored within the history variables. Within material subroutines this data is typically stored in element local coordinate systems. In order to properly map this information between models it is necessary to have the tensor data present on the *INITIAL_STRESS_SHELL card and have it stored in global coordinates. During mapping the data is then converted into the local coordinate system of the crash mesh. This data can be dumped into the dynain file that is created at termination time if the parameter FTENSR is set to 0 on the *INTERFACE_SPRINGBACK_LSDYNA card. Currently, the only material model that supports the mapping of element history tensor data is Material 190.

4. **IORTHO.** If IORTHO is set, correct mapping between non-matching meshes is invoked for the directions of orthotropic materials. A list of appropriate values for several materials is given here:

   IORTHO.EQ.1: materials 23, 122, 157, 234

   IORTHO.EQ.3: materials 22, 33, 36, 133, 189, 233, 243

   IORTHO.EQ.4: material 59

   IORTHO.EQ.6: materials 58, 104, 158

   IORTHO.EQ.8: materials 54, 55

   IORTHO.EQ.9: material 39

   IORTHO.EQ.10: material 82

IORTHO.EQ.13: materials 2, 86, 103

5. **Mapping mismatch.** Sometimes, during mapping, the two meshes (stamping mesh and crash mesh) do not match exactly, and therefore, not all elements of the new mesh obtain results from the old mesh. The message file contains the total number of crash elements that are or are not mapped. By default (PERCELE = 0), the calculation continues even when no elements are mapped. PERCELE > 0 allows you to define the minimum percentage of required mapped elements for the calculation to proceed. If a percentage less than PERCELE is mapped, the calculation stops with an error termination.

*INCLUDE_STAMPED_PART_SOLID_TO_SOLID_{OPTION}

The available options are:

<BLANK>

GENERAL

SET_LIST

Purpose: Map the final stress and strain tensors, history variables, and plastic strain of a solid part in one model (source part) to the corresponding solid part in a second model (target part). The total thickness of the target part is adjusted as necessary to match the final thickness of the source part. This feature was designed for the situation where the source part is a thick blank from a forming analysis and the target part is in a vehicle crash model. The mesh refinements of the two parts may differ. Differences in the spatial location of the target part and source part are allowed (see fields on Card 3). This keyword has several limitations which are discussed in the next paragraph and in Remark 2.

The default mapping uses the top surface to find the matching elements in each layer. In this case, it requires the solid model to be extruded from a shell part and the part to have the same number of layers in each location. Advantageously, this method can find the corresponding layer for mapping. Thus, this mapping feature applies only to 8-noded hexahedral and 6-noded pentahedral elements because it requires consistent, through-thickness orientation of solid element normals. When the solid part is not extruded, the default algorithm does not work. The keyword option GENERAL invokes an algorithm that finds the closest element in the source and maps the information. Thus, the part does not have to have the same number of layers in each location and can include tetrahedral elements. However, this approach might not find the right layer in the mapping.

The SET_LIST keyword option enables providing a list of parts as the target instead of a single part. This list is provided with Card 6. In this case, all the parts in the included file are considered for mapping to the target.

INCLUDE_STAMPED_PART_SOLID_TO_SOLID_*{OPTION} *must* be the final keyword in the input deck.

Card Summary:

Card 1. This card is required.

FILENAME

Card 2. This card is required.

PID	THICK	PSTRN	STRAIN	STRESS			

Card 3. This card is required.

N1SORC	N2SORC	N3SORC	N1TRGT	N2TRGT	N3TRGT		

Card 4. Include this blank card if using the SET_LIST keyword option.

Card 5. Include this blank card if using the SET_LIST keyword option.

Card 6. Include as many instantiations of this card as desired if using the SET_LIST keyword option. The next keyword ("*") card terminates this input.

PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	File name of the dynain file created by *INTERFACE_SPRINGBACK_LSDYNA in the source model.

Card 2	1	2	3	4	5	6	7	8
Variable	PID	THICK	PSTRN	STRAIN	STRESS			
Type	I	I	I	I	I			
Default	none	0	0	0	0			

VARIABLE	DESCRIPTION
PID	Part ID of the target part onto which the source model's results in FILENAME will be mapped. LT.0: Part ID of the target part is |PID| and the normals of the target part are flipped before mapping. A negative PID would be used if the target part's normals were oriented exactly opposite those of the source part. For the SET_LIST keyword option, this field is optional.
THICK	Flag to map part thickness. The thickness direction is determined from the element normals, hence the need for consistency in the element normals. The thicknesses of the target part elements are adjusted so the target part thickness matches the source part thickness at any given location. Currently, this variable is hardwired so that thickness mapping is always on.
PSTRN	Flag to map effective plastic strain. Currently setting this flag with any integer will map the effective plastic strain, and there is no other option. EQ.0: Map effective plastic strain. NE.0: Do not map effective plastic strain.
STRAIN	Flag to map the strain tensor. Currently setting this flag with any integer will map the tensorial strains, and there is no other option available. Note "STRFLG" in *DATABASE_EXTENT_BINARY must be set to "1" for output to d3plot as well as dynain files. EQ.0: Map strain tensor. NE.0: Do not map strain tensor.
STRESS	Flag to map stress tensor. Currently setting this flag with any integer will map the stresses and history variables, and there is no other option available. Only the history variables included in the dynain file specified by FILENAME are mapped; see "NSHV" in *INTERFACE_SPRINGBACK_LSDYNA for control of history variable output to dynain. EQ.0: Map stress tensor and history variables. NE.0: Do not map stress tensor and history variables.

Card 3	1	2	3	4	5	6	7	8
Variable	N1SORC	N2SORC	N3SORC	N1TRGT	N2TRGT	N3TRGT		
Type	I	I	I	I	I	I		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
N1SORC	First of 3 nodes needed to reorient the source part. No transformation if undefined.
N2SORC	Second of 3 nodes needed to reorient the source part. No transformation if undefined.
N3SORC	Third of 3 nodes needed to reorient the source part. No transformation if undefined.
N1TRGT	First of 3 nodes needed to reorient the target part. No transformation if undefined.
N2TRGT	Second of 3 nodes needed to reorient the target part. No transformation if undefined.
N3TRGT	Third of 3 nodes needed to reorient the target part. No transformation if undefined.

Include this blank card for the SET_LIST keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable								
Type								
Default								

Include this blank card for the SET_LIST keyword option.

Card 5	1	2	3	4	5	6	7	8
Variable								
Type								
Default								

Part List Card. Include as many instantiations of this card as desired if using the SET_-LIST keyword option. The next keyword ("*") card terminates this input.

Card 6	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
PID*i*	Part ID for the i^{th} target part

Remarks:

1. **Element normals.** Unless using the GENERAL keyword option, the solid element normals in the source model and the target model part must be consistent with each other and in the through-thickness direction of the part. The first three nodes of the solid determine the bottom face and the element normal is the normal to that face using the right-hand rule. Solid element normals can be displayed and modified using *EleTol* → *Normal* → *Solid* in LS-PrePost.

2. **Geometry limitations.** This keyword was designed with a particular application in mind: mapping results from a forming analysis of a thick blank. It is not designed to handle situations where the target part's geometry is dissimilar from the source part's geometry, such as where the target part has a hole but the source part does not. See *INITIAL_LAG_MAPPING for a more general solid-to-solid mapping capability.

Example:

The following keyword example maps the stress and strain tensors, history variables, and effective plastic strain from a stamped (source) solid blank mesh drawn.dynain with PID 1, onto a coarser mesh (target) s1.k with PID 2.

The mapping may be checked from the dynain file created after a single time step (note that the termination time is set to 0.0 for this purpose).

```
*KEYWORD
*PARAMETER_EXPRESSION
I elform           2
*INTERFACE_SPRINGBACK_LSDYNA
1,100
*DATABASE_EXTENT_BINARY
$#   NEIPH     NEIPS    MAXINT    STRFLG    SIGFLG    EPSFLG    RLTFLG    ENGFLG
       100       100         8         1         1         1         1         1
$#  CMPFLG    IEVERP    BEAMIP     DCOMP      SHGE     STSSZ    N3THDT   IALEMAT
         0         0         0         1         1         1         2         1
$# NINTSLD   PKP_SEN      SCLP     HYDRO     MSSCL     THERM    INTOUT    NODOUT
         0         0       1.0         0         0         0
$#    DTDT    RESPLT     NEIPB     QUADR     CUBIC
         0         0         0         0         0
*SET_PART
1
1
*CONTROL_TERMINATION
     0.000
*DATABASE_BINARY_D3PLOT
      0.50
$---------------------------------------------------------------------
*PART
PID 1 is of a source mesh (e.g. stamping dynain file)
$       PID       SID       MID
         1         1         1
PID 2 is a target mesh (no stress/strains, with normals consistently aligned)
         2         1         1
$---------------------------------------------------------------------
*MAT_PIECEWISE_LINEAR_PLASTICITY
$       MID        RO         E        PR      SIGY      ETAN      FAIL      TDEL
         1  7.83E-09  2.07E+05     0.333
$         C         P      LCSS      LCSR        VP
       0.0       0.0        11         0       0.0
$      EPS1      EPS2      EPS3      EPS4      EPS5      EPS6      EPS7      EPS8

$       ES1       ES2       ES3       ES4       ES5       ES6       ES7       ES8

*DEFINE_CURVE
        11
    0.0000000000E+00    3.8000500000E+02
    2.0000000000E-03    3.8854500000E+02
    1.5000000000E+00    8.3929800000E+02
$---------------------------------------------------------------------
*SECTION_SOLID
$    SECID    ELFORM      SHRF       NIP     PROPT   QR/IRID     ICOMP
         1   &elform
$---------------------------------------------------------------------
*INCLUDE
$ coarser solid mesh file with no stress/strain:
s1.k
*INCLUDE_STAMPED_PART_SOLID_TO_SOLID
$ dynain file name from stamping simulation (source); native file should be used.
```

```
../drawn.dynain
$       PID     THICK     PSTRN    STRAIN    STRESS
$  PID here is PID of the target mesh
        -2         0         0         0         0
$   N1SORC    N2SORC    N3SORC    N1TRGT    N2TRGT    N3TRGT

*END
```

*INCLUDE_TRIM

Purpose: The naive procedure to trim a part imported using the *INCLUDE keyword involves: (1) including the part using *INCLUDE; (2) designating the part for trimming using *CONTROL_FORMING_TRIMMING; and (3) defining trim curves using *DEFINE_TRIM_CURVE. By replacing *INCLUDE in step (1) with this keyword, namely, *INCLUDE_TRIM, LS-DYNA is directed to filter the included part while reading in the include file (as opposed to after reading in), thereby saving memory that would otherwise be allocated to store the (immediately) discarded data. This feature has been developed in conjunction with the *Ford Motor Company Research and Advanced Engineering Laboratory*.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							

VARIABLE	DESCRIPTION
FILENAME	File name of the part to be trimmed

Remarks:

Similar to the trimming method with *INCLUDE, the name of the file to be trimmed must be put into the FILENAME field. Unlike the *INCLUDE method, all information about the included part (such as stress and strain) should be in the named file (as opposed to split across several files). *INCLUDE_TRIM *cannot* be used unless the included file contains stress and strain information. Namely, the dynain file must contain *INITIAL_STRESS_SHELL and *INITIAL_STRAIN_SHELL cards; otherwise the *INCLUDE keyword should be used instead.

For solid element trimming, the keyword *INCLUDE_TRIM *must* be used. In other words, the dynain file from a previous process (for example, a forming simulation) must be included in a main trim file using *INCLUDE_TRIM (not *INCLUDE). Refer to the *DEFINE_CURVE_TRIM manual entry for more details.

For example, a drawn panel from a previous simulation can be included in a current simulation for trimming as follows,

```
*INCLUDE_TRIM
Drawnpanel.dynain
*CONTROL_FORMING_TRIMMING
      ⋮
*DEFINE_CURVE_TRIM_3D
```

```
        ⋮
*CONTROL_ADAPTIVE_CURVE
        ⋮
```

Performance Improvements:

Referring to the table below (parts courtesy of the *Ford Motor Company*), compared with *INCLUDE, the *INCLUDE_TRIM keyword reduces memory requirements by more than 50%. Levels of CPU time reductions exceed in some cases 50%.

Performance Improvements

		Roof	Hood Inr	B-Plr	Fender	BSA Otr	Door Otr	Wheel House (2 in 1)	Boxside Otr
#Element		410810	1021171	351007	189936	380988	315556	261702	1908369
CPU	old	7m26s	10m20s	3m11s	2m6s	5m45s	4m27s	2m52s	27m31s
	new	4m	9m18s	2m56s	1m22s	4m54s	3m35s	2m30s	13m59s
Memory (MW)	old	282	616	221	119	233	217	157	1150
	new	112	383	117	50	130	114	75	539

Revision Information:

This feature is available in LS-DYNA Revision 62207 or later releases, where the output of strain tensors for the shells is included. Prior Revisions do not include strain tensors for the shells.

*INCLUDE_UNITCELL

Purpose: This card creates a unit cell model with periodic boundary conditions using *CONSTRAINED_MULTIPLE_GLOBAL.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	INPT	OUPT	NEDOF					
Type	I	I	I					
Default	0	0	0					
Remarks	1		2					

Card 3	1	2	3	4	5	6	7	8
Variable	DX	DY	DZ	NEX	NEY	NEZ	NNPE	TOL
Type	F	F	F	I	I	I	I	F
Default	1.0	1.0	1.0	1	1	1	8	↓

Card 4	1	2	3	4	5	6	7	8
Variable	NOFF	EOFF	PNM					
Type	I	I	I					
Default	none	none	none					

Card 5	1	2	3	4	5	6	7	8
Variable	CNX	CNY	CNZ					
Type	I	I	I					
Default	none	none	none					

Node ID Cards. Included as many cards as needed. See Remark 2. Input is terminated at the next keyword ("*") card

Card 6	1	2	3	4	5	6	7	8
Variable	ECNX	ECNY	ECNZ					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
FILENAME	Name of the file containing the information about the unit cell
INPT	Type of input: EQ.0: Read *NODE information from the include file and add periodic boundary conditions to the include file. EQ.1: Create a unit cell mesh with periodic boundary conditions, and output to the include file.

VARIABLE	DESCRIPTION
OUPT	Type of output: EQ.1: Create a new main keyword file where the keyword *INCLUDE_UNITCELL is replaced by *INCLUDE with the include file name.
NEDOF	Number of extra nodal degrees of freedom (DOFs) for user-defined element. In the current implementation, the limit of NEDOF is 15.
DX	Length of the unit cell in the x-direction
DY	Length of the unit cell in the y-direction
DZ	Length of the unit cell in the z-direction
NEX	Number of elements along the x-direction
NEY	Number of elements along the y-direction
NEZ	Number of elements along the z-direction
NNPE	Number of nodes per element. The current implementation supports only 4-node tetrahedron or 8-node hexahedron elements.
TOL	Tolerance for searching for each pair of nodes in the periodic positions to create the periodic boundary conditions. This tolerance may be needed because numerical errors in the mesh can cause the coordinates of the pairs of nodes to *not* be exactly in the periodic positions. The default tolerance is computed based on the size of unit cell.
NOFF	Offset of node IDs
EOFF	Offset of element IDs
PNM	Part ID
CNX	Node ID of the 1st control point for the constraint in the x direction
CNY	Node ID of the 2nd control point for the constraint in the y direction
CNZ	Node ID of the 3rd control point for the constraint in the z direction
ECNX	Node ID of extra control point for the constraint in x direction of 3 extra nodal DOFs

VARIABLE	DESCRIPTION
ECNY	Node ID of extra control point for the constraint in y direction of 3 extra nodal DOFs
ECNZ	Node ID of extra control point for the constraint in z direction of 3 extra nodal DOFs

Remarks:

1. **Include File Field.** If INPT = 0, the geometry and discretization information for the unit cell are from the include file. In this case, the parameters set on Cards 3 and 4 are ignored.

2. **Extra Degrees of Freedom.** The extra degrees of freedom (DOFs) specified by NEDOF > 0 are represented by extra nodes with regular x, y and z DOFs. When NEDOF = 7, for example, the following chart shows the mapping from the extra DOFs to the regular ones of extra nodes:

Extra Node #	Extra DOFs	Regular DOFs
1	1	x
	2	y
	3	z
2	4	x
	5	y
	6	z
3	7	x

In this case, 3 control points for x, y, and z directions, respectively, need to be defined for each extra node.

*INCLUDE_WD

The *INCLUDE_WD keywords are needed during a weld line development calculation caused by including *INTERFACE_WELDLINE_DEVELOPMENT in the input deck. This feature outputs either the initial or final predicted welding curve. This feature requires the following *INCLUDE_WD keywords listed below in the order that they must appear in the input deck following *INCLUDE_WELDLINE_DEVELOPMENT:

*INCLUDE_WD_INITIAL_BLANK

*INCLUDE_WD_FINAL_PART

*INCLUDE_WD_WELDING_CURVE

*INCLUDE_WD_FINAL_PART

Purpose: Include the keyword format file that consists of geometry information for the final formed blank. This file is read to perform a weld line development calculation. This keyword must be used with *INTERFACE_WELDLINE_DEVELOPMENT and must follow *INCLUDE_WD_INITIAL_BLANK in the keyword deck. See *INTERFACE_WELDLINE_DEVELOPMENT for more details.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File name of the final formed blank in keyword format, consisting of *NODE and *ELEMENT_SHELL information

*INCLUDE_WD_INITIAL_BLANK

Purpose: Include the keyword format file that defines the geometry information for the initial sheet blank. This file is read to perform a weld line development calculation. This keyword must be used with and must directly follow *INTERFACE_WELDLINE_DEVELOPMENT (see there for more details).

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File name of the initial sheet blank in keyword format, consisting of *NODE and *ELEMENT_SHELL information.

*INCLUDE_WD_WELDING_CURVE

Purpose: Include the keyword format file that defines the welding curve (see *DEFINE_-CURVE_TRIM_3D) to perform a weld line development calculation. This keyword must be used with *INTERFACE_WELDLINE_DEVELOPMENT. Depending on the value of IOPTION on *INTERFACE_WELDLINE_DEVELOPMENT, this file defines either the final or initial welding curve. This keyword must follow *INCLUDE_WD_FINAL_PART directly. See *INTERFACE_WELDLINE_DEVELOPMENT for more details.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	File name of the welding curve in keyword format (see *DEFINE_-CURVE_TRIM_3D). If IOPTION = 1 on *INTERFACE_WELD-LINE_DEVELOPMENT, it should define the final welding curve. If IOPTION = -1, it should define the initial welding curve.

*INITIAL

The keyword *INITIAL provides a way of initializing velocities and detonation points. The keyword control cards in this section are defined in alphabetical order:

*INITIAL_AIRBAG_PARTICLE_POSITION

*INITIAL_ALE_MAPPING

*INITIAL_AXIAL_FORCE_BEAM

*INITIAL_CONTACT_WEAR

*INITIAL_CPG

*INITIAL_CRASHFRONT

*INITIAL_DETONATION

*INITIAL_DETONATION_GEOMETRY

*INITIAL_EOS_ALE

INITIAL_FATIGUE_DAMAGE_RATIO_*{OPTION}

INITIAL_FOAM_REFERENCE_GEOMETRY_*{OPTION}

*INITIAL_GAS_MIXTURE

*INITIAL_HISTORY_NODE

*INITIAL_HYDROSTATIC_ALE

INITIAL_IMPULSE_MINE

INITIAL_INTERNAL_DOF_SOLID_*OPTION

INITIAL_LAG_MAPPING_*{OPTION}

*INITIAL_MOMENTUM

INITIAL_PWP_DEPTH_*{OPTION}

*INITIAL_PWP_NODAL_DATA

*INITIAL_SOLID_VOLUME

*INITIAL_STRAIN_BEAM

*INITIAL_STRAIN_IGA_SHELL

INITIAL_STRAIN_SHELL_*{OPTION}

*INITIAL_STRAIN_SHELL_NURBS_PATCH

INITIAL_STRAIN_SOLID_*{OPTION}

*INITIAL_STRAIN_SOLID_NURBS_PATCH

*INITIAL_STRAIN_TSHELL

*INITIAL_STRESS_BEAM

INITIAL_STRESS_DEPTH_*{OPTION}

*INITIAL_STRESS_DES

*INITIAL_STRESS_IGA_SHELL

*INITIAL_STRESS_SECTION

INITIAL_STRESS_SHELL_*{OPTION}

*INITIAL_STRESS_SHELL_NURBS_PATCH

INITIAL_STRESS_SOLID_*{OPTION}

*INITIAL_STRESS_SOLID_NURBS_PATCH

*INITIAL_STRESS_SPH

*INITIAL_STRESS_TSHELL

INITIAL_TEMPERATURE_*OPTION

*INITIAL_TIED_CONTACT_DATA

*INITIAI_VAPOR_PART

*INITIAL_VEHICLE_KINEMATICS

There are two alternative sets of keywords for setting initial velocities. Cards from one set cannot be combined with cards from the other. **Standard velocity cards:**

*INITIAL_VELOCITY

*INITIAL_VELOCITY_NODE

*INITIAL_VELOCITY_RIGID_BODY

Alternative initial velocity cards supporting initial rotational about arbitrary axes and start times. **Alternative velocity cards:**

*INITIAL_VELOCITY_GENERATION

*INITIAL_VELOCITY_GENERATION_START_TIME

INITIAL_VOID_*OPTION

INITIAL_VOLUME_FRACTION_*{OPTION}

*INITIAL_VOLUME_FRACTION_GEOMETRY

*INITIAL_AIRBAG_PARTICLE_POSITION

Purpose: Initialize the position of an initial CPM air particle to the location specified.

Card 1	1	2	3	4	5	6	7	8
Variable	BAGID							
Type	I							
Default	none							

Particle Cards. The i^{th} card specifies the location of the i^{th} particle. The CPM solver expects one card for each particle. If fewer cards are supplied, the coordinates will be reused, and particles may share the same location at the beginning of the simulation.

Card 2	1	2	3	4	5	6	7	8
Variable		X	Y	Z				
Type		F	F	F				

VARIABLE	DESCRIPTION
BAGID	Airbag ID defined in *AIRBAG_PARTICLE_ID
X	*x* coordinate
Y	*y* coordinate
Z	*z* coordinate

*INITIAL_ALE_MAPPING

Purpose: This card initializes the current ALE run with data from the last cycle of a previous ALE run. A mapping file or files must be specified on the command line to provide the data to initialize the run. To map from a single mapping file, use map= on the command line (see Remark 6). To map from multiple mapping files, use map2= on the command line (see Remark 8). To map data histories (not just the last cycle) to a region of selected elements, see *BOUNDARY_ALE_MAPPING. This keyword is supported for ALE and S-ALE.

The following transitions are allowed:

1D → 2D	2D → 2D	3D → 3D
1D → 3D	2D → 3D	3D → 2D

Card 1	1	2	3	4	5	6	7	8
Variable	PID	TYP	AMMSID					
Type	I	I	I					
Default	none	none	none					

Card 2	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	VECID	ANGLE	SYM	TBEG	
Type	F	F	F	I	F	I	F	
Default	0.0	0.0	0.0	none	none	0	0.0	

VARIABLE	DESCRIPTION
PID	Part ID or part set ID.
TYP	Type of "PID" (see Remark 1): EQ.0: Part set ID (PSID). EQ.1: Part ID (PID).

VARIABLE	DESCRIPTION
AMMSID	Set ID of ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP. See Remark 1.
XO	Origin position in global x-direction. See Remarks 2 and 5.
YO	Origin position in global y-direction. See Remarks 2 and 5.
ZO	Origin position in global z-direction. See Remarks 2 and 5.
VECID	ID of the symmetric axis defined by *DEFINE_VECTOR. See Remarks 3 and 5.
ANGLE	Angle of rotation in degrees around an axis defined by *DEFINE_VECTOR for the 3D to 3D mapping. See Remark 4.
SYM	Treatment of the elements and nodes that are out of the mapping bounds (meaning the coordinates of their projections on the previous mesh are outside this mesh). SYM is a 6-digit parameter. Each digit represents a plane for a box that encloses the previous mesh. These planes are parallel to the previous coordinate system. The value of the digit determines the rule to be applied, so any combination of rules may be used. EQ.00000p: Rule for the X-plane along the lower previous mesh bound EQ.0000p0: Rule for the X-plane along the upper previous mesh bound EQ.000p00: Rule for the Y-plane along the lower previous mesh bound EQ.00p000: Rule for the Y-plane along the upper previous mesh bound EQ.0p0000: Rule for the Z-plane along the lower previous mesh bound EQ.p00000: Rule for the Z-plane along the upper previous mesh bound The value of p defines the rule to be applied to the given plane: p.EQ.0: Do nothing. p.EQ.1: Translational symmetry (direction of translation orthogonal to the box plane) p.EQ.2: Mirror-image symmetry about the box plane

VARIABLE	DESCRIPTION
	p.EQ.3: Continuity of boundary elements and nodes along the box's plane
TBEG	Time to start the run. It replaces the termination time of the previous run that generated the mapping file if TBEG is larger.

Remarks:

1. **Mapping of ALE multi-material groups.** The routines of this card need to know which mesh will be initialized with the mapping data, and more specifically, which multi-material groups. The first two fields, PID, and TYP, define the mesh. The third field, AMMSID, refers to a multi-material group list ID; see the *SET_MULTI-MATERIAL_GROUP_LIST card. The group list AMMSID should have as many elements as there are groups in the previous calculation (see *ALE_MULTI-MATERIAL_GROUP).

   Example: If the previous model has 3 groups, the current one has 5 groups, and the following mapping is wanted.

   Group 1 from the previous run → Group 3 in the current run
   Group 2 from the previous run → Group 5 in the current run
   Group 3 from the previous run → Group 4 in the current run

   The *SET_MULTI-MATERIAL_GROUP_LIST card should be set as follows:

```
*SET_MULTI-MATERIAL_GROUP_LIST
300
3,5,4
```

   In special cases, a group can be replaced by another. If Group 4 in the previous example is replaced by Group 3, the keyword setup has to be modified to have -3 instead of 4. The minus sign is a way for the code to know that the replacing group (-3 replaces 4) is a complement of Group 3:

```
*SET_MULTI-MATERIAL_GROUP_LIST
300
3,5,-3
```

2. **Coordinate system origin.** The location to which the data is mapped is controlled by the origin of the coordinate system (XO, YO, ZO).

3. **Symmetry axis.** For a mapping file created by a previous axisymmetric model, the symmetric axis orientation in the current model is specified by VECID. For a mapping file created by a 3D or 1D spherical model, VECID is read but ignored. For a mapping file created by a 1D plane strain model, the vector

specified with VECID orients the beams in the current mesh. For a 3D to 3D mapping, the vector is used if the parameter ANGLE is defined (see Remark 4).

4. **Rotating 3D data onto a 3D calculation.** For a mapping from a previous 3D run to a current 3D model, the previous 3D data will be rotated about the vector, VECID, through an angle specified in the ANGLE field.

5. **Plane strain and 3D to 2D.** The definitions of X0, Y0, Z0 and VECID change in the case of the following mappings:

   a) plane strain 2D (ELFORM = 13 in *SECTION_ALE2D) to plane strain 2D

   b) plane strain 2D to 3D

   c) 3D to 2D

   While VECID still defines the y-axis in the 2D domain, the first three parameters in *DEFINE_VECTOR, additionally, define the location of the origin. The last three parameters define a position along the y-axis. For this case, when 2D data is used in a 3D calculation, the point X0, Y0, Z0, together with the vector, VECID, define the plane.

6. **Mapping file.** Including the command line argument "map=" will invoke the creation of a mapping file. When the keyword *INITIAL_ALE_MAPPING is not in the input deck, but the argument "map=" is present on the command line, the ALE data from the last cycle is written in the mapping file. This file contains the following nodal and element data:

   - nodal coordinates (last step)
   - nodal velocities
   - part IDs
   - element connectivities
   - element centers
   - densities
   - volume fractions
   - stresses
   - plastic strains
   - internal energies
   - bulk viscosities
   - relative volumes
   - coupling penetrations from FSI

7. **Chained mappings.** To chain mapping operations so that LS-DYNA both reads and writes a mapping file, the command line argument "map1=" is necessary. If the keyword *INITIAL_ALE_MAPPING is in the input deck and the prompt map= is in the command line, the ALE data is read from the mapping file defined by map= to initialize the run. Data from the last cycle are written in the mapping file defined by map1=.

8. **Mapping from multiple mapping files.** The command-line option map2 = enables mapping from multiple mapping files in the run. The file specified with this command-line option must be a .txt file listing all the mapping files from which to map. Note that each specified mapping file should map to a different portion of the domain.

*INITIAL_AXIAL_FORCE_BEAM

Purpose: Initialize the axial force resultants in beam elements that are used to model bolts. This option works for beam type 9 (a Hughes-Liu type beam) with *MAT_SPOTWELD only and for beam type 1 with any material.

Card 1	1	2	3	4	5	6	7	8
Variable	BSID	LCID	SCALE	KBEND				
Type	I	I	F	I				
Default	none	none	1.0	0				

VARIABLE	DESCRIPTION
BSID	Beam set ID
LCID	Load curve ID defining preload force versus time. When the load curve ends or goes to zero, the initialization is assumed to be completed. See Remark 2 below.
SCALE	Scale factor on load curve.
KBEND	Bending stiffness flag EQ.0: Bending stiffness is negligible since all integration points are assigned the same axial stress. EQ.1: Bending stiffness is retained by keeping the axial stress gradient. EQ.2: Same as 1, but also allows for lining up several beams with prescribed axial force. See Remark 3.

Remarks:

1. **Damping.** To achieve convergence during explicit dynamic relaxation, the application of the damping options is very important. If contact is active, contact damping is recommended with a value between 10-20 percent. Additional damping applied using *DAMPING_PART_STIFFNESS also speeds up convergence where a coefficient of 0.10 is effective. If damping is not used, convergence may not be possible.

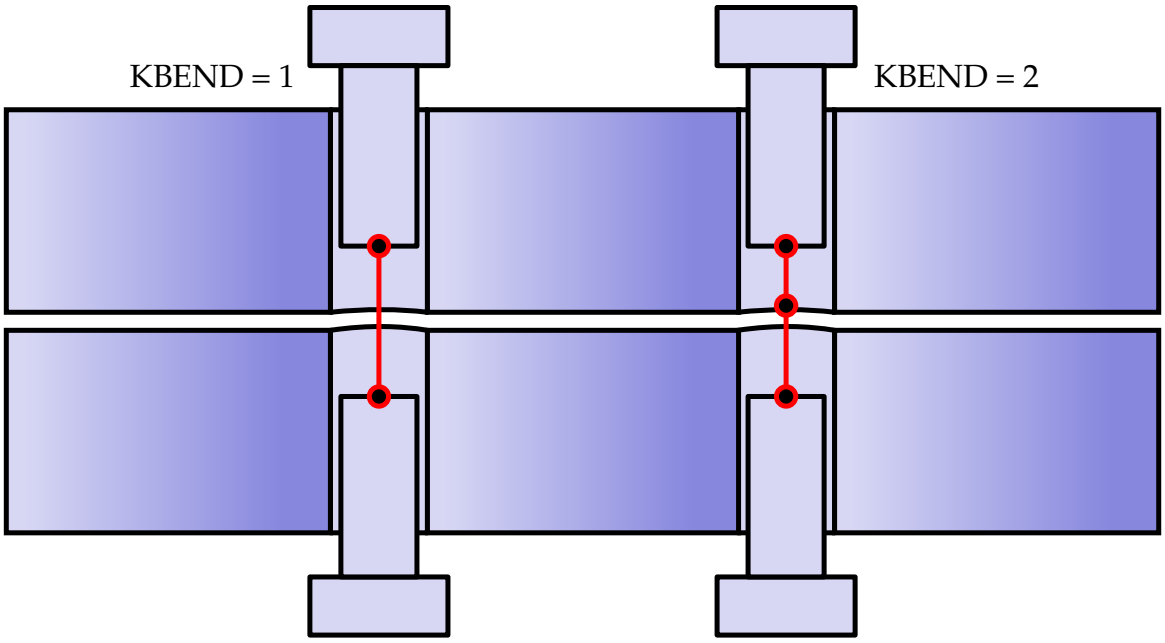


Figure 28-1. Illustration of modeling strategies for different values of KBEND

2. **Ramping.** When defining the load curve, LCID, a ramp starting at the origin should be used to increase the force to the desired value. The time duration of the ramp should produce a quasistatic response. When the end of the load curve is reached, or when the value of the load decreases from its maximum value, the initialization stops. If the load curve begins at the desired force value, that is, no ramp, convergence will take much longer, since the impulsive-like load created by the initial force can excite nearly every frequency in the structural system where force is initialized.

3. **Modeling Strategy.** When preloading bolts, users commonly reserve *one* beam element along the shank/threaded part of each bolt for the *INITIAL_AXIAL_-FORCE_BEAM keyword. The selected beam element must be long enough such that it does not reduce to zero length by the end of preload. The formulation of the beam causing crimping upon loading. Mesh connections, boundary conditions or contact prevent crimping. We illustrate this strategy with the left bolt in Figure 28-1. The beam to be load is red. Note that the loaded beam must be significantly longer than the sum of *(i)* the gap between the two plates and *(ii)* the two gaps between the plates and bolt heads. We use only one beam because otherwise the position of any node connecting two preloaded beam elements cannot be determined from the equations of equilibrium; the positions of intermediate nodes along the axis of the bolt are arbitrary. With KBEND = 2, you can load more than one beam as shown by the right bolt in the figure. LS-DYNA internally imposes additional constraints for intermediate nodes to make the problem well posed. The length reduction is displacement controlled to avoid excessive dynamic effects. Dynamic effects might otherwise occur if the bolt

heads impact the plates to be clamped at high speed. Note that the previously discussed differences apply even if only one beam element is used for the pre-load.

*INITIAL_CONTACT_WEAR

Purpose: Initialize contact wear for simulation of wear processes. Define as many cards as desired. See Remarks.

Note that this card is not supposed to be manually inserted; rather it is supposed to be generated by LS-DYNA during wear processes.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	NID	WDEPTH	NX	NY	NZ	ISEQ	NCYC
Type	I	I	F	F	F	F	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
CID	Contact Interface ID.
NID	Node ID.
WDEPTH	Wear depth, in units of length.
NX, NY, NZ	Direction vector for wear, internally normalized.
ISEQ	Simulation sequence number for the entire process.
NCYC	The wear on this card will be processed NCYC times to modify the worn geometry. This means that *one* LS-DYNA simulation is used to predict the wear for *NCYC* repetitions of the process to save simulation time. This number should be chosen with care; a negative number means that LS-DYNA will not apply this card. See Remarks below.

Remarks:

This card is not supposed to be manually inserted, but is automatically generated by LS-DYNA when simulating wear processes; see *CONTACT_ADD_WEAR and parameters NCYC on *INTERFACE_SPRINGBACK_LSDYNA and SPR/MPR on *CONTACT. A sequence of identical simulations, except for perturbation of the geometry of certain components due to wear, is undertaken. For a given contact interface and node ID, the corresponding node is perturbed by the wear depth in the direction of wear. If the cycle

number NCYC is negative, the geometry has been already processed in LS-PrePost and the card is ignored by LS-DYNA. If a node appears multiple times, the wear from the individual sequences is accumulated.

*INITIAL_CPG

Purpose: Initialize fluid quantities in the CPG domain.

Card 1	1	2	3	4	5	6	7	8
Variable	BAGID	ITYPE	IDDR	PREI	TEMPI			
Type	I	I	I	F	F			
Default	none	none	none	none	none			

Optional card.

Card 2	1	2	3	4	5	6	7	8
Variable	NGAS							
Type	I							
Default	0							

Initial Gas Fraction Card. Define NGAS cards.

Card 2.1	1	2	3	4	5	6	7	8
Variable	XFRAC*i*							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
BAGID	Airbag ID defined in *AIRBAG_CPG card
ITYPE	Defines the initialization type: EQ.1: Give initial pressure and temperature.

VARIABLE	DESCRIPTION
IDDR	ID of the region where the initial conditions will apply. See *DEFINE_CPG_REGION.
PREI/TEMPI	Initial pressure and temperature in the region defined by IDDR for ITYPE = 1
NGAS	Initial gas fractions for the gases given in the same order as in *AIRBAG_CPG. NGAS defined here cannot be larger than NGAS defined in *AIRBAG_CPG.
XFRAC*i*	Initial gas fractions for the different gases

*INITIAL_CRASHFRONT

Purpose: For some composite materials, the strength of an element is reduced if one of its neighboring elements fails due to damage. The damaged nodes are considered crash-front. In general, all nodes are initially perfect and there are no initial crashfront nodes. However, this card allows the user to initialize crashfront nodes for damage that may occur prior to the simulation, such as during the manufacturing process. Any elements connected to these crashfront nodes are initialized with reduced strength.

The crashfront node set can be initialized using the nodes that are in a set of segments, shell elements, parts, or nodes. This keyword works with material models 17, 54, 55, 58, 169, 261, and 262.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
SID	Set ID from which the initial crashfront nodes are defined.
STYPE	ID type of SID: EQ.0: segment set ID, EQ.1: shell element set ID, EQ.2: part set ID, EQ.3: part ID, EQ.4: node set ID.

***INITIAL_DETONATION** *{OPTION}*

Available options include:

<BLANK>

SET

Purpose: Define points to initiate the location of high explosive detonations in part IDs that use *MAT_HIGH_EXPLOSIVE_BURN (*MAT_008). Also, see *CONTROL_EXPLOSIVE_SHADOW. If no *INITIAL_DETONATION is defined, detonation occurs in all the high explosive elements at time = 0. The SET keyword option is supported only for ALE and S-ALE and enables specifying where the high explosive is to be initially lit with an element set.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	X	Y	Z	LT		MMGSET	
Type	I	F	F	F	F		I	
Default	all HE	0.	0.	0.	0.		0	

Acoustic Boundary Card. Additional card for PID = -1.

Card 2	1	2	3	4	5	6	7	8
Variable	PEAK	DECAY	XS	YS	ZS	NID		
Type	F	F	F	F	F	I		

VARIABLE	DESCRIPTION
PID	Part or part set ID of the high explosive to be lit, except in the case where the high explosive is modeled using an ALE formulation, in which case PID is the part, part set, or element set ID of the mesh where the high explosive material to be lit initially resides. However, two other options are available: LT.-1: |PID| is the ID of a part set (*SET_PART) EQ.-1: An acoustic boundary or *BOUNDARY_USA_SURFACE

VARIABLE	DESCRIPTION
	EQ.0: All high explosive materials are considered. GT.0: PID is a part ID unless using the SET keyword option. For the SET keyword option, it is an element set (*SET_BEAM for ALE 1D models, *SET_SHELL for ALE 2D models, or *SET_SOLID for ALE 3D models).
X	x-coordinate of detonation point, see Figure 28-2.
Y	y-coordinate of detonation point.
Z	z-coordinate of detonation point.
LT	Lighting time for detonation point. This time is ignored for an acoustic boundary.
MMGSET	ID of *SET_MULTI-MATERIAL_GROUP_LIST selecting the explosive ALE groups to be lit in the mesh defined by PID.
PEAK	Peak pressure, p_o, of the incident pressure pulse. See Remark 3.
DECAY	Decay constant, τ. See Remark 3.
XS	x-coordinate of standoff point; see Figure 28-2.
YS	y-coordinate of standoff point
ZS	z-coordinate of standoff point
NID	Reference node ID near structure

Remarks:

1. **Lighting time.** For solid elements (not acoustic) two options are available. If the control card option *CONTROL_EXPLOSIVE_SHADOW is not used, the lighting time for an explosive element is computed using the distance from the center of the element to the nearest detonation point, L_d; the detonation velocity, D; and the lighting time for the detonator, t_d:

$$t_L = t_d + \frac{L_d}{D}.$$

The detonation velocity for this default option is taken from the element whose lighting time is computed and does not account for the possibility that the detonation wave may travel through other explosives with different detonation velocities, or that the line of sight may pass outside the explosive material.

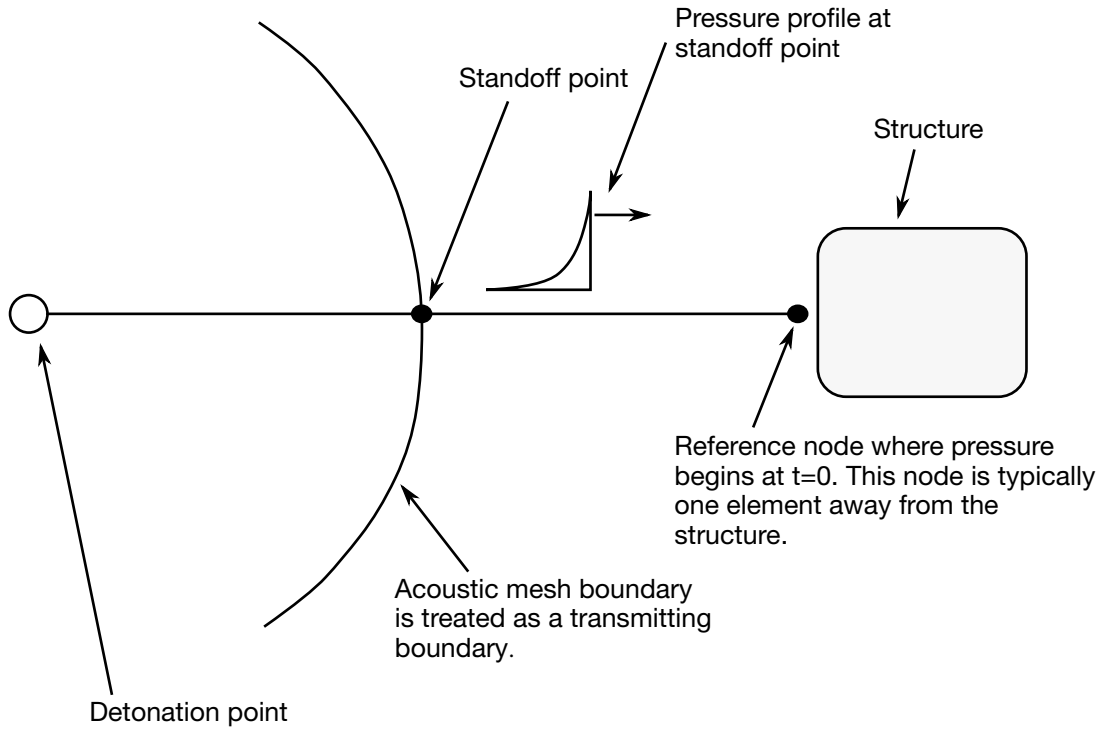


Figure 28-2. Initialization of the initial pressures due to an explosive disturbance is performed in the acoustic media. LS-DYNA automatically determines the acoustic mesh boundary and applies the pressure time history to the boundary. This option is only applicable to the acoustic element formulation; see *SECTION_SOLID.

If the control card option *CONTROL_EXPLOSIVE_SHADOW is defined, the lighting time is based on the shortest distance through the explosive material. If inert obstacles exist within the explosive material, the lighting time will account for the extra time required for the detonation wave to travel around the obstacles. The lighting times also automatically account for variations in the detonation velocity if different explosives are used. No additional input is required for this option, but care must be taken when setting up the input. This option works for two and three-dimensional solid elements. It is recommended that for best results:

a) Keep the explosive mesh as uniform as possible with elements of roughly the same dimensions.

b) Inert obstacles, such as wave shapers within the explosive, must be somewhat larger than the characteristic element dimension for the automatic tracking to function properly. Generally, a factor of two should suffice. The characteristic element dimension is found by checking all explosive elements for the largest diagonal.

c) The detonation points should be either within or on the boundary of the explosive. Offset points may fail to initiate the explosive. When LT is non-zero, the detonation point is fixed to the explosive material at $t = 0$ and moves as the explosive material moves prior to detonation.

d) Check the computed lighting times in the post-processor LS-PrePost. The lighting times may be displayed at time = 0., state 1, by plotting component 7 (a component normally reserved for plastic strain) for the explosive material. The lighting times are stored as negative numbers. The negative lighting time is replaced by the burn fraction when the element ignites.

2. **Line detonations.** Line detonations may be approximated by using a sufficient number of detonation points to define the line. Too many detonation points may incur significant initialization costs.

3. **Acoustic boundary.** The pressure as a function of time curve for the acoustic boundary is defined by:

$$p(t) = p_o e^{-\frac{t}{\tau}}.$$

*INITIAL_DETONATION_GEOMETRY

Purpose: Define detonation points that are selected based on a specified geometry. These points ignite high explosives defined by parts that use *MAT_HIGH_EXPLOSIVE_BURN (*MAT_008).

Also see *INITIAL_DETONATION and *CONTROL_EXPLOSIVE_SHADOW.

Card Summary:

Card 1. This card is required.

HEID	HETYP	MMGSET					

Card 2. This card is required.

GEOTYP	LT	DGEO	BOXID				

Card 3a. This card is included if GEOTYP = 1.

VID1							

Card 3b. This card is included if GEOTYP = 2.

VID1	VID2	VID3	VID4				

Card 3c. This card is included if GEOTYP = 3.

VID1	VID2	VID3					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	HEID	HETYP	MMGSET					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
HEID	ID specifying the high explosives to be lit

VARIABLE	DESCRIPTION
HETYP	Type of HEID: EQ.0: Part set ID (*SET_PART) EQ.1: Part (*PART) EQ.2: Element set ID (*SET_SOLID in 3D and *SET_SHELL in 2D)
MMGSET	ID of *SET_MULTI-MATERIAL_GROUP_LIST selecting the explosive ALE groups to be lit

Card 2	1	2	3	4	5	6	7	8
Variable	GEOTYP	LT	DGEO	BOXID				
Type	I	F	F	I				
Default	none	0.	↓	0				

VARIABLE	DESCRIPTION
GEOTYP	Type of geometry formed by the detonation points: EQ.1: Plane EQ.2: Cylindrical or truncated conical surface EQ.3: Spherical or ellipsoidal surface
LT	Lighting time for detonation point
DGEO	Maximum distance from the detonation geometry for determining which HE elements become detonation points. If the element center for the specified HE is less than this distance away from the detonation geometry, the element center becomes a detonation point. If zero or undefined, DGEO becomes the length of the largest specified HE element. Note that this condition can be limited to only elements inside a box specified with BOXID below.
BOXID	ID of a box (see *DEFINE_BOX). If used, the element center becomes a detonation point if it is both within the box and within the distance specified with DGEO relative to the detonation geometry.

Plane Geometry Card. This card is included if GEOTYP = 1.

Card 3a	1	2	3	4	5	6	7	8
Variable	VID1							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
VID1	ID of the vector (see *DEFINE_VECTOR) that gives the plane's normal vector. The tail of VID1 is a point in the plane.

Cylindrical Surface Geometry Card. This card is included if GEOTYP = 2.

Card 3b	1	2	3	4	5	6	7	8
Variable	VID1	VID2	VID3	VID4				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
VID1, …, VID4	IDs of vectors (see *DEFINE_VECTOR) for specifying the orientation and location of a cylinder or truncate cone with either circular or elliptical bases. The cylinder or truncated cone can also be oblique. For all cases, the tail of VID1 gives the center of one of the bases, the length of VID2 gives the length of the cylinder or truncated cone's axis, and the direction of VID2 gives the direction of the cylinder or truncated cone's axis. For a cylinder with circular bases, the length of VID1 is the radius of the circle. The bases lie in planes formed by VID1 and the cross product of VID1 with VID2. VID3 should not be input for this case. See Figure 28-3. For a cylinder with elliptical bases, the lengths of VID1 and VID3 are the lengths of the two axes. The direction of the axis with the length of VID1 is in the direction of VID1. The direction for the axis

VARIABLE	DESCRIPTION
	with the length given by VID3 is in the direction given by the cross product of VID1 with VID2. See Figure 28-4. For a truncated cone, you must specify VID4 in addition to VID1 and VID2. The bottom base determines if the bases are circular or elliptical based on whether you specify VID3 as previously discussed. VID4 gives the direction and length of the first axis of the second base. It does not need to be parallel to VID1. The second base is scaled uniformly from the first base. See Figure 28-5.

Spherical or Ellipsoidal Surface Geometry Card. This card is included if GEOTYP = 3.

Card 3c	1	2	3	4	5	6	7	8
Variable	VID1	VID2	VID3					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
VID1, …, VID3	IDs of vectors (see *DEFINE_VECTOR) for specifying the orientation and location of the sphere or ellipsoid. In each case, the center of the surface is given by the tail of VID1. For a sphere, only specify VID1. The length of VID1 is the radius of the sphere. For an ellipsoid with two axes of equal length, specify VID1 and VID2. The length of VID1 is used for the axes of equal length which are in the directions VID1 and the cross product of VID1 with VID2. The length of VID2 gives the length of the third axis which is orthogonal to VID1 and the cross product of VID1 with VID2. See Figure 28-6. For an ellipsoid with all axes of different length, specify VID1, VID2, and VID3. One axis is in the direction of VID1 with the length of VID1. Another axis is in the direction of the cross product of VID1 with VID2 with the length given by the length of VID3. The third axis is orthogonal to the other two axes and has a length equal to the length of VID2. See Figure 28-7.

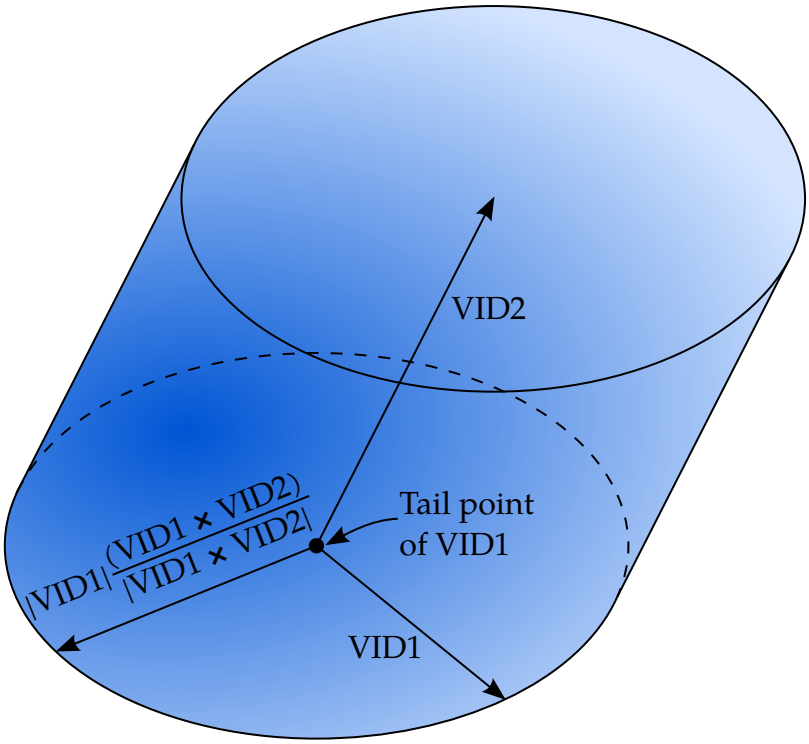


Figure 28-3. Oblique cylinder with circular bases (GEOTYP = 2)

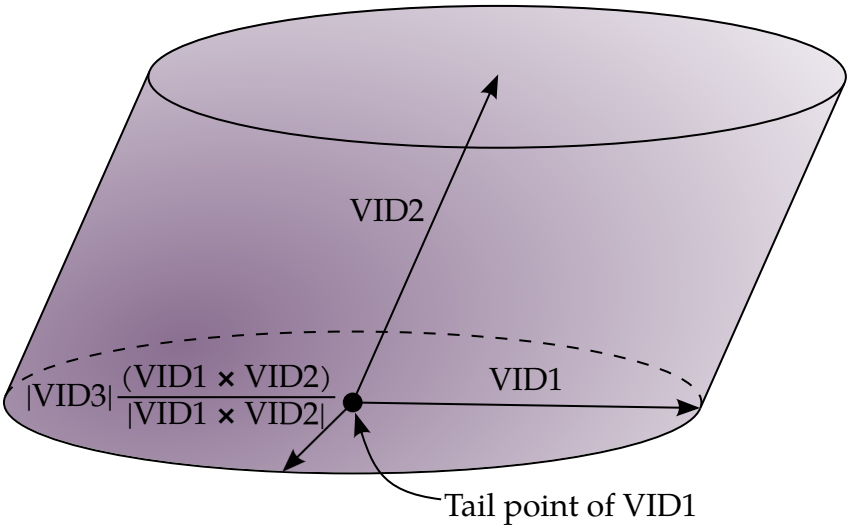


Figure 28-4. Oblique cylinder with elliptical bases (GEOTYP = 2)

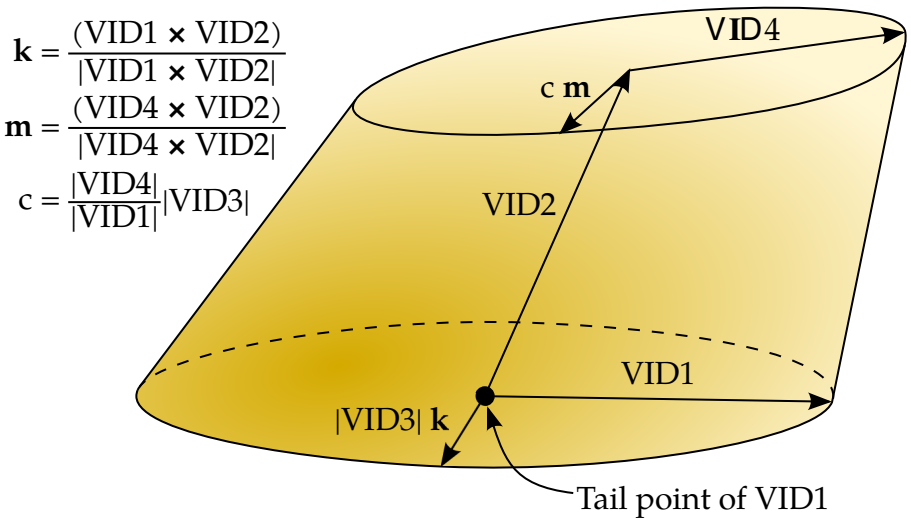


Figure 28-5. Oblique truncated cone with elliptical bases (GEOTYP = 2)

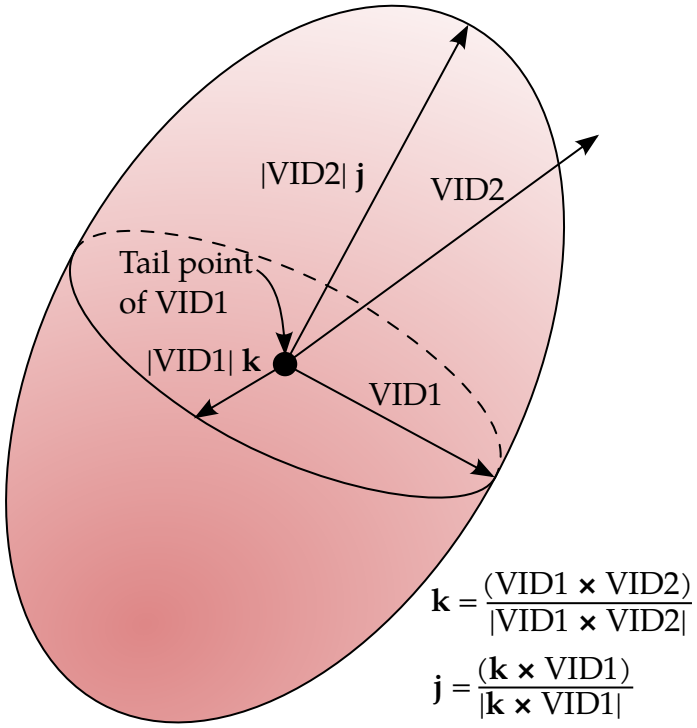


Figure 28-6. Ellipsoid with two equal axes (GEOTYP = 3)

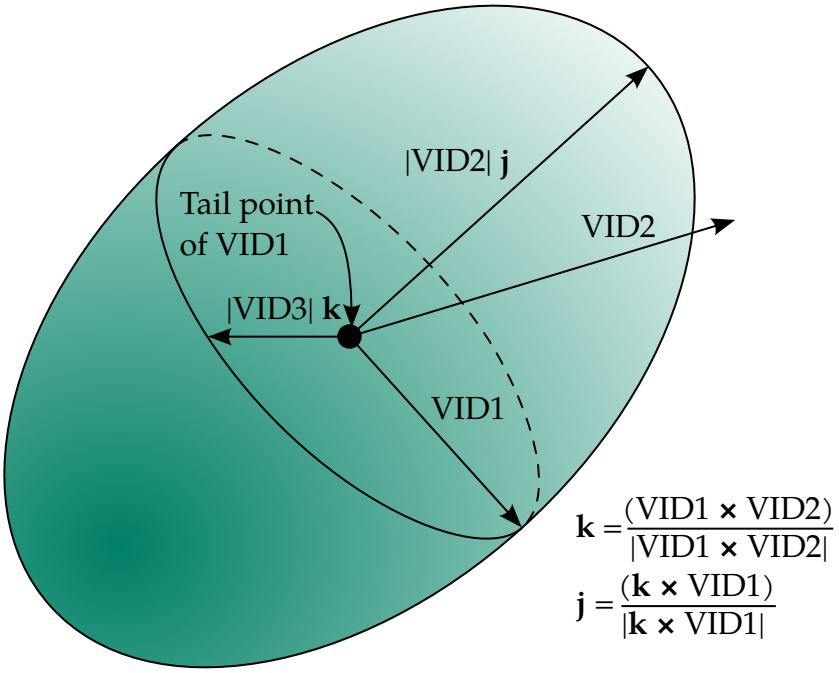


Figure 28-7. Ellipsoid with axes of different length (GEOTYP = 3)

*INITIAL_EOS_ALE

Purpose: Initialize the pressure in ALE elements that have materials with *EOS.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TYP	MMG	E0	V0	P0		
Type	I	I	I	F	F	F		
Default	none	none	none	0.0	0.0	0.0		

VARIABLE	DESCRIPTION
ID	Part, part set, or element set ID
TYP	Type of "ID": EQ.0: Part set ID EQ.1: Part ID EQ.2: Solid set ID for 3D case EQ.3: Shell set ID for 2D case EQ.4: Beam set ID for 1D case
MMG	Specifies the multi-material group. GT.0: ALE multi-material group LT.0: Set ID of ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP_LIST
E0	Initial internal energy per reference volume unit (as defined in *EOS). See Remark 1.
V0	Initial relative volume (as defined in *EOS). See Remark 1.
P0	Initial pressure. See Remark 2.

Remarks:

1. **Initialization with volume and energy.** For most *EOS, E0 and V0 should be used to initialize the pressure. If only the internal energy is initialized, V0 should be 1.0 (If V0 = 0.0, E0 will not be applied).

2. **Initial pressure with derived volume and energy.** For *EOS_001, *EOS_004 and *EOS_006, the initial pressure P0 can be input directly. An iterative method will compute the initial internal energy and relative volume. This approach is applied if E0 = 0.0 and V0 = 0.0.

*INITIAL_FATIGUE_DAMAGE_RATIO_{OPTION1}_{OPTION2}

Available options for *OPTION1* include:

<BLANK>

D3FTG

D3PLOT

Available options for *OPTION2* include:

<BLANK>

SCALE_FACTOR

Purpose: This card sets the initial damage ratio for fatigue analysis. The initial damage ratio may come from the previous loading cases. The initial damage ratio can be defined directly by the user or can be extracted from an existing binary database like D3FTG (using the option D3FTG) and D3PLOT (using the option D3PLOT). A scale factor for the damage ratio from each previous loading case can be defined if that loading case repeats using the SCALE_FACTOR keyword option

Card Summary:

Card Sets. Include as many sets of the following cards as needed to include the initial damage ratio from each loading condition (for multiple loading condition cases). This input terminates with the next keyword ("*") card.

Card 1a. Include this card if no keyword option is used (<BLANK>).

PID/SID	PTYP	DRATIO					

Card 1b. Include this card if *OPTION1* is D3FTG and *OPTION2* is unused (<BLANK>).

FILENAME

Card 1c. Include this card if *OPTION1* is D3FTG and *OPTION2* is SCALE_FACTOR.

SF	FILENAME

Card 1d. Include this card if *OPTION1* is D3PLOT and OPTION2 is blank.

FILENAME

Card 1e. Include if *OPTION1* is D3PLOT and *OPTION2* is SCALE_FACTOR.

SF	FILENAME

Card 2. Include this card if the D3PLOT keyword option is used.

NSTATE	NEIPHD	NEIPSD					

Data Card Definitions:

Initial Damage Ratio Card. Card 1 for no option (<BLANK>)

Card 1a	1	2	3	4	5	6	7	8
Variable	PID/SID	PTYP	DRATIO					
Type	I	I	F					
Default	none	0	0.0					

VARIABLE	DESCRIPTION
PID/SID	Part ID or part set ID for which the initial damage ratio is defined
PTYP	Type of PID/PSID: EQ.0: Part ID EQ.1: Part set ID
DRATIO	Initial damage ratio

D3FTG Card. Card 1 for the D3FTG keyword option with *OPTION2* unused (<BLANK>).

Card 1b	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	d3ftg							

VARIABLE	DESCRIPTION
FILENAME	Path and name of the existing binary database for fatigue information

D3FTG Card. Card 1 for the D3FTG keyword option with *OPTION2* as SCALE_FACTOR

Card 1c	1	2	3	4	5	6	7	8
Variable	SF	FILENAME						
Type	F	C						
Default	1.0	d3ftg						

VARIABLE	DESCRIPTION
SF	Scale factor
FILENAME	Path and name of the existing binary database for fatigue information

D3PLOT Card. Card 1 for the D3PLOT keyword option with *OPTION2* unused.

Card 1d	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	d3plot							

VARIABLE	DESCRIPTION
FILENAME	Path and name of the existing binary database for fatigue information

D3PLOT Card. Card 1 for the D3PLOT keyword option with *OPTION2* as SCALE_FACTOR

Card 1e	1	2	3	4	5	6	7	8
Variable	SF	FILENAME						
Type	F	C						
Default	1.0	d3plot						

VARIABLE	DESCRIPTION
SF	Scale factor
FILENAME	Path and name of the existing binary database for fatigue information

D3PLOT Card. Card 2 for the D3PLOT keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	NSTATE	NEIPHD	NEIPSD					
Type	I	I	I					
Default	1	1	1					

VARIABLE	DESCRIPTION
NSTATE	State ID in binary database (e.g. d3plot) for reading damage variables
NEIPHD	ID of additional integration point history variable which saves the damage for solid elements
NEIPSD	ID of additional integration point history variable which saves the damage for shell and thick shell elements

Remarks:

1. **Fatigue Problem Types.** This card works for both time-domain fatigue and frequency-domain fatigue problems.

2. **Transient Preload Cases.** The option D3PLOT can be used for users who want to use damage from transient preload cases (e.g. from GISSMO damage model) as the initial fatigue damage ratio. In this case, LS-DYNA reads the binary database defined in Card 1d or 1e (e.g. d3plot) and extracts the damage variables from the additional integration point history variables based on NSTATE, NEIPHD, and NEIPSD defined in Card 2.

*INITIAL_FIELD_SOLID

Purpose: This keyword is a simplified version of *INITIAL_STRESS_SOLID which can be used with hyperelastic materials. The keyword is used for history variable input. Data is usually in the form of the eigenvalues of diffusion tensor data. These are expressed in the global coordinate system.

> **NOTE:** As of LS-DYNA R5 in all contexts, other than *MAT_TISSUE_DISPERSED, this keyword is deprecated (and disabled). For all other materials this keyword has been superceded by *INITIAL_STRESS_SOLID.

Card Sets: Include as many pairs of Cards 1 and 2 as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NINT	NHISV					
Type	I	I	I					
Default	none	none	0					

Card 2	1	2	3	4	5	6	7	8
Variable	FLD1	FLD2	FLD3	FLD4	FLD5	FLD6	FLD7	FLD8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
EID	Element ID
NINT	Number of integration points (should correspond to the solid element formulation).
NHISV	Number of field variables. If NHISV exceeds the number of integration point field variables required by the constitutive model, only the number required is output; therefore, if in doubt, set NHISV to a large number.

VARIABLE	DESCRIPTION
FLD*n*	Data for the n^{th} field (history) variable. NOTE that *MAT_TISSUE_DISPERSED only use FLD1 to FLD3 since NHISV = 3.

Remarks:

Add as many cards as necessary. The keyword input ends at the next keyword ("*") card. For example for two elements it can look as:

```
*INITIAL_FIELD_SOLID
$EID       NINT     NHISV
    1          1         3
$FLD1      FLD2      FLD3
  0.1       0.8       0.1
$EID       NINT     NHISV
    2          1         3
$FLD1      FLD2      FLD3
  0.3       0.2       0.5
```

*INITIAL_FOAM_REFERENCE_GEOMETRY_{OPTION}

Available options include:

<BLANK>

RAMP

Purpose: Define the reference geometry of a foam for stress initialization. LS-DYNA initializes the stresses using the reference geometry when the REF flag is turned on in the *MAT input for the following hyperelastic material models: 2, 5, 7, 21, 23, 27, 31, 38, 57, 73, 77, 83, 91, 132, 179, 181, 183, and 189. Supported solid elements are the constant stress hexahedron (#1), the fully integrated S/R hexahedron (#2), the tetrahedron (#10), and the pentahedron (#15). This keyword is also supported for plane strain (shell #13) and axisymmetric solid (#14 and #15) elements. With this keyword, dynamic relaxation can be avoided once a deformed configuration is obtained, usually on the first run of a particular problem.

To use this keyword, define the geometry of the low-density foam in a deformed configuration. The stresses depend only on the deformation gradient matrix, F_{ij}:

$$F_{ij} = \frac{\partial x_i}{\partial X_j}$$

where x_i is the deformed configuration and X_j is the undeformed configuration.

Some shell and solid elements modeling the fabric and foam, respectively, share nodes. The reference geometry specified by either *INITIAL_FOAM_REFERENCE_GEOMETRY or *AIRBAG_REFERENCE_GEOMETRY will be applied to both the shell and solid elements that share these nodes. However, having different reference geometry for shell and solid elements sharing common nodes can be achieved by using *INITIAL_FOAM_REFERENCE_GEOMETRY for solid elements and *AIRBAG_SHELL_REFERENCE_GEOMETRY for shell elements.

Card Summary:

Card 1. This card is included if the RAMP option is used.

NDTRRG							

Card 2. Include as many of this card as needed. The next keyword ("*") card terminates this input.

NID	X	Y	Z			

Data Card Definitions:

RAMP Card. Additional card for the option of RAMP.

Card 1	1	2	3	4	5	6	7	8
Variable	NDTRRG							
Type	I							
Default	0							

Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z				
Type	I	F		F		F				
Default	none	0.		0.		0.				

VARIABLE	DESCRIPTION
NDTRRG	Number of time steps taken for an element to restore its reference geometry, for 3D solid elements only. The definition of NDTRRG allows an element to ramp up to its reference shape in NDTRRG time steps. Currently, LS-DYNA uses only one value of NDTRRG and applies it to all foam materials with reference geometries. If more than one NDTRRG is defined, the one defined last will be used.
NID	Node number
X	x-coordinate in reference configuration
Y	y-coordinate in reference configuration
Z	z-coordinate in reference configuration

*INITIAL_GAS_MIXTURE

Purpose: This command is used to specify (a) which ALE multi-material groups may be present inside an ALE mesh set at time zero, and (b) the corresponding reference gas temperature and density which define the initial thermodynamic state of the gases. The order of the species in the gas mixture corresponds to the order of different gas species defined in the associated *MAT_GAS_MIXTURE card. This card must be used together with a *MAT_GAS_MIXTURE card or, equivalently, a *MAT_ALE_GAS_MIXTURE card.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	MMGID	TEMP				
Type	I	I	I	F				
Default	none	0	none	none				

Card 2	1	2	3	4	5	6	7	8
Variable	R01	R02	R03	R04	R05	R06	R07	R08
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
SID	Set ID for initialization. This SID defines the ALE mesh within which certain ALE multi-material group(s) may be present at t = 0.
STYPE	Set type for the SID above: EQ.0: SID is a part set ID EQ.1: SID is a part ID
MMGID	ALE multi-material group ID of the material that may be present at t = 0 in the ALE mesh set defined by SID. For general ALE, it must be AMMGID. For S-ALE, either AMMGID or AMMG name (AMMGNM) could be used here. Please refer to *ALE_STRUC-

VARIABLE	DESCRIPTION
	TURED_MULTI-MATERIALS_GROUP for more details.
TEMP	Initial static temperature of the gas species occupying the ALE mesh. Note that all species in the mixture are assumed to be in thermal equilibrium (having the same T).
RO1-RO8	Initial densities of the ALE material(s), which may occupy some region (or all) of the aforementioned ALE mesh, for up to eight different gas species. The order of the density input corresponds to the order of the materials defined in the associated *MAT_GAS_MIXTURE card.

Remarks:

1. **Example.** Please see the example under the *MAT_GAS_MIXTURE card definition for an application of the *INITIAL_GAS_MIXTURE card.

2. **Initial pressure.** The temperature is assumed to be the initial temperature which together with the gas density, will define the initial pressure of the gas species using the perfect gas law,

$$P|_{t=0} = \rho|_{t=0}(C_P - C_V)T|_{t=0} \ .$$

The user should manually check the initial pressure for consistency.

3. **ALE multi-material groups.** Given an ALE mesh, this mesh may initially be occupied by one or more ALE multi-material groups (AMMG). For example, a background ALE mesh (H1) containing AMMG 1 may be partially filled with AMMG 2 using the volume filling command *INITIAL_VOLUME_FRACTION_GEOMETRY. Then there are two AMMGs to be initialized for this mesh H1. The commands look like the following.

```
$---------------------------------------------------------------------
$ One card is defined for each AMMG that will occupy some elements of a mesh
set
*INITIAL_GAS_MIXTURE
$      SID     STYPE     MMGID        T0
         1         1         1   298.15
$     RHO1      RHO2      RHO3      RHO4      RHO5      RHO6      RHO7
RHO8
    1.0E-9
*INITIAL_GAS_MIXTURE
$      SID     STYPE     MMGID        T0
         1         1         2   298.15
$     RHO1      RHO2      RHO3      RHO4      RHO5      RHO6      RHO7
RHO8
    1.2E-9
$---------------------------------------------------------------------
```

***INITIAL_HISTORY_NODE**_{OPTION}_

Available options include:

<BLANK>

SET

Purpose: Initialize certain history variable values on a nodal basis for shells, thick shells, and solids. The value of the history variable at an integration point is then determined by interpolating the nodal values as described in more detail in Remark 1.

Card Sets per NODE. Define as many nodes or node sets in this section as desired. The input is assumed to terminate when a new keyword ("*") card is detected.

Node Card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID/NSID	NHISV						
Type	I	I						
Default	none	none						

History Variable Cards. Include NHISV (see Card 1) cards, one card per history variable.

Card 2	1	2	3	4	5	6	7	8
Variable	HINDEX	VAL						
Type	I	F						
Default	none	0.0						

VARIABLE	DESCRIPTION
NID/NSID	Node ID or node set ID, see *SET_NODE_…
NHISV	Number of history variables to be initialized
HINDEX	Define the index in the history variable vector

VARIABLE	DESCRIPTION
VAL	Define the value of the history variable

Remarks:

1. **Interpolation Methods.** For shell elements, the values of the history variables assigned to its nodes are interpolated to the location of the in-plane integration points using the finite element shape functions associated to the individual element formulation. All the through thickness integration points will receive the same value as the one in the shell reference plane.

   For thick shell and solid elements, the values of the history variables assigned to its nodes are interpolated to the location of all the integration points using the finite element shape functions associated to the individual element formulation.

2. **Number of Nodes Required for Initialization.** Only integration points of an element where *at least one of* its associated nodes has a history value set with this keyword will be initialized. The values of the history variables of the uninitialized nodes are assumed to be 0.0.

3. **Keyword Limitations.** Note that initialization with nodes that are shared between two parts should be performed with caution since this keyword does not discriminate based on the part ID and more than one material may be involved. In the case of more than one material for a given node, LS-DYNA will issue an error saying that the history variable cannot be initialized. Also, this keyword should not be used for composite shells because of the material change through the thickness (see Remark 1).

4. **Comparison to *INITIAL_STRESS_SOLID/SHELL.** When defining initial values of history variables, this keyword serves a similar purpose to *INITIAL_STRESS_SOLID/SHELL. There are two big differences. The first is how the history variables are initialized. With this keyword you input the nodal values for the history variables which are then interpolated for the integration points. For *INITIAL_STRESS_SOLID/SHELL, you input the initial values at each integration point for an element. Another difference is that for this keyword only the desired history variables are initialized. For instance, if you want to initialize history variable 14, with INITIAL_STRESS_SHELL/SOLID you would have to initialize history variables 1 through 14, whereas with this keyword you would only initialize history variable 14:

   ```
   *INITIAL_HISTORY_NODE_SET
   nsid,1
   14,50.0
   ```

   or like this

```
*INITIAL_HISTORY_STRESS_SHELL_SET
sid,1,1,14
0.0

,,,,,50.0
```

*INITIAL_HYDROSTATIC_ALE

Purpose: When an ALE model contains one or more regular (not reservoir-type) ALE parts (ELFORM = 11 and AET = 0), this command may be used to initialize the hydrostatic pressure field in the regular ALE domain due to gravity. A *LOAD_BODY_*OPTION* keyword must be defined when using this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	ALESID	STYPE	VECID	GRAV	PBASE			
Type	I	I	I	F	F			
Default	none	0	none	0.0	0.0			

Multi-Material Layers Group Cards. Repeat card 2 as many times as the number of AMMG layers present in the model.

Card 2	1	2	3	4	5	6	7	8
Variable	NID	MMGBLO						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ALESID	ALESID is a set ID defining the ALE domain/mesh whose hydrostatic pressure field due to gravity is initialized by this keyword. See Remarks 2 and 4.
STYPE	ALESID set type (see Remark 4): EQ.0: Part set ID EQ.1: Part ID EQ.2: Solid set ID for 3D case EQ.3: Shell set ID for 2D case
VECID	Vector ID of a vector defining the direction of gravity

VARIABLE	DESCRIPTION
GRAV	Magnitude of the gravitational acceleration. For example, in metric units the value is usually set to 9.80665 m/s^2.
PBASE	Nominal or reference pressure at the top surface of all fluid layers. By convention, the gravity direction points from the top layer to the bottom layer. Each fluid layer must be represented by an ALE multi-material group ID (AMMGID or MMG). See Remark 1.
NID	Node ID defining the top of an ALE fluid (AMMG) layer
MMGBLO	AMMG ID of the fluid layer immediately below this NID. Each node is defined in association with one AMMG layer below it. See Remark 3. In case of S-ALE, AMMG name (AMMGNM) could be also used in place of AMMGID. See Remark 5.

Remarks:

1. **Pressure in multi-layer fluids.** For models using multi-layer ALE fluids, the pressure at the top surface of the top fluid layer is set to PBASE, and the hydrostatic pressure is computed as following

$$P = P_{\text{base}} + \sum_{i=1}^{N_{\text{layers}}} \rho_i g h_i \, .$$

2. **Limitations on element formulation.** This keyword applies only to regular ALE parts with ELFORM = 11 and AET = 0 on *SECTION_SOLID or *SECTION_ALE2D (non-reservoir-type parts). This keyword cannot be used to initialize reservoir-type ALE parts (AET = 4). Also, ramping functions are not supported, so the loading is applied in one step at $t = 0$. For initializing a reservoir-type ALE domain, see *ALE_AMBIENT_HYDROSTATIC.

3. **Limitation on EOS model.** The keyword only supports *EOS_GRUNEISEN and *EOS_LINEAR_POLYNOMIAL, but only in the following two cases:

$$c_3 = c_4 = c_5 = c_6 = 0, \qquad E_0 = 0$$
$$c_4 = c_5 > 0, \qquad c_1 = c_2 = c_3 = c_6 = 0, \qquad V_0 = 0.$$

4. **S-ALE usage.** When this feature is used with S-ALE, the part and part set options might not make much sense because all elements inside an S-ALE mesh are assigned to one single part ID. To prescribe the initial hydrostatic pressure to all elements in an S-ALE mesh, you can certainly use a part ID (STYPE = 1). Prescribing the initial hydrostatic pressure to only some elements requires ALESID referring to a solid set (STYPE = 2) for 3D or shell set (STYPE = 3) for 2D that contains the desired S-ALE elements. To generate the set, use *SET_SOLID_GENERAL or *SET_SHELL_GENERAL with OPTION = SALECPT.

5. **AMMG name for S-ALE.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, each AMMG can only be referred to by their AMMGID. The AMMGID for each AMMG is based on the order of appearance of the AMMG in the input deck. For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_MULTI-MATERIAL_-GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUC-TURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the field AMMGNM. Each AMMG defined with that keyword can then be referred with either its name or its AMMGID (which is again based on order of appearance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, then the AMMGIDs may change. Then, you must find all those references and change them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

Example:

Model Summary: Consider a model consisting of two ALE parts, air on top of water.

H1 = AMMG1 = Air part above

H2 = AMMG2 = Water part below

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ (non-ambient) ALE materials (fluids) listed from top to bottom:
$
$ NID AT TOP OF A LAYER SURFACE          ALE MATERIAL LAYER BELOW THIS NODE
$ TOP OF 1st LAYER -------> 1722         ----------------------------------------
$                                        Air above   = PID 1 = H1 = AMMG1 (AET=0)
$ TOP OF 2nd LAYER -------> 1712         ----------------------------------------
$                                        Water below = PID 2 = H2 = AMMG2 (AET=0)
$ BOTTOM ------------------------------------------------------------------------
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*INITIAL_HYDROSTATIC_ALE
$   ALESID     STYPE     VECID      GRAV     PBASE
        12         0        11   9.80665  101325.0
$      NID    MMGBLO
      1722         1
      1712         2
*SET_PART_LIST
        12
         1         2
*ALE_MULTI-MATERIAL_GROUP
         1         1
         2         1
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH       CID
        11       0.0       1.0       0.0       0.0       0.0       0.0
*DEFINE_CURVE
         9
               0.000               0.000
               0.001               1.000
              10.000               1.000
*LOAD_BODY_Y
$     LCID        SF    LCIDDR        XC        YC        ZC
         9   9.80665         0       0.0       0.0       0.0
```

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*INITIAL_IMPULSE_MINE

Purpose: Apply initial velocities to the nodes of a three-dimensional structure due to the impulse imparted by the detonation of a buried land mine. This keyword cannot be used with 2D models or with rigid parts. It is also not supported by MPP (see workaround in Remark 3). This feature is based on the empirical model developed by [Tremblay 1998].

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	MTNT	RHOS	DEPTH	AREA	SCALE	not used	UNIT
Type	I	F	F	F	F	F		I
Default	none	0.0	0.0	0.0	0.0	1.0		1
Remarks	1	4						

Card 2	1	2	3	4	5	6	7	8
Variable	X	Y	Z	NIDMC	GVID	TBIRTH	PSID	SEARCH
Type	F	F	F	I	I	F	I	F
Default	0.0	0.0	0.0	0	none	0.0	0	0.0

VARIABLE	DESCRIPTION
SSID	Segment set ID. See *SET_SEGMENT and Remark 1.
MTNT	Equivalent mass of TNT. See Remark 4.
RHOS	Density of overburden soil.
DEPTH	Burial depth from the ground surface to the center of the mine. This value must be positive.
AREA	Cross sectional area of the mine.
SCALE	Scale factor for the impulse.

VARIABLE	DESCRIPTION
UNIT	Unit system. This must match the units used by finite element model. EQ.1: inch, dozen slugs (i.e., lbf × s^2/in), second, PSI (default) EQ.2: meter, kilogram, second, Pascal EQ.3: centimeter, gram, microsecond, megabar EQ.4: millimeter, kilogram, millisecond, GPa EQ.5: millimeter, metric ton, second, MPa EQ.6: millimeter, gram, millisecond, MPa
X, Y, Z	*x*-, *y*-, and *z*- coordinates of mine center.
NIDMC	Optional node ID representing the mine center (see *NODE). If defined then X, Y and Z are ignored.
GVID	Vector ID representing the vertically upward direction, that is, the normal to the ground surface. See *DEFINE_VECTOR.
TBIRTH	Birth time. Impulse is activated at this time.
PSID	Part set ID identifying the parts affected by the mine. See *SET_-PART. If the segment set defined by SSID includes segments of more than one part, PSID may be used to load only segments of identified parts. Otherwise, if PSID is set to zero, the part affected by the mine defaults to the part comprised by the nodes of the segment set.
SEARCH	Limit the search depth into the structure. Initial nodal velocity is distributed from the segment to a depth equal to the SEARCH value. The value must be positive. If set to zero, the search depth is unlimited and extends through the part(s) identified by PSID.

Remarks:

1. **Orientation.** Segment normals should nominally point toward the mine.

2. **Element types.** The segments should belong to 3D thin shell, solid, or thick shell elements. This keyword cannot be used with 2D geometries.

3. **Workaround to use MPP.** Although this feature is unavailable in MPP, a simple workaround for this limitation involves performing the initialization in SMP and then using that initial velocity field in MPP.

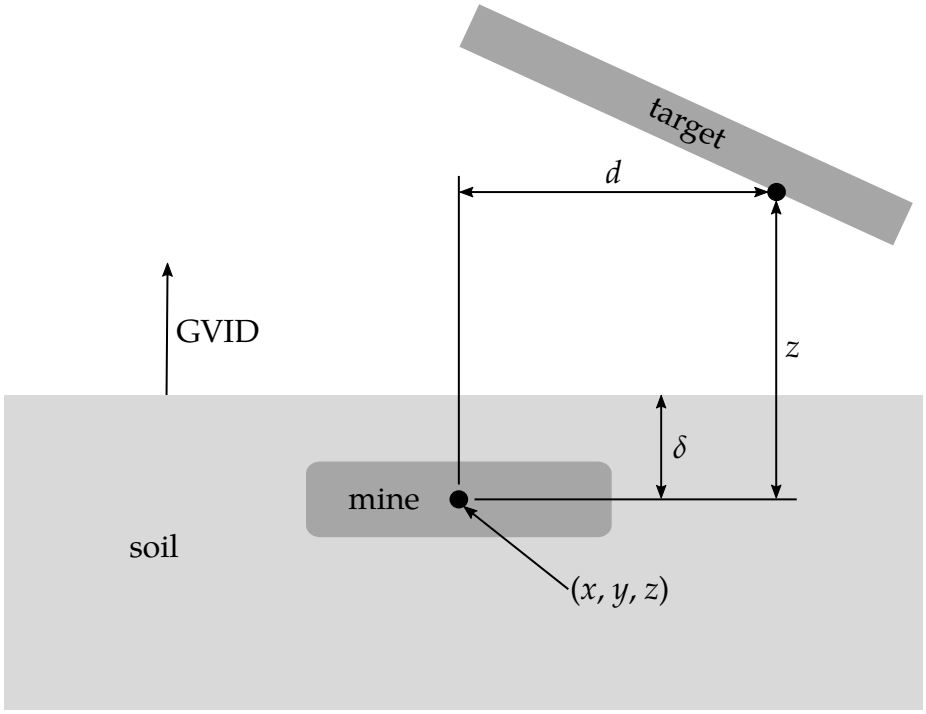


Figure 28-8. Schematic of the buried mine parameters.

4. **Equivalent mass of TNT.** Several methods can be used to approximate the equivalent mass of TNT for a given explosive. One method involves scaling the mass by the ratio of the squares of the Chapman-Jouguet detonation velocities given by:

$$M_{\text{TNT}} = M_e \frac{v_e^2}{v_{\text{TNT}}^2}$$

where M_{TNT} is the equivalent TNT mass and $v_{\text{TNT}}$is the Chapman-Jouguet detonation velocity of TNT. M_e and v_e are, respectively, the mass and C-J velocity of the explosive under investigation. LS-DYNA takes the density of "Standard" TNT to be $1.57 \frac{\text{g}}{\text{cm}^3}$ and v_{TNT} to be $0.693 \frac{\text{cm}}{\mu\text{s}}$

5. **Energy release.** This implementation assumes the energy release (heat of detonation) for 1 kilogram of TNT is 4.516 MJ.

6. **Error bounds.** Prediction of the impulse relies on an empirical approach which involves fitting curves to experimental results. The upper error bound is 1.8 times the predicted value and the lower is the predicted value divided by 1.8. Thus, if the predicted impulse is 10 kN-seconds, then the solution space ranges from 5.6 kN-sec to 18 kN-sec.

7. **Limits of the model's validity.** The computed impulse is valid when the following criteria are met:

$$0.106 \leq \frac{\delta}{z} \leq 1$$
$$6.35 \leq \frac{E/A}{\rho c^2 z} \leq 150$$
$$0.154 \leq \frac{\sqrt{A}}{z} \leq 4.48$$
$$0 \leq \frac{d}{z} \leq 19.3$$

where,

δ = the distance from the mine center to the ground surface (DEPTH)
z = the vertical distance from the mine center to the target point
E = the energy release of the explosive
A = the cross-sectional area of the mine (AREA)
ρ = the soil density (RHOS)
c = the wave speed in the soil
d = the lateral distance from the mine center to the target point.

See Figure 28-8.

References:

Tremblay, J.E., "Impulse on Blast Deflectors from a Landmine Explosion," DRDC Valcartier, DREV-TM-9814, (1998).

INITIAL_INTERNAL_DOF_SOLID_*OPTION

Available Options include:

TYPE3

TYPE4

Purpose: Initialize the internal degrees of freedom for solid element types 3 and 4.

Card 1	1	2	3	4	5	6	7	8
Variable	LID							
Type	I							
Default	none							

Value Cards. Include 12 cards for type 3 and 6 cards for type 4.

Card	1	2	3	4	5	6	7	8
Variable	VALX	VALY	VALZ					
Type	F	F	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
LID	Element ID.
VALX	x component of internal degree of freedom.
VALY	y component of internal degree of freedom.
VALZ	z component of internal degree of freedom.

Remarks:

1. **Internal DOF.** The internal degrees of freedom are specified in terms of the displacements of the corresponding mid-side nodes of the 20 node hex and the 10

node tetrahedron that are the basis of the type 3 and 4 solid elements, respectively.

*INITIAL_LAG_MAPPING_{OPTION}

The available options are

<BLANK>

WRITE

WRITE3DAXI

Purpose: This card initializes a 3D Lagrangian calculation with data from the last cycle of a preceding 2D or 3D Lagrangian calculation.

In its *INITIAL_LAG_MAPPING form (<BLANK> option), this keyword causes data to be read in from a *mapping file*. With the WRITE option active, this card is used to set which parts are written to the mapping file. The mapping file's filename is specified using the "lagmap=" command line argument (see Remarks 1 and 2). The option WRITE3DAXI causes LS-DYNA to output the mapping file for a 3D axisymmetric mesh as if it was generated by a 2D axisymmetric model.

The following transitions are allowed:

2D → 2D	3D → 3D
2D → 3D	3D → 2D

Card 1	1	2	3	4	5	6	7	8
Variable	SETID							
Type	I							

Mesh Mapping Card. Additional card for <BLANK> or WRITE3DAXI keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	XP	YP	ZP	VECID	ANGLE	NELANGL	TBEG	
Type	F	F	F	I	F	I	F	
Default	0.0	0.0	0.0	none	none	0	0.0	

VARIABLE	DESCRIPTION
SETID	Part set ID. See Remarks 3 and 4.
XP	*x*-position of a point belonging to the plane from which the 3D mesh is generated (only for a 2D to 3D mapping or 3Daxi to 3D mapping). See Remark 5.
YP	*y*-position of a point belonging to the plane from which the 3D mesh is generated (only for a 2D to 3D mapping or 3Daxi to 3D mapping). See Remark 5.
ZP	*z*-position of a point belonging to the plane from which the 3D mesh is generated (only for a 2D to 3D mapping or 3Daxi to 3D mapping). See Remark 5.
VECID	ID of the rotation axis (symmetric axis for a 2D to 3D mapping or 3Daxi to 3D mapping) defined by *DEFINE_VECTOR. See Remark 5.
ANGLE	Angle of rotation around an axis defined by *DEFINE_VECTOR. See Remark 5.
NELANGL	Mapping parameter. See Remark 5. GT. 0: For a 2D to 3D mapping, number of elements to create in the azimuthal direction for ANGLE EQ.-1: No mesh is generated or projected. EQ.-2: For a 3D to 3D mapping, ANGLE only rotates the data from the mapping file (not the current mesh). EQ.-3: No mesh is generated or projected aside from the boundary nodes of the current mesh being projected onto the boundary faces of the previous mesh.
TBEG	Time at which the current run starts if it is greater than the previous run's termination time. By default, the current run starts at the termination time of the previous run, which is stored in the mapping file. TBEG only replaces the termination time of the previous run if it exceeds that termination time.

Remarks:

1. **The mapping file as output.** In the absence of a *INITIAL_LAG_MAPPING card, adding a "lagmap=" argument to the command line causes LS-DYNA to write a mapping file. This file contains the following nodal and element data:

- nodal coordinates (initial and last steps)
- nodal velocities
- nodal temperatures (if *CONTROL_THERMAL_SOLVER is used)
- part IDs
- element connectivities
- element centers
- densities
- volume fractions
- stresses
- plastic strains
- internal energies
- bulk viscosities
- relative volumes
- strains (only if the strain tensor is flagged for output using the variable STRFLG in *DATABASE_EXTENT_BINARY)
- extra history variables

2. **The mapping file as input.** If the keyword INITIAL_LAG_MAPPING is in the input deck and the "lagmap=" argument is in the command line, then Lagrangian data is read from the mapping file defined by "lagmap=" to initialize the run.

3. **Part sets (write).** The part set, SETID, defines which parts are involved in the mapping. The WRITE option can be used to write data in the mapping file for *only* the parts specified by the set. If the keyword *INITIAL_LAG_MAPPING_WRITE is not included in the input deck, then *all* Lagrangian parts are written in the mapping file during the last cycle.

4. **Part sets (read).** The mapping initializes the data for every node and element defined by SETID within the domain swept by the 2D mesh or the region initially occupied by the previous 3D mesh. For nodes and elements outside of SETID it has no effect.

5. **Embedding.** The first point in the definition of the rotation axis VECID specifies the origin location for the mesh of the previous run in the current 3D space. The 2D to 3D mapping depends on whether or not a 3D mesh has already been defined. The 3D to 3D mapping requires a pre-existing mesh.

a) *No Mesh Case for a 2D to 3D mapping*. If there is no 3D mesh (no element with parts in SETID), the point (XP,YP,ZP) together with the symmetry axis (VECID) are used to generate a mesh. The point defines the plane in which the 2D mesh is embedded. The 3D mesh is generated by rotating the 2D mesh around the axis. The point (XP,YP,ZP) must not be on the symmetry axis. ANGLE defines the angle of rotation in degrees. The rotation is counterclockwise when viewed from the axis head. NELANGL is the number of elements to generate in the azimuthal direction.

b) *Pre-existing Mesh Case for a 2D to 3D mapping.* If there is a 3D mesh (elements with parts in SETID), the nodes should be within the domain swept by the initial positions of the 2D mesh. Then, the nodes are projected to new locations based on the positions of the previous run's last mesh. If NELANGL = -1, the nodes should be within the domain swept by the final positions of the 2D mesh and the mesh location is not modified.

c) *Pre-existing Mesh Case for a 3D to 3D mapping.* If there is a 3D mesh (elements with parts in SETID) and NELANGL ≠ -1, the nodes should be in the region initially occupied by the previous 3D mesh. Then, the nodes are mapped to new locations based on the previous run's last mesh positions. If ANGLE is defined, the pre-existing mesh is rotated by ANGLE about VECID. The first point in VECID is still the previous origin location and it can be used to translate the pre-existing mesh. However, if NELANGL = -1, the nodes should be in the region finally occupied by the previous 3D mesh and the mesh location is not modified.

d) *3D Axisymmetric to 3D (or 2D) mapping (keyword option WRITE3DAXI).* In the 3D axisymmetric model (1st step model with only solids and/or shells), the point (XP,YP,ZP) defines the radial plane of the 3D axisymmetric mesh used to generate the mapping file. The point (XP,YP,ZP) must not be on the symmetry axis defined by VECID. It is recommended to locate this point on one of the two radial boundaries of the 3D axisymmetric mesh as the nodes of these boundaries do not move out of their plane in the orthoradial direction. The solid faces and shell edges in the radial plane defined by the point (XP,YP,ZP) become shells and beams, respectively, in the mapping file as if a 2D axisymmetric model is run to output this file. The Remarks 5a) and 5b) are still valid to set up the 3D or 2D model of the 2nd step that reads the mapping file.

*INITIAL_MOMENTUM

Purpose: Define initial momentum to be deposited in solid elements. This keyword crudely simulates an impulsive type of loading.

Card	1	2	3	4	5	6	7	8
Variable	EID	MX	MY	MZ	DEPT			
Type	I	F	F	F	F			
Default	none	0.	0.	0.	0.			

VARIABLE	DESCRIPTION
EID	Element ID
MX	Initial x-momentum
MY	Initial y-momentum
MZ	Initial z-momentum
DEPT	Deposition time

Remarks:

Assuming an 8-noded brick element, the specified momentum is distributed equally to each node of the element, and the change in velocity due to this deposited momentum is added to the nodal velocity at the specified deposition time.

***INITIAL_PWP_DEPTH**_{OPTION}_

The available options include:

<BLANK>

SET

Purpose: Initialize pore water pressure in solid elements where a non-hydrostatic profile is required.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	LC						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
PID	Part ID or Part Set ID for the SET keyword option
LC	Curve of pore water pressure head (length units) as a function of *z*-coordinate

Remarks:

This feature overrides the automatically calculated hydrostatic pressure profile; the hydrostatic pressure profile is the default behavior for a part with pore fluid. The points in the curve must be ordered with the most negative *z*-coordinate first – this order looks "upside-down" on the page.

*INITIAL_PWP_NODAL_DATA

Purpose: Initialize nodal pore pressure data. This keyword is written by LS-DYNA to the dynain file if *CONTROL_PORE_FLUID is present, to enable subsequent analyses beginning from a state reached in a previous analysis. It is not expected that users will create or modify this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	NHISV	PID					
Type	I	I	I					
Default	none	0	0					

History Variable Cards. Define as many cards as necessary to define NHISV variables. For example, if NHISV = 25, then 5 cards are required.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
NID	Node ID
NHISV	Number of nodal pore pressure history variables.
PID	Part ID with which this node is associated for purposes of evaluating pore fluid related properties.
HISV*i*	Define NHISV history variables.

*INITIAL_SOLID_VOLUME

Purpose: Recalculate and reset initial volume of solid elements using material models with EOS before analysis if the original nodal position has been moved by nodal projections in contact initialization. This option eliminates calculation of non-physical initial hydrostatic pressure due to the nodal repositioning.

Card Sets.

Part Set Card. Define as many of this card as desired. The input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PSID	Part set ID.

*INITIAL_STRAIN_BEAM

Purpose: Initialize displacements of discrete beams for output to DISBOUT (see *DATABASE_DISBOUT). This keyword can be used for multi-stage analyses where the accumulated displacements of discrete beams are needed for post-processing. These displacements do not affect the solution. They are for output only. If a dynain file is requested, this keyword is automatically written to the dynain file provided that (a) the model contains discrete beams and (b) *DATABASE_DISBOUT is set.

Element Cards. Define initial displacements for as many beam elements as desired. This input terminates when the next keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	RDISP	SDISP	TDISP	RROT	SROT	TROT	
Type	I	F	F	F	F	F	F	
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
EID	Element ID (must be a discrete beam, i.e., ELFORM = 6 on *SECTION_BEAM).
RDISP	The initial displacement along the local *r*-axis
SDISP	The initial displacement along the local *s*-axis
TDISP	The initial displacement along the local *t*-axis
RROT	The initial rotation about the local *r*-axis
SROT	The initial rotation about the local *s*-axis
TROT	The initial rotation about the local *t*-axis

*INITIAL_STRAIN_IGA_SHELL

Purpose: Initialize strain tensor and thicknesses at in-plane integration points for isogeometric shell elements, defined via *IGA_SHELL. This keyword is primarily for multi-stage metal forming operations where the accumulated strain and the thickness change is of interest.

The strain tensors are defined at the inner and outer integration points and are used for post-processing only. There is no interpolation with this option, and the strains are defined in the global Cartesian coordinate system.

Note: The in-plane integration point locations, defined with R and S in this keyword, may not match the integration point locations used during the computation. In such cases, LS-DYNA uses the information of the closest data point in this keyword for the analysis.

Card Sets. Define as many IGA_SHELL elements in this section as desired, *one set of cards per element*. The input is assumed to terminate when a new keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NPLANE	ITHK	LARGE				
Type	I	I	I	I				
Default	none	none	0	0				

Strain Cards for LARGE = 0.

For the NPLANE integration points, define two pairs of cards below, one pair for the inner integration point and the other pair for the outer integration point, respectively.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					
Default	none	none	none					

Card 3	1	2	3	4	5	6	7	8
Variable	EPSXX	EPSYY	EPSZZ	EPSXY	EPSYZ	EPSZX	THKI	
Type	F	F	F	F	F	F	F	

Strain Cards for LARGE = 1.

For the NPLANE integration points, define two pairs of cards below, one pair for the inner integration point and the other pair for the outer integration point, respectively.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		EPSXX		EPSYY	
Type	F		F		F		F		F	

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	EPSZZ		EPSXY		EPSYZ		EPSZX		THKI	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID	IGA shell element ID
NPLANE	Number of in-plane integration points being output
ITHK	Flag for initialization of thicknesses at in-plane IPs: EQ.0: Off EQ.1: On
LARGE	Large format flag: EQ.0: Off EQ.1: On. Each strain field is twice as long for higher precision.

VARIABLE	DESCRIPTION
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to *IGA_2D_NURBS_XYZ patch definition)
S	Parametric s-coordinate of location of in-plane integration point (with respect to *IGA_2D_NURBS_XYZ patch definition)
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
EPS*ij*	Define the *ij* strain component. The strains are defined in the *global* cartesian system.
THKI	Thickness value at in-plane integration point

*INITIAL_STRAIN_SHELL_{OPTION}

The available options include:

<BLANK>

SET

Purpose: Initialize strain tensor for a shell element. This option is primarily for multi-stage metal forming operations where the accumulated strain is of interest.

These strain tensors are defined at the inner and outer integration points and are used for post-processing only. There is no interpolation with this option and the strains are defined in the global Cartesian coordinate system. The *DATABASE_EXTENT_BINARY flag STRFLG must be set to unity for this option to work. When the keyword option is blank, the strains at all the integration points can be defined by providing nonzero NPLANE and NTHICK and setting INTOUT flag of *DATABASE_EXTENT_BINARY to either "STRAIN" or "ALL."

Card Sets. Define as many shell elements in this section as desired, *one set of cards per element*. The input is assumed to terminate when a new keyword is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NPLANE	NTHICK	LARGE				ILOCAL
Type	I	I	I	I				I
Default	none	none	none	0				0

Ordering of Integration Points

When NPLANE and NTHICK are defined, include NPLANE × NTHICK cards below. For each through thickness point define NPLANE points. NPLANE should be either 1 or 4 corresponding to either 1 or 4 Gauss integration points. If four integration points are specified, they should be ordered such that their in-plane parametric coordinates are at:

$$\left(-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\quad \left(-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right)$$

respectively.

Strain Cards for LARGE = 0.

When NPLANE and NTHICK are not defined or when the SET option is used, define two cards below, one for the inner integration point and the other for the outer integration point, respectively. Otherwise define NPLANE × NTHICK cards, one card for each integration point.

Card 2	1	2	3	4	5	6	7	8
Variable	EPSxx	EPSyy	EPSzz	EPSxy	EPSyz	EPSzx	T	
Type	F	F	F	F	F	F	F	
Default	none	none	none	0.			0.	

Strain Cards for LARGE = 1.

When NPLANE and NTHICK are not defined or when the SET option is used, define two pairs of cards below, one pair for the inner integration point and the other pair for the outer integration point, respectively. Otherwise define NPLANE × NTHICK pairs of cards; one pair for each integration point.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	EPSxx		EPSyy		EPSzz		EPSxy		EPSyz	
Type	F		F		F		F		F	

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	EPSzx		T							
Type	F		F							

VARIABLE	DESCRIPTION
EID	Element ID or shell element set ID when the SET option is used.

VARIABLE	DESCRIPTION
NPLANE	Number of in-plane integration points being output (not read when the SET option is used).
NTHICK	Number of integration points through the thickness (not read when the SET option is used).
LARGE	Large format flag: EQ.0: off, EQ.1: on. Each strain field is twice as long for higher precision.
ILOCAL	Flag for coordinate system of strain components: EQ.0: global, EQ.1: local (not supported).
EPS*ij*	Define the *ij* strain component. The strains are defined in the GLOBAL Cartesian system.
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.

*INITIAL_STRAIN_SHELL_NURBS_PATCH

Purpose: Initialize strain tensor and thicknesses at in-plane integration points for isogeometric shell elements. This option is primarily for multi-stage metal forming operations where the accumulated strain and the thickness change is of interest.

The strain tensors are defined at the inner and outer integration points and are used for post-processing only. There is no interpolation with this option and the strains are defined in the global Cartesian coordinate system.

Card Sets. Define as many NURBS shell elements in this section as desired, *one set of cards per element*. The input is assumed to terminate when a new keyword is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NPLANE	ITHK	LARGE				
Type	I	I	I	I				
Default	none	none	0	0				

Strain Cards for LARGE = 0.

For the NPLANE integration points, define two pairs of cards below, one pair for the inner integration point and the other pair for the outer integration point, respectively.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					
Default	none	none	none					

Card 3	1	2	3	4	5	6	7	8
Variable	EPSXX	EPSYY	EPSZZ	EPSXY	EPSYZ	EPSZX	THKI	
Type	F	F	F	F	F	F	F	

Strain Cards for LARGE = 1.

For the NPLANE integration points, define two pairs of cards below, one pair for the inner integration point and the other pair for the outer integration point, respectively.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		EPSXX		EPSYY	
Type	F		F		F		F		F	

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	EPSZZ		EPSXY		EPSYZ		EPSZX		THKI	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID	NURBS element ID
NPLANE	Number of in-plane integration points being output
ITHK	Flag for initialization of thicknesses at in-plane IPs: EQ.0: off EQ.1: on
LARGE	Large format flag: EQ.0: off EQ.1: on. Each strain field is twice as long for higher precision.

VARIABLE	DESCRIPTION
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
S	Parametric s-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
EPS*ij*	Define the *ij* strain component. The strains are defined in the GLOBAL Cartesian system.
THKI	Thickness value at in-plane integration point

*INITIAL_STRAIN_SOLID_{OPTION}

The available options include:

<BLANK>

SET

Purpose: Initialize strain tensor at element center. This keyword can be used for multi-stage metal forming operations where the accumulated strain is of interest.

These strain tensors are defined at the element center and are used for post-processing only. The strains are defined in the global Cartesian coordinate system. The ones digit of STRFLG in *DATABASE_EXTENT_BINARY is automatically set to unity when *INITIAL_STRAIN_SOLID_{OPTION} is included in the input deck. This capability is not available for cohesive elements since it is based on displacements, not strains.

Card Sets. Define as many solid elements in this section as desired: *one pair of cards per element*. The input is assumed to terminate when a new keyword ("*") card is detected.

Element ID Cards.

Card 1	1	2	3	4	5	6	7	8
Variable	EID							
Type	I							
Default	none							

Strain Cards.

Card 2	1	2	3	4	5	6	7	8
Variable	EPSxx	EPSyy	EPSzz	EPSxy	EPSyz	EPSzx		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
EID	Element ID or solid element set ID when the SET keyword option is used.

VARIABLE	DESCRIPTION
EPS*ij*	Define the ij^{th} strain component. The strains are defined in the global cartesian system.

*INITIAL_STRAIN_SOLID_NURBS_PATCH

Purpose: Initialize strain at integration points for isogeometric solid elements.

These strain tensors are used for post-processing only. There is no interpolation with this option and the strains are defined in the global Cartesian coordinate system.

Card Sets. Define as many NURBs solid elements in this section as desired, *one set of cards per element*. The input is assumed to terminate when a new keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NIP	LARGE					
Type	I	I	I					
Default	none	none	0					

Strain Cards for LARGE = 0.

For each NIP integration points, define a pair of cards below.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					
Default	none	none	none					

Card 3	1	2	3	4	5	6	7	8
Variable	EPSXX	EPSYY	EPSZZ	EPSXY	EPSYZ	EPSZX		
Type	F	F	F	F	F	F		

Strain Cards for LARGE = 1.

For each NIP integration points, define a pair of cards below.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		EPSXX		EPSYY	
Type	F		F		F		F		F	

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	EPSZZ		EPSXY		EPSYZ		EPSZX			
Type	F		F		F		F			

VARIABLE	DESCRIPTION
EID	NURBS element ID
NIP	Number of integration points being output
LARGE	Large format flag: EQ.0: off EQ.1: on. Each strain field is twice as long for higher precision.
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
S	Parametric *s*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
T	Parametric *t*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
EPS*ij*	Define the *ij* strain component. The strains are defined in the GLOBAL Cartesian system.

*INITIAL_STRAIN_TSHELL_{OPTION}

The available options include:

<BLANK>

SET

Purpose: Initialize the strain tensors for thick shell elements.

Strain tensors are defined at the inner and outer integration points and are used for post-processing only. Strain tensors are defined in the global Cartesian coordinate system. The STRFLG flag on *DATABASE_EXTENT_BINARY must be set to unity for this keyword to work. Initialize as many elements as needed.

Card Sets. For each element, include a set of Cards 1, 2, and 3, where Card 2 is for the inner layer and Card 3 is for the outer layer. The input is assumed to terminate when a new keyword (*) card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID							
Type	I							
Default	none							

Strain Cards. Card 2 is the strain at the inner layer. Card 3 is the strain at the outer layer.

Cards 2, 3	1	2	3	4	5	6	7	8
Variable	EPSXX	EPSYY	EPSZZ	EPSXY	EPSYZ	EPSZX		
Type	F	F	F	F	F	F		
Default	0	0	0	0	0	0		

VARIABLE	DESCRIPTION
EID/SID	Element ID or thick shell element set ID when the SET option is used, see *SET_TSHELL.

VARIABLE	DESCRIPTION
EPS*ij*	Define the ij^{th} strain component. The strains are defined in the global cartesian system.

*INITIAL_STRESS_BEAM

Purpose: Initialize stresses, plastic strains, and history variables for Hughes-Liu beam elements and truss beam elements, or the axial force, moment resultants, and history variables for Belytschko-Schwer beam elements and discrete beams.

Card Summary:

Card Sets. Define as many beams in this section as desired. Each set consists of one Card 1 and several additional cards depending on variables NPTS, LARGE, NHISV, and NAXES. The input terminates when a new keyword ("*") card is detected.

Card 1. This card is required.

EID	RULE	NPTS	LOCAL	LARGE	NHISV	NAXES	

Card 2a. This card is included for Belytschko-Schwer beams if LARGE = 0.

F11	T11	M12	M13	M22	M23	PARM	

Card 2b.1. This card is included for Belytschko-Schwer beams if LARGE = 1.

F11	T11	M12	M13	M22

Card 2b.2. This card is included for Belytschko-Schwer beams if LARGE = 1. Include additional cards of this format to include all of the history variables.

M23	PARM	HISV1	HISV2	HISV3

Card 2c. Include this card NPTS times for Hughes-Liu or truss beams if LARGE = 0.

SIG11	SIG22	SIG33	SIG12	SIG23	SIG31	EPS	

Card 2d.1. Include NPTS sets of this card and Card 2d.2 for Hughes-Liu or truss beams if LARGE = 1.

SIG11	SIG22	SIG33	SIG12	SIG23

Card 2d.2. For each of the NPTS sets include additional cards of this format to include all of the history variables.

SIG31	EPS	HISV1	HISV2	HISV3

Card 3.1. This card is included if NAXES = 12.

AX1	AX2	AX3	AX4	AX5

Card 3.2. This card is included if NAXES = 12.

AX6	AX7	AX8	AX9	AX10

Card 3.3. This card is included if NAXES = 12.

AX11	AX12			

Card 1	1	2	3	4	5	6	7	8
Variable	EID	RULE	NPTS	LOCAL	LARGE	NHISV	NAXES	
Type	I	I	I	I	I	I	I	
Default	none	2	none	0	0	0	0	

VARIABLE	DESCRIPTION
EID	Element ID
RULE	Integration rule type number: EQ.1: 1 × 1 Gauss quadrature, EQ.2: 2 × 2 Gauss quadrature (default beam), EQ.3: 3 × 3 Gauss quadrature, EQ.4: 3 × 3 Lobatto quadrature, EQ.5: 4 × 4 Gauss quadrature.
NPTS	Number of integration points. For the Belytschko-Schwer resultant beam element, NPTS = 1.
LOCAL	Coordinate system for stresses: EQ.0: Stress components are defined in the global coordinate system. EQ.1: Stress components are defined in the local beam system. In the local system components SIG22, SIG33, and SIG23 are set to 0.0.
LARGE	Format size: EQ.0: Off,

VARIABLE	DESCRIPTION
	EQ.1: On. Each field is twice as long for higher precision.
NHISV	Number of additional history variables. Only available for LARGE = 1.
NAXES	Number of variables giving the beam's local axes (0 or 12). See Remark 2.

Belytschko-Schwer Card for LARGE = 0. Additional card for the Belytschko-Schwer beam.

Card 2a	1	2	3	4	5	6	7	8
Variable	F11	T11	M12	M13	M22	M23	PARM	
Type	F	F	F	F	F	F	F	

Belytschko-Schwer Cards for LARGE = 1. Additional cards for the Belytschko-Schwer beam. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 2b.1	1	2	3	4	5	6	7	8	9	10
Variable	F11		T11		M12		M13		M22	
Type	F		F		F		F		F	

Card 2b.2	1	2	3	4	5	6	7	8	9	10
Variable	M23		PARM		HISV1		HISV2		HISV3	
Type	F		F		F		F		F	

Hughes-Liu Cards for LARGE = 0. Additional cards for the Hughes-Liu or truss beam. Include NPTS additional cards, one per integration point. See Remark 1.

Card 2c	1	2	3	4	5	6	7	8
Variable	SIG11	SIG22	SIG33	SIG12	SIG23	SIG31	EPS	
Type	F	F	F	F	F	F	F	

Hughes-Liu Cards for LARGE = 1. Additional cards for the Hughes-Liu or truss beam. Include NPTS additional card sets, one per integration point. Include as many cards in one card set as necessary to collect NHISV (see Card 1) history variables. See Remark 1.

Card 2d.1	1	2	3	4	5	6	7	8	9	10
Variable	SIG11		SIG22		SIG33		SIG12		SIG23	
Type	F		F		F		F		F	

Card 2d.2	1	2	3	4	5	6	7	8	9	10
Variable	SIG31		EPS		HISV1		HISV2		HISV3	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
F11	Axial force resultant along local beam axis 1
T11	Torsional moment resultant about local beam axis 1
M12	Moment resultant at node 1 about local beam axis 2
M13	Moment resultant at node 1 about local beam axis 3
M22	Moment resultant at node 2 about local beam axis 2
M23	Moment resultant at node 2 about local beam axis 3
PARM	Generally not used.
SIG*ij*	Define the *ij* stress component

VARIABLE	DESCRIPTION
EPS	Effective plastic strain
HISV*n*	Define the *n*th history variable

Optional Local Axes Cards for NAXES = 12. Additional cards for definition of local axes values. These 12 values are internally used by LS-DYNA for the mapping between local beam element system and global coordinate system. They are automatically written to the dynain file if *INTERFACE_SPRINGBACK_LSDYNA or *CONTROL_STAGED_CONSTRUCTION is used.

Card 3.1	1	2	3	4	5	6	7	8	9	10
Variable	AX1		AX2		AX3		AX4		AX5	
Type	F		F		F		F		F	

Card 3.2	1	2	3	4	5	6	7	8	9	10
Variable	AX6		AX7		AX8		AX9		AX10	
Type	F		F		F		F		F	

Card 3.3	1	2	3	4	5	6	7	8	9	10
Variable	AX11		AX12							
Type	F		F							

VARIABLE	DESCRIPTION
AX*n*	The *n*th local axes value. See Remark 2.

Remarks:

1. **Axial stress for truss beams.** Note that only SIG11 is nonzero for truss elements since they only carry axial forces.

2. **NAXES.** NAXES = 12 indicates providing AX1, …, AX12 for determining the local axes of the beam. These values are useful in cases where an initial analysis is performed, the beam has rotated during the analysis, and a subsequent analysis is to be performed. LS-DYNA uses them to properly orient the beam in the subsequent analysis. *The user is not expected to enter these values.* LS-DYNA automatically writes this keyword with these values to a dynain file when indicated in the input deck (such as with *INTERFACE_SPRINGBACK_LSDYNA). These AX1, …, AX12 correspond to the $\hat{Y}$ terms seen in Equation 7.12 of the R14 Theory manual.

*INITIAL_STRESS_DEPTH_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Initialize solid element stresses where stress is a function of depth. This feature is intended only for material models whose stresses are updated incrementally and whose stress state does not depend on internal history variables.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	RO_G	ZDATUM	KFACT	LC	LCH	LCK0	
Type	I	F	F	F	I	I	I	
Default	none	none	none	0.0	opt	opt	opt	

VARIABLE	DESCRIPTION
PID/PSID	Part ID or Part Set ID for the SET option
RO_G	Stress per unit elevation above datum, which is usually ρ_g = density × gravity.
ZDATUM	z-coordinate of datum
KFACT	x and y-stresses = KFACT × z-stress
LC	Optional curve of stress as a function of the z-coordinate (ZDATUM is ignored with this option)
LCH	Optional curve of horizontal stress as function of the z-coordinate (KFACT is ignored with this option)
LCK0	Optional curve of K0 (ratio of horizontal stress to vertical stress) as a function z-coordinate. KFACT and LCH are ignored with this option. The x-axis of the curve is the z-coordinate, the y-axis is K0.

Remarks:

1. **Model Description.** With this keyword stress is calculated according to,

$$\sigma_z = \mathrm{RO_G} \times (\mathrm{Z_{element}} - \mathrm{ZDATUM}) ,$$

where $\mathrm{Z_{element}}$ is the z-coordinate of the centroid of the element. It is assumed that the z-axis points vertically upwards. For a 2D problem (axisymmetric or plane strain), replace z in this documentation with y. To generate compressive stresses, the datum should be above the highest element, that is, the datum should have a more positive z-coordinate. For instance, this is at the surface of the soil in geotechnics simulations. If the curve, LC, is defined, it overrides RO_-G and ZDATUM. Note that the points in the curve should be ordered with most negative z-coordinate first.

2. **Pore Water.** If pore water is present, the stresses input here are effective (meaning soil skeleton stresses only, not including the pore pressure). The pore water pressures will automatically be initialized to hydrostatic, or by *INITIAL_-PWP_DEPTH or *BOUNDARY_PWP_TABLE if those cards are present. Effective vertical stress at a given depth is usually equal to the buoyant weight of the soil between that depth and the surface, where buoyant weight is calculated from the density of the saturated soil minus the density of water. The density of saturated soil is the RO given on the *MAT card, and the density of water is the PF_RHO given on *BOUNDARY_PORE_FLUID or *CONTROL_PORE_FLUID. Note also that the density of saturated soil is somewhat higher than the density of dry soil, and the value of RO on the *MAT card should reflect this.

3. **Stress at Integration Points.** For fully integrated elements, the stress calculated for the element centroid is applied at all the integration points. There is no stress gradation up the height of the element. This is to ensure equilibrium with gravity loading, which is applied at the nodes and therefore results in stress that is uniformly distributed up the height of the element.

*INITIAL_STRESS_DES

Purpose: Initialize stresses and coordination number for DES elements.

Element Cards. Define as many DES elements in this section as desired. The new keyword ("*") terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	COOR
Type	I	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EID	DES particle ID
SIG*ij*	Define the ij^{th} stress component. Stresses are defined in the global cartesian system.
COOR	Define the coordination number of the DES element

*INITIAL_STRESS_IGA_SHELL

Purpose: Initialize stresses, history variables, and the effective plastic strain for isogeometric shell elements, defined via *IGA_SHELL. Materials that do not use an incremental formulation for the stress update may not be initializable with this card.

Note: The in-plane integration point locations, defined with R and S in this keyword, might not match the integration point locations used during the computation. In such cases, LS-DYNA uses the information of the closest data point in this keyword for the analysis. When the number of through-thickness integration points differs between this keyword and the actual analysis, LS-DYNA interpolates accordingly.

Card Sets per Element. Define as many IGA_SHELL elements in this section as desired. The input is assumed to terminate when a new keyword ("*") card is detected.

Element Card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NPLANE	NTHICK	NHISV	LARGE			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

Solid Mechanics Data Card for LARGE = 0.

The following set of cards: "Stress Cards" and "History Variable Cards" should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card 1. Additional Card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					

Stress Card 2. Additional Card for LARGE = 0.

Card 3	1	2	3	4	5	6	7	8
Variable	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPSP	
Type	F	F	F	F	F	F	F	

History Variable Cards. Additional Cards for LARGE = 0. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8
Variable	HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8
Type	F	F	F	F	F	F	F	F

Solid Mechanics Data Card for LARGE = 1.

The following set of cards: "Stress Cards" and "History Variable Cards" should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card 1. Additional Card for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		SIGXX		SIGYY	
Type	F		F		F		F		F	

Stress Card 2. Additional Card for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGZZ		SIGXY		SIGYZ		SIGZX		EPS	
Type	F		F		F		F		F	

History Variable Cards. Additional Cards for LARGE = 1. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID	IGA shell element ID
NPLANE	Number of in-plane integration points being output
NTHICK	Number of integration points through the thickness
NHISV	Number of additional history variables.
LARGE	Format size. See cards above. EQ.0: Off EQ.1: On
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to *IGA_2D_NURBS_XYZ-patch definition)
S	Parametric *s*-coordinate of location of in-plane integration point (with respect to *IGA_2D_NURBS_XYZ-patch definition)
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
SIG*ij*	Define the *ij* stress component. The stresses are defined in the *global* cartesian system.
EPS	Effective plastic strain
HISV*n*	Define the n^{th} history variable

*INITIAL_STRESS_SECTION

Purpose: Initialize (pre-load) the stress normal to a plane intersecting the model using a curve (*DEFINE_CURVE). The initialization is applied to a set of solid elements that are specified using the *DATABASE_CROSS_SECTION card. This option is compatible only with those material models that are incrementally updated (such as elastic, viscoelastic, and elastoplastic). Except as noted in Remark 3, rubbers, foams, and materials that are combined with equations-of-state cannot be initialized with this card.

Card 1	1	2	3	4	5	6	7	8
Variable	ISSID	CSID	LCID	PSID	VID	IZSHEAR	ISTIFF	
Type	I	I	I	I	I	I	I	
Default	none	none	none	none	none	0	0	

VARIABLE	DESCRIPTION
ISSID	ID for this card.
CSID	Cross-section ID. See *DATABASE_CROSS_SECTION.
LCID	Load curve ID defining preload stress as a function of time. When the load curve ends or goes to zero, the initialization is assumed to be completed. See Remark 2.
PSID	Part set ID. Stress is initialized on only those parts included in *both* PSID from this card and the PSID field from the associated *DATABASE_CROSS_SECTION card.
VID	Vector ID (see *DEFINE_VECTOR) specifies the direction normal to the cross section (CSID). VID *must* be set when the "SET" variant of *DATABASE_CROSS_SECTION (*DATABASE_CROSS_SECTION_SET) is used. If the cross section is defined using the PLANE option, the normal used in the definition of the plane is used when VID is left undefined.
IZSHEAR	Shear stress flag: EQ.0: Shear stresses are prescribed as zero during the time the curve is acting to prescribe normal stress.

VARIABLE	DESCRIPTION
	EQ.1: Shear stresses are allowed to develop during the time the curve is acting to prescribe normal stress. For implicit the section can also take bending and is identical to 2; see Remark 5.
	EQ.2: Shear and bending stresses are allowed to develop during the time the curve is acting to prescribe normal stress; see Remark 5.
ISTIFF	Artificial stiffness. Simulates additional linearly elastic "ghost" elements in the cross section. These elements prevent mesh distortion by stiffening up the structure.
	GT.0: Load curve ID defining stiffness fraction as a function of time. The stiffness of the ghost elements is the load curve value times the stiffness of the material in the part. Since the ghost element stress counteracts the preload stress the fraction should be low (1% or less). The total section stress is the preload stress minus the ghost element stress.
	LT.0: |ISTIFF| is the load curve ID for the stiffness fraction as a function of time. The preload stress is here automatically adjusted ($\pm$10% of original prestress values) such that the total section stress corresponds to the curve in LCID.

Remarks:

1. **Dynamic relaxation issues.** To achieve convergence during explicit dynamic relaxation, applying damping options is very important. If contact is active, contact damping is recommended with a value between 10 - 20 percent. Additional damping applied using *DAMPING_PART_STIFFNESS also speeds up convergence, where a coefficient of 0.10 is effective. If damping is not used, convergence may not be possible.

2. **Best practices for quasi-static load ramp-up.** When defining the load curve, LCID, a ramp starting at the origin should be used to increase the stress to the desired value. The time duration of the ramp should produce a quasi-static response. When the end of the load curve is reached, or when the value of the load decreases from its maximum value, the initialization stops. If the load curve begins at the desired stress value, that is, no ramp, convergence will take much longer, since the impulsive-like load created by the initial stress can excite nearly every frequency in the structural system where stress is initialized.

3. **Supported materials.** This option currently applies only to materials that are incrementally updated. Hyperelastic materials and those that require an equation of state are not currently supported. However, materials 57, 73, and 83 can be initialized with this approach.

4. **Supported elements.** Solid elements types 1, 2, 3, 4, 9, 10, 15, 16, 17, and 18 are supported. ALE elements are not supported. Element forms 16 and 17 with mid-side nodes may only work well if the area of the cross-section where the preload stress is applied does not vary. In other words, element forms 16 or 17 will not work to preload a tapered part.

5. **IZSHEAR for solid elements.** By default, IZSHEAR = 0, that is, *only* the stress normal to the section is prescribed while *all* other stress components are set to zero. Therefore, the stress tensor components with respect to a local system aligned with the section would have $\sigma_{xx} = \sigma$ (the prescribed value) and $\sigma_{yy} = \sigma_{zz} = \sigma_{xy} = \sigma_{xz} = \sigma_{yz} = 0$, which holds for each integration point of each element in the section. This initialization in turn means that these elements, thus the entire section, are infinitely weak in bending and shear which might have a potential side effect of unexpected and nonsensical deformations for certain loading and surrounding geometry conditions.

   Turning on IZSHEAR = 1 will for *explicit analysis* essentially relax the zero constraint on the shear stresses, resulting in a stress tensor where σ_{xy}, σ_{xz} and σ_{yz} are updated according to the laws of the material model, thus adding some shear stiffness to the section to avoid spurious deformations.

   For *implicit analysis* (and currently only for low order element formulations -1, -2, 1, 2, 10, 15 and 115 with sections defined using *DATABASE_CROSS_SECTION_PLANE or *DATABASE_CROSS_SECTION_SET), the following approach is taken for IZSHEAR = 1: Each collection of elements in a section (for example, of a bolt) is identified and seen as an entity for which only the *mean normal stress* of the section is prescribed, in contrast to imposing constraints on each integration point. For example, if the goal is to prestress 10 bolts modeled with solid elements, LS-DYNA will, based on element connectivity, identify the 10 unique sections (10 clusters of elements) corresponding to each bolt, and subsequently prescribe the mean stress in each of these sections independently. The advantage of this approach is that the section can globally resist bending and shear, and thus preserve the structural integrity of the bolt more adequately.

   The IZSHEAR = 1 approach for implicit is also available in explicit analysis through IZSHEAR = 2. Note that if this approach is used with *DATABASE_CROSS_SECTION_SET, the vector option must be used to define the normal direction. Using *DATABASE_CROSS_SECTION_SET leads to a better distribution of the stress than *DATABASE_CROSS_SECTION_PLANE by, for instance, imposing the stress for a sufficient length of the shank. Distributing

the stress with this method prevents elements from collapsing, thereby preserving structural integrity.

IZSHEAR = 2 can also be used when the initial stress (meaning stress at time zero) in the section is nonzero. For instance, bolts that are to be preloaded may have already gone through a sequence of load steps from prior simulations, resulting in a nonzero initial stress state. The load curve for the stress should still start at the origin and increase to the desired stress value. LS-DYNA will preserve stress continuity at the start of the simulation by internally scaling the curve appropriately

6. **Supported meshes.** We recommend using regular meshes for stress initialization. Stress is prescribed only on a cross-sectional plane of elements, and the remaining element stresses are calculated from a quasi-static analysis. For regular meshes, the remaining elements will balance the cross-section stress, leading to a smooth (or even constant) stress profile. Irregular meshes, however, may give local stress concentrations that do not achieve the desired overall normal stress, especially for IZSHEAR = 1 and IZSHEAR = 2.

*INITIAL_STRESS_SHELL_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Initialize stresses, history variables, and the effective plastic strain for shell elements. Materials that do not use an incremental formulation for the stress update may not be initializable with this card.

Card Sets per Element. Define as many shell elements or shell element sets in this section as desired. The input is assumed to terminate when a new keyword ("*") card is detected.

Element Card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID	NPLANE	NTHICK	NHISV	NTENSR	LARGE	NTHINT	NTHHSV
Type	I	I	I	I	I	I	I	I
Default	none	none	none	0	0	0	0	0

Ordering of Integration Points.

For each through thickness point define NPLANE points. NPLANE should be either 1 or 4 corresponding to either 1 or 4 Gauss integration points. If four integration points are specified, they should be ordered such that their in-plane parametric coordinates are at:

$$\left(-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\quad \left(-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),$$

respectively. It is not necessary for the location of the through thickness integration points to match those used in the elements which are initialized. The data will be interpolated by LS DYNA.

Solid Mechanics Data Card for LARGE = 0.

The following set of cards, "Stress Card" through "Tensor Cards," should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card. Additional Card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	T	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPS
Type	F	F	F	F	F	F	F	F

History Variable Cards. Additional Cards for LARGE = 0. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 3	1	2	3	4	5	6	7	8
Variable	HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8
Type	F	F	F	F	F	F	F	F

Tensor Cards. Additional card for LARGE = 0. Include as many cards as necessary to collect NTENSR (see Card 1) entries. Tensor cards contain only 6 entries per card.

Card 4	1	2	3	4	5	6	7	8
Variable	TENXX	TENYY	TENZZ	TENXY	TENYZ	TENZX		
Type	F	F	F	F	F	F		

Solid Mechanics Data Card for LARGE = 1.

The following set of cards, "Stress Card 1" through "Tensor Cards," should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card 1. Additional Card for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	T		SIGXX		SIGYY		SIGZZ		SIGXY	
Type	F		F		F		F		F	

Stress Card 2. Additional Card for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGYZ		SIGZX		EPS					
Type	F		F		F					

History Variable Cards. Additional Cards for LARGE = 1. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

Tensor Cards. Include as many pairs of Cards 5 and 6 as necessary to collect NTENSR entries. Note that Cards 5 and 6 must appear as pairs, and that Card 6 may include at most one value, as indicated below.

Card 5	1	2	3	4	5	6	7	8	9	10
Variable	TENXX		TENYY		TENZZ		TENXY		TENYZ	
Type	F		F		F		F		F	

Card 6	1	2	3	4	5	6	7	8	9	10
Variable	TENZX									
Type	F									

Thermal Data Cards for LARGE = 1.

For each element, thermal data cards come after the *entire* set of mechanical data cards. For each of the NTHINT thermal integration points, include the following set of cards.

Thermal Time History Cards. Additional cards for LARGE = 1. Include as many cards as needed to collect all the of NTHHSV time history variables per thermal integration point.

Card 7	1	2	3	4	5
Variable	THHSV1	THHSV2	THHSV3	THHSV4	THHSV5
Type	F	F	F	F	F

VARIABLE	DESCRIPTION
EID/SID	Element ID or shell set ID, see *SET_SHELL_…
NPLANE	Number of in plane integration points being output.
NTHICK	Number of integration points through the thickness.
NHISV	Number of additional history variables.
NTENSR	Number of components of tensor data taken from the element history variables stored.
LARGE	Format size. See cards above. EQ.0: off EQ.1: on
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
SIG*ij*	Define the *ij* stress component. The stresses are defined in the GLOBAL cartesian system.
EPS	Effective plastic strain.
HISV*n*	Define the n^{th} history variable.
TEN*ij*	Define the ij^{th} component of the tensor taken from the history variables. The tensor is defined in the GLOBAL Cartesian system. Define enough lines to provide a total of NTENSOR components, stored six components per line. This applies to material 190 only.
NTHINT	Number of thermal integration points.
NTHHSV	Number of thermal history variables per thermal integration point.

VARIABLE	DESCRIPTION
THHSV*n*	n^{th} history variable at the thermal integration point.

*INITIAL_STRESS_SHELL_NURBS_PATCH

Purpose: Initialize stresses, history variables, and the effective plastic strain for isogeometric shell elements. Materials that do not use an incremental formulation for the stress update may not be initializable with this card.

Card Sets per Element. Define as many NURBS shell elements in this section as desired. The input is assumed to terminate when a new keyword ("*") card is detected.

Element Card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NPLANE	NTHICK	NHISV	LARGE			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

Solid Mechanics Data Card for LARGE = 0.

The following set of cards: "Stress Cards" and "History Variable Cards" should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card1. Additional Card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					

Stress Card2. Additional Card for LARGE = 0.

Card 3	1	2	3	4	5	6	7	8
Variable	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPSP	
Type	F	F	F	F	F	F	F	

History Variable Cards. Additional Cards for LARGE = 0. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8
Variable	HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8
Type	F	F	F	F	F	F	F	F

Solid Mechanics Data Card for LARGE = 1.

The following set of cards: "Stress Cards" and "History Variable Cards" should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card 1. Additional Card for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		SIGXX		SIGYY	
Type	F		F		F		F		F	

Stress Card 2. Additional Card for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGZZ		SIGXY		SIGYZ		SIGZX		EPS	
Type	F		F		F		F		F	

History Variable Cards. Additional Cards for LARGE = 1. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID	NURBS Element ID
NPLANE	Number of in plane integration points being output.
NTHICK	Number of integration points through the thickness.
NHISV	Number of additional history variables.
LARGE	Format size. See cards above. EQ.0: off EQ.1: on
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to NURBS-patch definition)
S	Parametric *s*-coordinate of location of in-plane integration point (with respect to NURBS-patch definition)
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
SIG*ij*	Define the *ij* stress component. The stresses are defined in the GLOBAL cartesian system.
EPS	Effective plastic strain.
HISV*n*	Define the n^{th} history variable.

*INITIAL_STRESS_SOLID_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Initialize stresses and plastic strains for solid elements.

Material models 2, 5, 7, 21, 23, 27, 31, 38, 57, 73, 77, 83,132, 179, 181, 183, and 189 support a reference configuration (see *INITIAL_FOAM_REFERENCE_GEOMETRY and the variable REF in the material description), and if that feature is invoked, it will take precedence over *INITIAL_STRESS_SOLID. Furthermore, if FMATRX = 2 in *CONTROL_SOLID, *INITIAL_STRESS_SOLID is ignored for the aforementioned material models.

For *MAT_014 and any material that requires an equation-of-state (*EOS), the specified initial stresses are adjusted to be in accordance with the initial pressure calculated from the equation of state.

Card Sets per Element or Element Set. For this keyword, each data card set consists of an element or element set card and all of its corresponding data cards, both thermal and mechanical. For LARGE = 1, this can involve several (even tens of) cards per set. Include cards for as many solid elements or solid element sets as desired. The input is assumed to terminate when a new keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID	NINT	NHISV	LARGE	IVEFLG	IALEGP	NTHINT	NTHHSV
Type	I	I	I	I	I	I	I	I
Default	none	none	0	0	0	0	0	0

Ordering of Integration Points.

NINT may be 1, 8, or 14 for hexahedral solid elements, depending on the element formulation. If eight Gauss integration points are specified, they should be ordered such that their parametric coordinates are located at:

$$\left(-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\left(\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\left(\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\left(-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),$$

$$\left(-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\left(\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\left(\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\left(-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),$$

respectively. If eight points are defined for 1 point LS-DYNA solid elements, the average value will be taken.

NINT may be 1, 4, or 5 for tetrahedral solid elements, depending on the element formulation and NIPTETS in *CONTROL_SOLID. NINT may be 1 or 2 for pentahedral solid elements, depending on the element formulation.

Solid Mechanics Data Card for LARGE = 0.

Stress Card. Additional Card for LARGE = 0. This card should be included NINT times (one for each integration point).

Card 2	1	2	3	4	5	6	7	8
Variable	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPS	
Type	F	F	F	F	F	F	F	

Mechanical Data Cards for LARGE = 1.

The following set of cards "Stress Card 1" through "Additional History Cards." Should be included NINT times (one set for each integration point).

Stress Card 1. Additional cards for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	SIGXX		SIGYY		SIGZZ		SIGXY		SIGYZ	
Type	F		F		F		F		F	

Stress Card 2. Additional cards for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGZX		EPS		HISV1		HISV2		HISV3	
Type	F		F		F		F		F	

Additional History Cards. Additional cards for LARGE = 1. If NHISV > 3 define as many additional cards as necessary. NOTE: the value of IVEFLG (see Card 1) can affect the number of history variables on these cards.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	HISV4		HISV5		HISV6		HISV7		HISV8	
Type	F		F		F		F		F	

Thermal Data Cards for LARGE = 1.

For each element, thermal data cards come after the *entire* set of mechanical data cards. For each of the NTHINT thermal integration points, include the following set of cards.

Thermal Time History Cards. Additional cards for LARGE = 1. Include as many cards as needed to capture all the of NTHHSV time history variables per thermal integration point.

Card 5	1	2	3	4	5	6	7	8	9	10
Variable	THHSV1		THHSV2		THHSV3		THHSV4		THHSV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID/SID	Element ID or solid set ID; see *SET_SOLID_...
NINT	Number of integration points (should correspond to the solid element formulation).
NHISV	Number of additional history variables, which is typically equal to the number of history variables stored at the integration point + IVEFLG.
LARGE	Format size. If zero, NHISV must also be set to zero (this is the format used by LS-DYNA versions 970 and earlier), and, if set to 1, a larger format is used and NHISV is used.

VARIABLE	DESCRIPTION
IVEFLG	Initial Volume/energy flag (**only used in large format**): EQ.0: Last history variable is used as normal. EQ.1: Last history variable is used as the initial volume of the element. One additional history variable is required if IVEFLG = 1. EQ.2: Last two history variables are used to define the initial volume and the internal energy per unit initial volume. Two additional history variables must be allocated; see NHISV above, if IVEFLG = 2. If the initial volume is set to zero, the actual element volume is used.
IALEGP	The ALE multi-material group (AMMG) ID; only if the element is of ALE multi-material formulation (ELFORM = 11). In this case, each AMMG has its own sets of stress and history variables, so we must specify to which AMMG the stress data are assigned. For mixed elements, multiple cards are needed to complete the stress initialization in this element as each AMMG needs to have its own set of stress data. EQ.0: Assuming the element is fully filled by the AMMG that the element part belongs to. Please refer to *ALE_MULTI-MATERIAL_GROUP card. EQ.n: Assigning the stress to nth AMMG in that element.
SIG*ij*	Define the ij^{th} stress component. Stresses are defined in the GLOBAL Cartesian system.
EPS	Effective plastic strain
HISV*i*	Define NHISV history variables.
NTHINT	Number of thermal integration points
NTHHSV	Number of thermal history variables per thermal integration point
THHSV*n*	n^{th} thermal time history variable

Remarks:

1. **Cohesive Elements.** The elastic material model for cohesive elements is a total Lagrangian formulation, and the initial stress will therefore be ignored for it.

2. **Tensorial History Variables.** If the history variable field contains tensorial quantities, such as back stresses or similar tensors (such as for *MAT_003), then *INCLUDE_TRANSFORM with *DEFINE_TRANSFORMATION including a rotation (options MIRROR, ROTATE, POS6P, POS6N) should not be used. Only the actual stress tensor is transformed correctly at the moment.

*INITIAL_STRESS_SOLID_NURBS_PATCH

Purpose: Initialize stresses, effective plastic strain, and history variables for isogeometric solid elements. This command is not applicable to hyperelastic materials or any material model based on a Total Lagrangian formulation.

Card Sets. Define as many NURBSsolid elements in this section as desired, *one set of cards per element*. The input is assumed to terminate when a new keyword ("*") card is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	NINT	NHISV	LARGE				
Type	I	I	I	I				
Default	none	none	0	0				

Solid Mechanics Data Card for LARGE = 0.

The following set of cards, "Stress Cards" and "History Variable Cards," should be included NINT times (one set for each integration point).

Stress Card1. Additional card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	R	S	T					
Type	F	F	F					

Stress Card2. Additional card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPS	
Type	F	F	F	F	F	F	F	

History Variable Cards. Additional card for LARGE = 0. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 3	1	2	3	4	5	6	7	8
Variable	HISV1	HISV2	HISV3	HISV4	HISV5	HISV6	HISV7	HISV8
Type	F	F	F	F	F	F	F	F

Solid Mechanics Data Card for LARGE = 1.

The following set of cards, "Stress Cards" and "History Variable Cards," should be included NINT times (one set for each integration point).

Stress Card 1. Additional card for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	R		S		T		SIGXX		SIGYY	
Type	F		F		F		F		F	

Stress Card 2. Additional card for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGZZ		SIGXY		SIGYZ		SIGZX		EPS	
Type	F		F		F		F		F	

History Variable Cards. Additional card for LARGE = 1. Include as many cards as necessary to collect NHISV (see Card 1) history variables.

Card 4	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID	NURBS Element ID
NINT	Number of in integration points being output.
NHISV	Number of additional history variables.
LARGE	Format size: EQ.0: off EQ.1: on
R	Parametric *r*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
S	Parametric *s*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
T	Parametric *t*-coordinate of location of in-plane integration point (with respect to NURBS patch definition)
SIG*ij*	Define the *ij* stress component. The stresses are defined in the GLOBAL cartesian system.
EPS	Effective plastic strain.
HISV*n*	Define the n^{th} history variable.

*INITIAL_STRESS_SPH

Purpose: Initialize stresses and plastic strains for SPH elements. This command is not applicable to hyperelastic materials or any material model based on a Total Lagrangian formulation. For *MAT_005, *MAT_014, and any material that requires an equation-of-state (*EOS), the initialized stresses are deviatoric stresses, not total stresses.

Element Cards. Define as many SPH elements in this section as desired. The input is assumed to terminate when a new keyword is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPS
Type	I	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
EID	SPH particle ID
SIG*ij*	Define the ij^{th} stress component. Stresses are defined in the GLOBAL Cartesian system.
EPS	Effective plastic strain.

***INITIAL_STRESS_TSHELL**_*{OPTION}*

Available options include:

<BLANK>

SET

Purpose: Initialize stresses and plastic strains for thick shell elements.

Card Sets per Element. Define as many thick shell elements in this section as desired. The input is assumed to terminate when a new keyword is detected.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/SID	NPLANE	NTHICK	NHISV	LARGE			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

Ordering of Integration Points.

For each through thickness point define NPLANE points. NPLANE should be either 1 or 4 corresponding to either 1 or 4 Gauss integration points. If four integration points are specified, they should be ordered such that their in plane parametric coordinates are at:

$$\left(-\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},-\frac{\sqrt{3}}{3}\right),\quad \left(\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right),\quad \left(-\frac{\sqrt{3}}{3},\frac{\sqrt{3}}{3}\right)$$

respectively. It is not necessary for the location of the through thickness integration points to match those used in the elements which are initialized. The data will be interpolated by LS DYNA.

Data Card for LARGE = 0.

The following set of cards "Stress Card" and "History Cards." Should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card. Additional card for LARGE = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	T	SIGXX	SIGYY	SIGZZ	SIGXY	SIGYZ	SIGZX	EPS
Type	F	F	F	F	F	F	F	F

History Cards. Additional Card for LARGE = 0. Include as many History Cards as needed to define all NHIST history variables.

Optional	1	2	3	4	5	6	7	8
Variable	HISV1	HSIV2	HSIV3	HSIV4	HSIV5	HSIV6	HSIV7	HSIV8
Type	F	F	F	F	F	F	F	F

Data Card for LARGE = 1.

The following set of cards "Stress Cards" and "History Cards." Should be included NPLANE × NTHICK times (one set for each integration point).

Stress Card 1. Additional card for LARGE = 1.

Card 2	1	2	3	4	5	6	7	8	9	10
Variable	T		SIGXX		SIGYY		SIGZZ		SIGXY	
Type	F		F		F		F		F	

Stress Card 2. Additional card for LARGE = 1.

Card 3	1	2	3	4	5	6	7	8	9	10
Variable	SIGYZ		SIGZX		EPS					
Type	F		F		F					

History Cards. Additional Card for LARGE = 1. Include as many History Cards as needed to define all NHIST history variables.

Optional	1	2	3	4	5	6	7	8	9	10
Variable	HISV1		HISV2		HISV3		HISV4		HISV5	
Type	F		F		F		F		F	

VARIABLE	DESCRIPTION
EID/SID	Thick shell element ID or for the SET keyword option thick shell set ID (see *SET_TSHELL_…)
NPLANE	Number of in plane integration points.
NTHICK	Number of integration points through the thickness.
T	Parametric coordinate of through thickness integration point between -1 and 1 inclusive.
NHISV	Number of additional history variables.
LARGE	Format size. See keywords above. EQ.0: off EQ.1: on
SIG*ij*	Define the *ij* stress component. The stresses are defined in the GLOBAL Cartesian system.
EPS	Effective plastic strain

INITIAL_TEMPERATURE_*OPTION

Available options include:

NODE

SET

Purpose: Define initial nodal point temperatures using nodal set IDs or node numbers. These initial temperatures are used in a thermal only analysis or a coupled thermal/structural analysis. See also *CONTROL_THERMAL_SOLVER, *CONTROL_THERMAL_TIMESTEP, and CONTROL_THERMAL_NONLINEAR.

For thermal loading in a structural only analysis, see *LOAD_THERMAL_*OPTION*.

Node/Node set Cards. Include one card for each node or node set. This input ends at the next keyword ("*") keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID/NID	TEMP	LOC					
Type	I	I	I					
Default	↓	0.	0					
Remarks	1							

VARIABLE	DESCRIPTION
NSID/NID	Nodal set ID or nodal point ID, see also *SET_NODES: EQ.0: all nodes are included (set option only).
TEMP	Temperature at node or node set.
LOC	For a thick thermal shell, the temperature will be applied to the surface identified by LOC. See parameter, THSHEL, on the *CONTROL_SHELL keyword. EQ.-1: lower surface of thermal shell element EQ.0: middle surface of thermal shell element EQ.1: upper surface of thermal shell element

Remarks:

1. **SPH Particles.** This keyword can be used to define initial nodal point temperatures for SPH particles by using nodal set IDs or node numbers from SPH particles.

*INITIAL_TIED_CONTACT_DATA

Purpose: Initialize penalty-method tied contact. LS-DYNA writes this keyword to the dynain file if penalty-method tied contacts are present, to enable subsequent analyses beginning from a state reached in a previous analysis. It is not expected that users will create or modify this keyword.

Card 1	1	2	3	4	5	6	7	8
Variable	ILABEL	SNID	MN1	MN2	MN3	MN4		
Type	I	I	I	I	I	I		
Default	none	none	none	none	none	none		

Card 2	1	2	3	4	5	6	7	8
Variable	SS	TT	THK					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
ILABEL	ID of the instantiation of *CONTACT
SNID	ID of the tied node
MN1	ID of node 1 of the segment to which SNID is tied
MN2	ID of node 2 of the segment to which SNID is tied
MN3	ID of node 3 of the segment to which SNID is tied
MN4	ID of node 4 of the segment to which SNID is tied
SS	Initial isoparametric coordinate of the tied node within the segment in the range $-1 \le SS \le 1$

VARIABLE	DESCRIPTION
TT	Initial isoparametric coordinate of the tied node within the segment in the range $-1 \le TT \le 1$
THK	Initial offset of the tied node from the plane of the segment in length units

*INITIAL_VAPOR_PART

Purpose: Initialization of a part as a vapor material for *PART using *MAT_NULL and *EOS_PHASE_CHANGE. All elements in the part are initialized as a vapor material.

Card 1	1	2	3	4	5	6	7	8
Variable	PID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PID	Part ID of the part to be initialized as a vapor material

*INITIAL_VEHICLE_KINEMATICS

Purpose: Define initial kinematical information for a vehicle. In its initial orientation, the vehicle's yaw, pitch, and roll axes must be aligned with the global axes. Successive simple rotations are taken about these body fixed axes.

Card 1	1	2	3	4	5	6	7	8
Variable	GRAV	PSID	XO	YO	ZO	XF	YF	ZF
Type	I	I	F	F	F	F	F	F
Default	none	none	0.	0.	0.	0.	0.	0.

Card 2	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	AAXIS	BAXIS	CAXIS	IVADD	
Type	F	F	F	I	I	I	I	
Default	0.	0.	0.	0	0	0	2	

Card 3	1	2	3	4	5	6	7	8
Variable	AANG	BANG	CANG	WA	WB	WC		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
GRAV	Gravity direction code (see Remark 1): EQ.0: Gravity not considered EQ.1: Global $+x$ direction EQ.-1: Global $-x$ direction

VARIABLE	DESCRIPTION
	EQ.2: Global $+y$ direction EQ.-2: Global $-y$ direction EQ.3: Global $+z$ direction EQ.-3: Global $-z$ direction Note: This must be the same for all vehicles present in the model.
PSID	Part set ID
XO	x-coordinate of initial position of mass center
YO	y-coordinate of initial position of mass center
ZO	z-coordinate of initial position of mass center
XF	x-coordinate of final position of mass center
YF	y-coordinate of final position of mass center
ZF	z-coordinate of final position of mass center
VX	Global x-component of mass center velocity
VY	global y-component of mass center velocity
VZ	global z-component of mass center velocity
AAXIS	First rotation axis code. EQ.1: Initially aligned with global x-axis EQ.2: Initially aligned with global y-axis EQ.3: Initially aligned with global z-axis
BAXIS	Second rotation axis code
CAXIS	Third rotation axis code
IVADD	Flag for velocity overwrite / add: EQ.1: Add to pre-defined velocities. EQ.2: Overwrite pre-defined velocities (default).
AANG	Rotation angle about the first rotation axis (degrees)
BANG	Rotation angle about the second rotation axis (degrees)

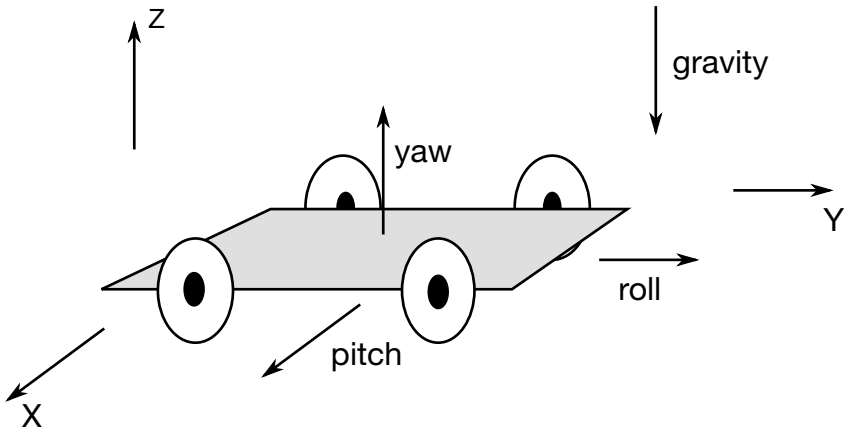


Figure 28-9. The vehicle pictured is to be oriented with a successive rotation sequence about the yaw, pitch, and roll axes, respectively. Accordingly, AAXIS = 3, BAXIS = 1, and CAXIS = 2. The direction of gravity is given by GRAV = -3.

VARIABLE	DESCRIPTION
CANG	Rotation angle about the third rotation axis (degrees)
WA	Angular velocity component for the *x* body-fixed axis (radian/second)
WB	Angular velocity component for the *y* body-fixed axis (radian/second)
WC	Angular velocity component for the *z* body-fixed axis (radian/second)

Remarks:

1. **GRAV.** The GRAV field is used to preserve the spatial location in the direction specified. To illustrate this feature, suppose a part set that represents a vehicle is initially oriented such that the wheels are on the ground (the ground plane) at *Z* = 100 and gravity is in the -*Z*-direction. To perform a roof drop simulation, the part set of the vehicle must be re-oriented, but the ground plane must be maintained. To do this, GRAV must be set to -3 which preserves *Z* = 100 as the lowest *Z*-coordinate, thereby maintaining the ground plane.

*INITIAL_VELOCITY

Purpose: Define initial nodal point velocities using nodal set IDs. This may also be used for sets in which some nodes have other velocities. See NSIDEX below.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	NSIDEX	BOXID	IRIGID	ICID			
Type	I	I	I	I	I			
Default	↓	{∅}	0	0	0			
Remark	1							

Card 2	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	VXR	VYR	VZR		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

Exempted Node Card. Additional card for NSIDEX > 0.

Card 3	1	2	3	4	5	6	7	8
Variable	VXE	VYE	VZE	VXRE	VYRE	VZRE		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
NSID	Nodal set ID, see *SET_NODES, containing nodes for initial velocity. If NSID = 0, the initial velocity is applied to all nodes.

VARIABLE	DESCRIPTION
NSIDEX	Nodal set ID, see *SET_NODE, containing nodes that are exempted from the imposed velocities and may have other initial velocities.
BOXID	All nodes in the box that belong to NSID are initialized. Nodes outside the box are not initialized. Exempted nodes are initialized to velocities defined by VXE, VYE, and VZE below, regardless of their location relative to the box.
IRIGID	Option to overwrite rigid body velocities defined on *PART_INERTIA or *CONSTRAINED_NODAL_RIGID_BODY_INERTIA cards. GE.1: Part set ID containing IDs of parts to overwrite. The center of gravity of a part must lie within box BOXID. If BOXID is not defined, then all parts specified in the set are overwritten. EQ.-1: Overwrite velocities for all rigid bodies defined with *PART_INERTIA or *CONSTRAINED_NODAL_RIGID_BODY_INERTIA that have a center of gravity within box BOXID. If BOXID is not defined, then all are overwritten. EQ.-2: Overwrite velocities for all rigid bodies defined with *PART_INERTIA or *CONSTRAINED_NODAL_RIGID_BODY_INERTIA.
ICID	Local coordinate system ID. The initial velocity is specified in the local coordinate system if ICID is greater than zero. Furthermore, if ICID is greater than zero, *INCLUDE_TRANSFORM does not rotate the initial velocity values specified by VX, VY, …, VZRE.
VX	Initial translational velocity in x-direction
VY	Initial translational velocity in y-direction
VZ	Initial translational velocity in z-direction
VXR	Initial rotational velocity about the x-axis
VYR	Initial rotational velocity about the y-axis
VZR	Initial rotational velocity about the z-axis
VXE	Initial velocity in x-direction of exempted nodes

VARIABLE	DESCRIPTION
VYE	Initial velocity in y-direction of exempted nodes
VZE	Initial velocity in z-direction of exempted nodes
VXRE	Initial rotational velocity in x-direction of exempted nodes
VYRE	Initial rotational velocity in y-direction of exempted nodes
VZRE	Initial rotational velocity in z-direction of exempted nodes

Remarks:

1. **Deck restrictions.** This generation input must not be used with *INITIAL_VELOCITY_GENERATION keyword.

2. **Multiple nodal velocity initializations.** If a node velocity is initialized on more than one input card set, then the last set input will determine its velocity. However, if the nodal velocity is also specified on a *INITIAL_VELOCITY_NODE card, then the velocity specification on this card will be used.

3. **Overwriting rigid body velocities.** Unless the option IRIGID is specified, initial velocities for rigid bodies given by *PART_INERTIA will overwrite generated initial velocities. The IRIGID option will cause the rigid body velocities specified on the *PART_INERTIA input to be overwritten. To directly specify the motion of a rigid body without using the keyword, *PART_INERTIA, which also requires the definition of the mass properties, use the keyword option, *INITIAL_VELOCITY_RIGID_BODY.

4. **Rigid body motion consistency.** Nodes that belong to rigid bodies must have motion consistent with the translational and rotational velocity of the center of gravity (c.g.) of the rigid body. During initialization, the translational and rotational rigid body momentum is computed based on the prescribed nodal velocity field. From this rigid body momentum, the translational and rotational velocities of the nodal points are calculated and reset to the new values. These new values may or may not be the same as the values prescribed for the nodes that make up the rigid body. Sometimes this occurs in single precision due to numerical round-off. If a problem like this happens, specify the velocity using the keyword *INITIAL_VELOCITY_RIGID_BODY.

5. **Mid-side nodes.** Mid-side nodes generated by *ELEMENT_SOLID_TET4TO10 will not be initialized because the node numbers are not known a priori to the user. Instead use *INITIAL_VELOCITY_GENERATION if you intend to initialize the velocities of the mid-side nodes.

6. **Rotational motion for deformable bodies.** Rotational motion can only be applied to nodes with rotational degrees of freedom. Element type and formulation determine the degrees of freedom for a node. For example, the nodes of solid formulation 1 have three translational degrees of freedom and no rotational degrees of freedom. Solid elements generally have no rotational degrees of freedom, while beam and shell elements typically have three rotational degrees of freedom.

7. **Rigidly rotating a deformable body.** To rigidly rotate a deformable body, the translational velocities, not rotational velocities, need to be set to specify the rigid body motion. For this application, we recommend using *INITIAL_VELOCITY_GENERATION instead.

8. **Intent of this keyword for deformable bodies.** For deformable bodies, this keyword is intended for initializing velocities that cannot be constructed from a simple rigid body motion. For instance, the velocities can come from a previous simulation.

*INITIAL_VELOCITY_GENERATION

Purpose: Define initial velocities for rotating and/or translating bodies. These velocities are invoked at time = 0 unless a start time or sensor switch is specified with *INITIAL_VELOCITY_GENERATION_START_TIME (see Remark 7). If a dynamic relaxation phase is invoked, these velocities do not apply during that phase.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	STYP	OMEGA	VX	VY	VZ	IVATN	ICID
Type	I	I	F	F	F	F	I	I
Default	{all}	↓	0.	0.	0.	0.	0	global

Card 2	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	NX	NY	NZ	PHASE	IRIGID
Type	F	F	F	F	F	F	I	I
Default	0.	0.	0.	0.	0.	0.	0	0

VARIABLE	DESCRIPTION
ID	Part ID, part set ID, or node set ID. If zero, STYP is ignored, and all velocities are set. WARNING for if IVATN = 0: If a part ID of a rigid body is specified, only the nodes that belong to elements of the rigid body are initialized. Nodes added with *CONSTRAINED_EXTRA_NODES are not initialized. Set IVATN = 1 to initialize the velocities of constrained nodes and parts.
STYP	Set type (see Remark 5): EQ.1: Part set ID (see *SET_PART) EQ.2: Part ID (see *PART) EQ.3: Node set ID (see *SET_NODE)
OMEGA	Angular velocity about the rotational axis. See Remark 6.

VARIABLE	DESCRIPTION
VX	Initial translational velocity in x-direction (see ICID below)
VY	Initial translational velocity in y-direction (see ICID below)
VZ	Initial translational velocity in z-direction (see ICID below)
IVATN	Flag for setting the initial velocities of constrained nodes and parts: EQ.0: Constrained parts are ignored. EQ.1: Constrained parts and nodes constrained to parts will be assigned initial velocities like the part to which they are constrained.
ICID	Local coordinate system ID. If ICID = 0, the specified translational velocities (VX,VY,VZ) and the direction cosines of the rotation axis (NX,NY,NZ) are in the global system; otherwise, they are in the local system specified by ICID. Therefore, if ICID is defined, *INCLUDE_TRANSFORM does not transform (VX,VY,VZ) and (NX,NY,NZ).
XC	Global x-coordinate on rotational axis
YC	Global y-coordinate on rotational axis
ZC	Global z-coordinate on rotational axis
NX	x-direction cosine. If set to -999, NY and NZ are interpreted as the 1st and 2nd nodes defining the rotational axis, in which case the coordinates of node NY are used as XC, YC, ZC. If ICID is defined, the direction cosine, (NX,NY,NZ), is projected along coordinate system ICID to yield the direction cosines of the rotation axis only if NX ≠ -999.
NY	y-direction cosine or the 1st node of the rotational axis when NX = -999.
NZ	z-direction cosine or the 2nd node of the rotational axis when NX = -999.
PHASE	Flag determining basis for initialization of velocity (see Remarks 6 and 7): EQ.0: Initial velocities are applied at $t = 0$ of the regular transient phase of the analysis and are based on the undeformed geometry. Rigid bodies whose velocities are initialized

VARIABLE	DESCRIPTION
	using this keyword should always use PHASE = 0. EQ.1: Initial velocities of deformable bodies are based on geometry that includes deformation incurred prior to the application of the initial velocities. That deformation could be due to a dynamic relaxation phase or due to a nonzero start time specified with *INITIAL_VELOCITY_GENERATION_START_TIME.
IRIGID	Controls hierarchy of initial velocities set with *INITIAL_VELOCITY_GENERATION versus those set with *PART_INERTIA / *CONSTRAINED_NODAL_RIGID_BODY_INERTIA when the commands conflict. EQ.0: *PART_INERTIA / *CONSTRAINED_NODAL_RIGID_BODY_INERTIA controls initial velocities. EQ.1: *INITIAL_VELOCITY_GENERATION controls initial velocities. This option does not apply if STYP = 3.

Remarks:

1. **Exclusions.** This generation input must not be used with *INITIAL_VELOCITY or *INITIAL_VELOCITY_NODE options.

2. **Order dependence.** The velocities are initialized in the order the *INITIAL_VELOCITY_GENERATION input is defined. Later input using the *INITIAL_VELOCITY_GENERATION keyword may overwrite the velocities previously set.

3. **Consistency for rigid body nodes.** Nodes that belong to rigid bodies must have motion consistent with the translational and rotational velocity of the rigid body. During initialization, the translational and rotational rigid body momentums are computed based on the prescribed nodal velocities. From this rigid body motion, the velocities of the nodal points are computed and reset to the new values. These new values may or may not be the same as the values prescribed for the node.

4. **SPH.** Prior to R15, SPH elements can only be initialized with node sets (STYP = 3). Starting with R15, SPH elements can additionally be initialized with parts and part sets.

5. **Constrained nodal rigid bodies.** Part IDs of *CONSTRAINED_NODAL_RIGID_BODYs that do not include the INERTIA option are not recognized by

the code in the case of STYP = 1 or 2. Use STYP = 3 (a node set ID) when initializing the velocity of such nodal rigid bodies.

6. **Rotating bodies.** When velocities for a rotating body (nonzero OMEGA) are initialized at $t = 0$ following a dynamic relaxation phase in which body forces (*LOAD_BODY) serve to preload the spinning body, velocities of deformable parts/nodes of the spinning body should be included in an *INITIAL_VELOCITY_GENERATION command with PHASE = 1. Furthermore, any rigid parts/nodes of the spinning body should be initialized using a second *INITIAL_VELOCITY_GENERATION command with PHASE = 0.

7. **Nonzero start time for initial velocity.** *INITIAL_VELOCITY_GENERATION_START_TIME can cause a delayed start time for the velocity. A delayed start time requires that the input deck contain an *INITIAL_VELOCITY_GENERATION command with PHASE = 1. Similar to the situation described in Remark 6, if there are any rigid body nodes whose velocity is being initialized, those nodes must be included in a second *INITIAL_VELOCITY_GENERATION command with PHASE = 0. Note that all velocity set with *INITIAL_VELOCITY_GENERATION begins at the start time if the input deck contains at least one *INITIAL_VELOCITY_GENERATION keyword with PHASE = 1. If the deck *only* includes one *INITIAL_VELOCITY_GENERATION with PHASE = 0, the start time is ignored, and the velocity is initialized at $t = 0$.

   The following example illustrates the initialization of translational velocity in the y-direction to 8.8 for node set 55 at $t = 13$. Node set 55 includes both rigid body nodes and deformable body nodes. Note that the same all-inclusive node set may be used for both *INITIAL_VELOCITY_GENERATION commands.

```
$ *INITIAL_VELOCITY_GENERATION
$#      id      styp     omega        vx        vy        vz     ivatn      icid
        55         3                           8.8
$#      xc        yc        zc        nx        ny        nz     phase    irigid
                                                                     0
*INITIAL_VELOCITY_GENERATION
$#      id      styp     omega        vx        vy        vz     ivatn      icid
        55         3                           8.8
$#      xc        yc        zc        nx        ny        nz     phase    irigid
                                                                     1
*INITIAL_VELOCITY_GENERATION_START_TIME
$#   stime
       13.
```

*INITIAL_VELOCITY_GENERATION_START_TIME

Purpose: Define a time to initialize velocities after time zero or provide a *SENSOR_SWITCH switch ID for a switch that causes the initialization of the velocities upon meeting the switch condition. Time zero starts after dynamic relaxation if used for initialization.

Card 1	1	2	3	4	5	6	7	8
Variable	STIME							
Type	F							
Default	0.0							

VARIABLE	DESCRIPTION
STIME	Start time or switch ID: GT.0.0: Start time at which LS-DYNA initializes the velocities. LT.0.0: |STIME| is a *SENSOR_SWITCH switch ID. When the switch condition is met, LS-DYNA initializes the velocities.

Remarks:

1. **One start time.** Only one *INITIAL_VELOCITY_GENERATION_START_TIME can be specified. Multiple start times are not allowed.

2. **Using start time.** All *INITIAL_VELOCITY_GENERATION commands adhere to the start time or switch condition provided the requirement is met that at least one of those commands has PHASE set to 1.

3. **Velocities of other nodes.** When *INITIAL_VELOCITY_GENERATION_START_TIME is active, nodes that are not part of the initial velocity generation definitions will be re-initialized with velocities as they were at $t = 0$.

*INITIAL_VELOCITY_NODE

Purpose: Define initial nodal point velocities for a node.

Card	1	2	3	4	5	6	7	8
Variable	NID	VX	VY	VZ	VXR	VYR	VZR	ICID
Type	I	F	F	F	F	F	F	I
Default	none	0.	0.	0.	0.	0.	0.	0

VARIABLE	DESCRIPTION
NID	Node ID
VX	Initial translational velocity in x-direction
VY	Initial translational velocity in y-direction
VZ	Initial translational velocity in z-direction
VXR	Initial rotational velocity about the x-axis
VYR	Initial rotational velocity about the y-axis
VZR	Initial rotational velocity about the z-axis
ICID	Local coordinate system ID. The specified velocities are in the local system if ICID is greater than zero. Furthermore, if ICID is greater than zero, *INCLUDE_TRANSFORM does not rotate the initial velocity values specificied by VX, VY, …, VZR.

See Remarks on *INITIAL_VELOCITY card.

*INITIAL_VELOCITY_RIGID_BODY

Purpose: Define the initial translational and rotational velocities for a rigid body or a nodal rigid body. Depending on the value of CMO (see *CONSTRAINED_NODAL_-RIGID_BODY and *MAT_RIGID), the velocities are either at the center of gravity or at an arbitrary point in space. This input overrides all other velocity input for the rigid body and the nodes which define the rigid body.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	VX	VY	VZ	VXR	VYR	VZR	ICID
Type	I	F	F	F	F	F	F	I
Default	none	0.	0.	0.	0.	0.	0.	0

VARIABLE	DESCRIPTION
PID	Part ID of the rigid body or the nodal rigid body.
VX	Initial translational velocity in global x-direction.
VY	Initial translational velocity in global y-direction.
VZ	Initial translational velocity in global z-direction.
VXR	Initial rotational velocity about the global x-axis.
VYR	Initial rotational velocity about the global y-axis.
VZR	Initial rotational velocity about the global z-axis.
ICID	Local coordinate system ID. The specified velocities are in the local system if ICID is greater than zero. Furthermore, if ICID is greater than zero, *INCLUDE_TRANSFORM does not transform the initial velocity values specified by VX, VY, …, VZR.

See Remarks 3 and 4 of the *INITIAL_VELOCITY input description.

***INITIAL_VOID**_*OPTION*

Available options include:

PART

SET

Purpose: Define initial voided part set IDs or part numbers. This command can be used only when ELFORM = 12 in *SECTION_SOLID. Void materials cannot be created during the calculation. Fluid elements which are evacuated, such as by a projectile moving through the fluid, during the calculation are approximated as fluid elements with very low densities. The constitutive properties of fluid materials used as voids must be identical to those of the materials which will fill the voided elements during the calculation. Mixing of two fluids with different properties is not permitted with this option.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID/PID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PSID/PID	Part set ID or part ID, see also *SET_PART:

Remarks:

This void option and multiple materials per element, see *ALE_MULTI-MATERIAL_-GROUP are incompatible and cannot be used together in the same run.

*INITIAL_VOLUME_FRACTION_{OPTION}

Available option:

<BLANK>

LSDA

NALEGP

Purpose: Define initial volume fractions of different materials in multi-material ALE / S-ALE elements. Without the NALEGP or LSDA option, the keyword allows up to 7 ALE multi-material groups. The NALEGP option adds in an additional card immediately after the keyword to let users input the number of ALE multi-material groups to be read in for each element. The LSDA option is for S-ALE only. It allows you to include the lsda file produced from a previous S-ALE simulation that contained either *INITIAL_VOLUME_FRACTION_GEOMETRY or *ALE_STRUCTURED_MESH_VOLUME_FILLING. The default name of the lsda file is salevfrc.lsda. This file is used to pre-fill and trim the mesh which can save simulation time.

Card Summary:

Card 1a. This card is included if there is no keyword option. Repeat this card as many times as desired (one card for each element volume fraction to be defined). This input ends at the next keyword ("*") card.

EID	VF1	VF2	VF3	VF4	VF5	VF6	VF7

Card 1b. This card is included if the keyword option is NALEGP. This card set (Card 1b, 1b.1, and 1b.2) is repeated as many times as desired (one set for each element volume fraction to be defined).

NALEGP							

Card 1b.1. This card is included if the option is NALEGP.

EID	VF1	VF2	VF3	VF4	VF5	VF6	VF7

Card 1b.2. This card is included if the option is NALEGP and NALEGP > 7. Repeat this card in each set enough times to define NALEGP volume fractions.

VF*i*	VF(*i*+1)	VF(*i*+2)	VF(*i*+3)	VF(*i*+4)	VF(*i*+5)	VF(*i*+6)	VF(*i*+7)

Card 1c. Include this card if the LSDA keyword option is used.

FILENAME

Data Card Definitions:

Volume Fraction Card. This card is included if no keyword option is used. Include as many of this card as desired. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	EID	VF1	VF2	VF3	VF4	VF5	VF6	VF7
Type	I	F	F	F	F	F	F	F
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
EID	Element ID.
VF1	Volume fraction of multi-material group 1, AMMGID = 1.
VF2	Volume fraction of multi-material group 2. Only needed in simulations with at least 3 material groups. Otherwise VF2 = 1 − VF1.
VF3	Volume fraction of multi-material group 3, AMMGID = 3.
VF4	Volume fraction of multi-material group 4, AMMGID = 4.
VF5	Volume fraction of multi-material group 5, AMMGID = 5.
VF6	Volume fraction of multi-material group 6, AMMGID = 6.
VF7	Volume fraction of multi-material group 7, AMMGID = 7.

NALEGP Card. Include this card for the NALEGP keyword option. This card set (Card 1b, 1b.1, and 1b.2) is repeated as many times as desired (one set for each element volume fraction to be defined).

Card 1b	1	
Variable	NALEGP	
Type	I	

VARIABLE	DESCRIPTION
NALEGP	Number of volume fractions.

NALEGP Volume Fractions Card. Include this card for NALEGP keyword option. This card set (Card 1b, 1b.1, and 1b.2) is repeated as many times as desired (one set for each element volume fraction to be defined).

Card 1b.1	1	2	3	4	5	6	7	8
Variable	EID	VF1	VF2	VF3	VF4	VF5	VF6	VF7
Type	I	F	F	F	F	F	F	F
Default	none	0.0	0.0	0.0	0.0	0.0	0.0	0.0

NALEGP Extra Volume Fractions Card. This card is included if the option is NALEGP and NALEGP > 7. Repeat this card in each set enough times to define NALEGP volume fractions.

Card 1b.2	1	2	3	4	5	6	7	8
Variable	VF*i*	VF(*i*+1)	VF(*i*+2)	VF(*i*+3)	VF(*i*+4)	VF(*i*+5)	VF(*i*+6)	VF(*i*+7)
Type	I	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
EID	Element ID.
VF1	Volume fraction of multi-material group 1, AMMGID = 1.
VF2	Volume fraction of multi-material group 2. Only needed in simulations with at least 3 material groups. Otherwise VF2 = 1 − VF1.
VF3	Volume fraction of multi-material group 3, AMMGID = 3.
VF4	Volume fraction of multi-material group 4, AMMGID = 4.
VF5	Volume fraction of multi-material group 5, AMMGID = 5.

VARIABLE	DESCRIPTION
VF6	Volume fraction of multi-material group 6, AMMGID = 6.
VF7	Volume fraction of multi-material group 7, AMMGID = 7.
VF(N)	Volume fraction of multi-material group N, AMMGID = N. Define NALEGP volume fractions.

LSDA File Card. This card is included for keyword option LSDA.

Card 1c	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A							

VARIABLE	DESCRIPTION
FILENAME	Name of the lsda file produced in a previous S-ALE run that contained either *INITIAL_VOLUME_FRACTION_GEOMETRY or *ALE_STRUCTURED_MESH_VOLUME_FILLING. The default name of the lsda file is salevfrc.lsda.

*INITIAL_VOLUME_FRACTION_GEOMETRY

Purpose: This is a volume-filling command for defining the volume fractions of various ALE multi-material groups (AMMG) that initially occupy several spatial regions in an ALE mesh. This applies only to ELFORMs 11 and 12 in *SECTION_SOLID and ALE-FORM 11 in *SECTION_ALE2D. For ELFORM 12, AMMGID 2 is void. See Remark 2.

Card Summary:

Card 1. This card is required.

FMSID	FMIDTYP	BAMMG	NTRACE				

Card 2. For each container, include this card (Container Card) and one geometry card (Card 3α). Include as many pairs as desired. This input ends at the next keyword ("*") card.

CNTTYP	FILLOPT	FAMMG	VX	VY	VZ		

Card 3a. Include this card if CNTTYP = 1.

SID	STYPE		XOFFST	ICUT			

Card 3b. Include this card if CNTTYP = 2.

SGSID		XOFFST	ICUT				

Card 3c. Include this card if CNTTYP = 3.

X0	Y0	Z0	XCOS	YCOS	ZCOS		

Card 3d. Include this card if CNTTYP = 4.

X0	Y0	Z0	X1	Y1	Z1	R1	R2

Card 3e. Include this card if CNTTYP = 5.

X0	Y0	Z0	X1	Y1	Z1	LCSID	

Card 3f. Include this card if CNTTYP = 6.

X0	Y0	Z0	R0				

Card 3g. Include this card if CNTTYP = 7.

IDFUNC							

Data Card Definitions:

Background ALE Mesh Card. Defines the background ALE mesh set & an AMMGID that initially fills it.

Card 1	1	2	3	4	5	6	7	8
Variable	FMSID	FMIDTYP	BAMMG	NTRACE				
Type	I	I	I	I				
Default	none	0	none	3				

VARIABLE	DESCRIPTION
FMSID	Background ALE (fluid) mesh SID to be initialized or filled with various AMMGs. This set ID refers to one or more ALE parts.
FMIDTYP	ALE mesh set ID type: EQ.0: FMSID is an ALE part set ID. EQ.1: FMSID is an ALE part ID.
BAMMG	The background fluid group ID or ALE Multi-Material group ID (AMMGID) that initially fills the entire ALE mesh region defined by FMSID. For S-ALE, AMMG name (AMMGNM) could also be used in place of AMMGID. See Remark 8.
NTRACE	Number of sampling points for volume filling detection. Typically, NTRACE ranges from 3 to maybe 10 (or more). The higher it is, the finer the ALE element is divided so that small gaps between 2 Lagrangian shells may be filled in. See Remark 4.

Container Card. Defines the container type and the AMMGID that fills the region defined by the container type.

Card 2	1	2	3	4	5	6	7	8
Variable	CNTTYP	FILLOPT	FAMMG	VX	VY	VZ		
Type	I	I	I	F	F	F		
Default	none	0	none	0	0	0		

VARIABLE	DESCRIPTION
CNTTYP	A "container" defines a Lagrangian surface boundary of a spatial region, inside (or outside) of which, an AMMG would fill up. CNTTYP defines the container geometry type of this surface boundary (or shell structure). EQ.1: The container geometry is defined by a part, a part set, or an element set (a solid, shell, or beam set). The parts (see *PART or *SET_PART) should be defined by shell elements in 3D (beam elements in 2D). If the parts are meshed with solids in 3D (shells in 2D), their boundaries define the container geometry (see Remark 7). EQ.2: The container geometry is defined by a segment set. EQ.3: The container geometry is defined by a plane: a point and a normal vector. EQ.4: The container geometry is defined by a conical surface: 2 end points and 2 corresponding radii (in 2D see Remark 6). EQ.5: The container geometry is defined by a cuboid or rectangular box: 2 opposing end points, minimum to maximum coordinates. EQ.6: The container geometry is defined by a sphere: 1 center point, and a radius. EQ.7: The container geometry is defined with a user-defined function implemented using *DEFINE_FUNCTION. The arguments of the function should be the coordinates of a point (x,y,z). The function should return 1.0 if the point is inside the geometry.

VARIABLE	DESCRIPTION
FILLOPT	A flag to indicate which side of the container surface the AMMG is supposed to fill. CNTTYP = 1, 2, and 3, the "head" side of a container surface/segment is defined as the side pointed to by the heads of the normal vectors of the segments ("tail" side refers to the opposite direction from the "head"). See Remark 5. Note that for CNTTYP = 1 and 2, the fluid interface can be offset from the container walls with XOFFST. XOFFST does not apply to the other container geometries. EQ.0: The "head" side of the geometry defined above will be filled with fluid (default). For CNTTYP = 4, 5, 6, and 7, the inside of the container is filled. EQ.1: The "tail" side of the geometry defined above will be filled with fluid. For CNTTYP = 4, 5, 6, and 7, the outside of the container is filled.
FAMMG	This defines the fluid group ID or ALE Multi-Material group ID (AMMGID), which will fill up the interior (or exterior) of the space defined by the "container". *The order of AMMGIDs is determined by the order in which they are listed under *ALE_MULTI-MATERIAL_GROUP card.* For example, the first data card under the *ALE_MULTI-MATERIAL_GROUP keyword defines the multi-material group with ID (AMMGID) 1, the second data card defines AMMGID = 2, and so on. In case of S-ALE, AMMG name (AMMGNM) could also be used in place of AMMGID. See Remark 8. LT.0: |FAMMG| is a *SET_MULTI-MATERIAL_GROUP_LIST ID listing pairs of group IDs. For each pair, the 2nd group replaces the first one in the "container".
VX	Initial velocity in the global *x*-direction for this AMMGID
VY	Initial velocity in the global *y*-direction for this AMMGID
VZ	Initial velocity in the global *z*-direction for this AMMGID

Part/Part Set Container Card. Additional card for CNTTYP = 1.

Card 3a	1	2	3	4	5	6	7	8
Variable	SID	STYPE	NORMDIR	XOFFST	ICUT			
Type	I	I	I	F	I			
Default	none	0	0	0.0	0			
Remark			obsolete					

VARIABLE	DESCRIPTION
SID	A set ID pointing to a part ID, part set ID, solid element set ID, shell element set ID, or beam set ID of the Lagrangian structure defining the "container" geometry to be filled
SSTYPE	Set ID type: EQ.0: Container SID is a Lagrangian part set ID (*SET_PART). EQ.1: Container SID is a Lagrangian part ID (*PART). EQ.2: Container SID is a Lagrangian solid set in 3D (*SET_SOLID) or a Lagrangian shell set in 2D (*SET_SHELL). EQ.3: Container SID is a Lagrangian shell set in 3D (*SET_SHELL) or a Lagrangian beam set in 2D (*SET_BEAM).
NORMDIR	Obsolete (see Remark 5).
XOFFST	|XOFFST| is the absolute length for offsetting the fluid interface from the default nominal fluid interface. The sign of XOFFST determines the direction in which the interface is offset. It is based on the normal vectors of the segments associated with the container. GT.0: The interface is offset along the positive direction of the segments of the container. LT.0: The interface is offset in the negative direction of the normal vectors of the segments of the container. This field is applicable to cases in which a container contains high-pressure fluid. The offset gives time to prevent leakage. In general, this may be set to roughly 5-10% of the ALE element width. It may be important only when ILEAK is turned ON to give the code time

VARIABLE	DESCRIPTION
	to "catch" the leakage (see *CONSTRAINED_LAGRANGE_IN_-SOLID). If ILEAK is not ON, this may not be necessary.
ICUT	Flag to activate computing the exact volume instead of an estimate when filling ALE elements cut by the Lagrangian structure: EQ.0: Inactive EQ.1: Active

Segment Set Container Card. Additional card for CNTTYP = 2.

Card 3b	1	2	3	4	5	6	7	8
Variable	SGSID	NORMDIR	XOFFST	ICUT				
Type	I	I	F	I				
Default	none	0	0.0	0				
Remark		obsolete						

VARIABLE	DESCRIPTION
SGSID	Segment Set ID defining the "container." See *SET_SEGMENT.
NORMDIR	Obsolete (see Remark 5).
XOFFST	|XOFFST| is the absolute length for offsetting the fluid interface from the default nominal fluid interface. The sign of XOFFST determines the direction in which the interface is offset. It is based on the normal vectors of the segments associated with the container. GT.0: The interface is offset along the positive direction of the segments of the container. LT.0: The interface is offset in the negative direction of the normal vectors of the segments of the container. This applies to cases where high-pressure fluid is contained within a container. The offset allows time to prevent leakage. In general, this may be set to roughly 5-10% of the ALE element width. It may be important only when ILEAK is turned ON to give the code time to "catch" the leakage (see *CONSTRAINED_LAGRANGE_IN_-

VARIABLE	DESCRIPTION
	SOLID). If ILEAK is not ON, this may not be necessary.
ICUT	Flag to activate computing the exact volume instead of an estimate when filling ALE elements cut by the Lagrangian structure: EQ.0: Inactive EQ.1: Active

Plane Card. Additional card for CNTTYP = 3.

Card 3c	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	XCOS	YCOS	ZCOS		
Type	F	F	F	F	F	F		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
X0, Y0, Z0	x, y and z coordinate of a spatial point on the plane.
XCOS, YCOS, ZCOS	x, y and z direction cosines of the plane normal vector. The filling will occur on the side pointed to by the plane normal vector (or "head" side).

Cylinder/Cone Container Card. Additional Card for CNTTYP = 4 (see Remark 6 for 2D).

Card 3d	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	X1	Y1	Z1	R1	R2
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
X0, Y0, Z0	x, y, and z coordinates of the center of the 1st base of the cone.

VARIABLE	DESCRIPTION
X1, Y1, Z1	x, y, and z coordinates of the center of the 2nd base of the cone.
R1	Radius of the 1st base of the cone
R2	Radius of the 2nd base of the cone

Rectangular Box Container Card. Additional Card for CNTTYP = 5.

Card 3e	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	X1	Y1	Z1	LCSID	
Type	F	F	F	F	F	F	I	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
X0, Y0, Z0	Minimum x, y, and z coordinates of the box.
X1, Y1, Z1	Maximum x, y, and z coordinates of the box.
LCSID	Local coordinate system ID, if defined, the box is aligned with the local coordinate system instead of the global coordinate system. See *DEFINE_COORDINATE_*OPTION*for details.

Sphere Container Card. Additional card for CNTTYP = 6.

Card 3f	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	R0				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
X0, Y0, Z0	x, y and z coordinate of the center of the sphere.
R0	Radius of the sphere

User Defined Container Card. Additional card for CNTTYP = 7.

Card 3g	1	2	3	4	5	6	7	8
Variable	IDFUNC							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
IDFUNC	Function ID (*DEFINE_FUNCTION) for the function of the coordinates (x,y,z) that gives the container geometry. If the point (x,y,z) is in the container, the function should return 1.0.

Remarks:

1. **Structure of data cards.** After Card 1, which defines the basic mesh filled by a certain fluid group (AMMGID), each "filling action" requires two additional lines of input: Cards 2 and 3α, where α depends on the CNTTYP value. At a minimum, this keyword requires three cards (1, 2, and 3α) for one "filling action."

   Each instance of this keyword can prescribe one or more "filling actions." The "filling actions" take place in the prescribed order, and the effects are cumulative. Later "filling actions" overwrite the previous ones. Therefore, any complex filling logic requires some planning. For example, the following card sequence with 2 "filling actions" is allowable:

```
*INITIAL_VOLIME_FRACTION_GEOMETRY
[Card 1]
[Card 2, CNTTYP = 1]
[Card 3a]
[Card 2, CNTTYP = 3]
[Card 3c]
```

   This sequence of cards prescribes a background ALE mesh with two "filling actions" to be executed. The 1st is a filling of a CNTTYP = 1 and the 2nd of CNTTYP = 3.

2. **Group IDs for ELFORM 12.** If ELFORM = 12, the single-material-and-void element formulation, is used in *SECTION_SOLID, then the non-void material defaults to AMMG = 1 and the void to AMMG = 2. These multi-material groups

are implied even though no *ALE_MULTI-MATERIAL_GROUP card is required.

3. **Using shells to divide space.** A simple ALE background mesh (for example, a cuboid mesh) can be constructed enveloping some Lagrangian shell structure (or container). The ALE region inside this Lagrangian shell container may be filled with one multi-material group (AMMG1), and the outside region with another (AMMG2). This approach simplifies the mesh generation requirements for ALE material parts with complex geometries.

4. **NTRACE.** The default number of sampling points is NTRACE = 3 in which case the total number is

$$(2 \times \text{NTRACE} + 1)^3 = 7^3$$

This means an ALE element is subdivided into $7 \times 7 \times 7$ regions. Each is to be filled in with the appropriate AMMG. An example of this application would be the filling of initial gas between multiple layers of Lagrangian airbag shell elements sharing the same ALE element.

5. **Interior/exterior fill setting.** To set which side of a container is to be filled: (1) define the shell (or segment) container with inward normal vectors; then (2) set the FILLOPT field on "Card 2" to 0, corresponding to the head of the normal, for the interior, and to 1, corresponding to the tail of the normal, for the exterior.

6. **Two-dimensional geometry.** If the ALE model is 2D (*SECTION_ALE2D instead of *SECTION_SOLID), CNTTYP = 4 defines a quadrilateral. In this case, the fields which, in the 3D case define a cone, are interpreted as the corner coordinates of a clockwise defined (inward normal) quadrilateral having the vertices: (X1, Y1), (X2, Y2), (X3, Y3), and (X4, Y4). The CNTTYP = 4 input fields X0, Y0, Z0, X1, Y1, Z1, R1, and R2 becomes X1, Y1, X2, Y2, X3, Y3, X4, and Y4 respectively. CNTTYP = 6 should be used to fill a circle.

7. **3D solid (2D shell) mesh for CNTTYP = 1.** If a part, *P*, defines the geometry container for CNTTYP = 1, is meshed with solids in 3D or shells in 2D, and is in the ALE group FAMMG (*P* should be in *ALE_MULTI-MATERIAL_GROUP and it should have the ALE formulation 11 in the *SECTION keyword), the ALE elements superimposed with *P* will be filled with FAMMG. The mesh of *P* will become a dummy rigid part.

8. **AMMG name for S-ALE.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, each AMMG can only be referred to by their AMMGID. The AMMGID for each AMMG is based on the order of appearance of the AMMG in the input deck. For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_MULTI-MATERIAL_-GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the

field AMMGNM. Each AMMG defined with that keyword can then be referred to with either its name or its AMMGID (which is again based on order of appearance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, then the AMMGIDs may change. Then, you must find all those references and change them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

9. **Files output for subsequent analyses.** This keyword causes file alevfrc.inc to be output for both ALE and S-ALE simulations at the end of the run. For S-ALE, it also outputs file salevfrc.lsda, which is an lsda version of alefrc.inc. alefrc.inc can be included with *INCLUDE for subsequent analyses while salevfrc.lsda can be included with *INITIAL_VOLUME_FRACTION_LSDA. These files pre-fill the volume fractions and store the trimmed mesh, saving time for subsequent analyses.

Example:

Consider a simple ALE model with ALE parts H1-H5 (5 AMMGs possible) and 1 Lagrangian shell (container) part S6. Only parts H1 and S6 initially have their meshes defined. We will perform 4 "filling actions". The volume filling results after each step will be shown below to clarify the concept used. The input for the volume filling looks like this.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ H1 = AMMG 1 = fluid 1 initially occupying whole ALE mesh= background mesh
$ H5 = AMMG 5 = fluid 5 fills below a plane = filling action 1 = CNTTYP=3
$ H2 = AMMG 2 = fluid 2 fills outside S6    = filling action 2 = CNTTYP=1
$ H3 = AMMG 3 = fluid 3 fills inside a cone = filling action 3 = CNTTYP=4
$ H4 = AMMG 4 = fluid 4 fills inside a box  = filling action 4 = CNTTYP=5
$ S6 =          Lagrangian shell container
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
         1         1
         2         1
         3         1
         4         1
         5         1
*INITIAL_VOLUME_FRACTION_GEOMETRY
$ The 1st card fills the whole pid H1 with AMMG 1=background ALE mesh
$    FMSID   FMIDTYP     BAMMG     <=== card 1: background fluid
         1         1         1
$ filling action 1 = AMMG 5 fill all elms below a plane
$ CNTTYPE   FILLOPT FILAMMGID     <=== card a  : container: CNTTYPE=3=plane
         3         0         5
$   X0, Y0,   Z0,    NX,  NY, NZ   <=== card b-3: details on container =plane
  25.0,20.0, 0.0,   0.0,-1.0,0.0
$ filling action 2: AMMG 2 fills OUTSIDE (FILLOPT=1) shell S6 (inward normals);
$ CNTTYPE   FILLOPT     FAMMG  <== card a  : container #1; FILLOPT=1=fill tail
         1         1         2
$    SETID   SETTYPE   NORMDIR  <== card b-1: details on container #1
         6         1         0
$ filling action 3 = AMMG 3 fill all elms inside a CONICAL region
$ CNTTYPE   FILLOPT     FAMMG  CNTTYP = 4 = Container = conical region
         4         0         3
$       X1        Y1        Z1        X2        Y2        Z2        R1        R2
     25.0      75.0       0.0      25.0      75.0       1.0       8.0       8.0
```

```
$ filling action 4 = AMMG 4 fill all elms inside a BOX region
$ CNTTYPE   FILLOPT  FFLUIDID                 : CNTTYP=5 = "BOX"
         5         0         4
$     XMIN      YMIN      ZMIN      XMAX      YMAX      ZMAX
      65.0      35.0       0.0      85.0      65.0       1.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

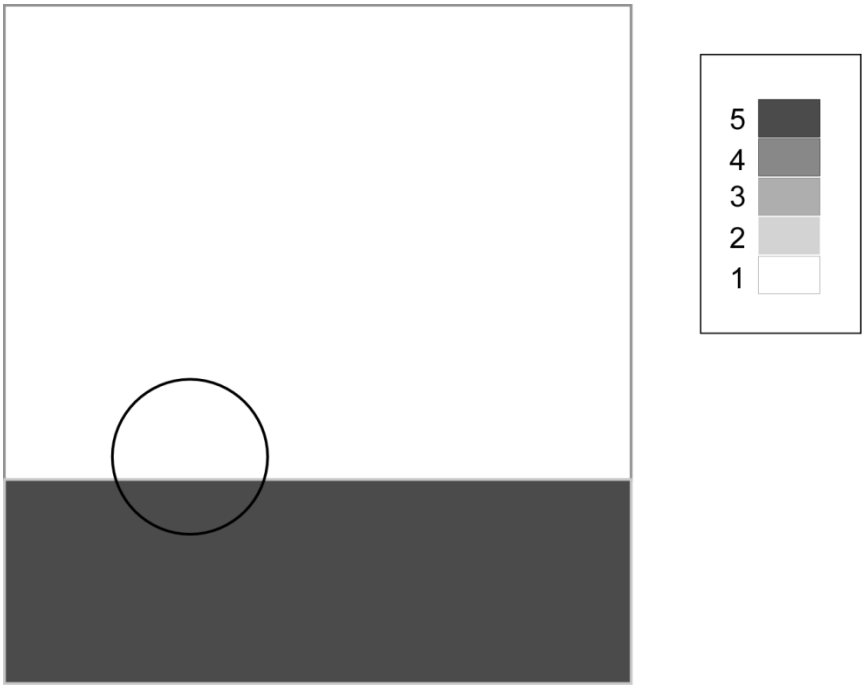


Figure 28-10. Before the 1st "filling action" the whole ALE mesh of part H1 is filled with AMMG 1 (white). After the 1st "filling action", AMMG 5 fills below the specified plane.

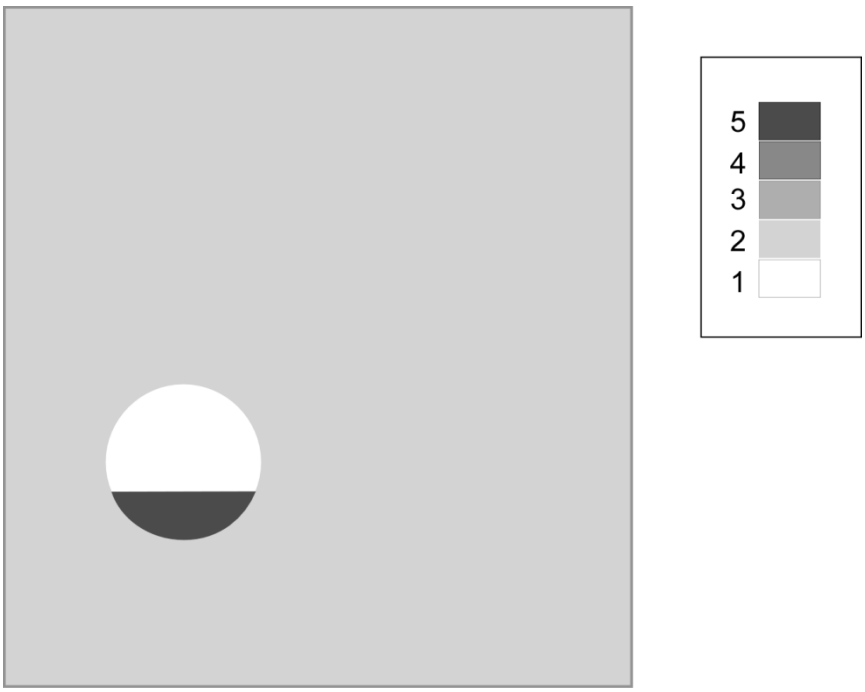


Figure 28-11. After the 1st and 2nd "filling actions", outside the shell (S6) is filled with AMMG 2.

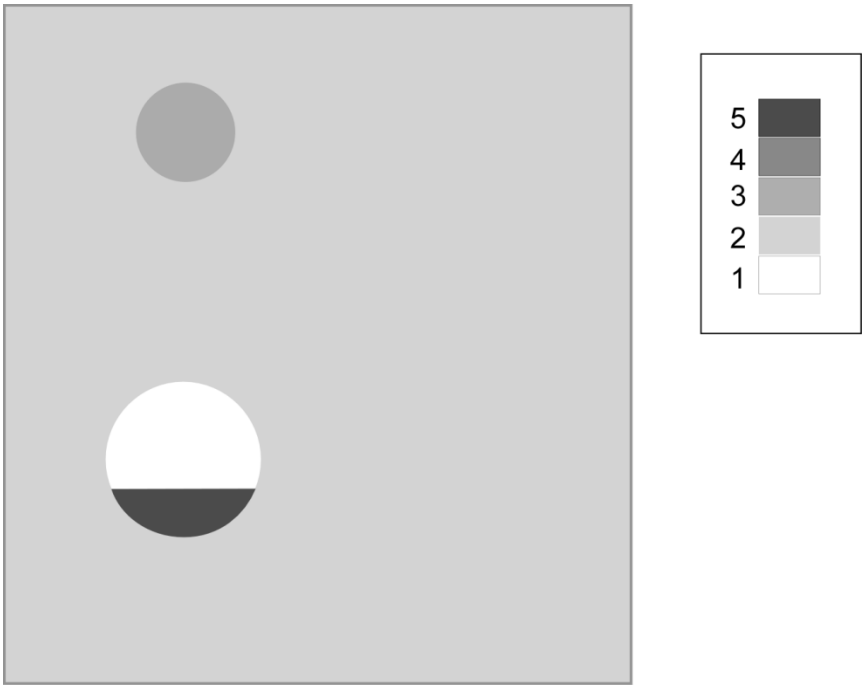


Figure 28-12. After the 1st, 2nd and 3rd "filling actions", it fills in the analytical sphere with AMMG 3.

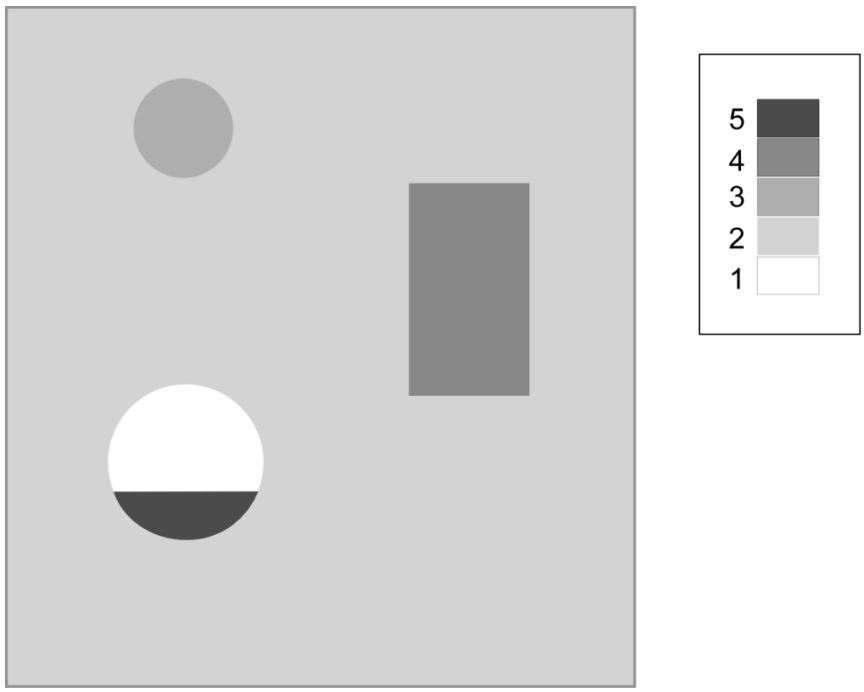


Figure 28-13. After the 1st, 2nd, 3rd and 4th "filling actions", the analytical box is filled with AMMG 4.

*INTEGRATION

In this section the user defined integration rules for beam and shell elements are specified. IRID refers to integration rule identification number on *SECTION_BEAM and *SECTION_SHELL cards respectively. Quadrature rules in the *SECTION_SHELL and *SECTION_BEAM cards need to be specified as a negative number. The absolute value of the negative number refers to user defined integration rule number. Positive rule numbers refer to the built in quadrature rules within LS-DYNA. The keyword cards in this section are:

*INTEGRATION_BEAM

*INTEGRATION_SHELL

*INTEGRATION_BEAM

Purpose: To support user defined through the thickness integration rules for the beam element.

Card 1	1	2	3	4	5	6	7	8
Variable	IRID	NIP	RA	ICST	K			
Type	I	I	F	I	I			
Default	none	0	0.0	0	0			

Standard Cross-Section Card. Additional card for ICST > 0.

Card 2	1	2	3	4	5	6	7	8
Variable	D1	D2	D3	D4	SREF	TREF	D5	D6
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	0.0	0.0	none	none

Quadrature Cards. Include NIP additional cards below for NIP ≠ 0.

Card 3	1	2	3	4	5	6	7	8
Variable	S	T	WF	PID				
Type	F	F	F	I				

VARIABLE	DESCRIPTION
IRID	Integration rule ID. IRID refers to IRID on *SECTION_BEAM card.
NIP	Number of integration points; see also ICST.
RA	Relative area of cross section, that is, the actual cross-sectional area divided by the area defined by the product of the specified thickness in the *s*-direction and the thickness in the *t*-direction. See also

VARIABLE	DESCRIPTION
	ICST below and Figure 29-1.
ICST	Standard cross section type. If this type is nonzero, then NIP and the relative area above should be input as zero. See shapes in Figures 29-3 through 29-24.

EQ.01: I-Shape	EQ.12: Cross
EQ.02: Channel	EQ.13: H-Shape
EQ.03: L-shape	EQ.14: T-Shape 2
EQ.04: T-shape	EQ.15: I-Shape 3
EQ.05: Tubular box	EQ.16: Channel 2
EQ.06: Z-Shape	EQ.17: Channel 3
EQ.07: Trapezoidal	EQ.18: T-Shape 3
EQ.08: Circular	EQ.19: Box-Shape 2
EQ.09: Tubular	EQ.20: Hexagon
EQ.10: I-Shape 2	EQ.21: Hat-Shape
EQ.11: Solid Box	EQ.22: Hat-Shape 2

K	Integration refinement parameter, k, for standard cross section types. Select an integer ≥ 0. See Figures 29-3 through 29-24.
D1-D6	Cross-section dimensions. See Figures 29-3 through 29-24.
SREF	s_{ref}, location of reference surface normal to s, for the Hughes-Liu beam only. This option is only useful if the beam is connected to a shell or another beam on its outer surface. Overrides NSLOC in *SECTION_BEAM, even if SREF = 0.
TREF	t_{ref}, location of reference surface normal to t, for the Hughes-Liu beam only. This option is only useful if the beam is connected to a shell or another beam on its outer surface. Overrides NTLOC in *SECTION_BEAM, even if TREF = 0.
S	Normalized s-coordinate of integration point, $-1 \le s \le 1$.
T	Normalized t-coordinate of integration point, $-1 \le t \le 1$.
WF	Weighting factor, A_{ri}, that is the area associated with the integration point divided by actual cross sectional area $A_{ri} = A_i/A$; see Figure 29-2.

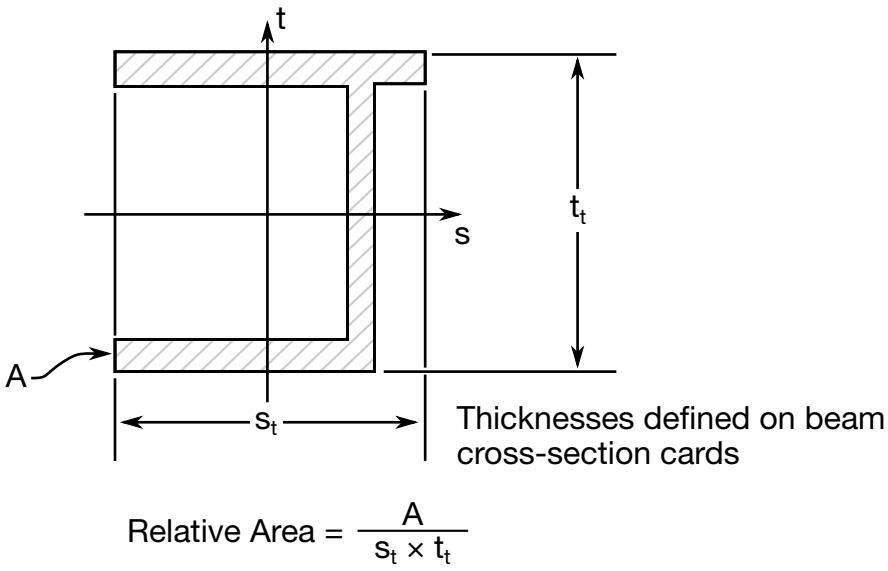


Figure 29-1. Definition of relative area for user defined integration rule.

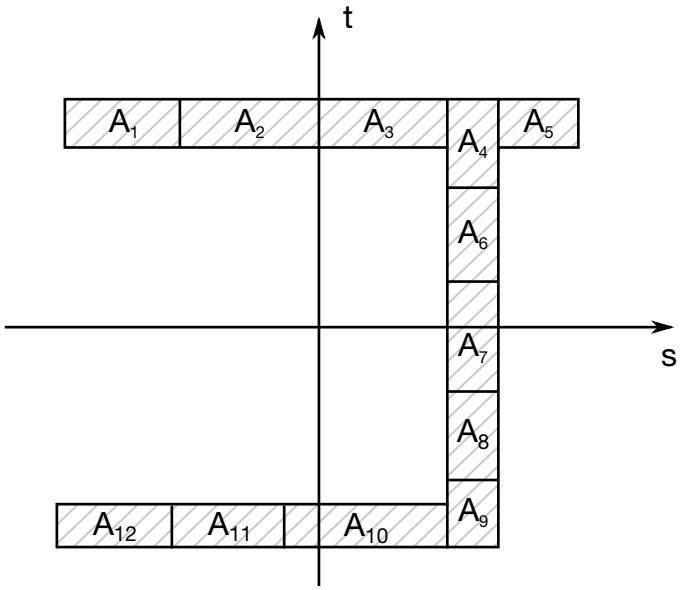


Figure 29-2. Definition of integration points for user defined integration rule.

VARIABLE	DESCRIPTION
PID	Optional PID, used to identify material properties for this integration point. If zero, the PID referenced on *ELEMENT_BEAM will be used.

Remarks:

The input for standard beam section types is defined below. In the following figures the dimensions are shown on the left and the location of the integration points are shown on

the right. If a quantity is not defined in the sketch, then it should be set to zero in the input. The input quantities include:

D1 - D6	=	Dimensions of section
k	=	Integration refinement parameter (an integer ≥ 0)
s_{ref}	=	location of reference surface normal to *s*, Hughes-Liu beam only
t_{ref}	=	location of reference surface normal to *t*, Hughes-Liu beam only

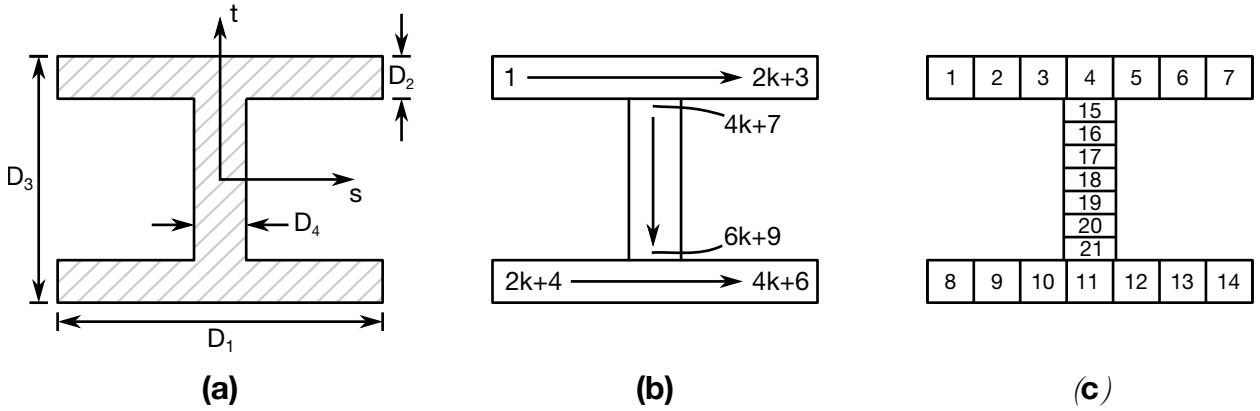


Figure 29-3. Type 1: I-Shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

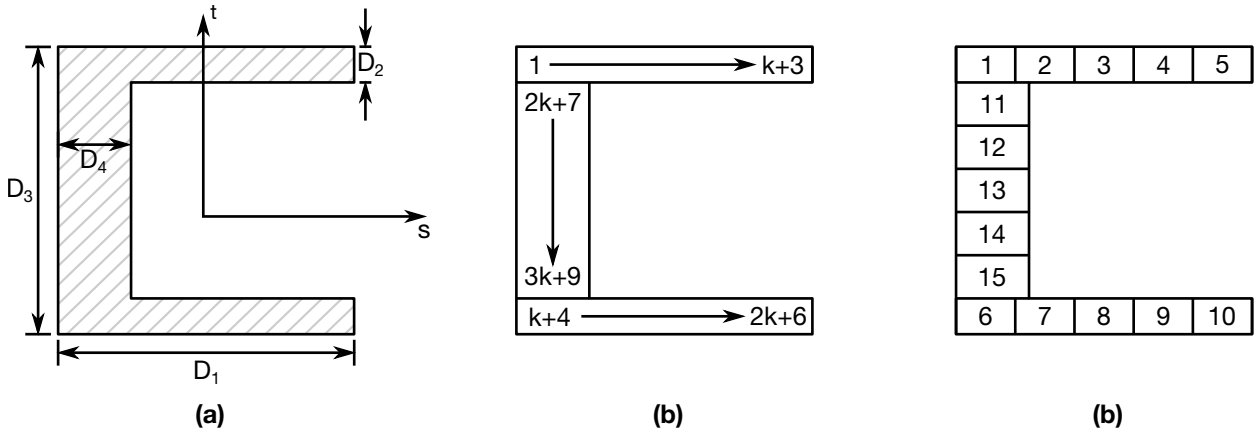


Figure 29-4. Type 2: Channel. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

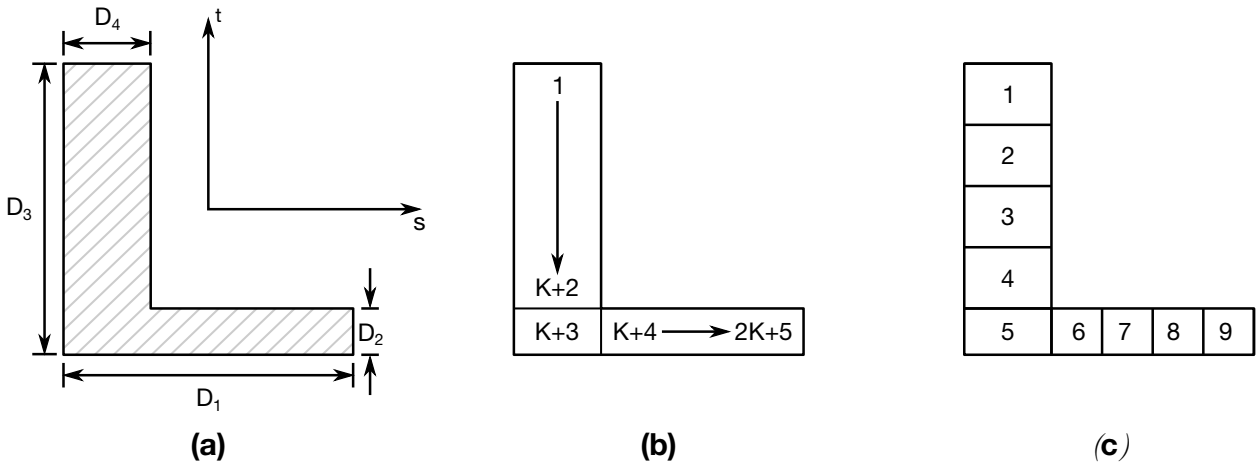


Figure 29-5. Type 3: L-shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

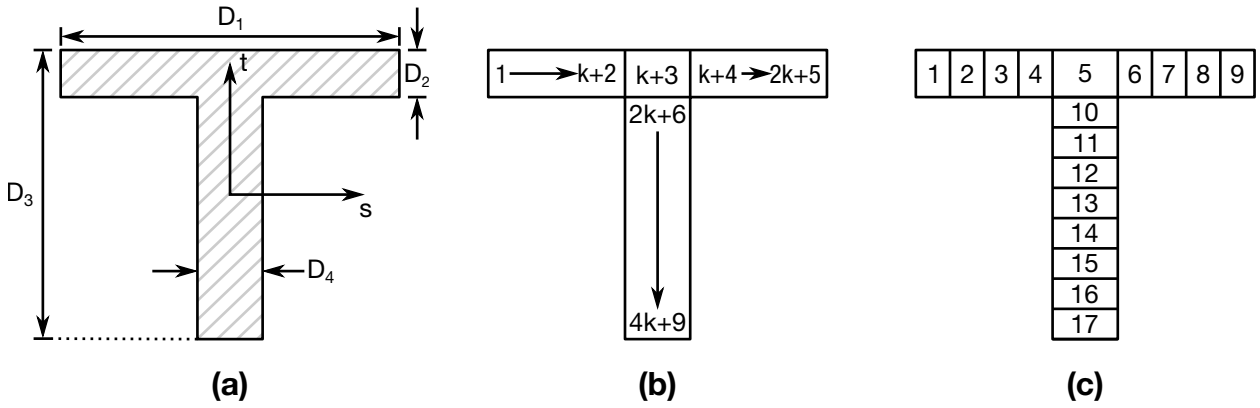


Figure 29-6. Type 4: T-shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

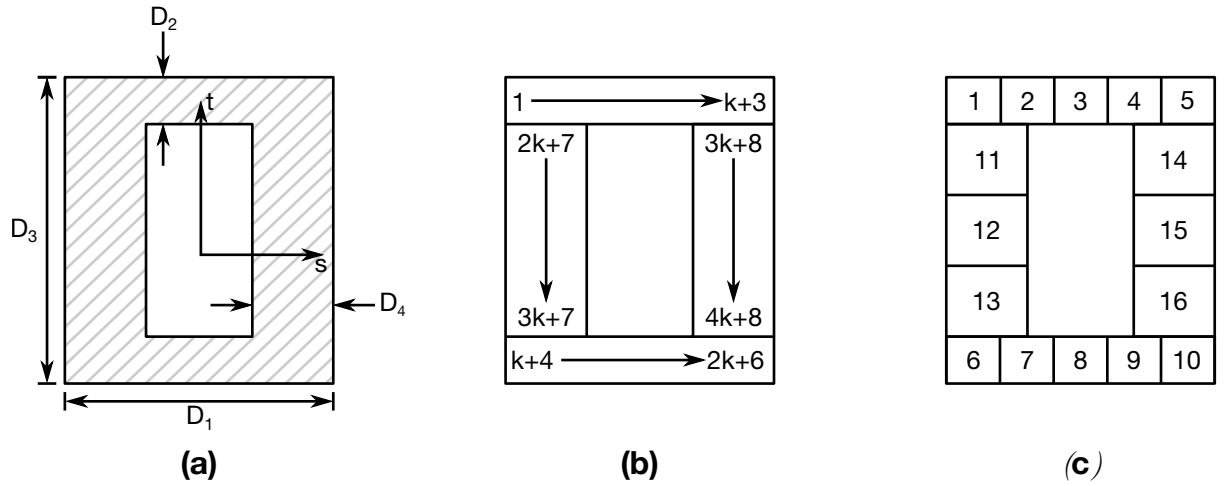


Figure 29-7. Type 5: Box-shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

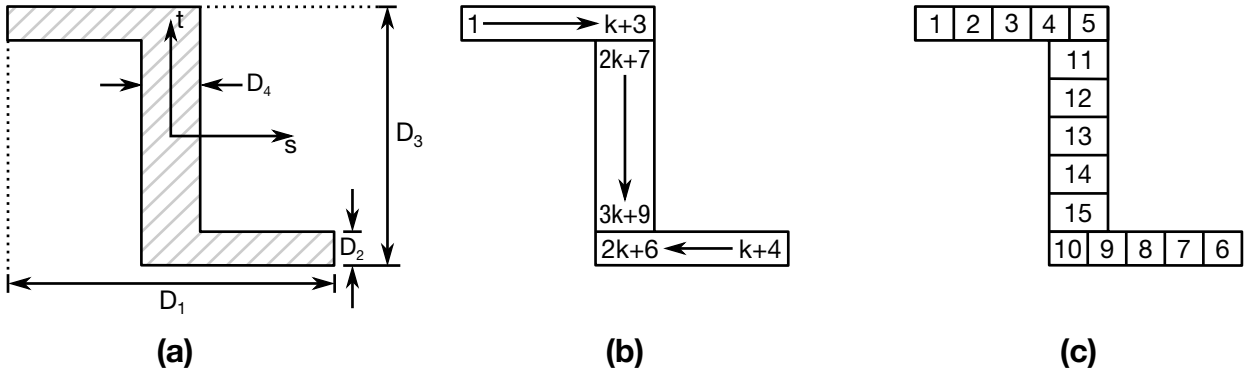


Figure 29-8. Type 6: Z-shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

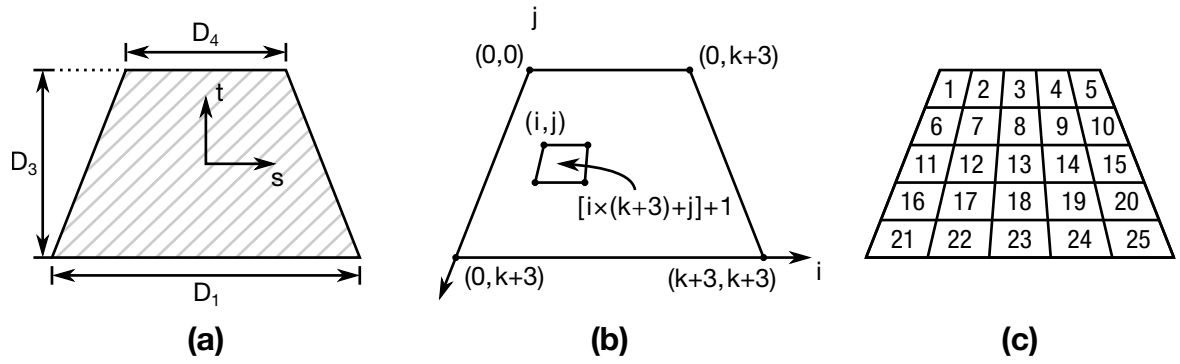


Figure 29-9. Type 7: Trapezoidal. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

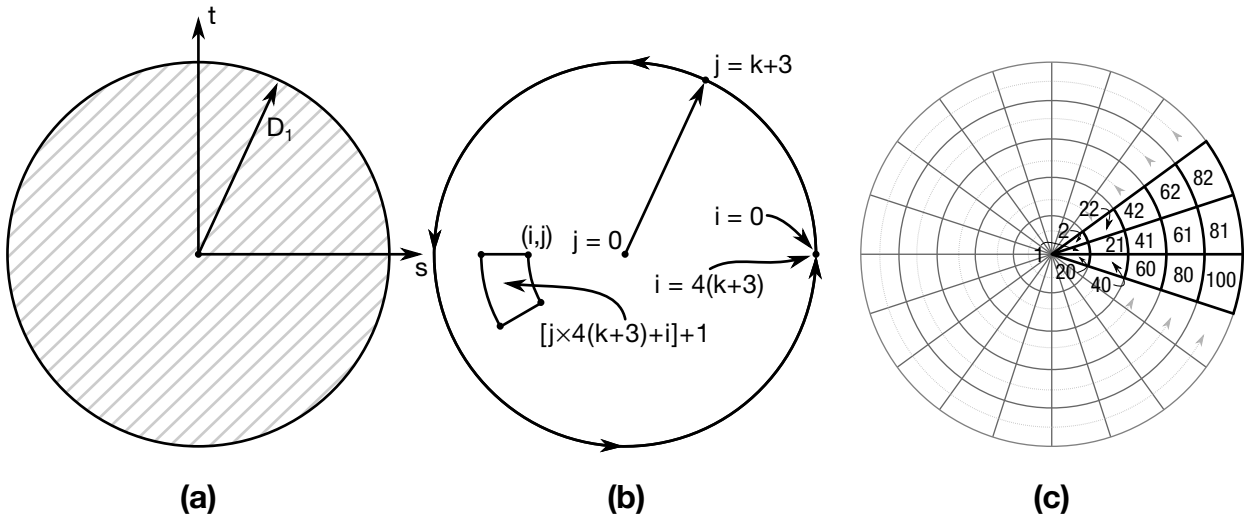


Figure 29-10. Type 8: Circular. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

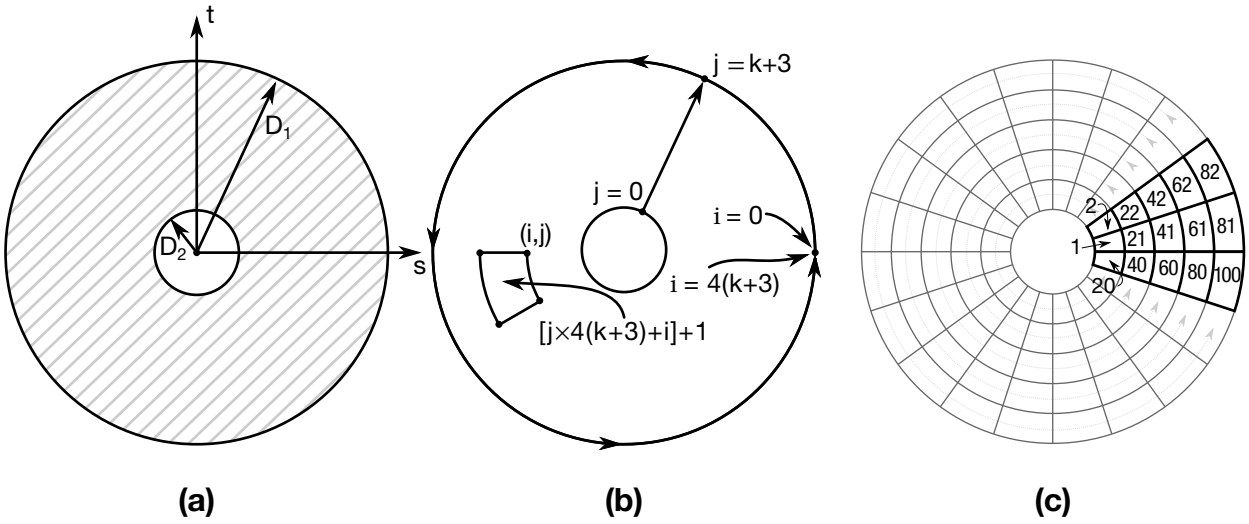


Figure 29-11. Type 9: Tubular. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

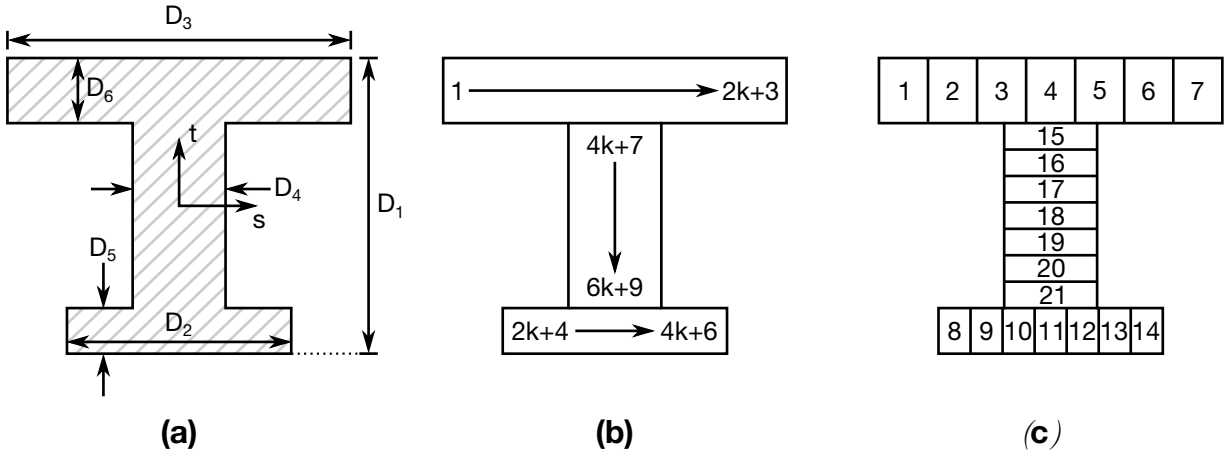


Figure 29-12. Type 10: I-Shape 2. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

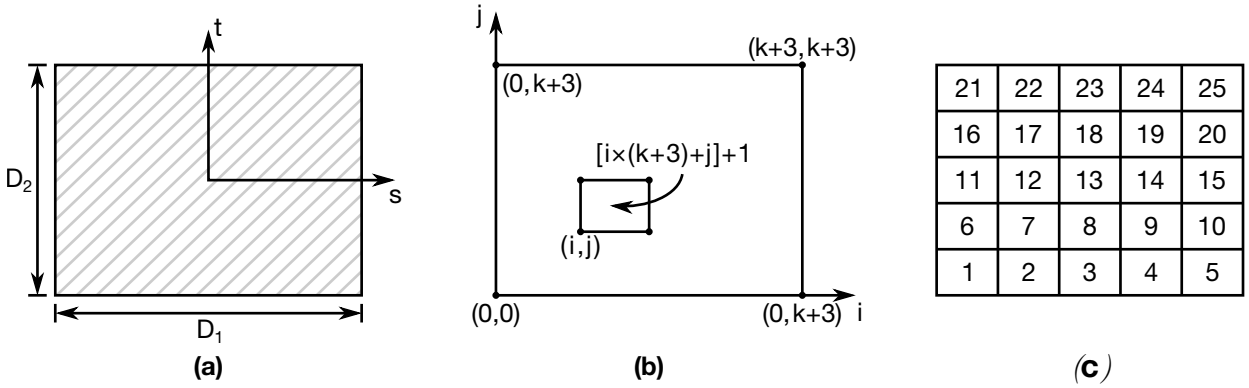


Figure 29-13. Type 11: Solid Box. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

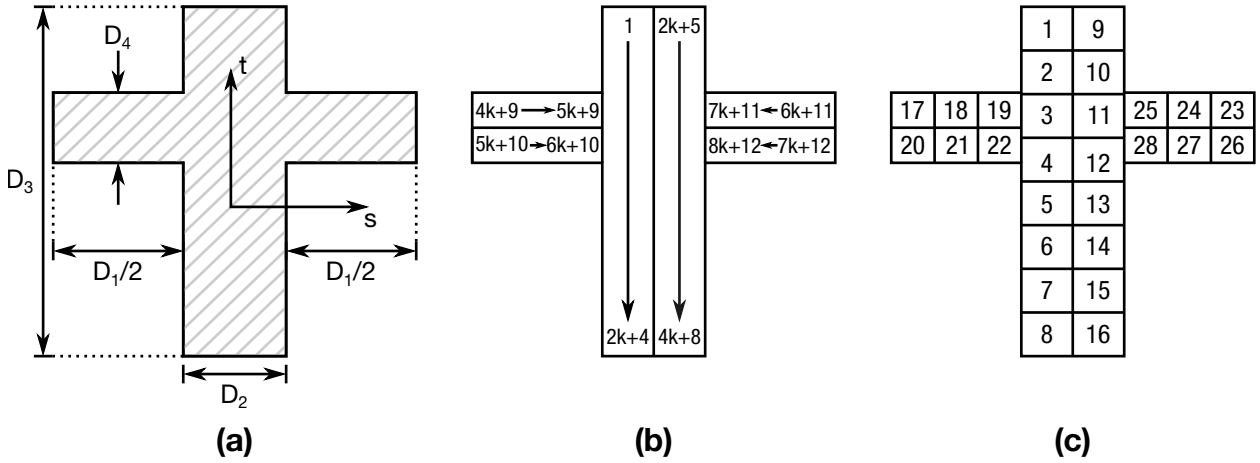


Figure 29-14. Type 12: Cross. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

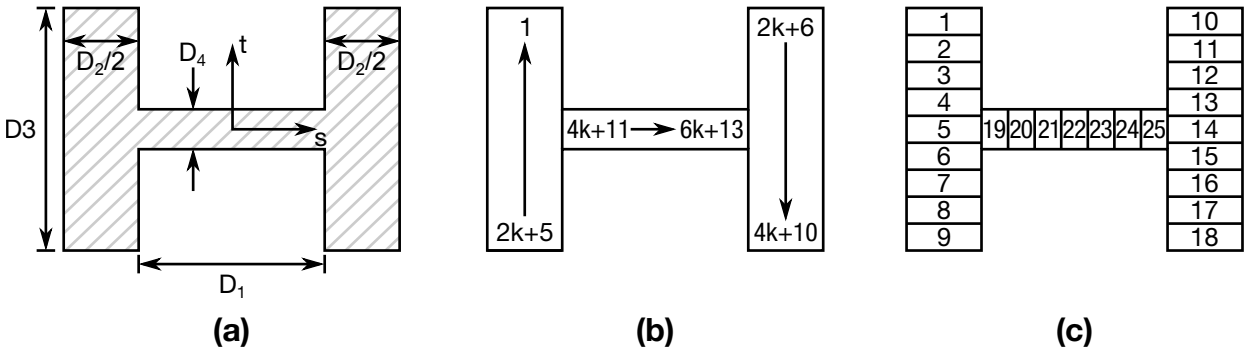


Figure 29-15. Type 13: H-Shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

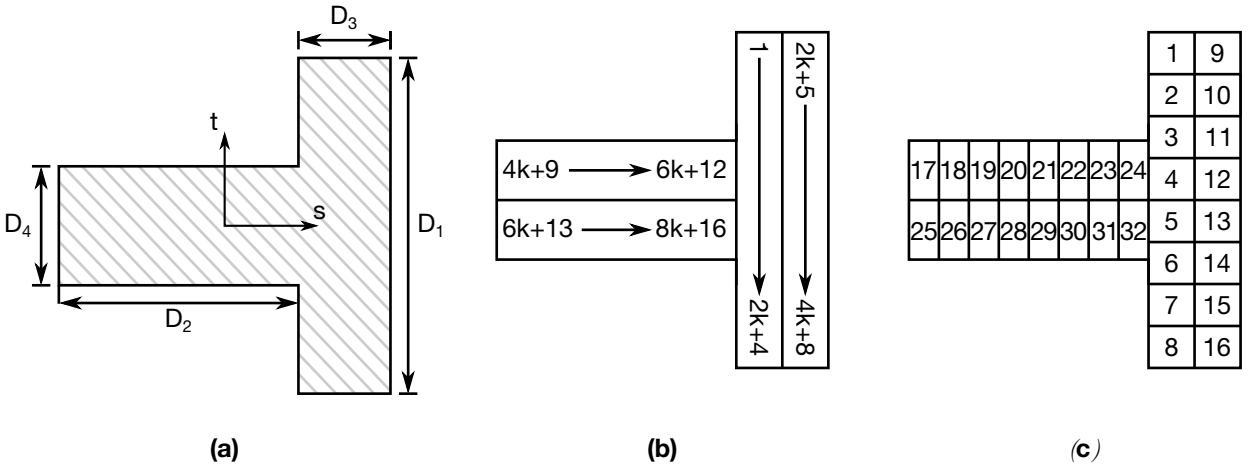


Figure 29-16. Type 14: T-Shape 2. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

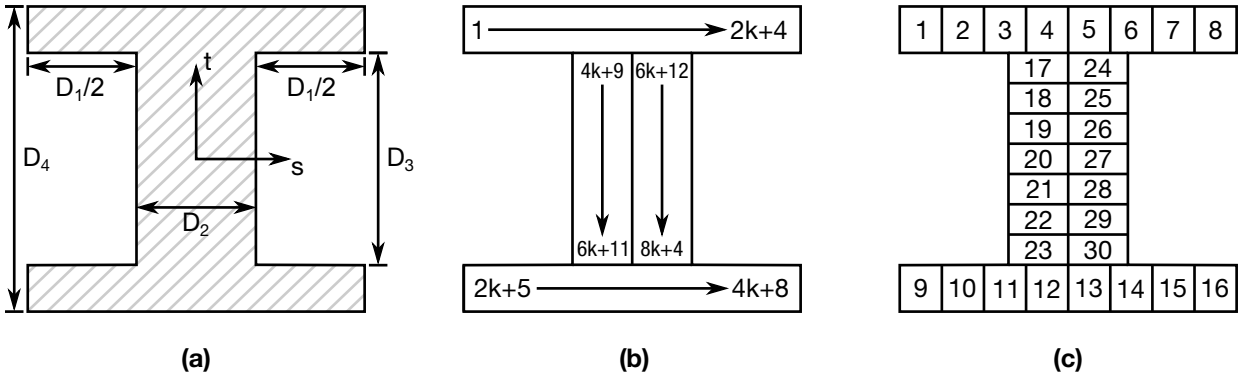


Figure 29-17. Type 15: I-Shape 3. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

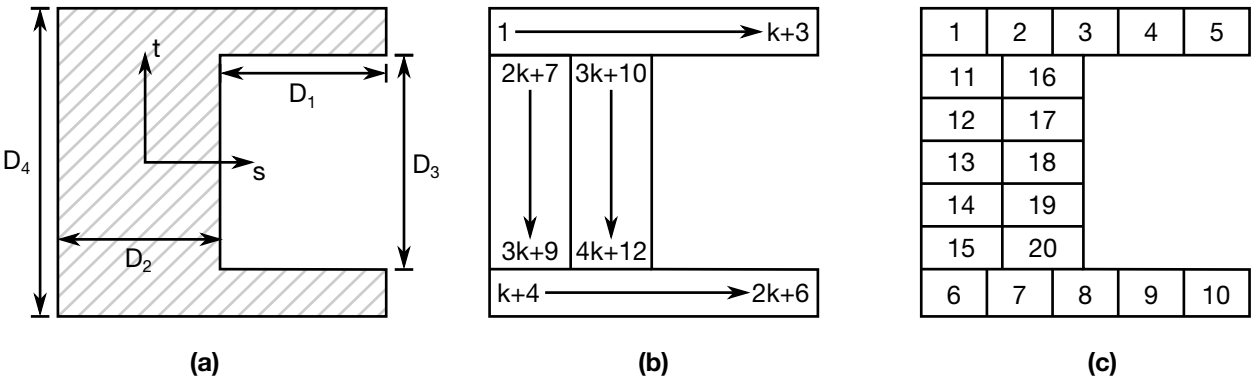


Figure 29-18. Type 16: Channel 2. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

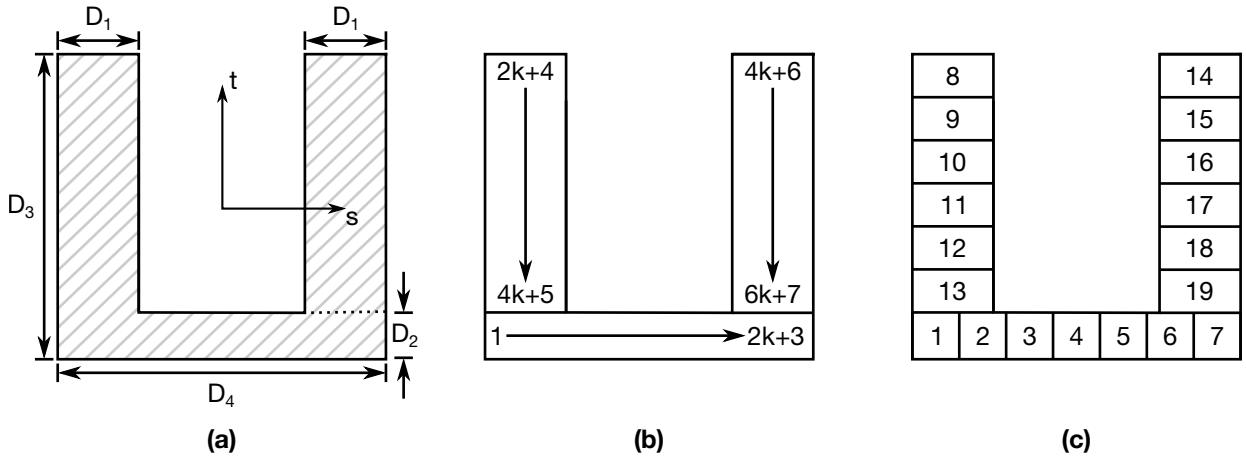


Figure 29-19. Type 17: Channel 3. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

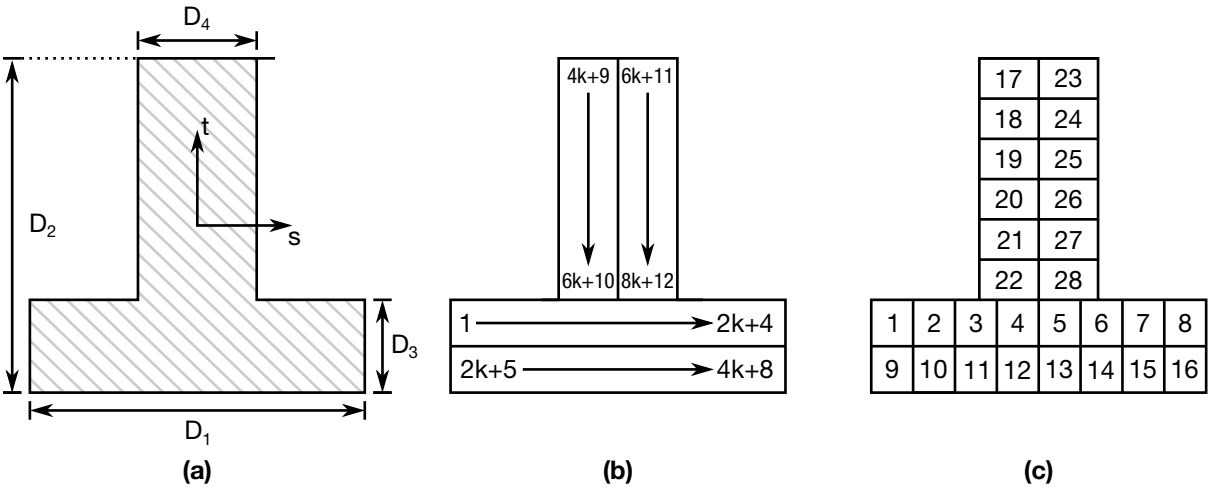


Figure 29-20. Type 18: T-Shape 3. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

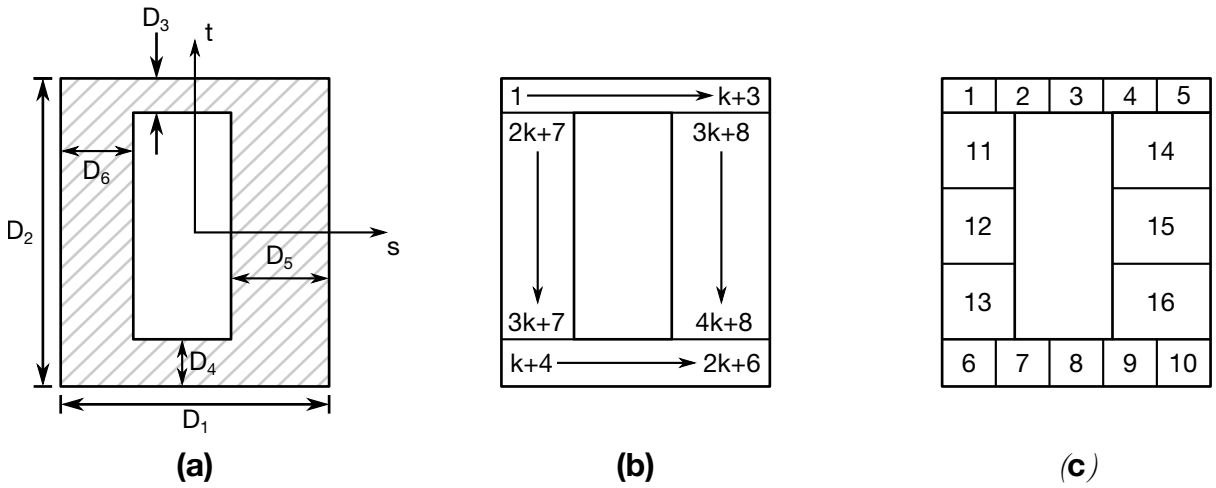


Figure 29-21. Type 19: Box Shape 2. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

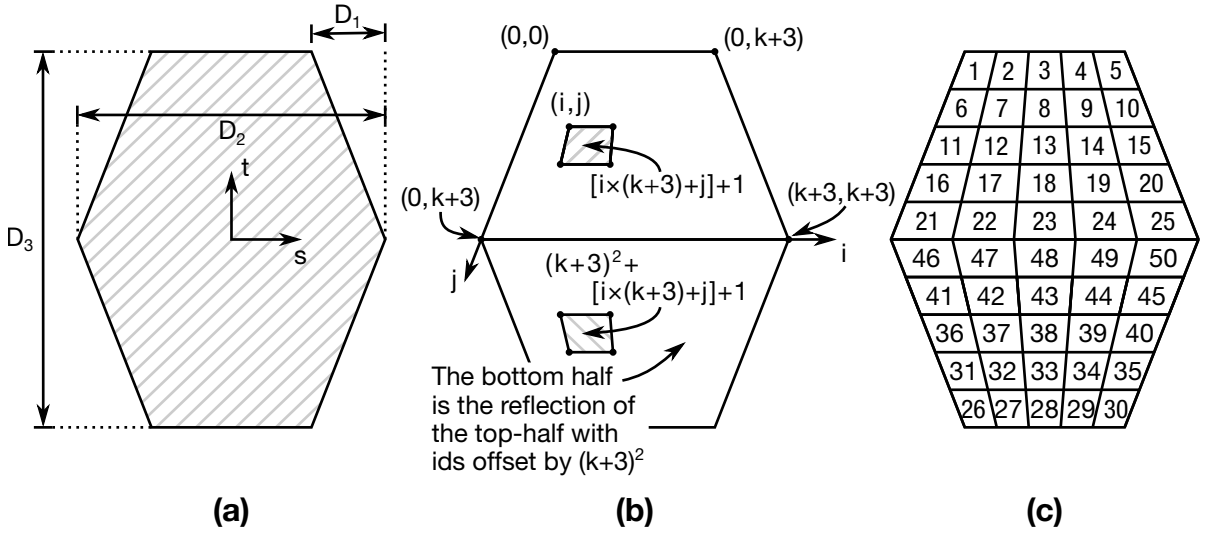


Figure 29-22. Type 20: Hexagon. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

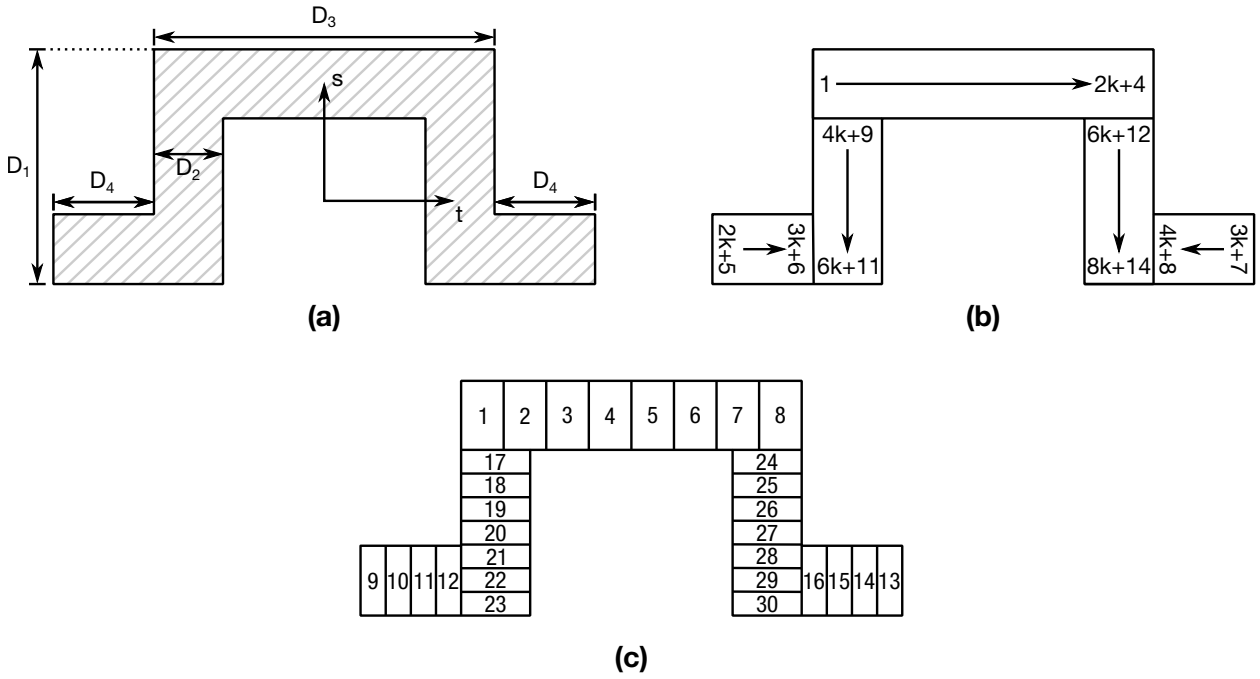


Figure 29-23. Type 21: Hat-Shape. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

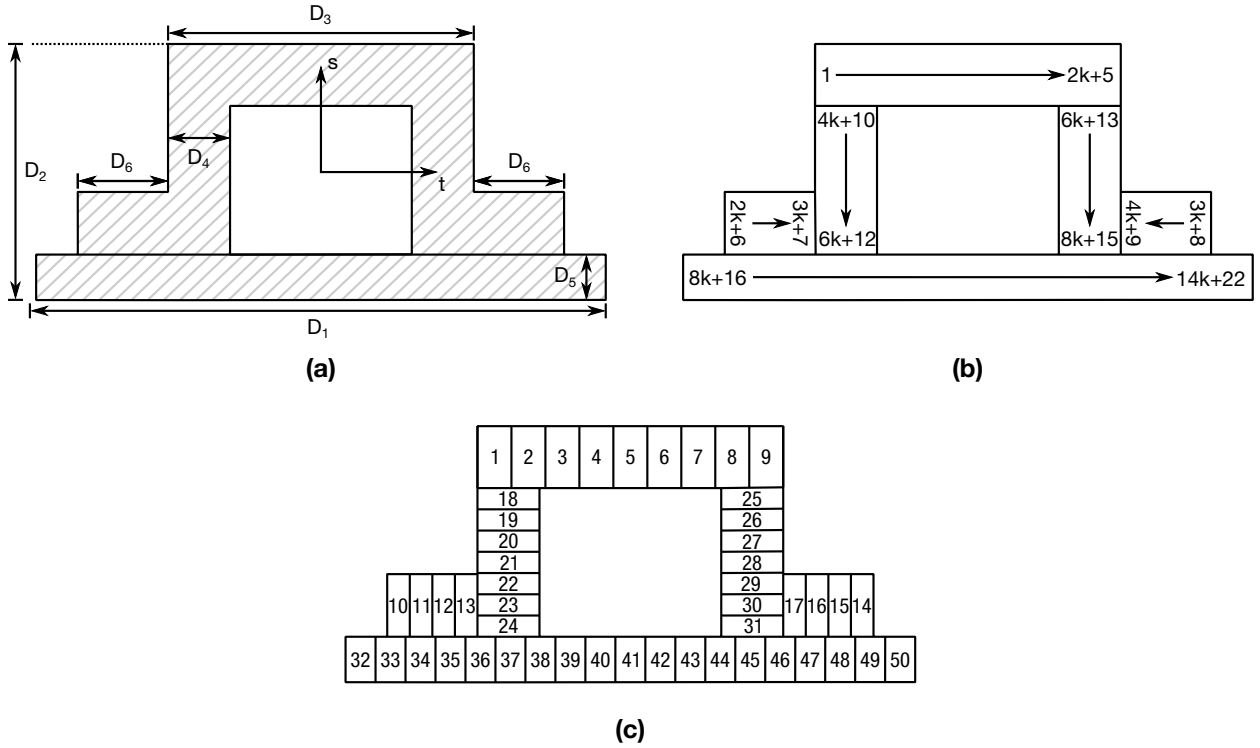


Figure 29-24. Type 22: Hat-Shape 2. (a) Cross section geometry. (b) Integration point numbering. (c) Example for k = 2.

*INTEGRATION_SHELL

Purpose: Define user defined through the thickness integration rules for the shell element. This option applies to three-dimensional shell elements with three or four nodes (*SECTION_SHELL types 1-11, ±16, 17, 18, 20, 21, 23-27, 30, 41, 42, and 101-105), to eight node thick shell types 1, 2, and 6 (*SECTION_TSHELL), and to all IGA shell types (*SECTION_IGA_SHELL). See *PART_COMPOSITE for a simpler alternative to *PART + *SECTION_SHELL + *INTEGRATION_SHELL.

Card 1	1	2	3	4	5	6	7	8
Variable	IRID	NIP	ESOP	FAILOPT				
Type	I	I	I	I				

Define NIP cards below if ESOP = 0.

Card 2	1	2	3	4	5	6	7	8
Variable	S	WF	PID					
Type	F	F	I					

VARIABLE	DESCRIPTION
IRID	Integration rule ID (IRID refers to IRID on *SECTION_SHELL card).
NIP	Number of integration points
ESOP	Equal spacing of integration points option: EQ.0: Integration points are defined below, EQ.1: Integration points are equally spaced through thickness such that the shell is subdivided into NIP layers of equal thickness.
FAILOPT	Treatment of failure when mixing different constitutive types, which do and do not include failure models, through the shell thickness. For example, consider the case where a linear viscoelastic material model, which does not have a failure option, is mixed with a composite model, which does have a failure option. Note: If the failure option includes failure based on the time step size of

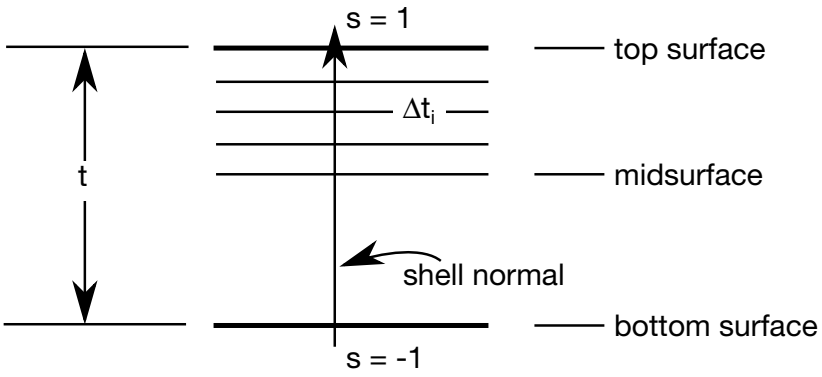


Figure 29-25. For the user defined shell integration rule the ordering of the integration points is arbitrary.

VARIABLE	DESCRIPTION
	the element, element deletion will occur regardless of the value of FAILOPT. EQ.0: Element is deleted when the layers which include failure, fail. EQ.1: Element failure cannot occur since some layers do not have a failure option.
S	Coordinate of integration point in range -1 to 1.
WF	Weighting factor. This is typically the thickness associated with the integration point divided by actual shell thickness, that is, the weighting factor for the i^{th} integration point = $\Delta t_i/t$ as seen in Figure 29-25.
PID	Optional part ID if different from the PID specified on the element card. The average mass density for the shell element is based on a weighted average of the density of each layer that is used through the thickness. When modifying the constitutive constants through the thickness, it is often necessary to defined unique part IDs without elements that are referenced only by the user integration rule. These additional part IDs only provide a density and constitutive constants with local material axes (if used) and orientation angles taken from the PID referenced on the element card. In defining a PID for an integration point, it is okay to reference a solid element PID. The material type through the thickness can vary.

*INTERFACE

Interface definitions may be used to define surfaces, nodal lines, and nodal points for which the time histories of nodal coordinates and if applicable (*INTERFACE_LINKING_NODE/EDGE) and available (beams, shells), nodal rotations, are saved at some user specified frequency. This data may then be used in subsequent analyses as an interface ID in the *INTERFACE_LINKING_DISCRETE_NODE as constraining nodes, in *INTERFACE_LINKING_SEGMENT as constraining segments and in *INTERFACE_LINKING_EDGE as the constraining edge for a series of nodes. This capability is especially useful for studying the detailed response of a small member in a large structure.

For the first analysis, the member of interest need only be discretized sufficiently that the displacements and velocities on its boundaries are reasonably accurate. After the first analysis is completed, the member can be finely discretized in the region bounded by the interfaces. Finally, the second analysis is performed to obtain highly detailed information in the local region of interest. When beginning the first analysis, specify a name for the interface segment file using the Z = parameter on the LS-DYNA execution line. When starting the second analysis, the name of the interface segment file created in the first run should be specified using the L = parameter on the LS-DYNA command line. Following the above procedure, multiple levels of sub-modeling are easily accommodated. The interface file may contain a multitude of interface definitions so that a single run of a full model can provide enough interface data for many component analyses. The interface feature represents a powerful extension of LS-DYNA's analysis capabilities. The keyword cards for this purpose are:

*INTERFACE_COMPENSATION_3D

*INTERFACE_COMPENSATION_TRIMMING_CURVE

*INTERFACE_COMPONENT_FILE

INTERFACE_COMPONENT_*OPTION

INTERFACE_LINKING_DISCRETE_NODE_*OPTION

*INTERFACE_LINKING_EDGE

*INTERFACE_LINKING_FILE

*INTERFACE_LINKING_FORCES

*INTERFACE_LINKING_NODE

*INTERFACE_LINKING_SEGMENT

INTERFACE_SPRINGBACK_*OPTION1_OPTION2

Interface definitions may also be employed to define soil-structure interfaces in earthquake analysis involving non-linear soil-structure interaction where the structure may be non-linear, but the soil outside the soil-structure interface is assumed to be linear. Free-field earthquake ground motions are required only at the soil-structure interface for such analysis. The keyword cards for this purpose are:

*INTERFACE_SSI

*INTERFACE_SSI_AUX

*INTERFACE_SSI_AUX_EMBEDDED

*INTERFACE_SSI_STATIC

As of R13, two interface keywords are available for performing a two-stage SPG analysis. The first keyword is for specifying that the current analysis is the first stage while the second keyword is for specifying the analysis is the second stage. These keywords are:

*INTERFACE_SPG_1

*INTERFACE_SPG_2

*INTERFACE_ACOUSTIC

Purpose: Create an interface file of boundary motions that can be used by *BOUNDARY_ACOUSTIC_INTERFACE for weakly coupled acoustic fluid-structure interaction in SSD analyses (see *CONTROL_IMPLICIT_SSD_DIRECT).

Card 1	1	2	3	4	5	6	7	8
Variable	IFID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
IFID	Segment set ID of structural faces for this interface. This structural IFID can be used by *BOUNDARY_ACOUSTIC_INTERFACE.

Remarks:

1. **Acoustic Interface Filename.** The motions of the structural segments will be written to the LSDB database "baifac.db" for every SSD solution step invoked with *CONTROL_IMPLICIT_SSD_DIRECT. Multiple interfaces from a single analysis may be saved in one interface file if they reference different IFID.

***INTERFACE_BLANKSIZE**_*OPTION*

Available options include:

DEVELOPMENT

INITIAL_TRIM

INITIAL_ADAPTIVE

SCALE_FACTOR

SYMMETRIC_PLANE

Purpose: This keyword causes LS-DYNA to run a blank-size development calculation instead of a finite element calculation. The input for this feature consists of (1) the result of a completed metal forming simulation, (2) the corresponding initial blank, and (3) the desired result from the simulation in the form of a boundary curve or full mesh. From these inputs, the *INTERFACE_BLANKSIZE method adjusts the initial blank so that the resulting formed piece more closely matches the target. Iterating may systematically improve the blank's starting geometry. This feature is available for both SMP and MPP but requires a double precision executable. A GUI for using this available in LS-PrePost as of version 4.2 under *APPLICATION → Metal Forming → Blank Size/Trim line*.

> **NOTE:** When this card is present, LS-DYNA *does not* proceed to the finite element simulation.

This keyword requires one or all three keyword options in the input deck, each corresponding to a different kind of forming operation:

1. **DEVELOPMENT.** This option takes as its target either a full mesh or a minimum boundary. It adjusts the blank so that the product more closely approximates the target. The computed blank-boundary is written to a file called trim-curves.ibo, which contains a *DEFINE_CURVE_TRIM_3D keyword.

2. **INITIAL_TRIM.** This option adjusts the blank so that trimming and mesh-refinement are mapped back onto the initial blank.

3. **INITIAL_ADAPTIVE.** This option reads in (1) the input mesh for a flanging simulation as well as (2) the adapted mesh calculated during flanging simulation. It maps the refinement back to the initial blank.

Card Set for *INTERFACE_BLANKSIZE_DEVELOPMENT.

Development Parameter Card.

Card 1	1	2	3	4	5	6	7	8
Variable	IOPTION		IADAPT	MAXSIZE	REF	SPACE	MAXGAP	ORIENT
Type	I		I	F	I	F	F	I
Default	-2		1	30.0	0	2.0	30.0	none

Target Card. See "Target curves (target.xyz)" in Figure 30-1.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME1							
Type	A80							

Final blank Card. See "Final blank (final.k)" in Figure 30-1.

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME2							
Type	A80							

Initial blank Card. See "Initial blank (initial.k)" in Figure 30-1.

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME3							
Type	A80							

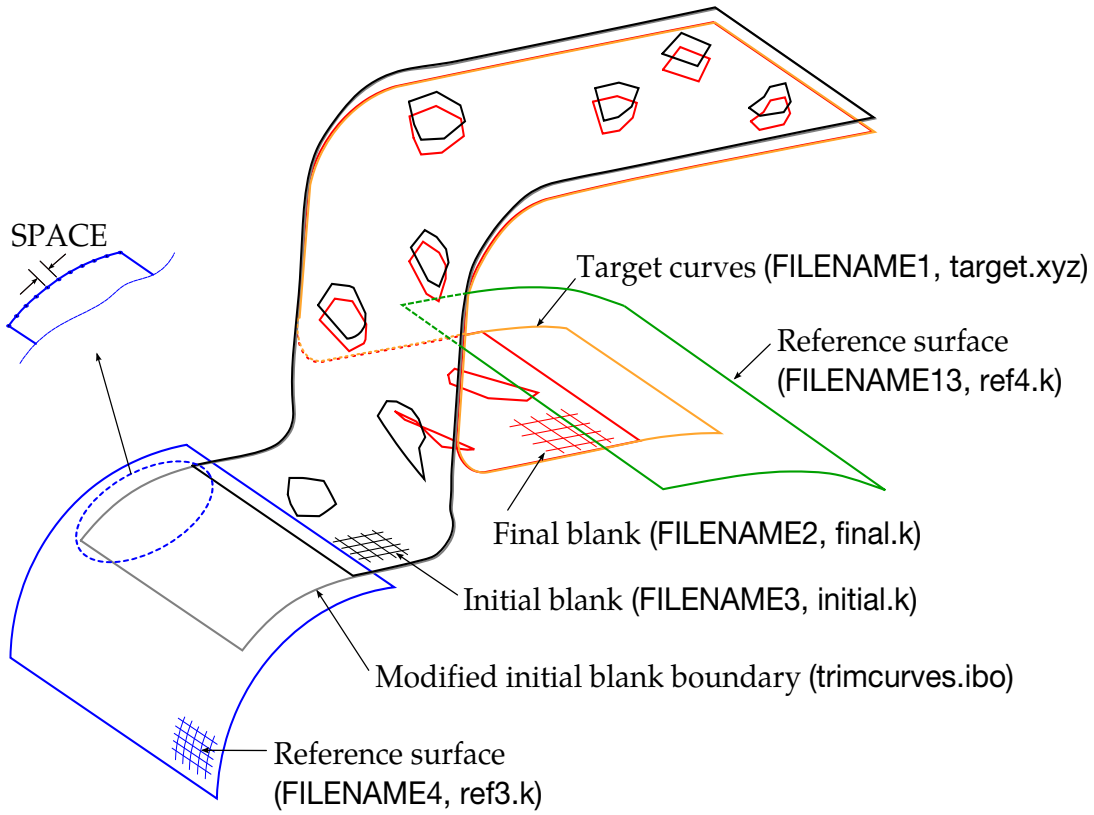


Figure 30-1. Trim curve development using a reference surface. See Example II.

Reference Surface Card. See "Reference surface (ref3.k)" in Figure 30-1and Example II.

Card 5	1	2	3	4	5	6	7	8
Variable	FILENAME4							
Type	A80							

Reference Surface Card. See "Reference surface (ref4.k)" in Figure 30-1 and Example II.

Card 6	1	2	3	4	5	6	7	8
Variable	FILENAME13							
Type	A80							

VARIABLE	DESCRIPTION
IOPTION	Target curve definition input type:

VARIABLE	DESCRIPTION
	EQ.1: (Entire) blank mesh in keyword format.
	EQ.2: Consecutive position coordinates of blank boundary loop curve in XYZ format, defined by *DEFINE_TARGET_BOUNDARY. The blank mesh's normal vector and the closed boundary curve are consistently oriented according to the right-hand rule, see Figure 30-2. Note the target boundary curves must have enough points for a successful prediction of the initial blank size.
	EQ.-2: Consecutive position coordinates of blank boundary loop curve in XYZ format, defined by *DEFINE_TARGET_BOUNDARY. The blank mesh's normal vector and the closed boundary curve are consistently oriented according to the left-hand rule, see Figure 30-2. Note the target boundary curves must have enough points for a successful prediction of the initial blank size.
	NOTE: Starting in August 2015, the curve direction is desensitized, meaning *both* IOPTION = 2 and IOPTION = -2 give the same results.
	In LS-PrePost 4.0,use menu option *GeoTol* → *ID Measure* to show the flow direction of a boundary curve. To reverse the curve direction, use *Curve* → *Rever(se)*.
IADAPT	Adaptive mesh control flag. If IADAPT = 1, the number of elements in the meshes initial (FILENAME3) and simulated blank (FILENAME2) can be different, so it is not necessary to use the sheet blank from the file adapt.msh (created by setting IOFLAG = 1 in *CONTROL_ADAPTIVE) for the initial blank mesh.
MAXSIZE	The maximum change in initial blank size in each iteration. It is used to limit the blank size change in each iteration during mapping to avoid convergence problems when the initial blank is curved.
REF	Flag to indicate trim curve projection to a reference surface (mesh); see Figure 30-1:

VARIABLE	DESCRIPTION
	EQ.0: No projection. EQ.1: The trim curves will be projected to the reference surface. In addition, the mesh file for the reference surface is given in FILENAME4.
SPACE	Point spacing distance on the reference surface for the projected curve; see Figure 30-1. If the gap between two neighboring points in the modified trimming curve is larger than this value, extra nodes will be added in the final blank and projected to the reference surface. A smaller value should be used for large reference surface curvature.
MAXGAP	When REF is set to 1, the nodes from the final blank will be projected to the reference surface. However, if the distance between the nodes and the surface is larger than MAXGAP, the nodes will not be projected.
ORIENT	A flag to control iterative optimization efficiency or to include a reference surface file for the final formed blank. EQ.1: Activates a new algorithm to potentially reduce the number of iterations during the iterative blank size optimization loop. This activation should be used in conjunction with the keyword option SCALE_FACTOR (0.75~0.9). EQ.2: To include a reference surface FILENAME13 (in keyword mesh format) for the simulated (formed) blank. This is useful when the formed blank is smaller than intended and extended surfaces are not flat. See Example II and Figure 30-1.
FILENAME1	Target input file name. When a blank mesh is used (IOPTION = 1), the keyword file must contain *NODE and *ELEMENT_SHELL keywords. When using the blank boundary curve for target (|IOPTION| = 2), the file must consist of *DEFINE_TARGET_BOUNDARY. See the Target Boundaries (and IGES) in the Remarks section.
FILENAME2	Simulated (formed or flanged) sheet blank mesh in keyword format. This mesh can be obtained from the final state of any downstream process simulation.

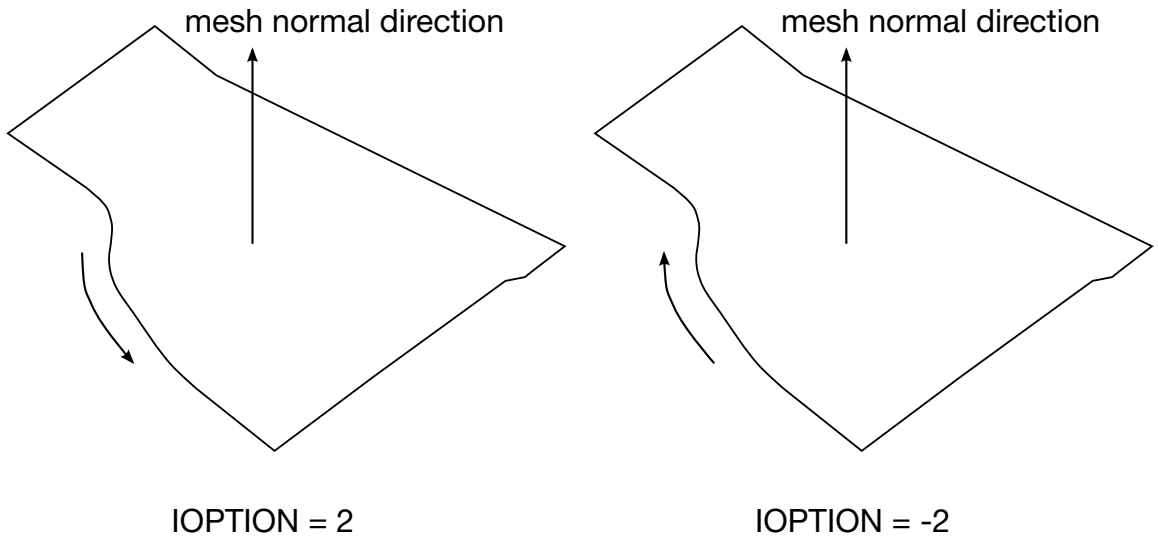


Figure 30-2. Differences between IOPTION 2 and -2.

VARIABLE	DESCRIPTION
FILENAME3	Initial sheet blank mesh in keyword format. This can be the first state mesh from any process simulation prior to FILENAME2 simulation. Set IADAPT = 1 if adaptive refinement is used in any simulation.
FILENAME4	Reference surface onto which adjustments to the blank's trim curves in its initial state are projected (ref3.k in Figure 30-1). This surface is typically a curved extension of the initial blank and must be defined as a mesh in keyword format. This file name must be defined when REF is set to 1. Also see extended reference surface for the final state (formed state), FILENAME13.
FILENAME13	Reference surface onto which adjustments to the blank's trim curves in its final state are projected (ref4.k in Figure 30-1). This surface is typically a curved extension of the formed blank and must be defined as mesh in keyword format. This file name must be defined when ORIENT is set to 2.

Card Set for *INTERFACE_BLANKSIZE_INITIAL_TRIM.

Initial Flat blank. FILENAME5 should specify the mesh of a blank that has been refined during a subsequent forming operation. FILENAME5 is usually set to the adapt.msh output. See "OP10 Initial Adapted blank mesh" in Figure 30-3.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME5							
Type	A80							

Formed blank. FILENAME6 specifies a dynain file from a forming simulation. See "OP10 Final blank" in Figure 30-3.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME6							
Type	A80							

Trimmed Formed blank. A dynain file from a trimming simulation that started with the state given in FILENAME6. See "OP20 Final blank" in Figure 30-3.

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME7							
Type	A80							

Trimmed Flat blank (output). This field specifies the name for the file to which the trimmed flat blank is written. See "OP20 Flat New" in Figure 30-3.

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME8							
Type	A80							

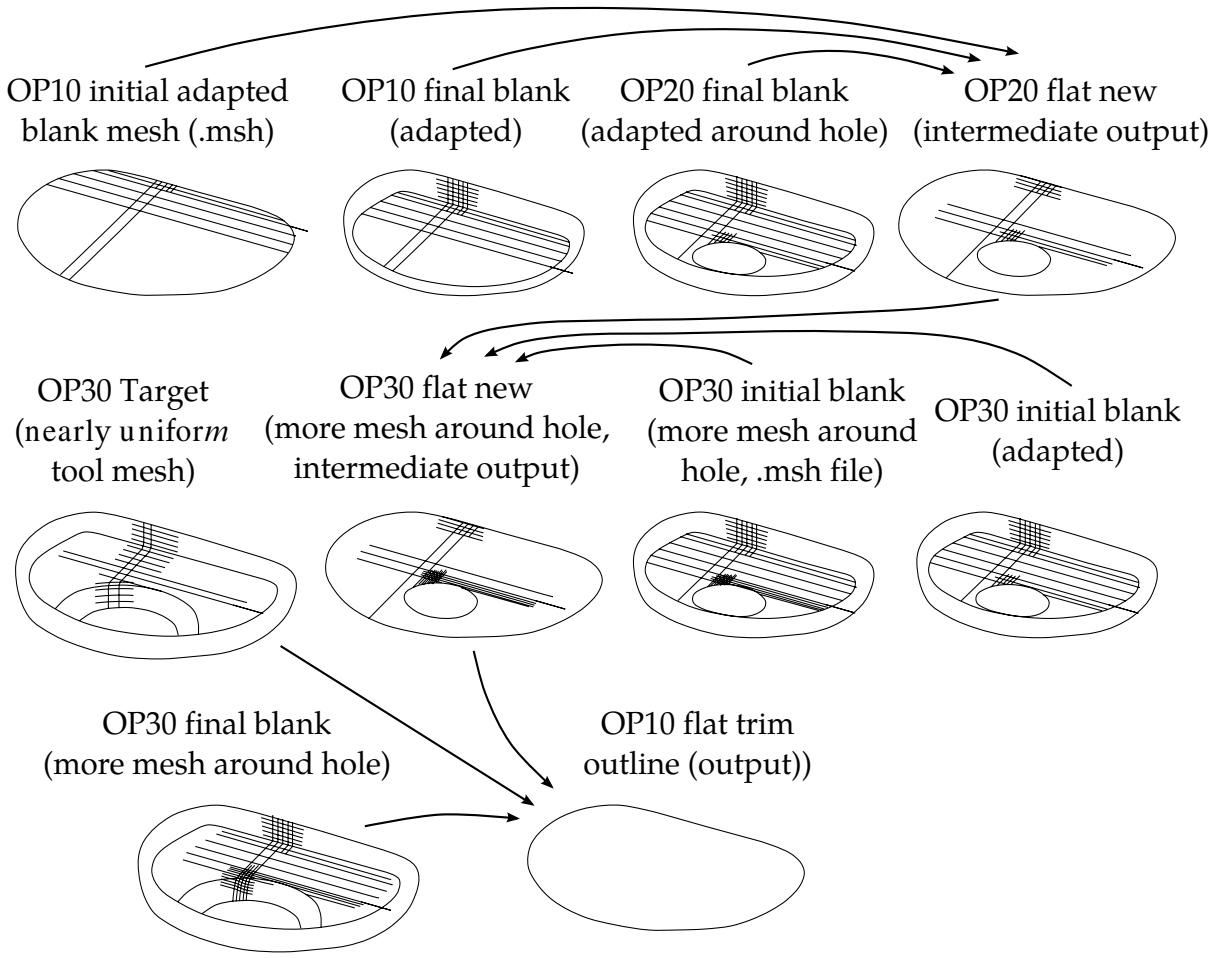


Figure 30-3. Inputs and outputs (*Courtesy of Metal Forming Analysis Corp.*). For exposition of labels OP10, OP20, and OP30 see Figure 30-4.

VARIABLE	DESCRIPTION
FILENAME5	Initial blank in its flat configuration *with* adapted mesh in keyword format. FILENAME5 usually points to an adapt.msh file. For example see OP10 in Figures 30-4, 30-22 and 30-3, in which FILENAME5 is the adapt.msh from a draw-forming calculation.
FILENAME6	Final formed blank in keyword format. This is usually the dynain file corresponding to the adapt.msh file mentioned above for FILENAME5; see Figures 30-4, 30-22 and 30-3.
FILENAME7	A trimmed blank in keyword format. This file should be derived from FILENAME6 as the dynain file from a trimming simulation. See OP20 in Figures 30-4, 30-22 and 30-3.
FILENAME8	This field specifies the name for the file in which the trimmed flat blank is to be written. See OP20 in Figures 30-4, 30-22 and 30-3.

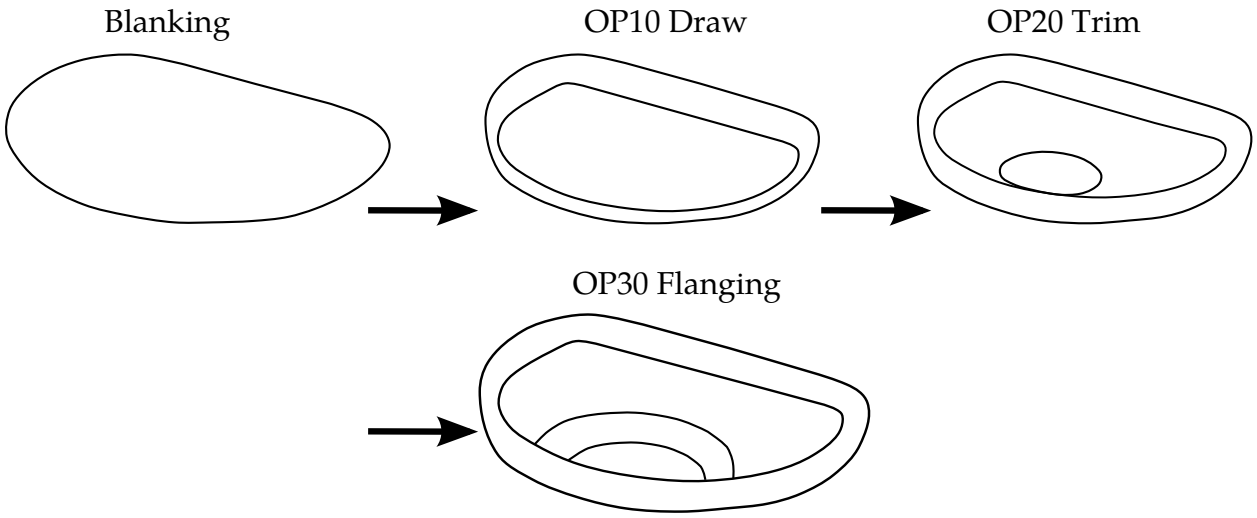


Figure 30-4. A stamping process consists of draw, trim and flanging (*Courtesy of Metal Forming Analysis Corporation*). The labels OP10, OP20, and OP30 are used *extensively* in the ensuing discussion.

Card Set for *INTERFACE_BLANKSIZE_INITIAL_ADAPTIVE.

Flat blank. FILENAME9 specifies the mesh of a blank in a flat configuration serving as the basis for a two-step metal forming process. The second-stage simulation produces an adapt.msh file, from which the refinements are to be mapped onto the flat-blank of FILENAME9. See, for example, "OP20 Flat New" in Figure 30-3 where FILENAME9 points to result, FILENAME8, of an INITIAL_TRIM calculation.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME9							
Type	A80							

Initial blank. FILENAME10 is the result from the first stage of a two-step process. See, for example, "OP30 Initial blank" in Figure 30-3 where FILENAME10 has been formed and trimmed and refined along the way.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME10							
Type	A80							

Adapted Initial blank. FILENAME11 contains a refined version of the mesh in FILENAME10. It is expect to name the adapt.msh file from some operation performed on the mesh of FILENAME10. See, for example, "OP30 Initial blank (with more mesh)" in Figure 30-3, where the adapt.msh file comes from a flanging simulation.

Card 3	1	2	3	4	5	6	7	8
Variable	FILENAME11							
Type	A80							

Refined Flat blank (output). This field specifies the name of the file to which the refined flat blank is written. The blank's mesh is refined to *exactly* match the forming process. See, for example, "OP30 Flat New" in Figure 30-3.

Card 4	1	2	3	4	5	6	7	8
Variable	FILENAME12							
Type	A80							

VARIABLE	DESCRIPTION
FILENAME9	FILENAME9 should point to the blank defined by FILENAME8, as in Figures 30-4, 30-22 and 30-3.
FILENAME10	Initial-stage result file name. This may be extracted from d3plot files using LS-PrePost or it may be generated by LS-DYNA as a dynain file. See, for example, "OP30 Initial Blank" in Figure 30-3 where FILENAME10 has been formed, trimmed and refined along the way. To obtain from d3plot file the necessary mesh in keyword format using LS-PrePost4.0 select *POST* → *OUTPUT* → *Dynain ASCII* and check the box for *Exclude strain and stress*.
FILENAME11	FILENAME11 contains a refined version of the mesh in FILENAME10. FILENAME11 can be obtained from adapt.msh file from the same operation performed on the mesh of FILENAME10. See, for example, "OP30 Initial Blank (with more mesh)" in Figure 30-3, where the adapt.msh file comes from a flanging calculation.
FILENAME12	This field specifies the name of the file to which the refined flat blank is written. The blank's mesh is refined to *exactly* match the forming process. See, for example, "OP30 Flat New" in Figure 30-3.

Card Set for *INTERFACE_BLANKSIZE_SCALE_FACTOR.

Scale Factor Card. Define one card for each curve. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	IDCRV	SF	OFFX	OFFY	OFFZ			
Type	I	F	F	F	F			
Default	1	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
IDCRV	Curve ID in the order of appearance as in FILENAME1 in the target card, as defined by *DEFINE_TARGET_BOUNDARY.
SF	Scale factor for the IDCRV defined above. It defines a fraction of the changes required for the predicted initial blank shape. For example, if SF is set to 0.0, the corresponding IDCRV will be excluded from the calculation (although the original initial curve still will be output); on the other hand, if SF is set to 1.0, full change will be applied to obtain the modified initial blank that reflects the forming process. A SF of 0.5 will apply 50% of the changes required to map the initial blank. This feature is especially important for inner holes that are small and hole boundary expansions are large, so the predicted initial hole can avoid "crisscross" situation. An example is provided in Scale Factor and Symmetric Plane.
OFFX, OFFY, OFFZ	Translational move of the target curve. This is useful when multiple target curves (e.g. holes) and formed curves are far away from each other. Input values of OFFX, OFFY and OFFZ helps establish one-to-one correspondence between each target curve and formed curve. OFFX.EQ.-10000.0: offset values are automatically calculated.

Card Set for *INTERFACE_BLANKSIZE_SYMMETRIC_PLANE.

Symmetric Plane Card.

Card 1	1	2	3	4	5	6	7	8
Variable	X0	Y0	Z0	V1	V2	V3		
Type	F	F	F	F	F	F		
Default	0.0	0.0	0.0	1.0	0.0	0.0		

VARIABLE	DESCRIPTION
X0, Y0, Z0	*x*, *y*, *z* coordinates of a point on the symmetric plane. See example in Scale Factor and Symmetric Plane.
V1, V2, V3	Vector components of the symmetric plane's normal. See example in Scale Factor and Symmetric Plane.

Remarks and Examples

Inverse Methods for Optimizing Blank Size and Trim Lines:

Finding the minimal practicable blank size and developing an optimal set of trim lines is an integral part of the die engineering process. This card, *INTERFACE_BLANKSIZE, is one of several features that have been developed to solve the inverse problem: that is, to calculate an initial blank or blank boundary that will yield a desired product based mostly on the target geometry of that final product.

1. **The One-Step Method.** The *CONTROL_FORMING_ONESTEP card is suitable for early blank-size estimates. It invokes the total-strain theory of plasticity, thereby bypassing the, as of yet, undetermined specific details of the forming process.

2. **Unflanging.** Once the product design and process plan are complete, die development begins with addendum and binder creation, followed by secondary tooling development. In this stage, *CONTROL_FORMING_UNFLANGING can be used to develop trim lines for the secondary tooling; final (or intermediate) desired flange shapes are unfolded onto the addendum or binder to obtain the corresponding flange shapes in its initial shape. It also implements failure criteria to arrive at suggested final flange curves, with strains and thickness output on the unfolded flanges.

3. **Interface Blanksize.** This card, *INTERFACE_BLANKSIZE, can be used to *accurately* determine the optimal initial blank. To do so it requires an initial configuration, the corresponding simulated configuration, and a desired target configuration. This method takes into account the entire metal-forming process.

   a) One application of this keyword is to map trim curves between dies to calculate the trim curves needed for all trim dies. An example of the application can be found in Figures 30-18 through 30-21.

	Compulational Cost	Accuracy	Information Required	Physical Process
Blanksize	Full simulation	Exact	Full Simulation	Any
Unflanging	Fast	Approximate	Process Geometry	Inverse Flanging
Onestep	Fast	Approximate	None (*Path independent*)	Any

Table 30-1. Comparison of inverse methods.

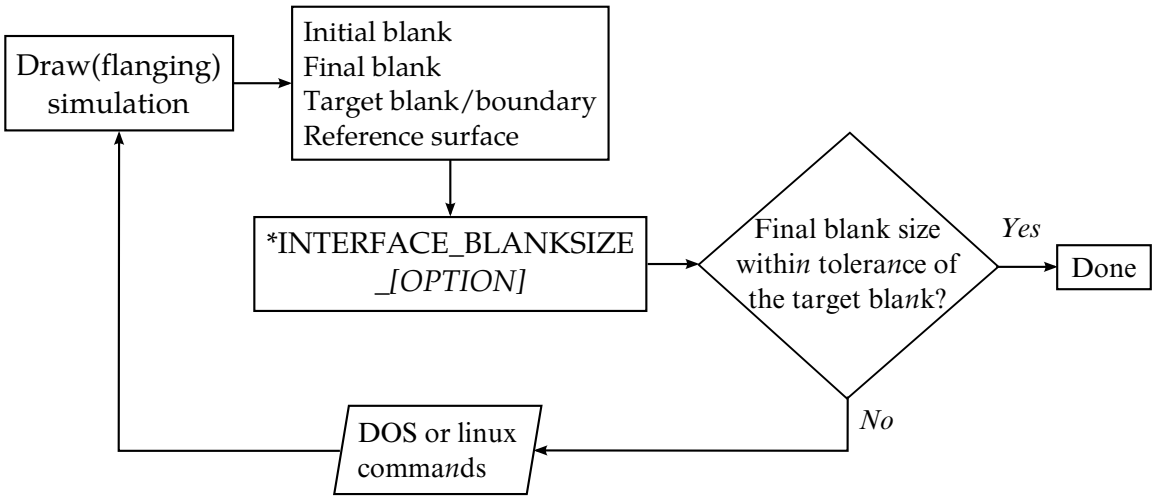


Figure 30-5. An iterative blank size development process.

b) The keyword can also be used to determine the precise minimal initial blank needed for a draw panel whose blank edge must be at specified distances from the edge of the draw beads.

Iterative Workflow:

The "interface blanksize" command produces an adjusted initial blank. It is not to be expected that the adjustment will be exact. However, this initial blank shape can be run through a second simulation to see if the final shape is close enough to the target blank. If it is not close enough, then the results from the second simulation can be used to repeat the process. Iterations can proceed until the final shape is within the range of the target shape. This iterative blank size development process is schematically presented in Figure 30-5 and exemplified in Figures 30-8 through 30-17.

Target Boundaries (and IGES):

When IOPTION = 2, or -2, a file with the keyword *DEFINE_TARGET_BOUNDARY must be present. This keyword can now be created from an IGES file using LS-PrePost 4.1 (or 4.2). To convert from IGES curves to keyword *DEFINE_TARGET_BOUNDARY in LS-PrePost 4.1, use menu option *Curve* → *Convert*→ *Method (To Keyword)* → *Select *DEFINE_TARGET_BOUNDARY*; pick the curves then hit *To Key*; write out the keyword file using *File* → *Save as* → *Save Keyword As*; and select *Output Version* as *V971_R7*. In LS-PrePost 4.2, select *Application* → *Metal Forming* → *Blanksize Trimline* to import the target curve directly in IGES format, the initial and final blank mesh and then write out a complete LS-DYNA input deck.

In addition, the target curves should be projected onto the final blank mesh if they do not exactly lie on the mesh surface. This can be done with LS-PrePost 4.1 using the menu

option *GeoTol* → *Project* → *Project*, select *Closest Projection*, select *Project to Elements*, then define the destination mesh and source curves, and hit *Apply*.

Computed Initial Blank Boundaries (and IGES):

Computed boundary curves are written with *DEFINE_CURVE_TRIM_3D keyword into a file called trimcurves.ibo. The format of this file follows the keyword's specification. LS-PrePost4.0 can convert the computed curve to IGES. See Figure 30-10. After hitting *Apply*, the curves will show up in the graphics window, and *File* → *Save as* → *Save Geom as* can be used to write the curves out in IGES format. In the LS-PrePost 4.2 GUI, under *Results*, trimcurves.ibo can be directly imported into the graphics window for viewing and to save in either STEP or IGES format.

To convert IGES to the *DEFINE_CURVE_TRIM_3D keyword format, import the IGES file into LS-PrePost4.0 and follow the procedures shown in Figure 30-11. After finishing step 2, "*curves have been converted to keyword format*" will be reported in the command prompt. Then use *File* → *Save* → *Save keyword* to write out the keyword file.

Support for Multi-Stage Processes with the Development Option:

Original Implementation.

Prior to Revision 88708 the development option required that the final blank (FILENAME2) differ from the initial blank (FILENAME3) by no more than a deformation and mesh refinement. In practice, this means that the two meshes must come from the same process simulation. For example, in a draw, trim and flanging process, the trimmed panel mesh is used for flanging simulation. Therefore, with the original implementation, LS-DYNA required that the initial blank state (FILENAME3) be trimmed when the final blank state (FILENAME2) is flanged on trimmed panel. Failure to observe this limitation may result in error termination.

Enhanced Implementation.

A more recent improvement to the blank size development (Revision 88708) removes the requirement that initial (FILENAME3) and final (FILENAME2) blanks must be from the same process simulations. The initial and final blank states *may* differ by a trimming process. This allows trimming and other process such as flanging to occur between the initial and final blanks, without the need of invoking the INITIAL_TRIM and INITIAL_-ADAPTIVE options. For example, the initial blank can be the blank mesh from "Blanking" in Figure 30-4, and the final blank can be the blank mesh from "Flanging," which is also in Figure 30-4.

Scale Factor and Symmetric Plane:

An example of using various scale factors ranging from 0.0 to 1.0 on a model involving an initial hole shape is shown in Figure 30-25. The target curve is given in targetline.k. All nodes along the symmetric plane are constrained by the SYMMETRIC_PLANE option. The symmetric plane is defined going through point coordinates $(-76, 2.63844, 0.38)$ with plane normal vector of $(1.0, 0.0, 0.0)$. A complete input is provided below:

```
*KEYWORD
*INTERFACE_BLANKSIZE_DEVELOPMENT
$  IOPTION              IADAPT
        -2                   1
$ target boundary curves
targetline.k
$ final formed mesh
final.k
$ initial mesh
initial.k
*INTERFACE_BLANKSIZE_SCALE_FACTOR
$    IDCRV        SF
         1       0.2
*INTERFACE_BLANKSIZE_SYMMETRIC_PLANE
$       X0         Y0         Z0         V1         V2         V3
       -76   2.63844       0.38        1.0        0.0        0.0
*END
```

If targetline.k consists of multiple curves, the following format can be used:

```
*INTERFACE_BLANKSIZE_SCALE_FACTOR
$    IDCRV        SF
         1       0.2
         2       0.8
         3       1.0
         ⋮         ⋮
```

Example I: Simple Example of the Development Option

Given the initial and final blank configuration and a target, this option calculates a new initial blank outline, corresponding to the target final blank boundary. In this example note that IADAPT = 1, meaning initial and final blank meshes may differ by an adaptively operation. The input and output files are detailed below, and output results are shown in Figure 30-6.

```
*KEYWORD
*INTERFACE_BLANKSIZE_DEVELOPMENT
$  IOPTION              IADAPT
         2                   1
$  input file for target mesh, or target position coordinates
targetpoints.k
$  input file for formed mesh
final.k
$  input file for initial blank mesh
initial.k
*END
```

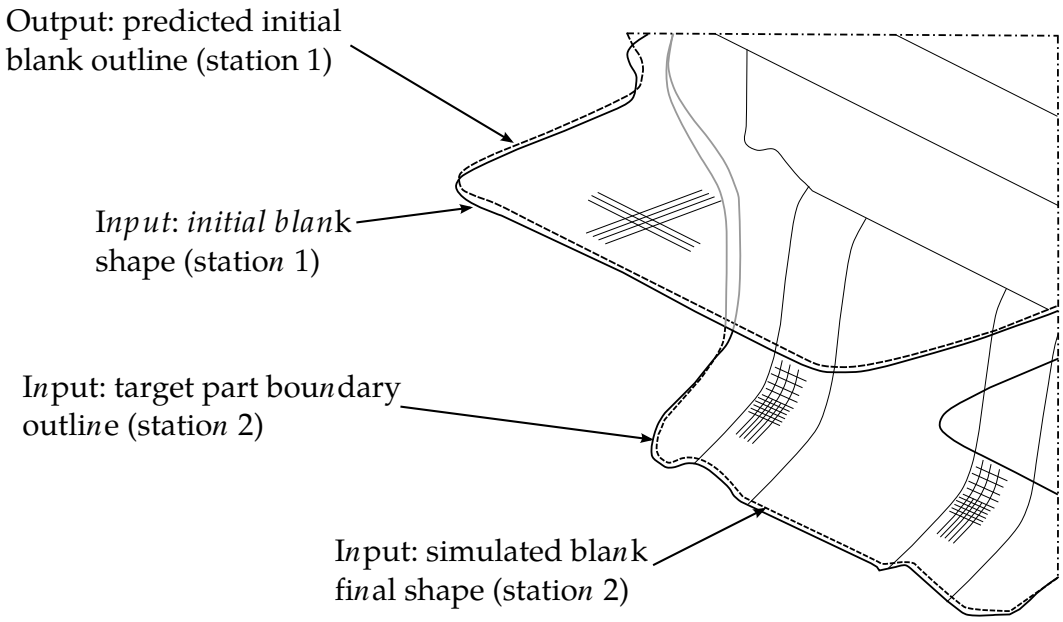


Figure 30-6. Blank size development in a progressive die with IADAPT = 1 in Example I.

The file, targetpoints.k partially shown below, was generated from IGES using *LS-Pre-Post 4.1*.

```
*KEYWORD
*DEFINE_TARGET_BOUNDARY
    -1.83355e+02    -5.94068e+02    -1.58639e+02
    -1.80736e+02    -5.94071e+02    -1.58196e+02
    -1.78126e+02    -5.94098e+02    -1.57813e+02
    -1.75546e+02    -5.94096e+02    -1.57433e+02
    -1.72888e+02    -5.94117e+02    -1.57026e+02
          ⋮               ⋮               ⋮
    -1.83355e+02    -5.94068e+02    -1.58639e+02
*END
```

The output is the modified initial blank outline in the file trimcurves.ibo.

Example II: The Reference Surface feature for the Development Option

For an initial blank that is not flat, the fields REF and FILENAME4 can be used to define a surface onto which changes in the boundary are needed. This is important when the adjusted boundary is not a simple tangential extension of the initial blank.

In a keyword example below, REF is set to "1" and the reference file for the extended initial shape is given as ref3.k. The maximum change between the initial and final blank size is set to be 20.0 mm per iteration. Point spacing distance (SPACE) of the calculated trim curve on the reference surface is set at 2.0 mm. Note that the inner holes and outer boundary curves are defined in target.xyz. The holes do not necessarily need to exist in the initial or final blank mesh. Also, since ORIENT is set to 2, a reference surface mesh

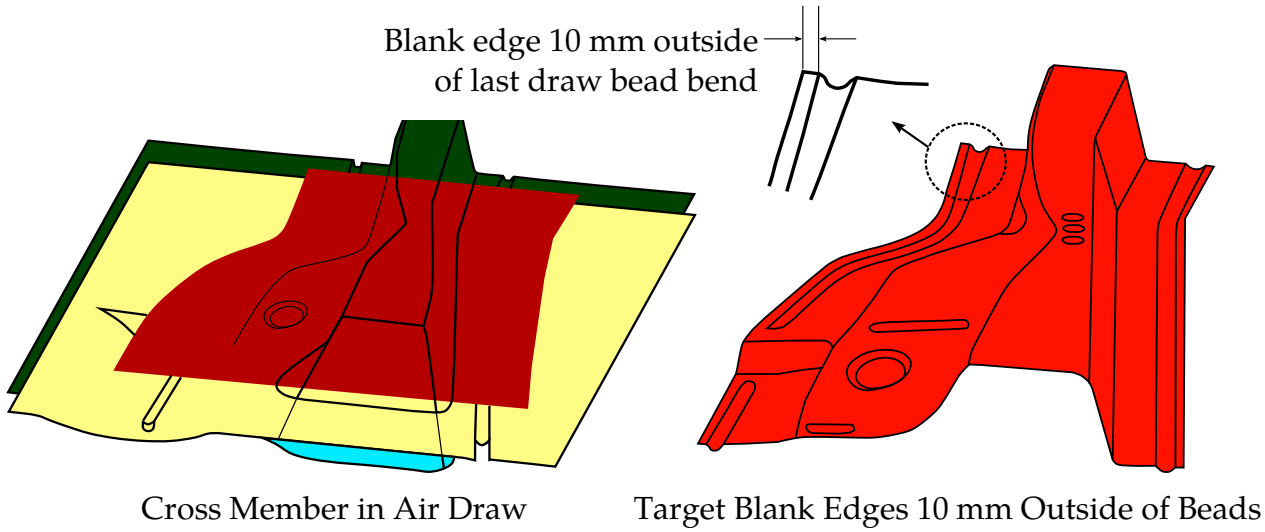


Figure 30-7. NUMISHEET 2005 cross member in Exmaple III.

file (ref4.k) is provided for the final (formed) state. The input details and output results are shown in Figure 30-1.

```
*KEYWORD
*INTERFACE_BLANKSIZE_DEVELOPMENT
$        1         2         3         4         5         6         7         8
$  IOPTION              IADAPT   MAXSIZE       REF     SPACE              ORIENT
        -2                   1    20.000         1       2.0                   2
$  input file for target mesh:
target.xyz
$  input file for formed mesh:
final.k
$  input file for initial blank mesh:
initial.k
$  reference file for extended initial shape:
ref3.k
$  reference file for extended final shape:
Ref4.k
*END
```

Example III: Development Feature Applied to a Draw Die with Physical Bead

In this example, which was created from NUMISHEET 2005, the DEVELOPMENT option has been used to design a blank such that, when formed, the edge is a specified distance outside of the last bend of a draw bead. In Figure 30-7, the tooling and blank set up is shown on the left. The right side of the figure shows the target blank, whose left and right edges everywhere are made 10 mm outside of the last bending radius of the draw beads. This setup is prototypical of one method to ensure a very stable and high-quality stamping process.

The first step towards setting up this analysis was to use *CONTROL_FORMING_ONESTEP to unfold the target blank and thereby obtain an initial guess of a flat blank, as shown to the left in Figure 30-12. The flat blank is then formed as one would usually do

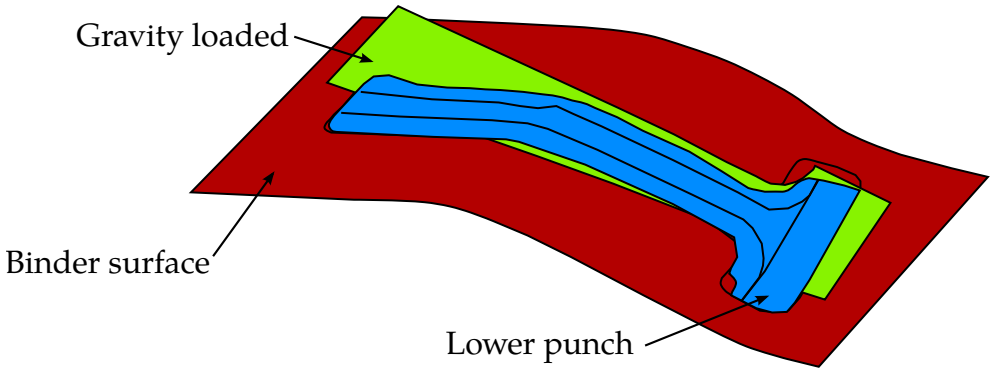


Figure 30-8. NUMISHEET 2008 B-pillar; Gravity loaded blank.

in a regular forming simulation as shown on the right side of the figure. The formed blank (Iteration 0) turns out to be larger than the target.

Next, the DEVELOPMENT is applied to generate a new and better initial blank that will lead to the target blank. The flat blank is used as input for the "initial blank mesh", the formed blank is used as input for the "simulated mesh", and the target blank mesh, or boundary points, is used to define the target.

In Figure 30-13 (left) the improved initial blank, called *the first compensated blank,* is superimposed onto the original one-step unfolded result. The one-step unfolded result is somewhat larger than the developed blank. When formed the improved blank (Iteration 1) nearly overlaps the target blank, shown in Figure 30-13 (right). If the final formed blank still deviates from the target, another iteration would ensue, until satisfactory results are obtained.

Example IV: Iterating with the Development Option

Because the NUMISHEET 2008 B-pillar model involves neither trimming nor flanging, it exmplifies the DEVELOPMENT option in its most direct use case. This model, illustrated in Figure 30-8, simulates a draw-die's action on a gravity-stressed flat blank. The B-pillar undergoes a stamping process including gravity, binder closing, and being drawn. In this example the DEVELOPMENT option is used to calculate the geometry for an initial blank that will exactly satisfy the design specification for a final formed panel. To highlight the efficiency of this feature, we start with an initial blank whose formed product deviates from the specification by a wide margin and then iterate using the DEVELOPMENT feature.

The target and the optimal initial blank are shown in Figure 30-14. The intial guess intentionally deviates from the optimal initial blank as shown in the left panel of Figure 30-15; the formed blank is compared with the target in the right panel.

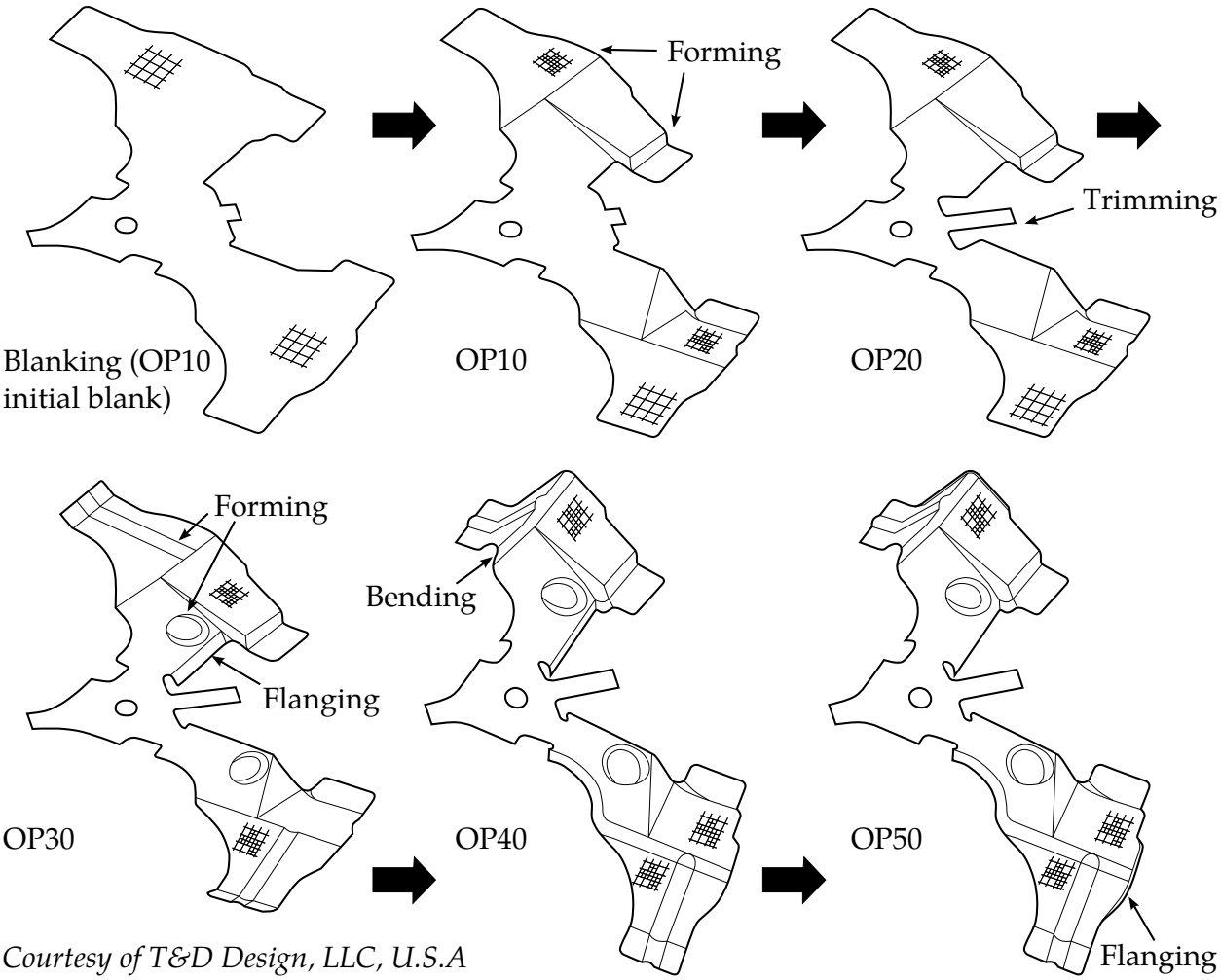


Figure 30-9. Enhanced DEVELOPMENT feature on a progressive die. Even though trimming occurs at OP20, the algorithm requires as input only OP10, OP50, and a target geometry.

In the first iteration a new initial blank is computed, and illustrated in the left panel of Figure 30-16, bearing the label *1st compensated blank*. The simulation is repeated using the first compensated blank, and in the right panel the result is compared to the target. The formed blank is narrower in the notched areas as compared with the target.

The second iteration is shown in the left panel of Figure 30-17 bearing the label, *2nd compensated blank*. Again, the simulation is repeated, but this time using the second compenated blank, and the result is compared to the target in the right panel of Figure 30-17. The resulting product is a good match to the target.

Because the initial blank intentionally deviated from its ideal shape by a large margin, this example requires two iterations to converge. Generally this processed is bootstrapped with the *CONTROL_FORMING_ONESTEP card, which calculates an initial guess by approximately unfolding the target shape.

Example V: Development Feature Applied to Flanging Process

In this example, which is schematically illustrated in Figure 30-18, the NUMISHEET 2002 fender outer is flanged along the hood line. The development feature adjusts the initial blank's boundary so that the formed piece matches the specified target flanged shape as shown in Figure 30-19. For demonstration purposes, the trimmed blank shape intentionally deviates from the optimal configuration by a large amount. This error is indicated in Figure 30-20 by the label *initial guess trim curves*. In Figure 30-20 the flanged product is shown to deviate substantially from the flanged target along the boundary. As shown in Figure 30-21 after one iteration the correct initial blank boundary is obtained.

Alternatively, *CONTROL_FORMING_UNFLANGING can be used to unfold the flanged target onto the addendum to obtain the initial blank size, or a starting guess for this process.

Example VI: Enhanced-Development Feature Applied to Progressive Die Process

In the example of Figure 30-9, *courtesy of T&D Design, LLC, U.S.A*, the blank-shape for a five stage progressive die process is calculated. Because this process involves a trimming step, the development capability prior to Revission 88708 does not support this example.

In this example, an initial blank at OP10, undergoes trimming, reforming, bending and flanging to arrive at the blank in OP50. In Figure 30-23 the computed product is compared with the specified target. A blank size development calculation produces the modified OP10 initial blank outline. The updated blank is used in a verification simulation. As seen in Figure 30-24, the blank size development feature produces a good result.

Trim lines are not optimized by the development feature, so trimming should only occur along the boundary of the target blank. The modified OP10 blank requires some refitting in the trimmed area.

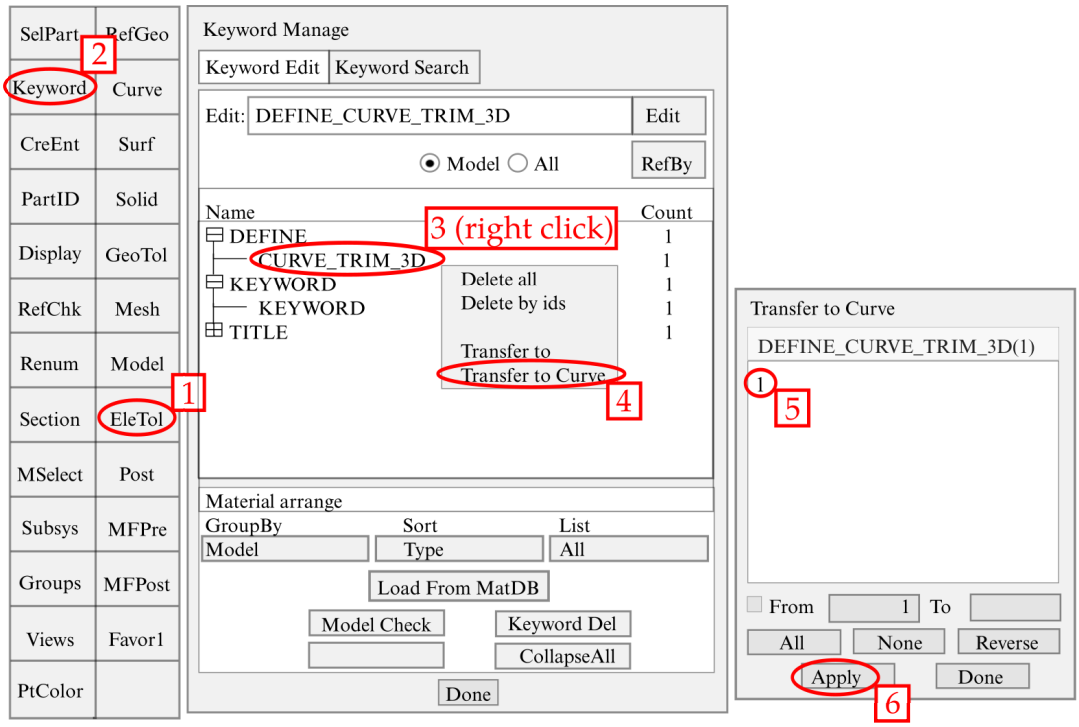


Figure 30-10. Converting trimcurves.ibo to IGES format in LSPP4.0.

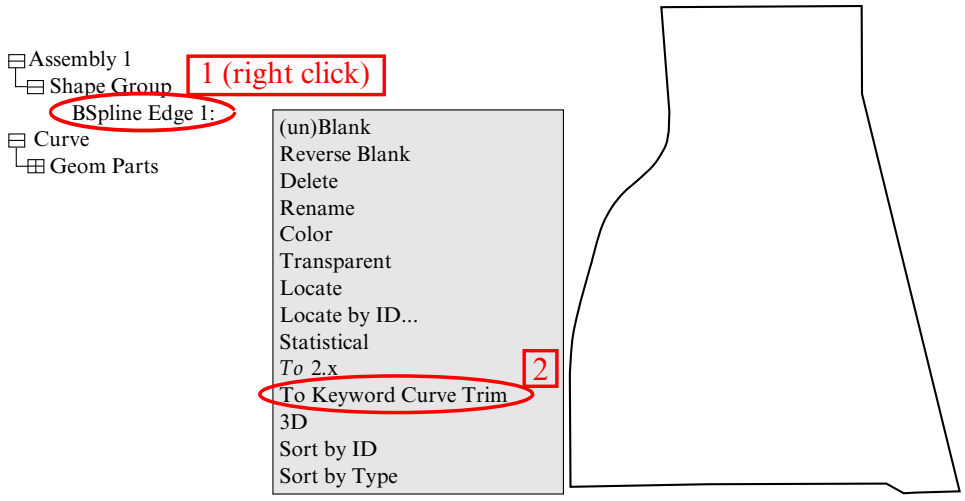


Figure 30-11. Converting IGES file to *DEFINE_CURVE_TRIM_3D.

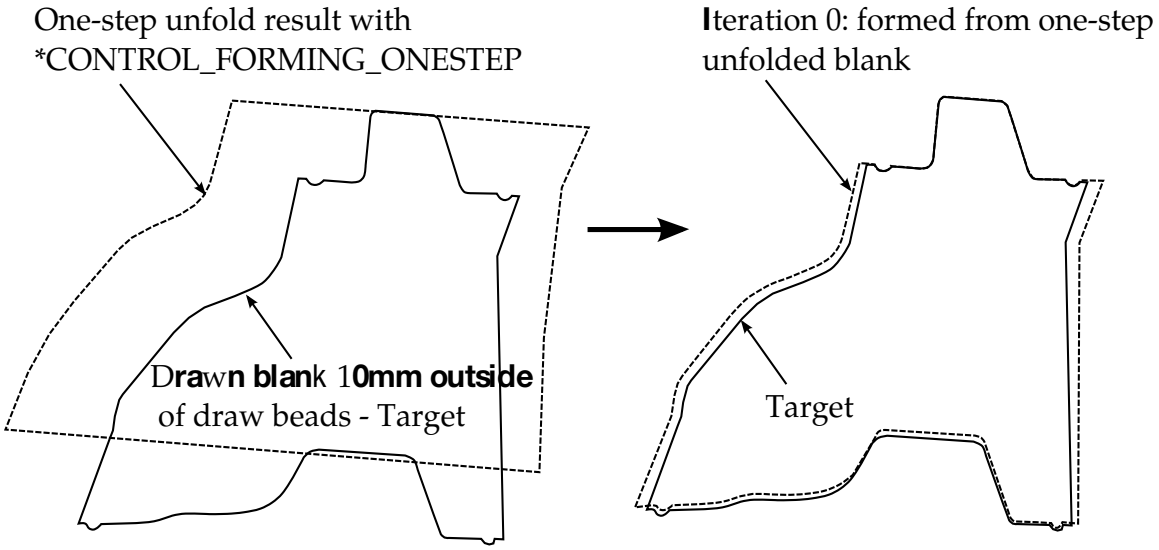


Figure 30-12. Initial blank calculation and baseline formed blank.

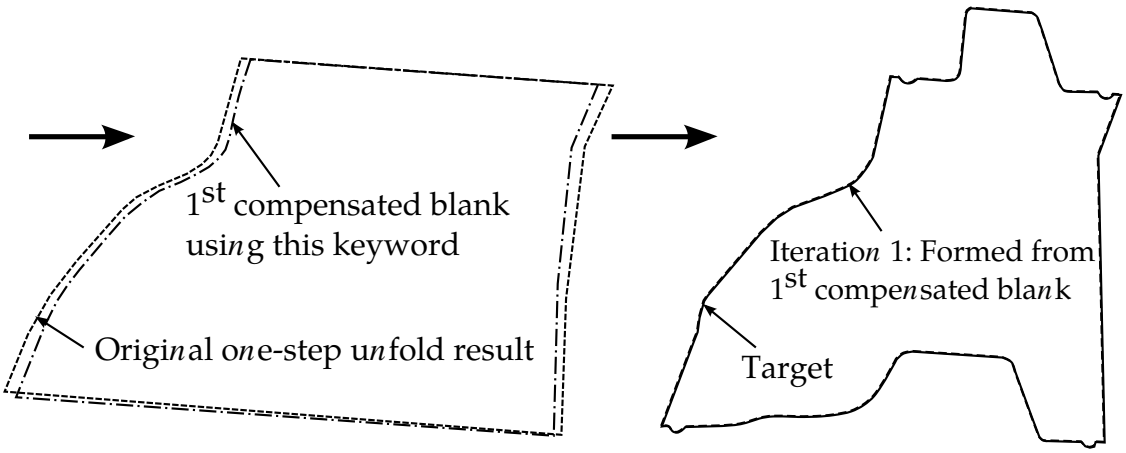


Figure 30-13. The first compensated blank and the final confirmation run.

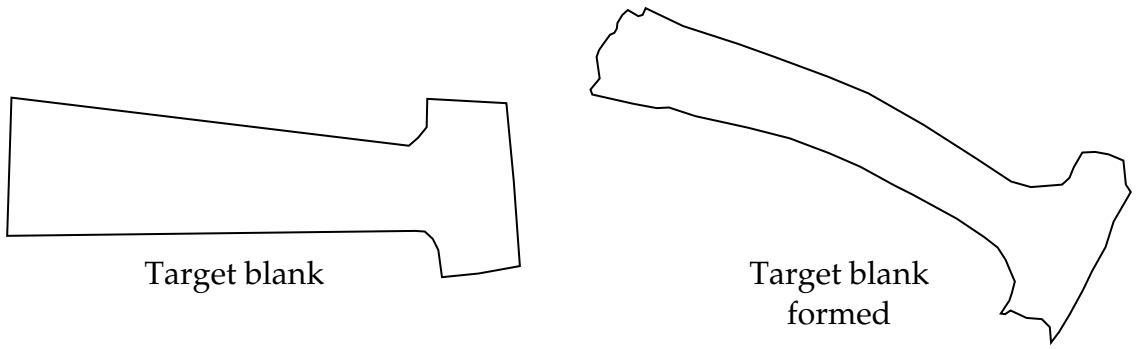


Figure 30-14. Assumed target blanks.

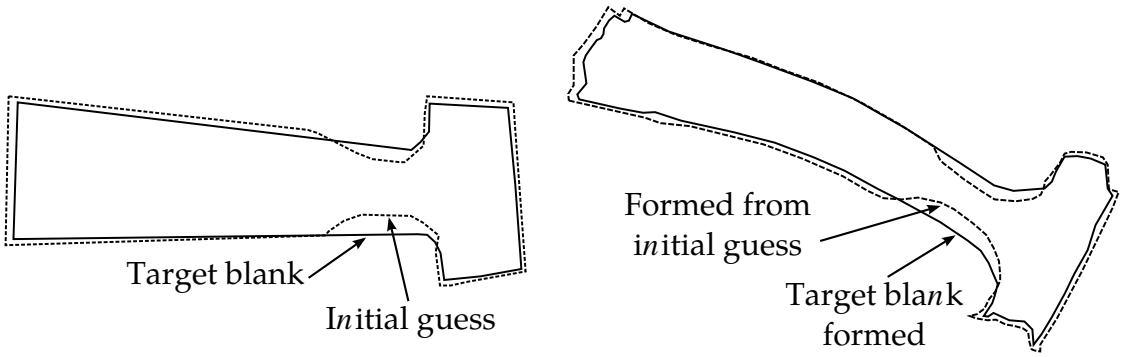


Figure 30-15. Iteration 0 results comparison with target.

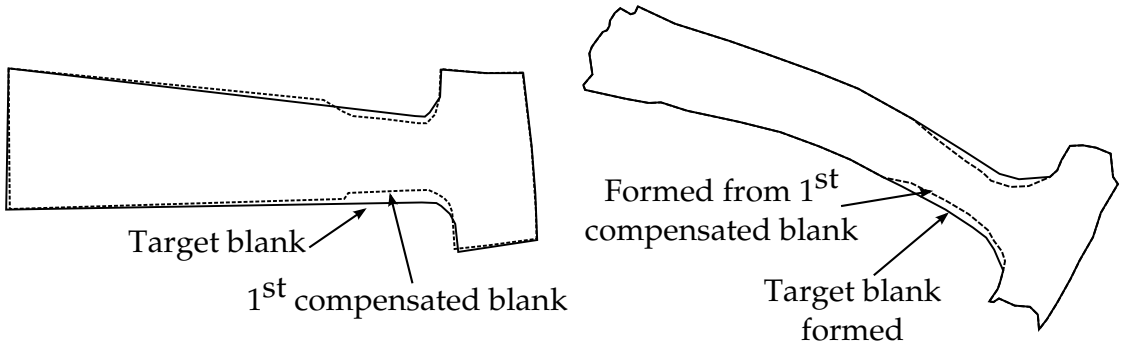


Figure 30-16. Iteration 1 results.

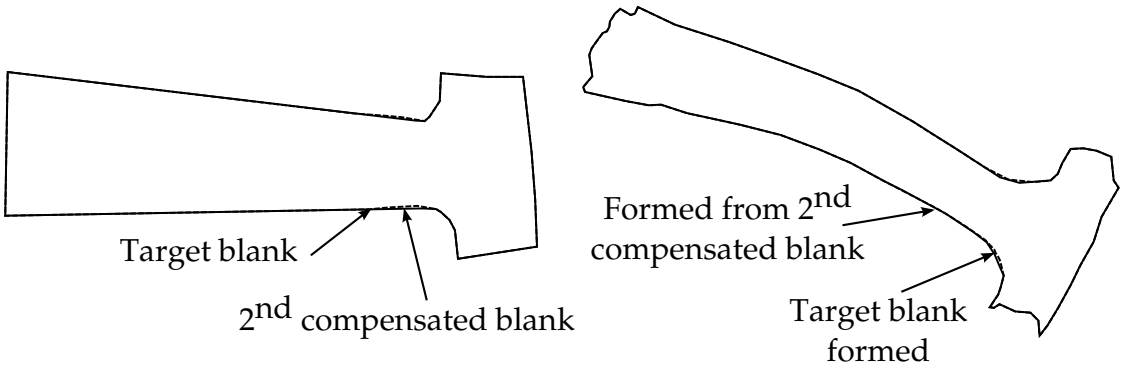


Figure 30-17. Iteration 2 results.

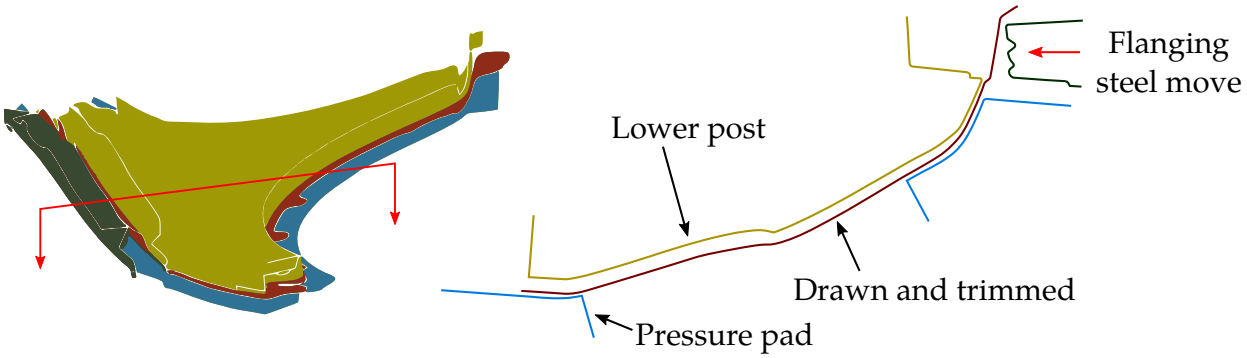


Figure 30-18. The flanging process on NUMISHEET 2002 fender outer.

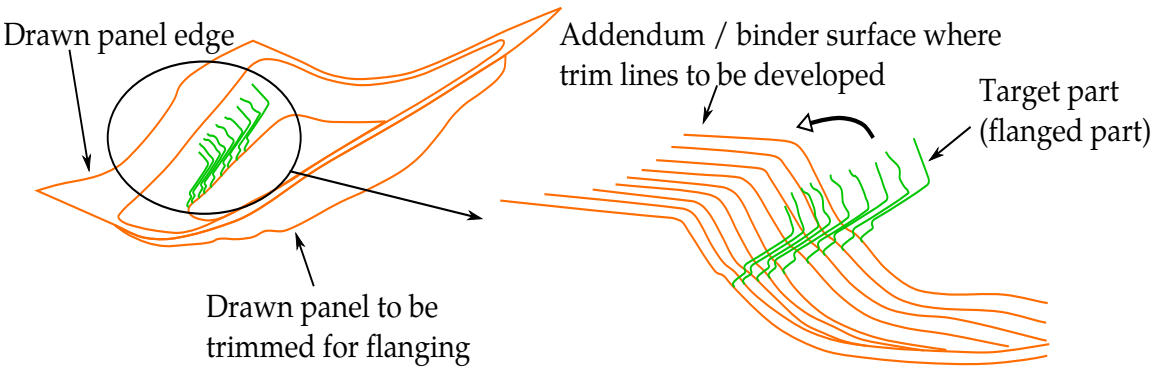


Figure 30-19. Multiple section view showing the target part and addendum surfaces.

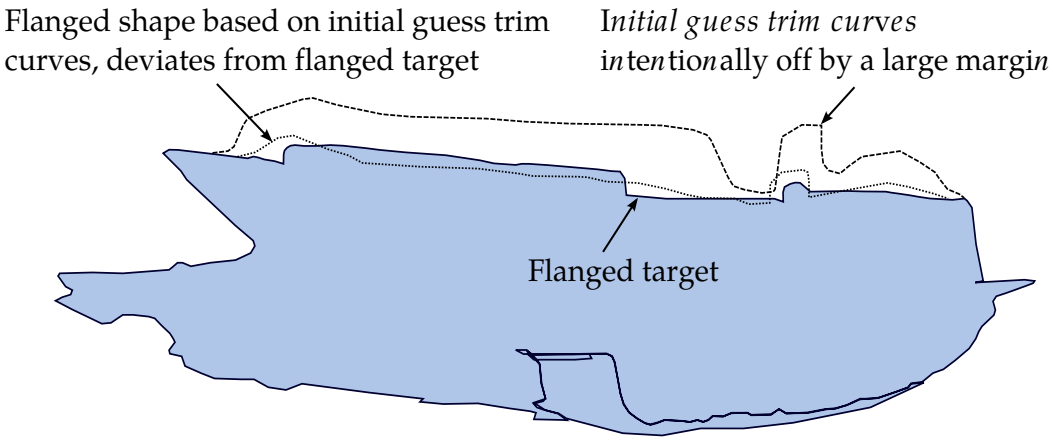


Figure 30-20. Initial trim curves intentionally made to be off by a large margin.

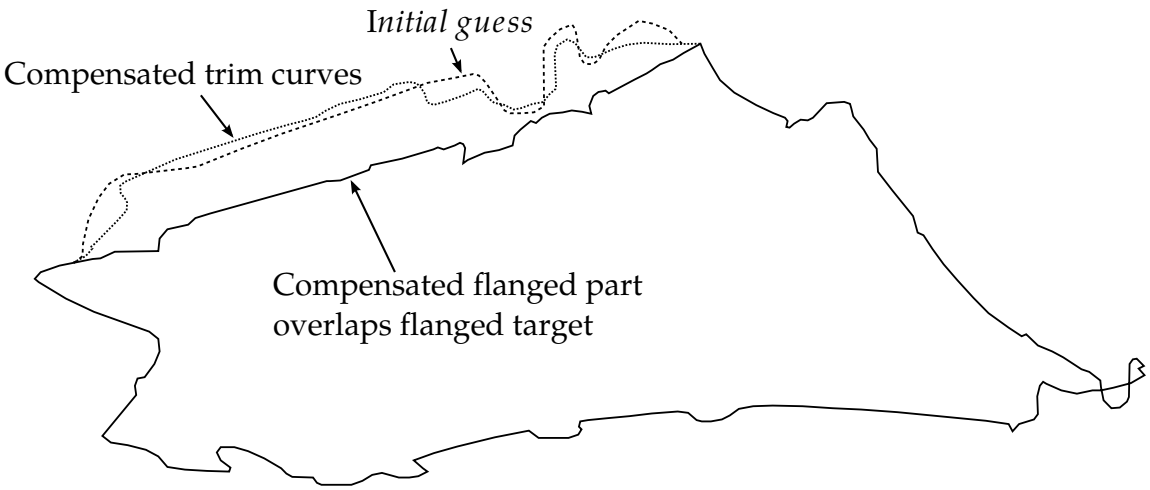
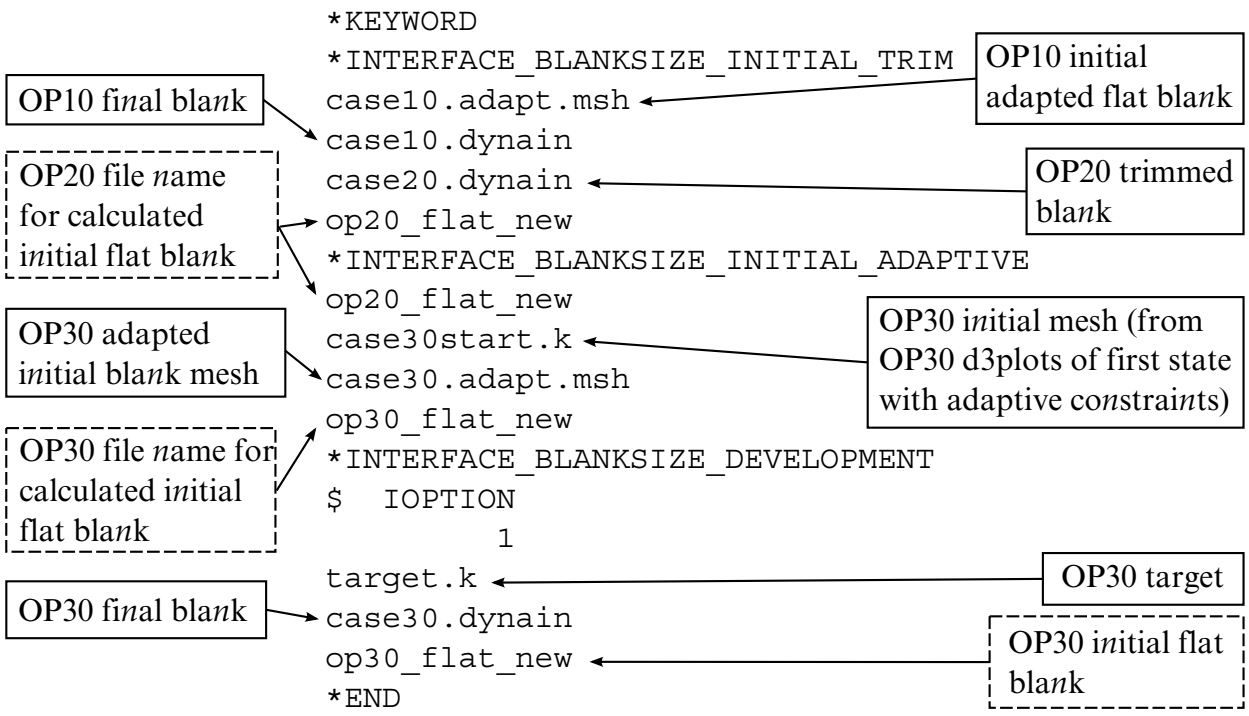


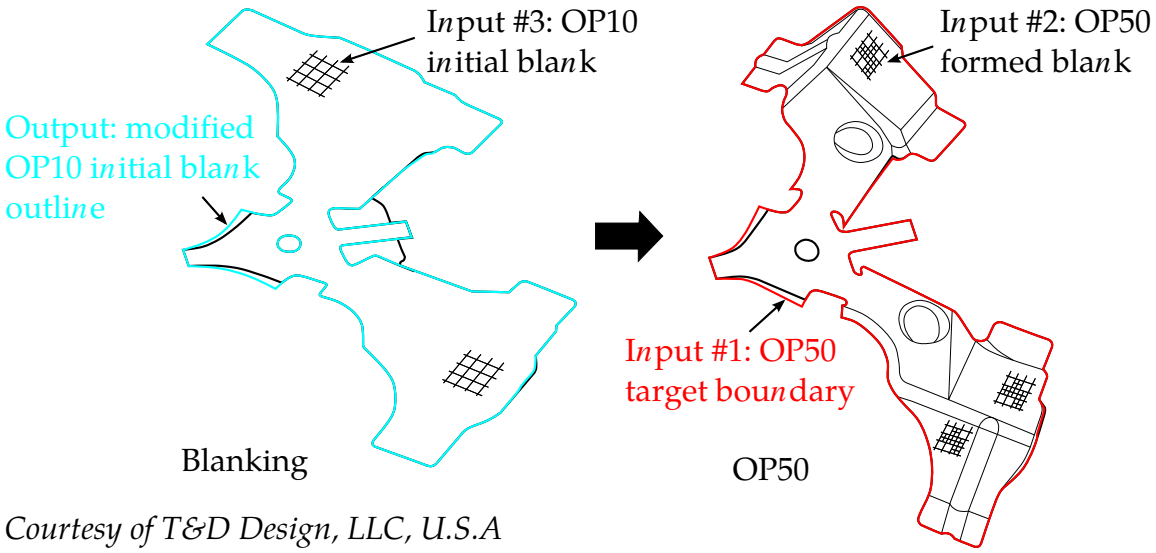
Figure 30-21. Compensated trim curves overlap with the target curves.



User inputs LS-DYNA intermediate output files

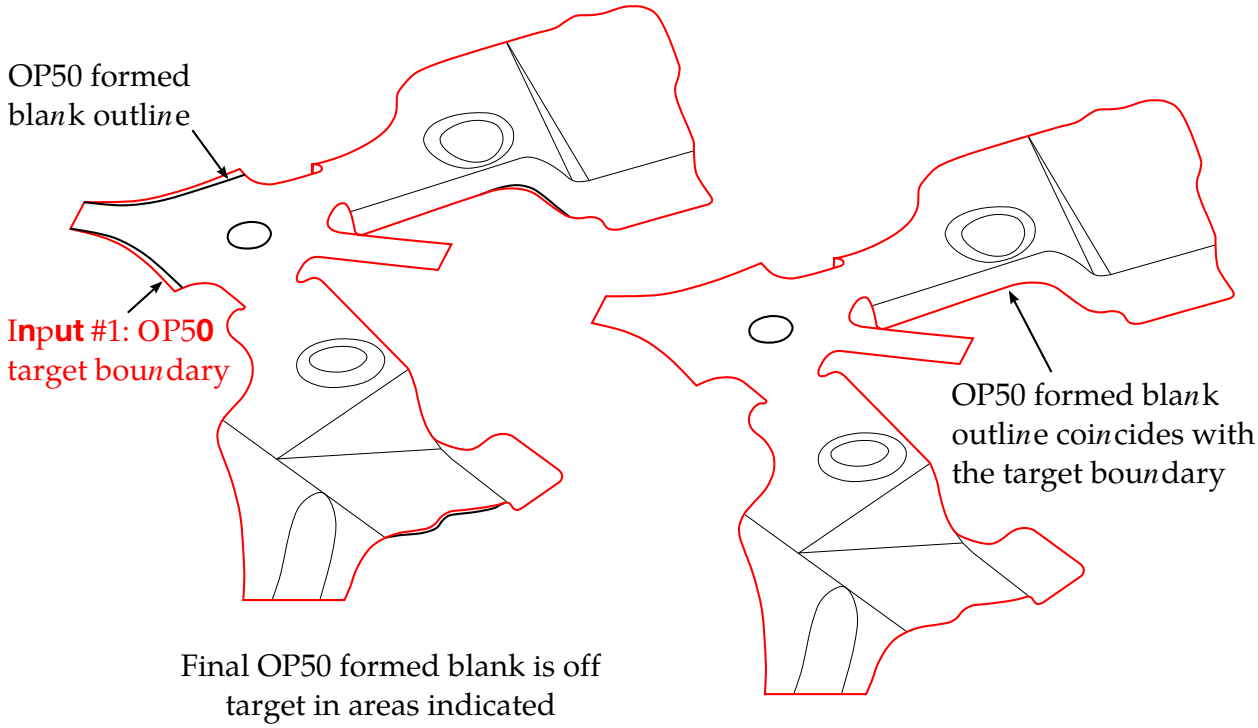
LS-DYNA simulation output: new trim line OP10 (file "trimcurves.ibo")

Figure 30-22. File structures for a multi-process blank development.



Courtesy of T&D Design, LLC, U.S.A

Figure 30-23. Inputs and output for the enhanced DEVELOPMENT feature.



Courtesy of T&D Design, LLC, U.S.A

Figure 30-24. Verification simulation on a progressive die process.

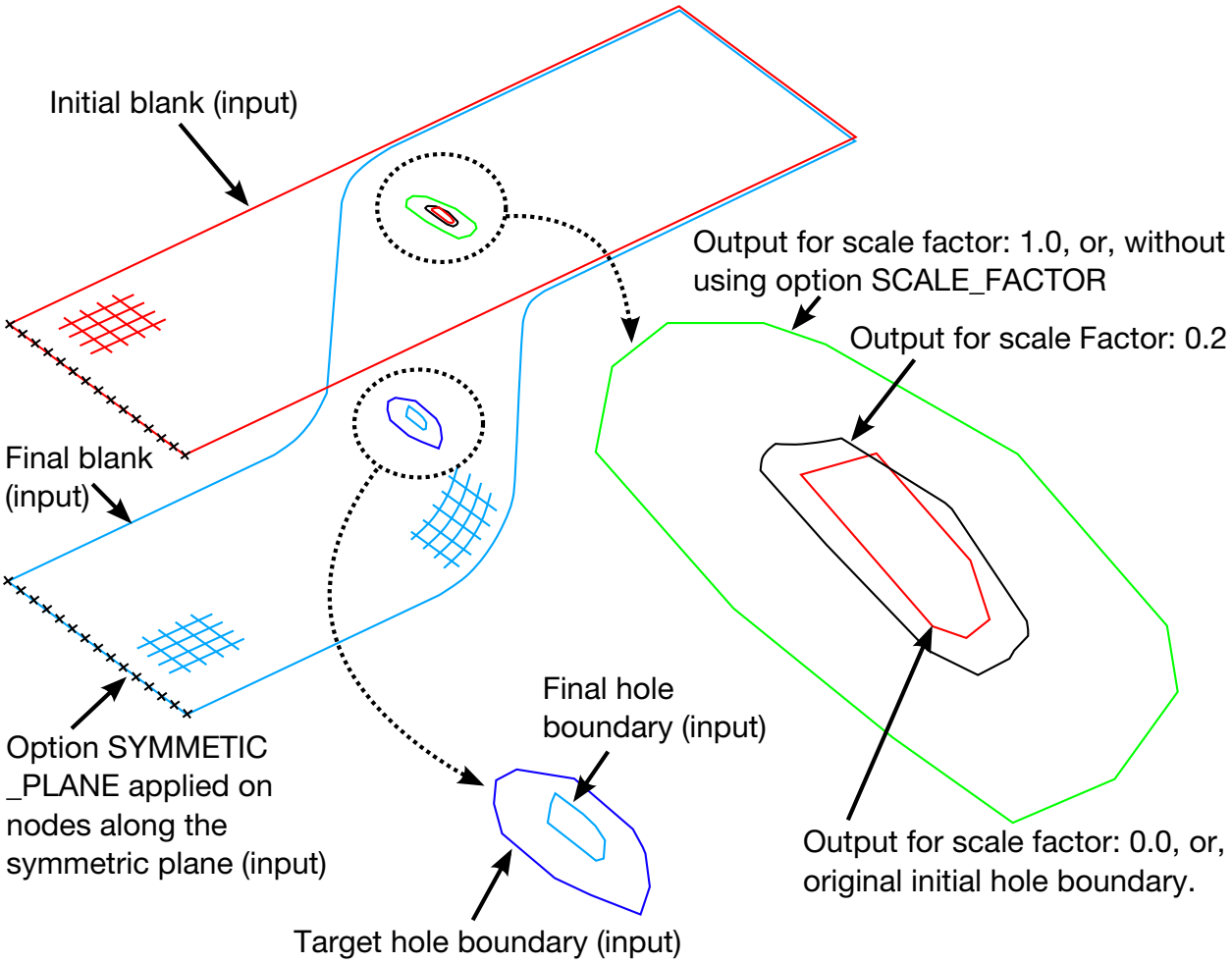


Figure 30-25. Options SCALE_FACTOR and SYMMETRIC_PLANE.

***INTERFACE_COMPENSATION_3D** *{OPTION}*

Available options include:

<BLANK>

ACCELERATOR

MULTI_STEPS

LOCAL_SMOOTH

PART_CHANGE

REFINE_RIGID

FLANGE

Purpose: The springback compensation module automatically modifies the rigid tools to compensate for springback deviations. Because springback compensation is a nonlinear problem, this module uses an iterative approach that usually requires two to three iterations to bring the deviation below the tolerance. This keyword requires a double-precision version.

To use this module, you need to provide the following information: a) the target, which is the deformed part with the original unmodified tool; b) the current tool file at its home or final position; c) the part shape formed with the current tool; d) the part shape after springback with the current tool; and e) a bridging file from the previous compensation (for the first compensation the target file is used, then a file named disp.tmp is generated and will be used for the next iteration). To make the inclusion order-independent, five keywords are used to include the files mentioned above:

a) *INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE

b) *INCLUDE_COMPENSATION_CURRENT_TOOLS

c) *INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK

d) *INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK

e) *INCLUDE_COMPENSATION_COMPENSATED_SHAPE

Depending on the die setup, different kinds of compensation are required. For example, the binder may not be allowed to be modified. Various compensation methods are therefore available. The most used methods are 6, 7, and 8. Linear and nonlinear methods are also available to modify the tool surface outside the blank outline.

The applications for this card include: (1) calculating the deviation of the stamped part from its intended design and automatically compensating the tool to minimize the deviation; (2) mapping the existing trim curve to the modified tool; and (3) automatically detecting undercut. Note that the current methods sometimes fail to eliminate undercut.

NOTE: The keyword *INTERFACE_COMPENSATION_3D replaces *INTERFACE_COMPENSATION_NEW starting in R11.

This feature is implemented and available in LS-PrePost under Application → Metal Forming →Easy Setup.

The ACCELERATOR option speeds up the convergence rate of compensation. This option also allows for a much simpler user interface. See Remark 7.

The MULTI_STEPS option is for tooling compensation of the next die process based on the target blank shape, compensated blank shape for the next step, and current tools. This feature is useful in line die process/tooling compensation. See Remark 8.

The LOCAL_SMOOTH option smooths a tool's local area mesh, which could become distorted because of a bad or coarse mesh of the original tool's surface, a non-constant gap between tooling pairs (for example, flanging post and flanging steel), or a few compensation iterations. This option also requires using *SET_NODE_LIST_SMOOTH. See Remark 10.

The PART_CHANGE is intended for when the compensation analysis has already been performed, but the part or formed blank has undergone a small modification since the analysis was performed. If the modification is small enough, the final compensated tool can be modified using the changed part or formed blank shape, thereby eliminating the need to run a new compensation iteration loop. This option is used with *INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE and *INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL. See Remark 13.

The REFINE_RIGID option refines the rigid tool mesh and aligns the nodes along the user-provided trim curves. The required inputs are the original tool mesh and the trim curves. This option ensures that subsequent compensation of the tool extends to the trim curves. Note that this option does not physically change the tool's shape, but the refined and realigned tool mesh along the trim curves greatly improves convergence of the compensation in the iterative process. This option only needs to be done once before the iterative springback compensation begins. See Remark 14.

The FLANGE option can be used to compensate for springback in a flanging process. See Remark 16.

Card Summary:

Card 1a.1. Include this card if the keyword option is <BLANK>, MULTI_STEPS, or LOCAL_SMOOTH.

METHOD	SL	SF	ELREF	PSIDP	UNDCT	ANGLE	NLINEAR

Card 1a.2. Include this card if the keyword option is <BLANK>, MULTI_STEPS, or LOCAL_SMOOTH.

TANGENT						FXNSID	TRRNSID

Card 1a.3. This card is read if the keyword option is <BLANK>, MULTI_STEPS, or LOCAL_SMOOTH. It is optional. This card is needed if springback is performed on a post or through clamping.

VX	VY	VZ	GAPO				

Card 1b. Include this card if the ACCELERATOR option is used.

ISTEPS	TOLX	TOLY	TOLZ	OPTION			

Card 1c. Include this card if the PART_CHANGE option is used.

MAXGAP							

Card 1d.1. Include this card if the REFINED_RIGID option is used.

FILENAME1

Card 1d.2. Include this card if the REFINE_RIGID option is used.

FILENAME2

Card 1e. This card is included if the FLANGE option is used. A maximum of two of this card may be input and the input ends with the next keyword ("*") card.

PID	VX	VY	VZ				

Data Card Definitions:

Include this card if the keyword option is <BLANK>, MULTI_STEPS, and LOCAL_-SMOOTH.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	METHOD	SL	SF	ELREF	PSIDP	UNDCT	ANGLE	NLINEAR
Type	I	F	F	I	F	F	F	I
Default	6	5.0	0.75	1	none	0	0.0	1

VARIABLE	DESCRIPTION
METHOD	There are several extrapolation methods for the addendum and binder outside of trim lines. See Remark 1 for a discussion of available methods.
SL	The smooth level parameter controls the smoothness of the modified surfaces. A large value makes the surface smoother. Typically, the value ranges from 5 to 10. If springback is large, the transition region is expected to be large. However, by using a smaller value of SL, the region of transition can be reduced.
SF	Shape compensation scale factor. The value scales the springback amount of the blank, and the scaled amount is used to compensate the tooling. GT.0: Compensate in the opposite direction of the springback; LT.0: Compensate in the punch moving direction (for undercut). This scale factor determines how much the shape deviation is compensated, and a favorable value can reduce the number of iterations. However, the right value is case dependent. Usually, SF is chosen between the range of 0.75 and 1.25.
ELREF	Element refinement option: EQ.1: Special element refinement is used with the tool elements (default); EQ.2: Special element refinement is turned off.
PSIDP	Define the part set ID for primary parts of the tooling. Properly

VARIABLE	DESCRIPTION
	choosing the parts for the primary side is important, as it affects the kinds of modifications made to the tooling. Usually, only one side of the tool is chosen as the primary side, and modifications to the other side (the secondary side) depend solely on changes to the primary side. This specification allows the two sides to be coupled while maintaining a constant (tool) gap between them. If both sides are chosen as primary, the gap between them might change and become inhomogeneous. When using METHOD 7, the choice of primary side affects results for three-piece draw models. At this time, when the punch and binder are chosen as the primary side, the binder region is not changed. Otherwise, when the die is chosen as the primary side, the binder is changed since the changes extend to the edges of the primary tool.
UNDCT	Tool undercut treatment option: EQ.0: No check (default) EQ.1: Check and fix undercut.
ANGLE	An angle defining the undercut.
NLINEAR	Activate nonlinear extrapolation

Include this card if the keyword option is <BLANK>, MULTI_STEPS, and LOCAL_-SMOOTH.

Card 1a.2	1	2	3	4	5	6	7	8
Variable	TANGENT						FXNSID	TRRNSID
Type	I						I	I
Default	0						0	0

VARIABLE	DESCRIPTION
TANGENT	A flag to maintain tangency during the compensation. Set TAN-GENT = 1 to maintain the tangential transition between the compensated and non-compensated areas of the rigid tool (for example, between addendum and binder in METHOD 7), and between the rigid tool area at the trim curves and the addendum part

VARIABLE	DESCRIPTION
	of the tool. See Remark 15.
FXNSID	Node set ID specifying the nodes that are fixed during springback compensation. If 0, these nodes are internally determined.
TRRNSID	Node set ID specifying the transitional region during compensation. If 0, these nodes are internally determined.

This card is read if the keyword option is <BLANK>, MULTI_STEPS, and LOCAL_SMOOTH. It is optional. This card is needed if springback is performed on a post. See Remark 17.

Card 1a.3	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	GAP0				
Type	F	F	F	F				
Default	0.0	0.0	0.0	0.0				

VARIABLE	DESCRIPTION
VX, VY, VZ	Components of the direction the blank needs to move to sit on the post
GAP0	Distance between the initial position of the blank and the post

This card is included if the ACCELERATOR option is used.

Card 1b	1	2	3	4	5	6	7	8
Variable	ISTEPS	TOLX	TOLY	TOLZ	OPTION			
Type	I	F	F	F	I			
Default	0	0.5	0.5	0.5	1			

VARIABLE	DESCRIPTION
ISTEPS	Number of compensation iterations for an accelerated compensation calculation. See Remark 7.

VARIABLE	DESCRIPTION
TOLX	Part deviation tolerance between current blank and target blank shape in global x-direction.
TOLY	Part deviation tolerance between current blank and target blank shape in global y-direction.
TOLZ	Part deviation tolerance between current blank and target blank shape in global z-direction.
OPTION	Compensation acceleration method. Currently available only for method 1.

This card is included if the PART_CHANGE option is used.

Card 1c	1	2	3	4	5	6	7	8
Variable	MAXGAP							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
MAXGAP	Maximum gap between the original part and changed part

This card is used if the REFINE_RIGID option is used:

Card 1d.1	1	2	3	4	5	6	7	8
Variable	FILENAME1							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME1	A keyword file name of the rigid tool mesh to be refined. This should be the tooling mesh used in the forming or flanging simulation, before any compensation is done. The refined rigid tool

VARIABLE	DESCRIPTION
	mesh will be in the file rigid_refined.tmp. See Remark 14.

This card is included if the REFINE_RIGID option is used.

Card 1d.2	1	2	3	4	5	6	7	8
Variable	FILENAME2							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FILENAME2	A keyword file name with trim curves defined using *DEFINE_CURVE_TRIM_3D. The curves will be used to refine and realign the FILENAME1 to improve the convergence in the iterative compensation process. The refined rigid tool mesh will be in the file rigid_refined.tmp. See Remark 14.

This card is included if the FLANGE option is used. A maximum of two instances of this card may be input, and the input ends with the next keyword ("*") card.

Card 1e	1	2	3	4	5	6	7	8
Variable	PID	VX	VY	VZ				
Type	I	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
PID	Part ID of the flanging steel to be compensated
VX, VY, VZ	Vector components of the flanging steel's (PID) moving direction

Remarks:

1. **Overview of compensation methods** . After trimming, only a limited portion of the tool is directly related to the springback of the blank. Modifications of the rigid tool outside the trimmed region involve extrapolation. Unfortunately, extrapolating is unstable and tends to generate non-smooth surfaces. To resolve this problem, eight smoothing algorithms are implemented. The most frequently used methods are methods 7, 8, and -8, while the others are used only occasionally.

   a) *Method 7*. If the punch is chosen as the primary side, the binder will not be changed. Aside from the region inside the punch opening, the rest of the model is untouched. Smoothing has little effect on method 7. The smoothness of the modified tool depends on the magnitude of the springback and the size of the addendum region. This method is nonlinear and, therefore, necessitates an iterative solve.

      Advantages: The binder remains unchanged.

      Disadvantages: The change is limited to inside the addendum region, and the modified surface may not be smooth if the springback magnitude is large and the transition is small.

   b) *Method 6.* The smoothness and the transition region of the modified surface depend on the springback magnitude and the smoothing factor. If the springback magnitude is large, the transition region is increased automatically. On the other hand, the transition region is smaller if the springback magnitude is small. At the same time, a larger smoothing factor results in a smaller transition region. Like method 7, this method is also nonlinear.

      Advantages: The smoothness of the modified surfaces can be controlled.

      Disadvantages: It is impossible to limit the transition region, and the binder surface (and therefore the draw beads) could change if the springback is large.

   c) *Method 3.* This method is similar to method 6; however, it is linear and requires no iteration.

   d) *Method 8.* This method is an enhanced version of method 6. It can account for addendum and binder changes. Usually, PSIDP includes the upper tooling, including the addendum and binder (in an air draw).

   e) *Method -8.* This method is a modification of Method 8. It is used for trim die nesting (from the drawn panel shape). See Remark 9.

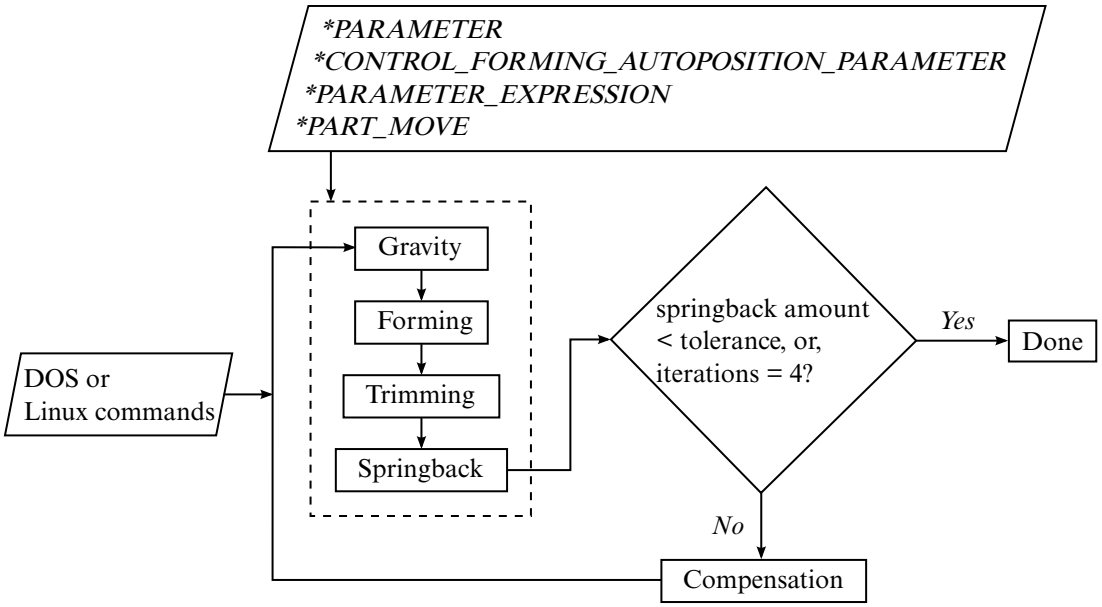


Figure 30-26. Iterative compensation flow chart

f) *Method 9*. This method is intended for compensation of the trim die. After draw forming, the draw shell is put on the trim post (or nest). Due to springback, a separation may exist between the shell and the blank. However, a too-large gap causes quality issues when the upper tool moves down and closes it. This option morphs the post and makes sure the gap between the blank and the post is within tolerance.

g) *Methods 1, 2, 4, and 5*. These methods are deprecated and may be removed in the future. They are included only for maintaining backward compatibility.

2. **Preventing undercut.** When the draw wall is steep, it is likely that undercut occurs. Because undercut is not acceptable in real-world die manufacturing, it must be prevented. The compensation code can automatically detect undercut and issue a warning message. Additionally, it writes a list of undercut elements to a file called blankundercut.tmp, so you can easily identify which elements may be problematic. If the undercut affects only a few elements, it is possible to fix the problem manually. Undercut can be reduced by compensating for springback only in the moving direction of the punch (by using a negative scale factor). This method is not 100% reliable, and more robust solutions are being studied.

3. **Iterative springback compensation.** Figure 30-26 illustrates the iterative springback compensation algorithm as applied to a typical stamping process. The first stamping process simulation is done following gravity → forming → trimming → springback (ITERATION 0). The stamping process simulation is set up using eZ-Setup in LS-PrePost. By using the parameterized automatic

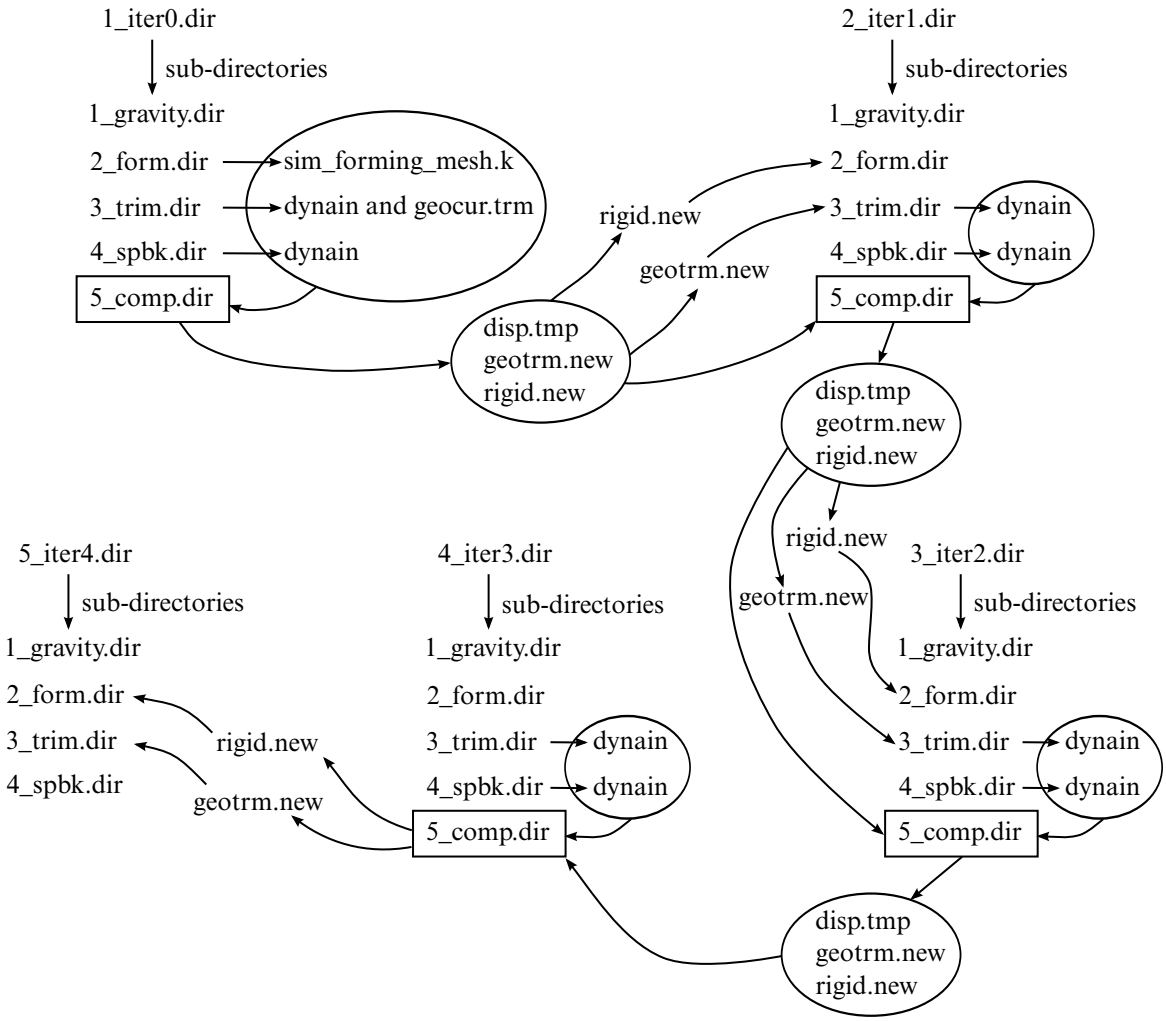


Figure 30-27. File structure for compensation

tool/blank positioning feature, the process simulation is fully automated (no user intervention required). Based on the calculated springback amount, tooling geometry is compensated through a compensation run. The stamping process simulation is conducted again, automatically, based on the new compensated tooling, followed by a second tooling compensation (ITERATION 1). Iterations 2, 3, and 4 follow the same pattern. The iteration process is repeated until the blank springback shape conforms to the target shape or until it reaches 4 iterations (typically required to achieve part tolerance). With some shell scripting, this loop can be automated. These tools allow you to toggle between the single and double precision versions.

Tracking the files involved in the iterative process can be daunting, especially at the advanced stage of the iterations. Figure 30-27 indicates what is written to storage during the process.

An input deck defining a springback compensation model is given below. The keyword file blank0.k includes node and element information for the blank

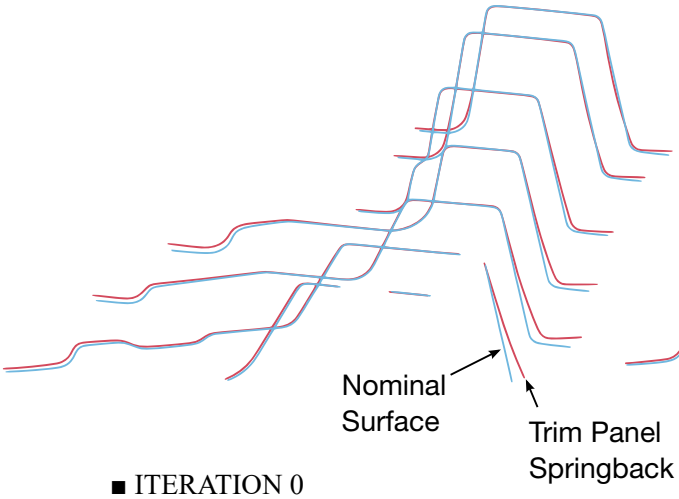
before springback (after forming and trimming) with adaptive constraints (if any exist). The keyword file spbk.k includes node and element information for the blank after springback, with adaptive constraints (if they exist). The blank shapes before and after springback (blank0.k and spbk.k) may be based on either the original die design (ITER0) or an intermediate compensated die design (e.g., the n^{th} iteration).

The keyword file reference0.k is the blank before springback for iteration 0 (ITER0). This file is blank0.k and should *not* change from iteration to iteration. For iteration 0, the file reference1.k is also the same as blank0.k, but for iteration 1, reference1.k should be the display.tmp generated from the compensation calculation during iteration 0, and so on and so forth for the subsequent iterations.

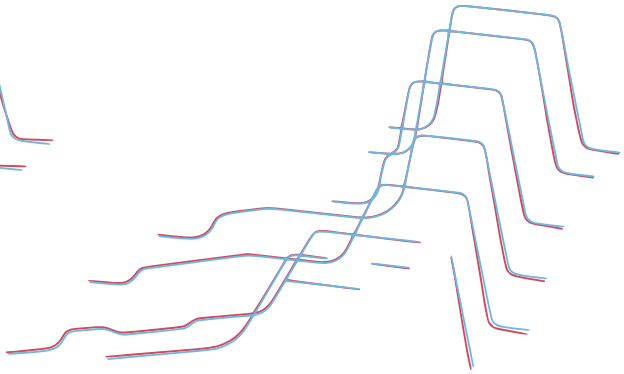
The keyword input tools.k must contain the mesh information for all of the stamping tools in their home positions. Compensated tools are written to rigid.new with the original constant gap being maintained among the tools. During the baseline calculation, iteration 0, a keyword file called geocur.trm is generated during a trimming simulation based on trimming curve input (usually in IGES format). During the compensation run of ITER1, geocur.trm is used to generate new trim curves, which are output to geotrm.new. The new curves conform to the current compensated tools and are used for the next iteration. The new trim file, geotrm.new, is also in keyword format and contains *DEFINE_CURVE_TRIM_3D.

The example below models a three-piece air-draw process. The upper die cavity (including binder) has part ID 2, which is included in the part set ID 1 and is used for variable PSIDP. Method 8 compensates all the tools included in file tools.k based on the compensated shape for the upper cavity.

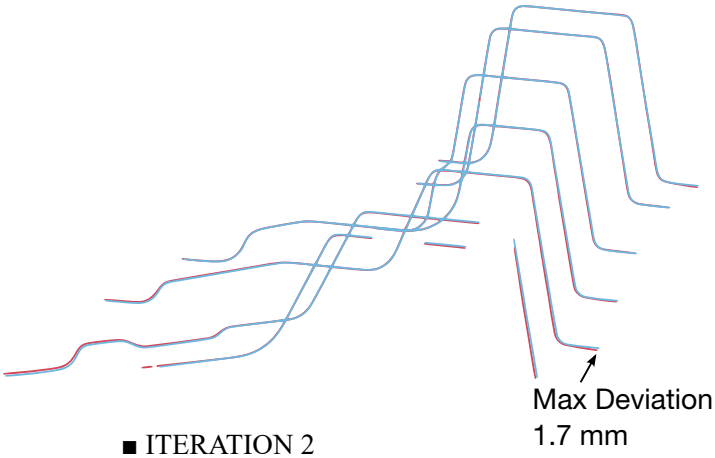
```
*KEYWORD
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*INTERFACE_COMPENSATION_3D
$   METHOD        SL        SF     ELREF     PSIDP    UNDRCT    ANGLE NLINEAR
         8    10.000     1.000         0         1         0      0.0       1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
blank0.k
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINBACK
spbk.k
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
reference0.k
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
reference1.k
*INCLUDE_COMPENSATION_CURRENT_TOOLS
tools.k
*INCLUDE_COMPENSATION_TRIM_CURVE
geocur.trm
*SET_PART_LIST
$      PSID
         1
$      PID
         2
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
*END
```



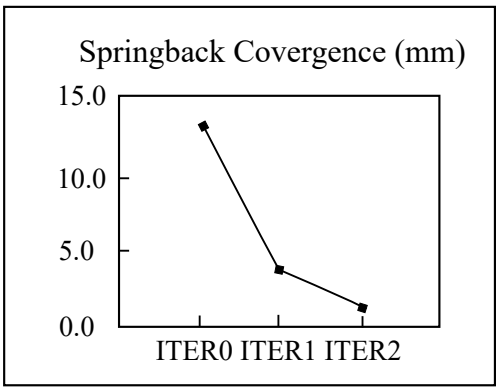
■ ITERATION 0
■ Max. springback 13 mm on flanges



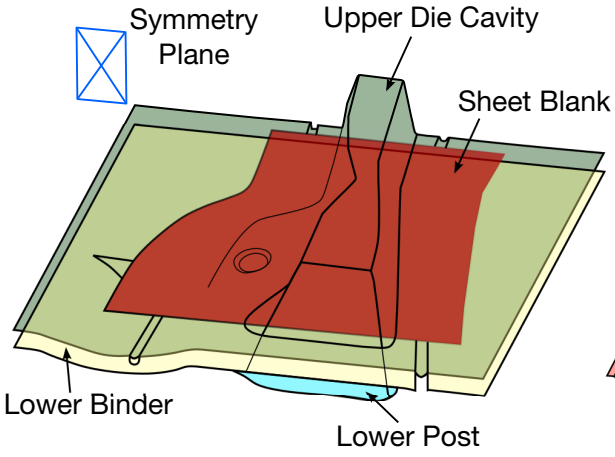
■ ITERATION 1
■ Springback reduced to < 3.5 mm



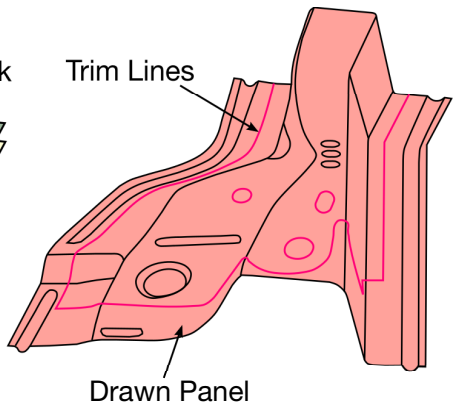
■ ITERATION 2
■ Springback reduced to < 1.7 mm



■ Compensation convergence history



2005 NUMISHEET Cross Member Model



Drawn Panel and Trim Lines

Figure 30-28. Iterative springback compensation on NUMISHEET2005 Xmbr(red – springback, blue – design intent)

4. **A NUMISHEET2005 example.** In Figure 30-28, the NUMISHEET2005 cross member is compensated following the flow chart. In two iterations, the springback is reduced from 13mm to 1.7mm. Further iterations would reduce the part deviation down to a specific design target. Typically, four iterations are needed.

5. **Iterative compensation applied during die construction.** The blank shape after springback can be obtained from the actual experimental shape of the springback panel, if available. For example, in hard tool construction, the trimmed panel can be scanned using white light technology, and the panel shape can be written to an STL file. The STL format can be easily converted to keyword format, and the trimmed panel can be used as a rigid tool onto which the baseline (ITER0) trimmed panel (deformable) can be "pushed" using element normal pressure and *CONTROL_IMPLICIT_FORMING with IOPTION set to 1. In this scenario, adaptive refinement is disabled to maintain the one-to-one correspondence between element and node information. An advantage of this method is that the springback shape used for compensation will match the actual panel springback; therefore, the best tooling compensation result is expected. An example of such is shown in Figures 30-29 and 30-30.

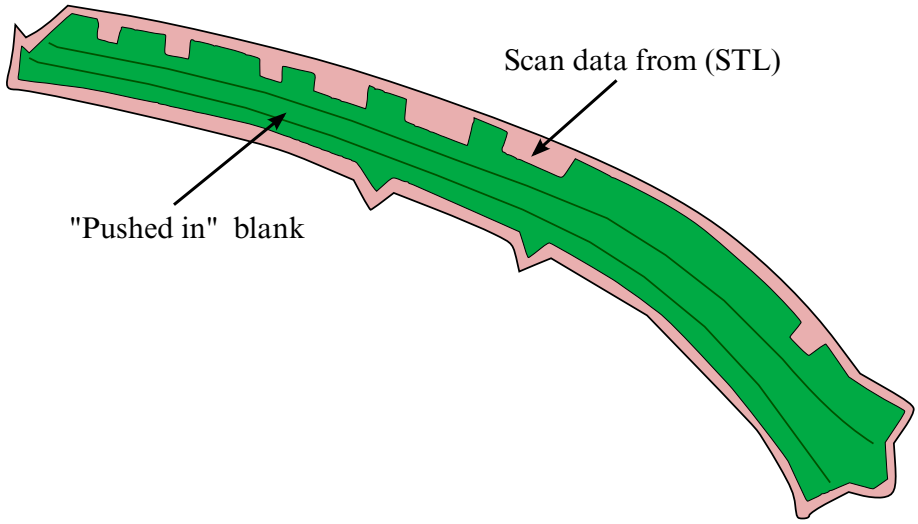


Figure 30-29. A trimmed panel "pushed" onto the scan data (rigid body).

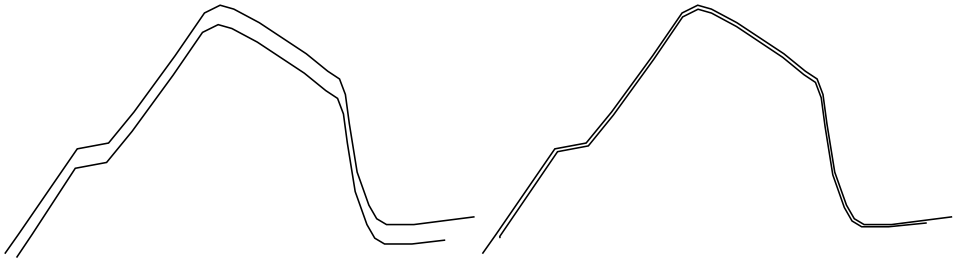


Figure 30-30. Section showing the "push" results – before and after.

6. **Compensation of localized regions.** Compensation of a localized tooling region is possible using *INCLUDE_COMPENSATION_CURVE. To do this in the previous example input, incorporate the following lines:

```
*INCLUDE_COMPENSATION_CURVE
curves.k
```

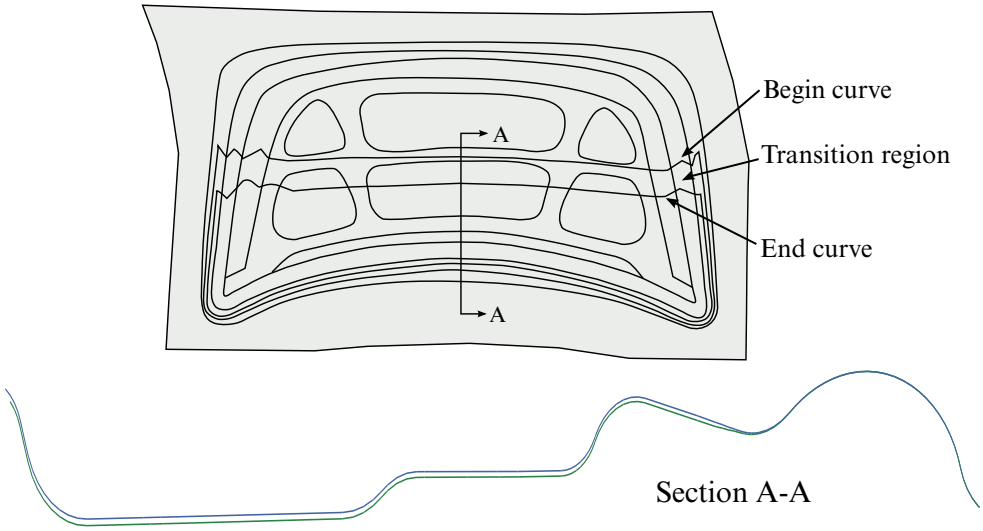


Figure 30-31. Two curves defining a localized area of a decklid inner panel.

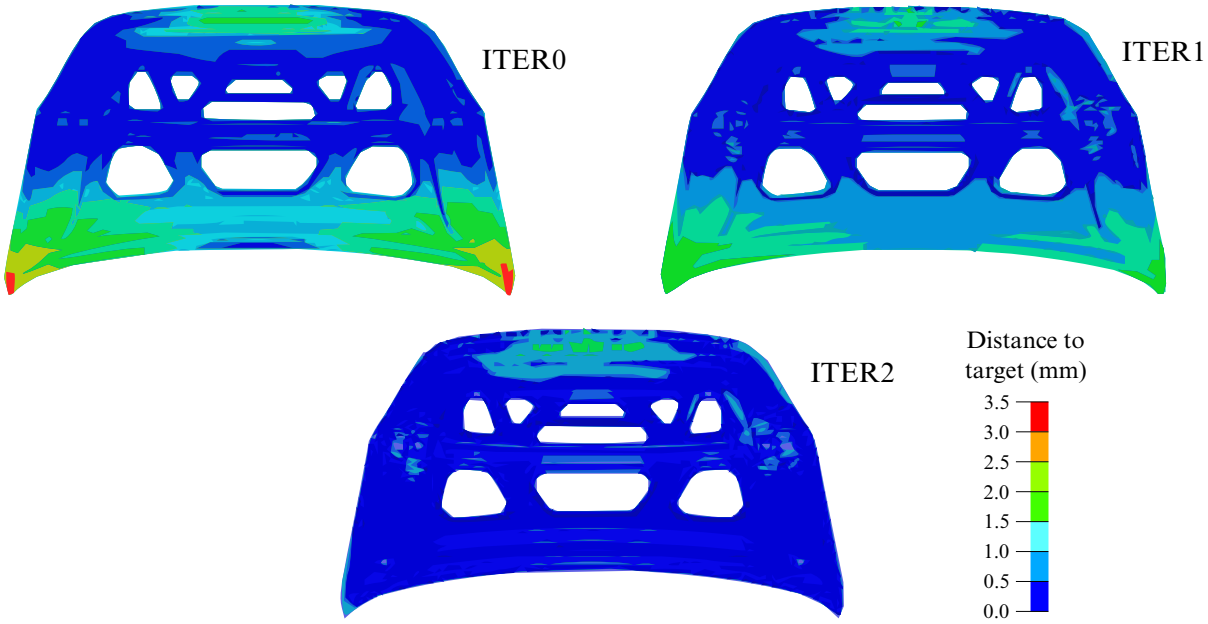


Figure 30-32. Iterative compensation for a localized (back light) region.

The file curves.k defines the two enclosed "begin" and "end" curves using *DEFINE_CURVE_COMPENSATION_CONSTRAINT_BEGIN/END. In Figure 30-31, the NUMISHEET2005 deck lid inner panel is compensated locally in the horizontal area above the back light. Tangency of the compensated tool is maintained at the "End Curve" as shown in section A-A. Figure 30-32 also includes color contours of the part-separation distance throughout the iterations between

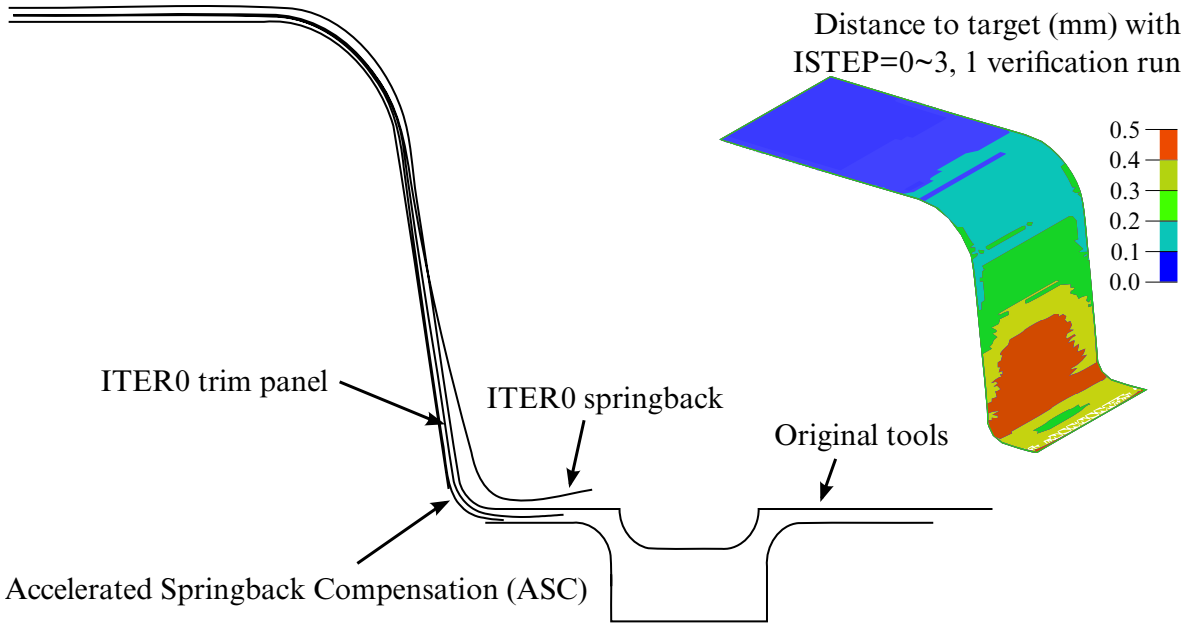


Figure 30-33. Accelerated Springback Compensation.

the compensated panel and the target design intent. Part tolerance is achieved in two iterations.

7. **Accelerated springback compensation.** The option ACCELERATOR can be used in conjunction with *INCLUDE_COMPENSATION_ORIGINAL_DYNAIN and *INCLUDE_COMPENSATION_SPRINGBACK_INPUT to compensate springback with a faster convergence rate and a simplified user interface. A complete example is provided below. The example uses springback input file spbk.dyn and a trimmed panel with file name case20trimmed.dynain (including all stress and strain tensors and adaptive constraints). The variable ISTEPS is increased from 0 to 3, representing 3 compensation iterations. ISTEPS = 0 represents the baseline springback simulation (ITER0), while ISTEPS = 1, 2, 3 represent the compensation iterations. This feature requires you to change only one variable (ISTEPS), and then submit the same input file to continue the next iteration.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D_ACCELERATOR
$    ISTEPS      TOLX      TOLY      TOLZ    OPTION
         3      0.20      0.20       0.2         1
*INCLUDE_COMPENSATION_ORIGINAL_DYNAIN
./case20trimmed.dynain
*INCLUDE_COMPENSATION_SPRINGBACK_INPUT
./spbk.dyn
*END
```

This algorithm generates many scratch files, including a file named acceltmp.tmp. *Do not delete them*. They are used to pass data between steps. A file named compensation.info is generated and updated after each ISTEPS calculation. It contains iteration information, including maximum deviations in the x,

y, and z directions. When the maximum deviation is within the tolerances specified in the TOLX, TOLY, and TOLZ fields, a message appears in the file indicating the compensation iterations have converged, along with a message bootstrapping the next step. A file spbk.new is generated in the same directory and is used by *INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK with the scale factor for the tool compensation set to one. After the compensation, a verification calculation may be needed.

Currently, mesh coarsening and checking are not supported in the accelerated mode. Also, the dynain file from the previous die process is not necessary.

An example of this feature is shown for a simple channel type of draw (one-half model) in Figure 30-33, which converged in three iterations, while four iterations were needed for the non-accelerated compensation.

8. **Line die compensation.** The option MULTI_STEPS can be used together with *INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP to enable compensation of tools for the next die process. An example is given below. In this example the target blank, named reference0.tmp, and the current tool named rigid.tmp come from the first die process. The disp.tmp file comes from the compensation in the second die process step. For example, a flanging die compensation can be a second die process step, preceded by a redraw die process as the first die process step.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D_MULTI_STEPS
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$   METHOD        SL        SF     ELREF      PSID    UNDRCT     ANGLE   NLINEAR
         8     6.000      1.00         1         1         0         0         1
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
reference0.tmp
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE_NEXT_STEP
disp.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
rigid.tmp
*SET_PART_LIST
$      PSID
         1
$       PID
         2
*END
```

9. **Compensation of trim dies (trim die nesting).** The trim die can be compensated using the drawn panel's springback shape when METHOD is set to -8. In the example below, which is also shown in Figure 30-34, the draw panel, state1.k, is taken as the blank before springback, and the draw panel springback shape, state2.k, is taken as the blank after springback. The tool shape for the draw process, drawtool.k, is used as the current tool. After the simulation, the algorithm creates a compensated tool named rigid.new, which can be used for the trim die shape.

```
*KEYWORD
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

```
*INTERFACE_COMPENSATION_3D
$    METHOD        SL        SF      ELREF       PSID    UNDRCT      ANGLE   NLINEAR
        -8    10.000     1.000          2          1         0        0.0         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
state1.k
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
state2.k
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
ref0.tmp
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
ref1.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
drawtool.k
*INCLUDE_COMPENSATION_TRIM_CURVE
orginaltrim.k
*SET_PART_LIST
$       PSID
          1
$       PID
          3
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*END
```

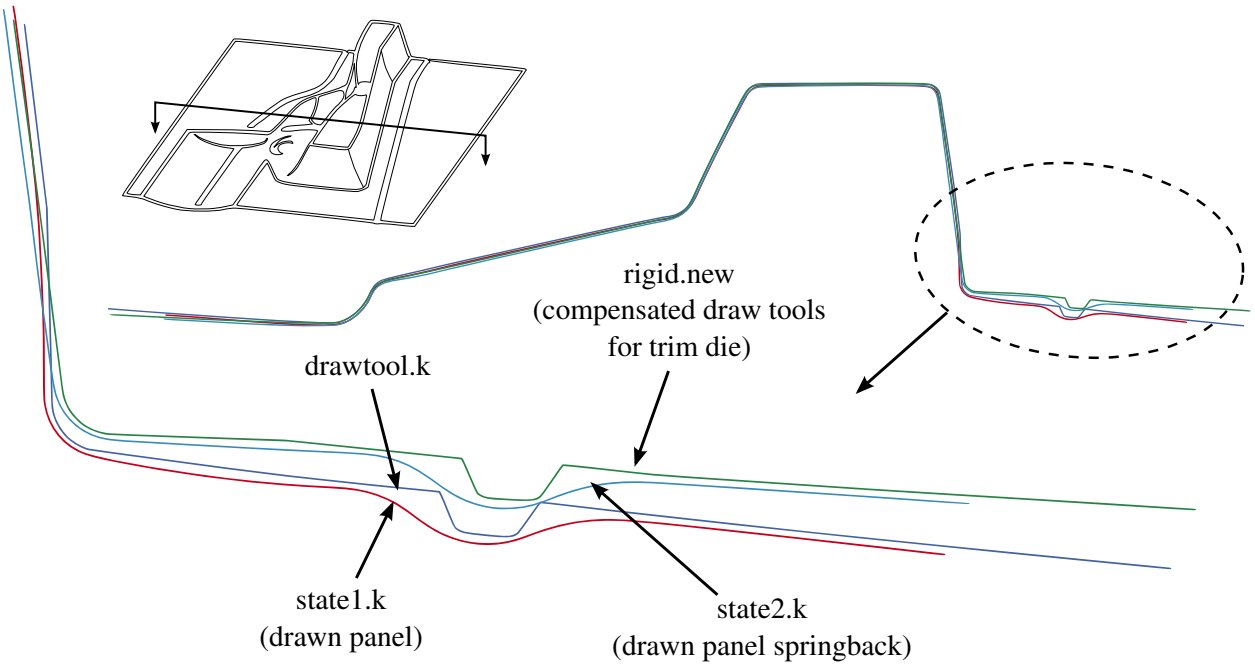


Figure 30-34. Trim die compensation with drawn panel springback shape.

10. **Local smoothing of the tooling mesh.** The keyword option LOCAL_-SMOOTH can be used, along with a few more keywords, to smooth and restore the distorted tooling mesh after iterative compensation. In the example below, *INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL includes an original tool, rigid.tmp, which has a good and smooth mesh. *INCLUDE_COMPENSATION_3D_RIGID_TOOL includes a compensated tool, rigidnew.bad, which could have a distorted mesh. *SET_NODE_LIST_SMOOTH defines a node set that includes a distorted local area in the distorted mesh. Each node set defines a region needing smoothing. The node set should not include any boundary

nodes of the tooling parts, otherwise position of the tooling may be altered. Smoothed tooling is stored in a file called rigid.new. In this example, method 7 is active, the variable ELREF is set to 2, and PSID is left as undefined. Note also that the *INCLUDE keyword is *not* supported here. For example, *SET_NODE_-LIST_SMOOTH must not be in a separate file that is included in the main input file.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D_LOCAL_SMOOTH
$    METHOD        SL        SF      ELREF       PSID     UNDRCT      ANGLE    NLINEAR
         7    10.000     1.000          2                     0        0.0          1
*INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL
rigid.tmp
*INCLUDE_COMPENSATION_NEW_RIGID_TOOL
rigidnew.bad
*SET_NODE_LIST_SMOOTH
         1
     61057     61058     61059     61060     61061     61062     61063     61064
...
*SET_NODE_LIST_SMOOTH
         2
     56141     56142     56143     56144     56145     56146     56147     56148
...
*END
```

In the example shown in Figures 30-35 and 30-36, the local mesh is smoothed in the draw-bead area of the NUMISHEET2005 cross member. In this case, the die gap is not maintained throughout the tooling surface. Typically, this occurs in the draw bead regions, where male beads have lower bending radii (missing upper radii) and female beads have only upper bending radii (missing lower radii). Two node sets are defined for the local areas of the left and right female draw beads (Figure 0-12) that require smoothing. It is important to include some nodes from the relatively flat portion of the binder immediately off the bend radii in the node sets. Figure 30-38 shows the smoothed results.

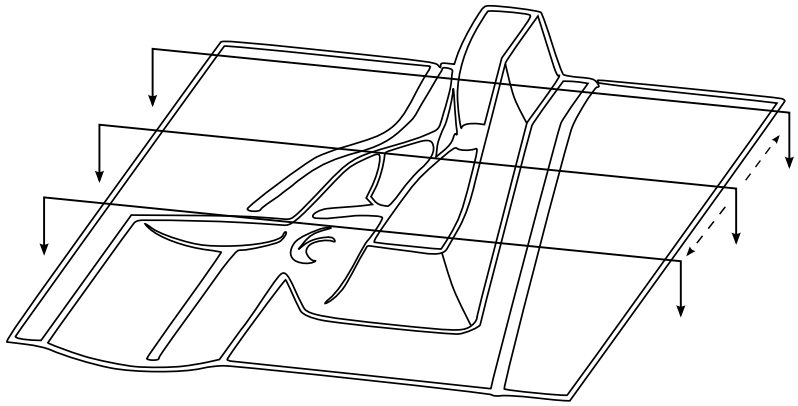


Figure 30-35. The NUMISHEET 2005 cross member.

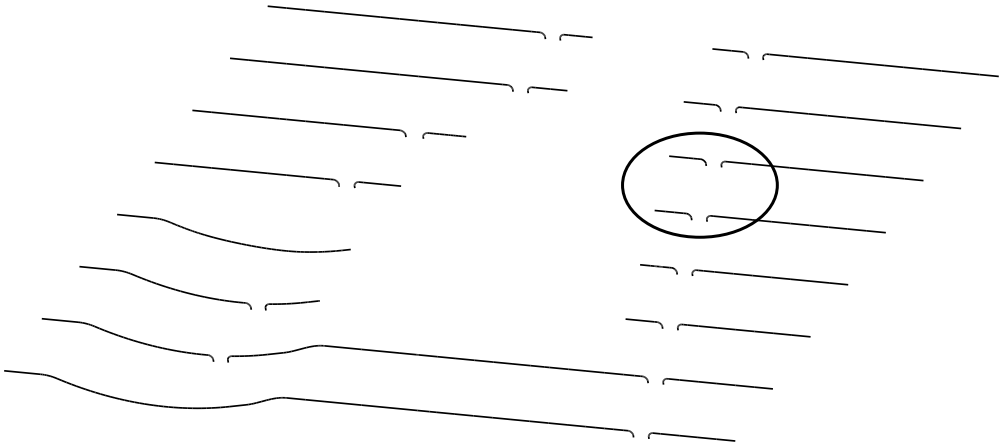


Figure 30-36. Multiple sections cut on the lower binder.

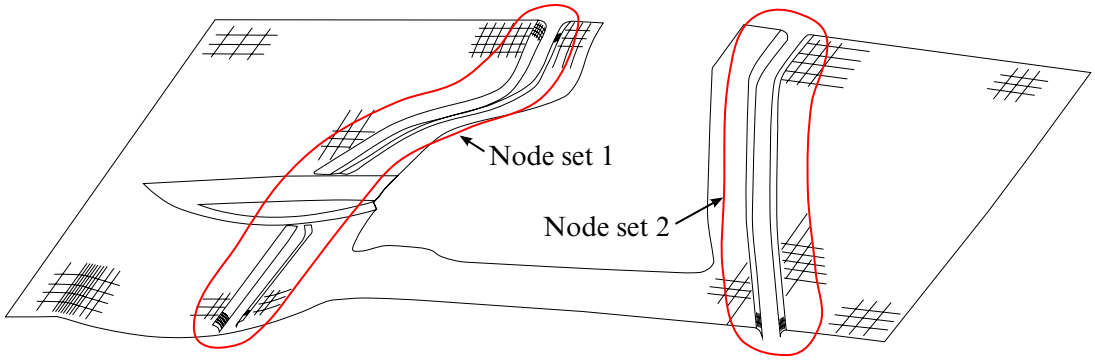


Figure 30-37. Local smoothing - two node sets defined including some nodes on the relatively flat binder area for both left and right draw beads.

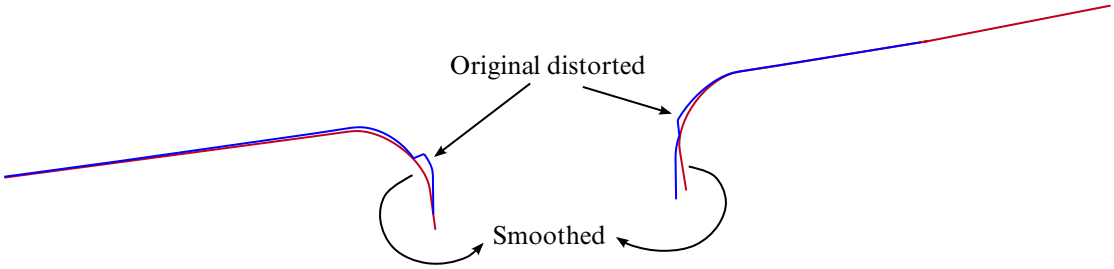


Figure 30-38. Comparison between original and smoothed tooling mesh.

In another example, a corner of a flanging die on a fender outer is smoothed. The mesh becomes distorted after a few compensation iterations, as shown in Figure 30-39. Figure 30-40 shows the result of local smoothing.

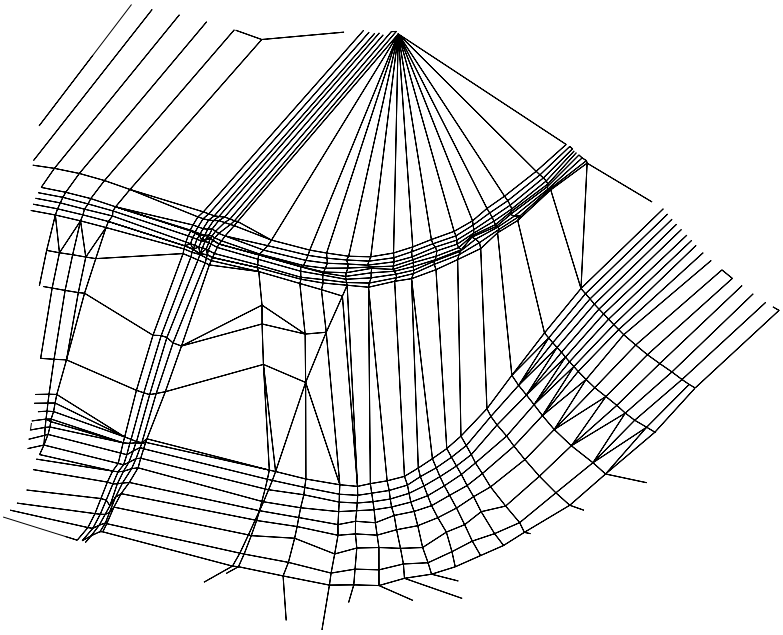


Figure 30-39. Original distorted tooling mesh.

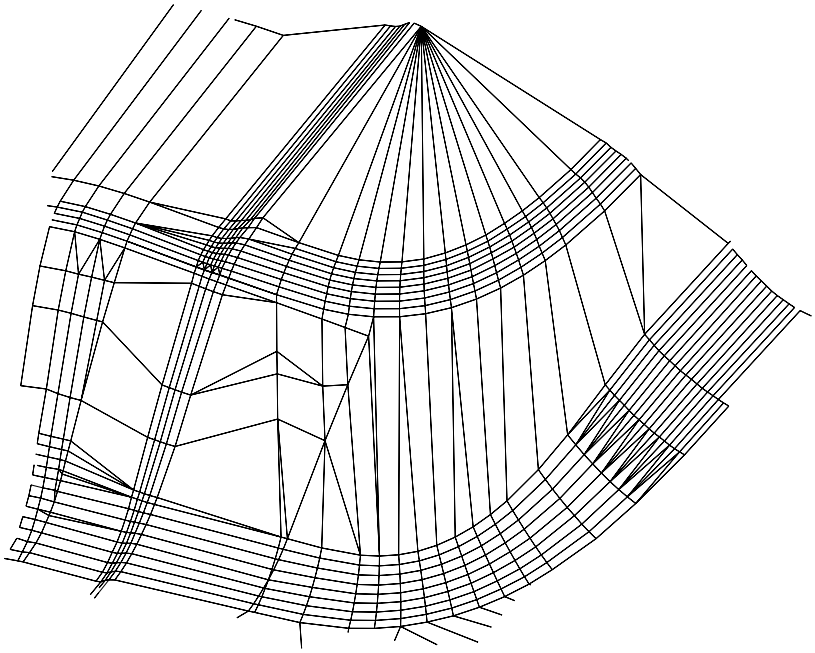


Figure 30-40. Smoothed tooling mesh.

11. **Compensation with a symmetry boundary condition.** To enforce a symmetry boundary condition, see the example at the end of the manual page for *INCLUDE_COMPENSATION_SYMMETRIC_LINES.

12. **Global compensation using the original tooling mesh.** For some tooling meshes, the compensated die surfaces are distorted. *INCLUDE_COMPENSATION_ORIGINAL_TOOL causes the compensation code to use the original

tooling mesh (starting in the second compensation) to extrapolate the addendum and binder in the compensated tooling surfaces. This feature minimizes cumulative error compared with using the last compensated tooling mesh and is therefore preferred.

A keyword example is included below. In it, part ID 3 (included in part set ID 1) is compensated after iteration 3 (ITER3), using method 8, with a scale factor of 0.5. The dynain files from the trimmed ITER3 are taken as the before state (see the BEFORE_SPRINGBACK option), and the dynain file from the springback calculation is taken as the after state. The desired blank shape is given by the dynain from the trimmed ITER0 output. The compensated shape is taken from the disp.tmp file of the last compensation run. The current tools are also from the last compensation iteration. The original tools are taken from the tooling mesh in ITER0. The mapped trim lines of the last compensation give the updated trim curves, geotrm.new.

```
*KEYWORD
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*INTERFACE_COMPENSATION_3D
$   METHOD        SL         SF      ELREF       PSID    UNDRCT     ANGLE   NLINEAR
         8    10.000      0.500          2          1         1       0.0         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
../7_iter3.dir/2_trim.dir/dynain
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
../7_iter3.dir/3_spbk.dir/dynain
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
../1_iter0.dir/2_trim.dir/dynain
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
../6_compensation.dir/disp.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
../6_compensation.dir/rigid.new
*INCLUDE_COMPENSATION_ORIGINAL_TOOLS
../1_iter0.dir/sim_forming_mesh.k
*INCLUDE_COMPENSATION_TRIM_CURVE
../6_compensation.dir/geotrm.new
*SET_PART_LIST
$      PSID
         1
$      PID
         3
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*END
```

Note that in an automatic compensation-loop calculation, input files disp.tmp, rigid.new, and geotrm.new, which are the default file names of the previous compensation, should not be in the same directory as the current compensation run, as these files *will* be overwritten.

13. **Updating the compensated tool after a small part shape change.** Often, a part undergoes a small shape change due to, for instance, a product change. If the shape change does not significantly alter the springback results, use the PART_CHANGE keyword option to update the compensated tools with the part mesh (inside the trim lines) or formed blank shape without going through another iterative compensation loop. This feature updates the compensated tool

shape within the specified MAXGAP. Changes to the geometry involving sharp corners and transitions without fillets are not permissible. A keyword example is provided below, where a maximum gap of 5 mm is specified between the original shape and the modified product shape. The updated part file name is updatepart.tmp, and the output file for the new rigid tool is newrigid.k.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D_PART_CHANGE
$   MAXGAP
       5.0
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
../1_iter0.dir/2_trim.dir/dynain
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
../6_compensation.dir/disp.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
../6_compensation.dir/rigid.new
*INCLUDE_COMPENSATION_UPDATED_BLANK_SHAPE
./updatedpart.tmp
*INCLUDE_COMPENSATION_UPDATED_RIGID_TOOL
$ file name to output the new rigid tools
./newrigid.k
*END
```

14. **Tool mesh refinement for better convergence.** The following keyword example refines the elements and aligns the nodes in the original tool mesh file, rnominaltools.k, at the provided trim curves, trimcurve106.k, defined using *DEFINE_CURVE_TRIM_3D. The refined rigid tool mesh is output as rigid_refined.tmp, which can be used to start the iterative springback compensation process. The aligned nodes are output to a file named bndnd0.tmp.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D_REFINE_RIGID
rnorminaltools.k
trimcurve106.k
*END
```

15. **Maintaining tangency.** To maintain tangency between the addendum and binder (method 7) in the draw die and tangential transitions immediately off the rigid tool at the trim curve area to the addendum, the variable TANGENT must be set to 1. In the keyword example below, the rigid_refined.tmp (from the keyword option REFINE_RIGID) is used as both the current and the original tool. The aligned nodes file bndnd0.tmp is included under *INCLUDE_COMPENSATION_TRIM_NODE, which is used for the tangency calculation. Note that the variable TANGENT replaces the keyword *INCLUDE_COMPENSATION_TANGENT_CONSTRAINT.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D
$   METHOD        SL        SF     ELREF      PSID    UNDRCT     ANGLE   NLINEAR
         7    10.000       1.0         2         1         0       0.0         1
$  TANGENT
         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
blank0A.tmp
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
spbkA.tmp
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
```

```
blank0A.tmp
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
blank0A.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
rigid_refined.tmp
*INCLUDE_COMPENSATION_ORIGINAL_TOOLS
rigid_refined.tmp
*INCLUDE_COMPENSATION_TRIM_NODE
bndnd0.tmp
*INCLUDE_COMPENSATION_TRIM_CURVE
../../1_iter0.dir/4_comp.dir/newtrmcurve.tmp
*INCLUDE_COMPENSATION_SYMMETRIC_LINES
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
     SYMID     SYMXY        X0        Y0
         1         2     -76.0       0.0
*SET_PART_LIST
$     PSID
         1
$      PID
         4
*END
```

16. **Compensation for the flanging process.** The FLANGE keyword option is used to compensate flanging tools for springback during the flanging process. The following keyword example defines flanging steels with PIDs 6 and 4 moving in the directions defined by the vectors [0.0,0.0, −1.0] and [0.0, −16.696, −5.674], respectively. The lower flanging post with part ID 2 is selected as the primary tool surface mesh, and the flanging steels are compensated while maintaining the necessary tool gaps as in the original tool mesh iter0tools.k. Note that in the compensation of the flanging process, an even tool gap everywhere is not required, so flanging steels do not have to follow the flanging post, for example, in the flanging steel radius area, etc.

```
*KEYWORD
*INTERFACE_COMPENSATION_3D
$   METHOD        SL        SF     ELREF      PSID    UNDRCT     ANGLE   NLINEAR
         8    10.000       1.0         2         1         0       0.0         1
*INCLUDE_COMPENSATION_BLANK_BEFORE_SPRINGBACK
blank0A.tmp
*INCLUDE_COMPENSATION_BLANK_AFTER_SPRINGBACK
spbkA.tmp
*INCLUDE_COMPENSATION_DESIRED_BLANK_SHAPE
blank0A.tmp
*INCLUDE_COMPENSATION_COMPENSATED_SHAPE
blank0A.tmp
*INCLUDE_COMPENSATION_CURRENT_TOOLS
iter0tools.k
*INCLUDE_COMPENSATION_ORIGINAL_TOOLS
iter0tools.k
*INTERFACE_COMPENSATION_3D_FLANGE
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
$      PID        VX        VY        VZ
         6         0         0       -1.
         4         0   -16.696    -5.674
*SET_PART_LIST
$     PSID
         1
$      PID
         2
*END
```

17. **Springack on a post.** Card 1a.3 is intended for the case where springback is performed on a post. It removes the initial gap between the initial blank and the post by using the provided information of the direction the blank would move to be on the post and the gap between them. If springback is performed based on single point constraints, this card is not needed.

*INTERFACE_COMPENSATION_TRIMMING_CURVE

Purpose: Update the original trimming curves based on the compensation from *INTERFACE_COMPENSATION_3D.

Card 1	1	2	3	4	5	6	7	8
Variable	OTRCNAME							
Type	A80							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	UTRCNAME							
Type	A80							
Default	none							

Card 3	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	DEPTH	IF3D			
Type	F	F	F	F	I			
Default	0.0	0.0	0.0	none	0			

VARIABLE	DESCRIPTION
OTRCNAME	File name for the original trimming curve
UTRNAME	File name for the updated trimming curve due to compensation
VX, VY, VZ	Components of the trimming vector for 2D trimming

VARIABLE	DESCRIPTION
DEPTH	For 2D trimming, depth of the trimming. For 3D, it is the minimum normal distance from the trimming curve to the blank element.
IF3D	Flag indicating whether the trimming is 2D or 3D: EQ.0: 2D trimming EQ.1: 3D trimming

*INTERFACE_COMPONENT_FILE

Purpose: Allow for the specification of the file where the component interface data should be written and the optional use of a new binary format for that data.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

Optional Card.

Card 2	1	2	3	4	5	6	7	8
Variable	FORMAT	NDOF						
Type	I	I						
Default	2	0						

VARIABLE	DESCRIPTION
FILENAME	Name of the file where the component data will be written
FORMAT	File format to use: EQ.1: Use old binary file format EQ.2: Use new LSDA file format
NDOF	Number of degrees of freedom per node that get written to the file: NE.3: Write as many as the model contains (3 or 6) EQ.3: Only write 3 even if the model has 6

Remarks:

If Z = is used on the command line, this card will be ignored. If this card is in effect, the new LSDA file format is the default format to be used. The new format has certain advantages and one possible drawback:

1. It allows for the use of the_TITLE modifier on all *INTERFACE_COMPONENT inputs so that subsequent *INTERFACE_LINKING cards can refer to components by a user-specified ID.

2. It is fully portable between machines with different precision and byte order.

3. It maintains the full precision of the coordinate vector. The internal coordinate vector has been in double precision for quite some time, even for single precision executables. The old binary format writes 32 bit data for single precision executables, losing some precision in the process.

4. Because of the maintained precision, the new format files will be significantly larger when running in single precision.

Of course, the new file format cannot be used for subsequent analysis with older versions of LS-DYNA, particularly those with a Product ID less than 50845. Executables that can read the new format for *INTERFACE_LINKING analysis automatically detect whether the new or old format is in use.

INTERFACE_COMPONENT_*OPTION1_{OPTION2}

Available values for OPTION1 include:

NODE

SEGMENT

SPH

OPTION2 only allows the value:

TITLE

Purpose: Create an interface for use in subsequent linking calculations. This command applies to the first analysis for storing interfaces in the interface file specified either by "Z=**isf1**" on the execution line or by the *INTERFACE_COMPONENT_FILE command. The output interval used to write data to the interface file is controlled by OPIFS on *CONTROL_OUTPUT. If OPIFS is not specified, the interval defaults to $1/10^{th}$ the value of DT specified in *DATABASE_BINARY_D3PLOT.

This capability allows the definition of interfaces that isolate critical components.

With the NODE and SEGMENT option, a database is created that records the motion of the interfaces. In later calculations the isolated components can be reanalyzed with arbitrarily refined meshes with the motion of their boundaries specified by the database created by this input. The interfaces defined here become the reference surfaces in the tied interface options.

With the SPH option, a database is created that records the FSI coupling forces of the interfaces. In later calculations the isolated components can be reanalyzed with arbitrarily refined meshes with external forces extracted from the database created by this input.

Each definition consists of a set of cards that define the interface. Interfaces may consists of a set of segments for later use with *INTERFACE_LINKING_SEGMENT, an ordered line of nodes for use with *INTERFACE_LINKING_EDGE, an unordered set of nodes for use with *INTERFACE_LINKING_NODE, or an unordered set of structure SPH particles for use with *INTERFACE_LINKING_FORCES.

Title Card. Additional card for TITLE keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	Title						
Type	I	A70						
Default	none	none						

VARIABLE	DESCRIPTION
ID	ID for this interface in the linking file
Title	Title for this interface

Card 2	1	2	3	4	5	6	7	8
Variable	SID	CID	NID					
Type	I	I	I					

VARIABLE	DESCRIPTION
SID	Set ID, see *SET_NODE (for NODE and SPH options) or *SET_-SEGMENT (for SEGMENT option).
CID	Coordinate system ID.
NID	Node ID.

Remarks:

1. **CID and NID.** CID and NID are optional. If CID appears, the transformation matrix for this coordinate system is written to the linking file at each output state. If NID appears, the displacement of this node is also written to the file. This information is then available to be used by the *INTEFACE_LINKING_-NODE_LOCAL. If either of these is non-zero, then the linking file will be written in the LSDA format, as the old format cannot support this optional output.

2. **Linking file format.** If the old-style binary format is used for the linking file (see *INTERFACE_COMPONENT_FILE), then the ID values are ignored and all components are numbered according to their input order, starting from 1.

3. **Dynamic relaxation.** *INTERFACE_COMPONENT_... and *INTERFACE_-LINKING_... are supported during dynamic relaxation. If both runs (meaning the one that creates the linking file and a subsequent run that uses the linking file) have dynamic relaxation, or neither does, then the linked interface follows the motion of the component interface through the whole simulation as expected. If only the first run has dynamic relaxation, the motion of the linked interface during the second run follows the ”dynamic relaxation converged” state of the first run. If only the second run has dynamic relaxation, then during dynamic relaxation, the linked interface does not move at all but is tied to the initial coordinates of the first run.

*INTERFACE_DE_HBOND

Purpose: Define the failure models for bonds linking various discrete element (DE) parts within one heterogeneous bond (*DEFINE_DE_HBOND).

Card 1	1	2	3	4	5	6	7	8
Variable	IID							
Type	I							
Default	none							

Bond Definition Cards. For each bond definition, include an additional card. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PTYPE1	PTYPE2	FRMDL	FRGK	FRGS	DMG
Type	I	I	I	I	I	F	F	F
Default	none	none	none	none	1	none	none	1.0

VARIABLE	DESCRIPTION
IID	Interface ID. All interfaces should have a unique ID
PID1	First part ID.
PID2	Second part ID. PID1 and PID2 define the bonds that this fracture model is applied to. There are three combinations as Case a: PID1.EQ.0 This is the default model for all bonds, overriding the default model defined in Card 1.1 of *DEFINE_DE_HBOND. Case b: PID1.GT.0 and PID2.EQ.0 This model is applied to the bonds within part PID1, instead of the default model. Case c: PID1.GT.0 and PID2.GT.0

VARIABLE	DESCRIPTION
	This model is applied to the bonds between parts PID1 and PID2 only, but not to those within part PID1 or part PID2 (as in case b).
	Notes: 1. The default fracture model is applied to all parts that are not specified in case b. 2. The fracture model of the part with a smaller part id is applied to the bonds between two different parts if not specified in case c.
PTYPE1	First part type: EQ.0: DES part set EQ.1: DES part
PTYPE2	Second part type: EQ.0: DES part set EQ.1: DES part
FRMDL	Fracture model. (same as FRMDL in Card 1.1 of keyword *DEFINE_DE_HBOND.)
FRGK	Fracture energy release rate for volumetric deformation. (same as FRGK in Card 1.1 of keyword *DEFINE_DE_HBOND.)
FRGS	Fracture energy release rate for shear deformation. (same as FRGS in Card 1.1 of keyword *DEFINE_DE_HBOND.)
DMG	Continuous damage development. (same as DMG in Card 1.1 of keyword *DEFINE_DE_HBOND.)

INTERFACE_DE_HBOND EXAMPLE:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  *INTERFACE_DE_HBOND
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$  Using DE_HBOND to bond 4 parts with different failure models
$
*SET_PART_LIST_TITLE
DES BOND PARTS
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid       da1       da2       da3       da4
         1
$      id1       id2       id3       id4
       101       102       103       104
*SET_PART_LIST_TITLE
DES HBOND SUB PARTS
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid       da1       da2       da3       da4
         2
$      id1       id2
       103       104
*INTERFACE_DE_HBOND
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      iid
         1
$     pid1      pid2    ptype1    ptype2     frmdl      frgk      frgs       dmg
       101         0         1         0         1     1.0E1     1.0E1       1.0
         2         0         0         0         1     1.0E3     1.0E3       1.0
       102       103         1         1         2     1.0E2     1.0E2       1.0
$
*DEFINE_DE_HBOND
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid     stype    bdform      idim
         1         2         2         3
$   pbk_sf    pbs_sf      frgk      frgs     bondr     alpha       dmg     frmdl
         1         1     1.0E0     1.0E0       2.5      0.01       1.0         1
$   precrk    cktype                itfid
         0         0                    1
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ the fracture models for the bonds within each part are determined as
$
$   pid1    pid2    frgk    comments
$    101     101   1.0E1    Line 1 in the interface card
$    102     102   1.0E0    default (defined in hbond card)
$    103     103   1.0E3    Line 2 in the interface card
$    104     104   1.0E3    Line 2 in the interface card
$
$ for the bonds between two parts:
$    101     102   1.0E1    taken from Part 101 (smaller part id)
$    101     103   1.0E1    taken from Part 101 (smaller part id)
$    101     104   1.0E1    taken from Part 101 (smaller part id)
$    102     103   1.0E2    Line 3 in the interface card
$    102     104   1.0E0    taken from Part 102 (smaller part id)
$    103     104   1.0E3    taken from Part 103 (smaller part id)
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

INTERFACE_LINKING_DISCRETE_NODE_*OPTION

Available options include:

NODE

SET

Purpose: Link node(s) to an interface in an existing interface file. This link applies to all element types. The interface file is specified using *INTERFACE_LINKING_FILE or by including "L=**filename**" on the execution line.

With this command, nodes in a node set must be given in the same order as they appear in the interface file. This restriction does not apply to the more recent keyword *INTERFACE_LINKING_NODE_…

Card 1	1	2	3	4	5	6	7	8
Variable	NID/NSID	IFID						
Type	I	I						

VARIABLE	DESCRIPTION
NID	Node ID or Node set ID to be moved by interface file (see *NODE or *SET_NODE)
IFID	Interface ID in interface file

*INTERFACE_LINKING_EDGE

Purpose: Link a series of nodes to an interface in an existing interface file. The interface file is specified using *INTERFACE_LINKING_FILE or by including "L=**filename**" on the execution line.

Card	1	2	3	4	5	6	7	8
Variable	NSID	IFID						
Type	I	I						

VARIABLE	DESCRIPTION
NSID	Node set ID to be moved by interface file.
IFID	Interface ID in interface file.

Remarks:

The set of nodes defined will be constrained to follow the movement of the interface IFID, which should correspond to a curve output on a previous analysis via *INTERFACE_-COMPONENT_NODE. The order of the nodes in the first analysis is important. The nodes, in the order specified in the first analysis, represent a curve. Each node in set NSID will be tied to the point on the curve nearest to its initial position, and then will be constrained to follow that point on the curve for the duration of the analysis. This option is intended to be used with beam or shell elements, as both translational and rotational degrees of freedom are constrained.

*INTERFACE_LINKING_FILE

Purpose: Allow for the specification of the file from which the component interface data should be read.

Card 1	1	2	3	4	5	6	7	8
Variable	Filename							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
FNAME	Name of the file from which the component data will be read

Remarks:

If L= is used on the command line, this card will be ignored. There is no option to specify the file format, as the file format is automatically detected.

*INTERFACE_LINKING_FORCES

Purpose: Link segments to an interface in an existing interface file. The interface file is specified using *INTERFACE_LINKING_FILE or by including "L=**filename**" on the execution line.

Segment Set ID Card. Include as many cards as desired. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	IFID						
Type	I	I						

VARIABLE	DESCRIPTION
SSID	Segment set on which to apply forces from interface file
IFID	Interface ID in interface file

Remarks:

The set of segments defined will receive external forces coming from the interface IFID, which should correspond to an interface output on a previous analysis using the *INTERFACE_COMPONENT_SPH keyword. Positions from the interface file will be projected to the closest element in the segment set, and forces will be interpolated to their corner nodes.

INTERFACE_LINKING_NODE_*OPTION

Available options include:

SET

LOCAL

SET_LOCAL

Purpose: Link nodes(s) to an interface in an existing interface file. This link applies to all element types. The interface file is specified using *INTERFACE_LINKING_FILE or by including "L=**filename**" on the execution line.

> **WARNING:** Nodes constrained by the linking file should not be assigned additional constraints.

Node/Set ID Card. Include as many cards as desired. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID/NSID	IFID	FX	FY	FZ	SPLINE	DEATH	
Type	I	I	I	I	I	I	F	

VARIABLE	DESCRIPTION
NID	Node ID or Node set ID to be moved by interface file. See *NODE or *SET_NODE.
IFID	Interface ID in interface file
FX	ID of a *DEFINE_FUNCTION which determines the x direction displacement scale factor. See Remarks.
FY	ID of a *DEFINE_FUNCTION which determines the y direction displacement scale factor. See Remarks.
FZ	ID of a *DEFINE_FUNCTION which determines the z direction displacement scale factor. See Remarks.
SPLINE	Interpolation option for recorded sample points: LE.1: Linear interpolation. This interpolation has continuity in displacements but jumps in velocity and acceleration. It

VARIABLE	DESCRIPTION
	may result in oscillatory behavior if there are insufficient sample points. EQ.2: Quadratic interpolation. This interpolation has continuity in velocities but jumps in accelerations. It exhibits intermediate smoothness. EQ.3: Cubic interpolation. This interpolation has continuity in accelerations. It is the smoothest level of interpolation.
DEATH	Death time for the linking interface. Specifying a death time that is less than the simulation time is intended for cases where a pulse or similar event is initiated using the linking interface. Note that if the interface linking file lasts for less time than the current simulation, then, after the simulation time exceeds the last time in the linking file, the motion is locked to the last recorded value.

Card 2. This card appears after Card 1 when the LOCAL keyword option is used.

Card 2	1	2	3	4	5	6	7	8
Variable	LCID	LNID	USEC	USEN				
Type	I	I	I	I				

VARIABLE	DESCRIPTION
LCID	Local coordinate system ID for transforming displacements.
LNID	Local node ID for transforming displacements
USEC	Flag to indicate the use of the coordinate system in the linking file during displacement transformation. See Remarks. EQ.0: Linking file coordinate system is ignored. EQ.1: Linking file coordinate system is used.
USEN	Flag to indicate the use of the node displacement in the linking file during displacement transformation. See Remarks. EQ.0: Node displacement is not used. EQ.1: Node displacement is used.

Remarks:

This keyword constrains a set of nodes to follow the displacement of the interface having ID IFID in the linking file. Note that usually *INTERFACE_COMPONENT_NODE generates the linking file in a previous analysis.

The order of the nodes is not important. Each node in set NID will be tied to the nearest node in IFID using a bucket sort during the initialization phase. Nodes not found are reported and subsequently not constrained. Translational degrees of freedom are constrained. If the constrained analysis has rotational degrees of freedom, then the rotation degrees of freedom will be likewise constrained, and the linking file *must* include rotational degrees of freedom.

The displacements in the linking file can be scaled upon input so that

$$\mathbf{u}_{\text{constrained}} = \begin{bmatrix} f_{\text{FX}}(\dots) & 0 & 0 \\ 0 & f_{\text{FY}}(\dots) & 0 \\ 0 & 0 & f_{\text{FZ}}(\dots) \end{bmatrix} \mathbf{u}_{\text{linking file}}$$

where $f_{\text{FX}}(\dots)$ is the *DEFINE_FUNCTION function having ID FX and so on. When a scaling function is not specified, the corresponding component is imported unscaled as if the scaling function had a constant value of unity. These functions may take either 0, 1, 3, or 4 input arguments. The functions FX, FY, and FZ may be different and they may take different numbers of arguments. The data passed into the scaling function depends on the number of arguments that the function takes and the possibilities can be broken down into four cases:

1. *0 variables*. A function taking no inputs is constant over space and time. LS-DYNA evaluates such a function at the start of the calculation and uses that value for the duration of the run.

2. *1 variables.* LS-DYNA passes in the simulation time at each step and the resulting value is applied to all nodes in the set.

3. *3 variables.* LS-DYNA passes in the *initial position* of each constrained node as an (x, y, z) triple at the start of the calculation and then uses the result for the duration of the run.

4. *4 variables*. LS-DYNA passes in the *current* simulation time and the *initial position* of each of the constrained nodes as an (x, y, z, t) tuple. This function is updated at each time step. Using scaling functions of 4 variables may result in a performance penalty as each function must be evaluated for every constrained node every cycle.

If time-dependent scaling functions are used, then the constrained nodes must start with coordinates identical to the constraining nodes in the linking file.

The LOCAL option and the values of the LCID, LNID, USEC, USEN flags, which was designed in conjunction with Honda R&D Co., Ltd., allow for the interface displacements to be transformed in various ways. By default, the scale factors FX, FY, and FZ act on the nodal displacements in the global coordinate system of the constrained calculation. This may be undesirable depending on how the global coordinate system of the linked calculation is defined. The most general transformation rule is:

$$\mathbf{u}_{\text{constrained}} = \mathbf{Q}_2 \begin{bmatrix} f_{\text{FX}}(\dots) & 0 & 0 \\ 0 & f_{\text{FY}}(\dots) & 0 \\ 0 & 0 & f_{\text{FZ}}(\dots) \end{bmatrix} \mathbf{Q_1}(\mathbf{u}_{\text{linked}} - \mathbf{c_1}) + \mathbf{c_2}$$

where,

$\mathbf{c}_1$ = The displacement of the NID node in the linking file
$\mathbf{c_2}$ = The displacement of node LNID
$\mathbf{Q}_1$ = Rotation into the local coordinates of the linking file
$\mathbf{Q}_2$ = Rotation into the local coordinate system; if unset the inverse of $\mathbf{Q}_1$

If USEC = 0, then $\mathbf{Q}_1$ is the identity rotation and any coordinate system in the linking file is ignored. If USEN = 0, then $\mathbf{c}_1$ is set to $\mathbf{0}$ and NID in the linking file is ignored.

*INTERFACE_LINKING_SEGMENT

Purpose: Link segments to an interface in an existing interface file. The interface file is specified using *INTERFACE_LINKING_FILE or by including "L=**filename**" on the execution line.

Segment Set ID Card. Include as many cards as desired. Input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	IFID	DEATH					
Type	I	I	F					

VARIABLE	DESCRIPTION
SSID	Segment set to be moved by interface file
IFID	Interface ID in interface file
DEATH	Death time for this interface. The default is infinity.

Remarks:

The set of segments defined will be constrained to follow the movement of the interface IFID, which should correspond to an interface output on a previous analysis using *INTERFACE_COMPONENT_SEGMENT. The behavior will be the same as if set SSID is the SURFA side of a *CONTACT_TIED_SURFACE_TO_SURFACE with IFID as the SURFB side. Translational movement will be constrained but not rotations. By specifying the DEATH parameter, it is possible to release the constraints in this interface at a given time. This is useful when the interface is intended to provide a short-term impulse to a system.

*INTERFACE_SPG_1

Purpose: Indicate that the current analysis is the first stage of a two-stage SPG analysis. This keyword does not affect the results of the SPG analysis.

The two-stage SPG analysis feature is available beginning with R13. Starting with R15, more internal variables other than the stresses and effective plastic strain can be transferred between stages by setting ISPGHIS = 1.

This card is optional.

Card 1	1	2	3	4	5	6	7	8
Variable	ISPGHIS							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ISPGHIS	Option to transfer more internal variables required by the material model other than the stresses and effective plastic strain: EQ.0: Only transfer the stresses and effective plastic strain. EQ.1: Include all other internal variables required by the material model.

Remarks:

1. **Output file.** When this keyword is included, LS-DYNA generates an ASCII file named 1234spg at normal termination. This file contains information (stress, strain, total displacement, etc.) about the SPG nodes at the end of the calculation.

2. **Dynain file for the second stage.** The second stage also requires a dynain file as input. Therefore, the first stage input file must include *INTERFACE_SPRINGBACK_LSDYNA to output this file. The part set for *INTERFACE_SPRINGBACK_LSDYNA must include the SPG parts since the SPG coordinates at termination are needed.

*INTERFACE_SPG_2

Purpose: Indicate the current analysis is the second stage of a two-stage SPG analysis.

The two-stage SPG analysis is only available for versions R13 and later. Starting with R15, more internal variables other than the stresses and effective plastic strain can be transferred between stages by setting ISPGHIS = 1.

This card is optional.

Card 1	1	2	3	4	5	6	7	8
Variable	ISPGHIS							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ISPGHIS	Option to read more internal variables other than the stresses and effective plastic strain if ISPGHIS = 1 in the first stage: EQ.0: Read only the stresses and effective plastic strain. EQ.1: Read all other internal variables required by the material model. The output file from the first stage only includes these internal history variables if ISPGHIS = 1 on *INTERFACE_SPG_1.

Remarks:

To perform a second stage analysis, the ASCII file 1234spg generated in the first stage calculation must be copied to the current folder and renamed as 1234spg0. The dynain file generated in the first stage calculation must also be included in the input deck for the second stage (see *INCLUDE_*{OPTION}*). The element connectivity and nodal coordinates for the second stage are taken from the dynain file.

***INTERFACE_SPRINGBACK**_*OPTION1_{OPTION2}*

Available options included for *OPTION1* are:

LSDYNA

NASTRAN

SEAMLESS

EXCLUDE (see Remark 9)

and for *OPTION2:*

<BLANK>

NOTHICKNESS (see Remark 1)

Purpose: This keyword causes LS-DYNA to, upon termination, write the final state of the system to an output file so that the results can be used as input for other subsequent calculations, such as trimming, springback, flanging, and hemming. Also, if a sense switch "swb", "swc", or "swd" is issued while the analysis is in progress, the current state will be written to an output file (see "SENSE SWITCH CONTROLS" in the Getting Started section for details). In addition to the analysis results, the output can optionally contain additional nodal constraints. The keyword name, which contains SPRINGBACK, is due to historical reasons as this feature is not limited to springback applications. All the options, except SEAMLESS, control the output of the forming information.

With the SEAMLESS option, LS-DYNA automatically runs the springback analysis after the termination time of the explicit or implicit dynamic analysis. See Remarks for Seamless Springback for more details.

Card Summary:

Card 1. This card is required.

PSID	NHSV	FTYPE		FTENSR	NTHHSV	RFLAG	INTSTRN

Card 2. This card is optional. For LS-DYNA to interpret the card as this card, include the string "OPTCARD" in the first column. Otherwise, the input reader assumes the second card is Card 4.

OPTC	SLDO	NCYC	FSPLIT	NDFLAG	CFLAG	HFLAG	

Card 3.1. This card is optional. To include it, Card 2 and 3.2 must also be included. For LS-DYNA to interpret the card as this card, include the string "OPTCARD" in the first column. Otherwise, the input reader assumes the third card is Card 4.

OPTC	DTWRT						

Card 3.2. This card is optional. It must be included if using Card 3.1. For LS-DYNA to interpret the card as this card, include the string "OPTCARD" in the first column.

OPTC	NMWRT	IVFLG					

Card 4. Include as many instantiations of this card as needed. The next keyword ("*") card terminates this input.

NID	TC	RC					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	NHSV	FTYPE		FTENSR	NTHHSV	RFLAG	INTSTRN
Type	I	I	I		I	I	I	I

VARIABLE	DESCRIPTION
PSID	Part set ID. See *SET_PART.
NHSV	Number of shell or solid history variables (beyond the six stresses and effective plastic strain) to be initialized in the interface file. For solids, LS-DYNA writes one additional state variable (initial volume). If NHSV is nonzero, the element formulations, unit system, and constitutive models should not change between runs. If NHSV exceeds the number of integration point history variables required by the constitutive model, LS-DYNA only writes the number required. Thus, if in doubt, set NHSV to a large number, such as 100.
FTYPE	File type (see Remark 6): EQ.0: ASCII EQ.1: Binary EQ.2: Both ASCII and binary.

VARIABLE	DESCRIPTION
	EQ.3: LSDA format (for LSDYNA option only) EQ.10: ASCII large format (see *INITIAL_STRESS_SHELL) EQ.11: Binary large format EQ.12: Both ASCII and binary large format
FTENSR	Flag for dumping tensor data from the element history variables into the dynain file. EQ.0: Don't dump tensor data from element history variables EQ.1: Dump any tensor data from element history variables into the dynain file in GLOBAL coordinate system. Currently, only Material 190 supports this option.
NTHHSV	Number of thermal history variables.
RFLAG	Flag to carry over reference quantities, such as the reference geometry for hyperelastic materials: EQ.0: Default, do not output. EQ.1: Output reference coordinates and nodal masses.
INTSTRN	Output of strains for shells; see also *INITIAL_STRAIN_SHELL. EQ.0: Only at the innermost and outermost integration points, EQ.1: At all through-thickness integration points.

Irregular Optional Card 1. The keyword reader will interpret the card following Card 1 as optional Card 2 if the first column of the card is occupied by the string "OPTCARD". Otherwise, it is interpreted as the first Node Card; see below.

Card 2	1	2	3	4	5	6	7	8
Variable	OPTC	SLDO	NCYC	FSPLIT	NDFLAG	CFLAG	HFLAG	
Type	A	I	I	I	I	I	I	

VARIABLE	DESCRIPTION
SLDO	Output of solid element data as EQ.0: *ELEMENT_SOLID, or EQ.1: *ELEMENT_SOLID_ORTHO (only for anisotropic

VARIABLE	DESCRIPTION
	material).
NCYC	Number of process cycles this simulation corresponds to in the simulation of wear processes; see Remark 7.
FSPLIT	Flag for splitting of the dynain file (only for ASCII format): EQ.0: dynain file written in one piece. EQ.1: Output is divided into two files: dynain_geo including the geometry data and dynain_ini including initial stresses and strains.
NDFLAG	Flag to dump nodes into dynain file: EQ.0: Default, dump only SPH and element nodes EQ.1: Dump all nodes
CFLAG	Output contact state: EQ.0: Default, do not output. EQ.1: Output contact state, currently only Mortar segment pair information and selected tied contacts with restrictions; see Remark 8.
HFLAG	Output hourglass state, only valid for FTYPE = 3: EQ.0: Default, do not output. EQ.1: Output hourglass stresses for carrying over to the next simulation.

Irregular Optional Card 2. This card must have the string "OPTCARD" in the first column for LS-DYNA to interpret it as an optional card instead of the first Node Card.

Card 3.1	1	2	3	4	5	6	7	8
Variable	OPTC	DTWRT						
Type	A	F						

Irregular Optional Card 3. This card must have the string "OPTCARD" in the first column for LS-DYNA to interpret it as an optional card instead of the first Node Card.

Card 3.2	1	2	3	4	5	6	7	8
Variable	OPTC	NMWRT	IVFLG					
Type	A	I	I					

VARIABLE	DESCRIPTION
DTWRT	Time interval between two consecutive writes of the dynain.lsda file.
NMWRT	Maximum number of dynain.lsda files created. To save space, LS-DYNA periodically overwrites these files.
IVFLG	Flag to *not* write out the initial volume for solid elements. See IVEFLG on *INITIAL_STRESS_SOLID. EQ.0: IVEFLG = 1 for the written *INITIAL_STRESS_SOLID keywords. The output file contains the initial volume. EQ.1: IVEFLG = 0 for the written *INITIAL_STRESS_SOLID keywords. The output file does not contain the initial volume. .

Node Cards. Define a list of nodal points and constraints. For the SEAMLESS Option the constraints are used for the springback analysis. This input ends at the next keyword ("*") card.

Card 4	1	2	3	4	5	6	7	8
Variable	NID	TC	RC					
Type	I	F	F					

VARIABLE	DESCRIPTION
NID	Node ID, see *NODE.
TC	Translational Constraint: EQ.0: No constraints

VARIABLE	DESCRIPTION
	EQ.1: Constrained x displacement
	EQ.2: Constrained y displacement
	EQ.3: Constrained z displacement
	EQ.4: Constrained x and y displacements
	EQ.5: Constrained y and z displacements
	EQ.6: Constrained z and x displacements
	EQ.7: Constrained x, y, and z displacements
RC	Rotational constraint:
	EQ.0: No constraints
	EQ.1: Constrained x rotation
	EQ.2: Constrained y rotation
	EQ.3: Constrained z rotation
	EQ.4: Constrained x and y rotations
	EQ.5: Constrained y and z rotations
	EQ.6: Constrained z and x rotations
	EQ.7: Constrained x, y, and z rotations

Remarks:

1. **NOTHICKNESS option.** The NOTHICKNESS option is available when the keyword's first option is either LSDYNA or NASTRAN. With the NOTHICKNESS option the shell element thickness is not output.

2. **Filenames.** The file name for the LSDYNA option is dynain and for NASTRAN is nastin.

3. **Trimming.** Trimming is available for the adaptive mesh, but it requires manual intervention. To trim an adaptive mesh, use the following procedure:

   a) Generate the file, dynain, using the keyword *INTERFACE_SPRINGBACK_LSDYNA.

   b) Prepare a new input deck including the dynain file.

   c) Add the keyword *ELEMENT_TRIM to this new deck.

d) Add the keyword *DEFINE_CURVE_TRIM to this new deck.

e) Run this new input deck with i=**input_file_name**. The adaptive constraints are eliminated by remeshing and the trimming is performed.

f) In case this new trimmed mesh is needed, run a zero termination time job and output the file generated using the keyword, *INTERFACE_SPRINGBACK_LSDYNA.

4. **Temperature.** The file new_temp_ic.inc will be created for a thermal solution and a coupled thermal-mechanical solution. The file new_temp_ic.inc is a KEYWORD include file containing *new temperature initial conditions* for the nodes belonging to the PSID.

   a) For thermal user materials it is possible to dump thermal history variables. See the NTHHSV field.

5. **Velocity.** Nodal velocities from the final state are included in the dynain file when using the LSDA format (FTYPE = 3), or during a staged construction analysis (see *CONTROL_STAGED_CONSTRUCTION), but are otherwise not output.

6. **FTYPE.** The choice of format size in field FTYPE is only available for shell stresses and shell history data; see parameter LARGE on *INITIAL_STRESS_SHELL. For solid and beam elements, the large format is always written to dynain; that is LARGE is automatically set to 1 on *INITIAL_STRESS_SOLID and *INITIAL_STRESS_BEAM, respectively.

   When you include a dynain file generated with a single precision executable in a subsequent double precision analysis, or vice versa, the dynain file should be in either ASCII or LSDA format. The binary file format is not compatible between single and double precision executables.

7. **NCYC.** When simulating wear processes, this represents the number of process cycles this particular simulation corresponds to and *INITIAL_CONTACT_WEAR cards are generated accordingly in the dynain file (only ASCII format supported). Cards will only be generated for nodes in contact interfaces associated with a *CONTACT_ADD_WEAR, and having SAPR or SBPR set to 2 on the first card on *CONTACT. This wear data is in a subsequent simulation accounted for NCYC times when modifying the worn geometry, or alternatively processed in LS-PrePost.

8. **CFLAG.** Transferring contact state, particularly tied information, may be important for multistage simulations. Setting CFLAG to 1 will output Mortar contact segment pairs for sliding, tied, tiebreak, and tied weld options. The resulting dynain.lsda, which must use the LSDA format (FTYPE = 3), can be

included in a subsequent simulation to restore the contact state. The LS-DYNA version or core count as wells as the contact type (for instance changing from tied weld to simple tied) may be changed between simulations if the contact ID and node IDs making up the segments remain unchanged. If parts or elements are removed between simulations and these happen to contain segments that are in the dynain.lsda file, these pairs will be simply ignored. This option provides a flexibility that is appealing for multistage processes. For sliding contact, typically the ignore and friction state is transferred. Concerning non-Mortar tied contacts, the contacts recommended along with the implicit accuracy option, IACC on *CONTROL_ACCURACY, are supported for implicit-implicit transitions and non-groupable option. More specifically, the supported contact formulations are

*CONTACT_TIED_NODES_TO_SURFACE

*CONTACT_TIED_NODES_TO_SURFACE_OFFSET

*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET

*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET

9. **EXCLUDE.** This option is used to limit what data will be output to the LSDYNA dynain file. The input format is completely different and consists of any number of keyword cards WITHOUT the leading *. These cards and their associated data will not be output. For example:

```
*INTERFACE_SPRINGBACK_EXCLUDE
BOUNDARY_SPC_NODE
CONSTRAINED_ADAPTIVITY
```

would output all the normal dynain data, except for the SPC and adaptive constraints. The currently recognized keywords that can be excluded are:

BOUNDARY_SLIDING_PLANE

BOUNDARY_SPC_NODE

CONSTRAINED_ADAPTIVITY

DEFINE_COORDINATE_NODES

DEFINE_COORDINATE_VECTOR

ELEMENT_BEAM

ELEMENT_SHELL

ELEMENT_SOLID

INITIAL_STRAIN_SHELL

INITIAL_STRAIN_SOLID

INITIAL_STRESS_BEAM

INITIAL_STRESS_SHELL

INITIAL_STRESS_SOLID

INITIAL_TEMPERATURE_NODE

INITIAL_VELOCITY_NODE

NODE

REFERENCE_GEOMETRY

Remarks for Seamless Springback:

When seamless springback is invoked, the solution automatically and seamlessly switches from explicit or implicit dynamic to implicit static mode at the termination time and continues to run the static springback analysis. Seamless springback can be activated in the original LS-DYNA input deck or later using a small restart input deck. In this way, the user can decide to continue a previous analysis by restarting to add the implicit springback phase. (Another alternative approach to springback simulation is to use the keyword *INTERFACE_SPRINGBACK_LSDYNA to generate a dynain file after forming, and then perform a second simulation running LS-DYNA in fully implicit mode for springback. See Appendix P for a description of how to run an implicit analysis using LS-DYNA.)

The implicit springback phase begins when the forming simulation termination time ENDTIM is reached, as specified with the keyword *CONTROL_TERMINATION. Since the springback phase is static, its termination time can be chosen arbitrarily (unless material rate effects are included). The default choice is 2.0 × ENDTIM and can be changed using the *CONTROL_IMPLICIT_GENERAL keyword; see fields DT0 and NSBS.

Since the springback analysis is a static simulation, a minimum number of essential boundary conditions or Single Point Constraints (SPC's) can be input to prohibit rigid body motion of the part. These boundary conditions can be added for the springback phase using the input option on the *INTERFACE_SPRINGBACK_SEAMLESS keyword above.

If no boundary conditions are added with the SEAMLESS option, an eigenvalue computation is automatically performed using the Inertia Relief Option to find any rigid body modes and then automatically constrain them out of the springback simulation (see

*CONTROL_IMPLICIT_INERTIA_RELIEF). This approach introduces no artificial deformation and is recommended for many simulations.

An "SPR" option is available for several *CONTROL_IMPLICIT keywords to further control the implicit springback phase. Generally, default settings can be used, in which case the SPR option for *CONTROL_IMPLICIT keywords is not necessary.

To obtain accurate springback solutions, a nonlinear springback analysis must be performed. In many simulations, this iterative equilibrium search will converge without difficulty. If the springback simulation is particularly difficult, either due to nonlinear deformation, nonlinear material response, or numerical precision errors, a multi-step springback simulation will be automatically invoked. In this approach, the springback deformation is divided into several smaller, more manageable steps.

Two specialized features in LS-DYNA are used to perform multi-step springback analyses. The addition and gradual removal of artificial springs is performed by the artificial stabilization feature. Simultaneously, the automatic time step control is used to guide the solution to the termination time as quickly as possible, and to persistently retry steps where the equilibrium search has failed. By default, both of these features are active during a seamless springback simulation. However, the default method attempts to solve the springback problem in a single step. If this is successful, the solution will terminate normally. If the single step springback analysis fails to converge, the step size will be reduced, and artificial stabilization will become active. Defaults for these features can be changed using the following keywords:

- *CONTROL_IMPLICIT_GENERAL,
- *CONTROL_IMPLICIT_AUTO, and
- *CONTROL_IMPLICIT_STABILIZATION.

*INTERFACE_SSI_{OPTION}_ID

Purpose: This card creates a tied-contact soil-structure interface for use in a transient analysis of a soil-structure system subjected to earthquake excitation. This card allows the analysis to start from a static state of the structure, as well as to read in ground motions recorded on the interface in an earlier analysis.

Available options are:

<BLANK>

OFFSET

CONSTRAINED_OFFSET

LS-DYNA implements the effective seismic input method [Bielak and Christiano (1984)] for modeling the interaction of a nonlinear structure with a linear soil foundation subjected to earthquake excitation. Note that any nonlinear portion of the soil near the structure may be incorporated with the structure into a larger generalized structure, but the soil is assumed to behave linearly beyond a certain distance from the structure.

The effective seismic input method couples the dynamic scattered motion in the soil, which is the difference between the motion in the presence of the structure and the free-field motion in its absence, with the total motion of the structure. This replaces the distant earthquake source with equivalent effective forces adjacent to the soil-structure interface and allows truncation of the large soil domain using a non-reflecting boundary (e.g. *MAT_PML_ELASTIC) to avoid unnecessary computation. These effective forces can be computed using the free-field ground motion at the soil-structure interface, thus avoiding deconvolution of the free-field motion down to depth.

Nonlinear behavior of the structure may be modeled by first carrying out a static analysis of the soil-structure system, and then carrying out the transient analysis with only the structure initialized to its static state. Because the transient analysis employs the dynamic scattered motion in the soil, the soil cannot have any static loads only it – only the structure is subjected to static forces. Consequently, the structure must be supported by the static reactions at the soil-structure interface. Additionally, the soil nodes at the interface must be initialized to be compatible with the initial static displacement of the structure. LS-DYNA will do these automatically if the soil-structure interface is identified appropriately in the static analysis and reproduced in the transient analysis.

Thus, soil-structure interaction analysis under earthquake excitation may be carried out in LS-DYNA as follows:

1. Carry out a static analysis of the soil-structure system (e.g. using dynamic relaxation; see *CONTROL_DYNAMIC_RELAXATION), with the soil-structure interface identified using *INTERFACE_SSI_STATIC_ID

Optionally, carry out a free-field analysis to record free-field motions on the future soil-structure interface, using either *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED, for surface-supported or embedded structures respectively.

2. Carry out the transient analysis as a full-deck restart job (see *RESTART), with only the structure initialized to its static stress state (see *STRESS_INITIALIZATION), and the same soil-structure interface identified using *INTERFACE_SSI_ID with the same ID as in static analysis:

   a) The structure mesh must be identical to the one used for static analysis.

   b) The soil mesh is expected to be different from the one used for static analysis, especially because non-reflecting boundary models may be used for transient analysis.

   c) The meshes for the structure and the soil need not match at the interface.

   d) Only the structure must be subjected to static loads, via *LOAD_BODY_PARTS

   e) The earthquake ground motion is specified using *LOAD_SEISMIC_SSI:

      i) The NODE, SET, and POINT options allow specifying the ground motion at particular nodes, node sets, or coordinate points, respectively, on the soil-structure interface.

      ii) The AUX option employs motions recorded from a previous analysis using *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED.

      iii) The DECONV option takes a motion specified at or below the top soild surface of the model and uses the SHAKE algorithm to deconvolve it to all points on the interface.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	STRID	SOILID	STRPR	SOILPR				
Type	I	I	I	I				
Default	none	none	0	0				

Optional card (superseded by LOAD_SEISMIC_SSI_AUX; see Remark 6)

Card 3	1	2	3	4	5	6	7	8
Variable	GMSET	SF	BIRTH	DEATH	MEMGM			
Type	I	F	F	F	I			
Default	none	1.	0.	10^{28}	2500000			

VARIABLE	DESCRIPTION
ID	Soil-structure interface ID. This is required and must be unique amongst all the contact interface IDs in the model.
HEADING	A descriptor for the given ID.
STRID	Segment set ID of base of structure at soil-structure interface.
SOILID	Segment set ID of soil at soil-structure interface.
STRPR	Include the structure side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files: EQ.1: Structure side forces included. Note that for ncforc the forces for this side will be listed under surfa.
SOILPR	Include the soil side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files: EQ.1: Soil side forces included. Note that for ncforc the forces for this side will be under surfb.
GMSET	Identifier for set of recorded motions from *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED

VARIABLE	DESCRIPTION
SF	Recorded motion scale factor.
BIRTH	Time at which specified recorded motion is activated.
DEATH	Time at which specified recorded motion is removed: EQ.0.0: Default set to 10^{28}
MEMGM	Size in words of buffer allocated to read in recorded motions

Remarks:

1. **Tied Contact Interface.** A tied contact interface (*CONTACT_TIED_SURFACE_TO_SURFACE) is created between the structure and the soil using the specified segment sets, with the soil segment set as the reference surface and the structure segment set as the tracked surface. Naturally, the two segment sets should not have merged nodes and can be non-matching in general. However, the area covered by the two surfaces should match.

2. **Keyword Options.** The options OFFSET and CONSTRAINED_OFFSET create the corresponding tied surface-to-surface contact interface.

3. **Interface ID.** The soil-structure interface ID is assigned as the ID of the generated contact interface.

4. **Soil Segment Set Orientation.** It is assumed that the soil segment set is oriented toward the structure.

5. **Multiple Soil-Structure Interfaces.** Multiple soil-structure interfaces are allowed, such as for bridge analysis.

6. **Obsolete Card.** Optional Card 3 has been superseded by *LOAD_SEISMIC_SSI_AUX, which allows for reading multiple ground motion files in the same model.

7. **Reading Recorded Motions.** The recorded motions are read in from a binary file named gmbin by default, but a different filename may be chosen using the option GMINP on the command line (see Getting Started, Execution Syntax).

8. **Recorded Motions.** If the motions from *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED were recorded on a segment set, then the free-field motions on each node in the soil segment set of the soil-structure interface are calculated from the nearest segment of the segment set used to record the motions. If, however, the motions were recorded on a node set, then the motions

on the soil segment set nodes are found by interpolation as is done for *LOAD_-SEISMIC_SSI.

***INTERFACE_SSI_AUX**_*{OPTION}*

Available options are:

<BLANK>

NODE

Purpose: This card records the motion at a free surface, or on a set of nodes on a free surface, for the purpose of using the recorded motion as a free-field motion in a subsequent interaction analysis using *INTERFACE_SSI. By default, this card records motions on a segment set defining a surface, but can record motions on a node set using the option NODE. Only one of *INTERFACE_SSI_AUX and *INTERFACE_SSI_AUX_EMBEDDED is to be used for a particular soil-structure interface.

Card 1	1	2	3	4	5	6	7	8
Variable	GMSET	SETID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
GMSET	Identifier for this set of recorded motions to be referred to in *INTERFACE_SSI. Must be unique.
SETID	Segment set or node set ID where motions are to be recorded.

Remarks:

1. **Output File.** The motions on the specified segment set or node set is recorded in a binary file named gmbin by default, but a different filename may be chosen using option GMOUT on the command line (see Getting Started, Execution Syntax).

2. **Output Interval.** The output interval for the motions may be specified using the parameter GMDT on the *CONTROL_OUTPUT card, with the default value being one tenth of the output interval for d3plot states.

***INTERFACE_SSI_AUX_EMBEDDED**_*{OPTION1}*_*{OPTION2}*

Purpose: This card creates a tied-contact interface and records the motions and contact forces. The recorded data can then be used as free-field motion and reactions in a subsequent soil-structure interaction analysis using *INTERFACE_SSI, where the structure is embedded in the soil after part of the soil has been excavated. Only one of *INTERFACE_SSI_AUX and *INTERFACE_SSI_AUX_EMBEDDED is to be used for a particular soil-structure interface.

Available options for *OPTION1* are:

<BLANK>

OFFSET

CONSTRAINED_OFFSET

OPTION2 allows an optional ID to be given:

ID

ID Card. Additional card for ID keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	GMSET	STRID	SOILID	STRPR	SOILPR			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

VARIABLE	DESCRIPTION
ID	Soil-structure interface ID. This is required and must be unique amongst all the contact interface IDs in the model.
HEADING	A descriptor for the given ID.

VARIABLE	DESCRIPTION
GMSET	Identifier for this set of recorded motions to be referred to in *INTERFACE_SSI. Must be unique.
STRID	Segment set ID at base of soil to be excavated
SOILID	Segment set ID at face of rest of the soil domain.
STRPR	Include the structure side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files: EQ.1: Structure side forces included. Note that for ncforc the forces for this side will be listed under surfa.
SOILPR	Include the soil side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files: EQ.1: Soil side forces included. Note that for ncforc the forces for this side will be listed under surfb.

Remarks:

1. **Output Binary File.** The motions on the specified segment set or node set is recorded in a binary file named gmbin by default, but a different filename may be chosen using the option GMOUT on the command line (see Getting Started, Execution Syntax).

2. **Output Interval.** The output interval for the motions may be specified using the parameter GMDT on the *CONTROL_OUTPUT card, with the default value being one tenth of the output interval for d3plot states.

*INTERFACE_SSI_STATIC_{OPTION}_ID

Purpose: This card creates a tied-contact soil-structure interface in order to record the static reactions at the base of the structure, which are to be used in a subsequent dynamic analysis of the soil-structure system subjected to earthquake excitation. This card is intended to be used with the initial static analysis of the structure subjected to gravity loads.

Available options are:

<BLANK>

OFFSET

CONSTRAINED_OFFSET

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	STRID	SOILID	STRPR	SOILPR				
Type	I	I	I	I				
Default	none	none	0	0				

VARIABLE	DESCRIPTION
ID	Soil-structure interface ID. This is required and must be unique amongst all the contact interface IDs in the model.
HEADING	A descriptor for the given ID.
STRID	Segment set ID of base of structure at soil-structure interface.
SOILID	Segment set ID of soil at soil-structure interface.
STRPR	Include the structure side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files:

VARIABLE	DESCRIPTION
	EQ.1: Structure side forces included. Note that for ncforc the forces for this side will be listed under surfa.
SOILPR	Include the soil side in the *DATABASE_NCFORC and the *DATABASE_BINARY_INTFOR interface force files: EQ.1: Soil side forces included. Note that for ncforc the forces for this side will be listed under surfb.

Remarks:

See *INTERFACE_SSI_ID. The ID used for a particular interface in the static analysis must also be used for the same interface identified using *INTERFACE_SSI_ID during dynamic analysis.

*INTERFACE_THICKNESS_CHANGE_COMPENSATION

Purpose: Change the original tool gap between the upper and lower tools to variable gaps based on the formed blank thickness from a forming simulation. *This keyword must be used with* *INCLUDE_COMPENSATION_CURRENT_TOOLS *and* *INCLUDE_COMPENSATION_BEFORE_SPRINGBACK. This feature is useful for hot-stamping simulations, where physical contact between the upper/lower tools and the sheet blank is required to ensure proper cooling rates. The rigid tools with the variable gaps will be output to the file rigid.new.

Compensation Card.

Card 1	1	2	3	4	5	6	7	8
Variable	IFLG	THK0						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
IFLG	Activation flag; set to "1" to invoke the option.
THK0	Initial thickness of the sheet blank

Remarks:

1. **Overview.** This keyword modifies the gap between tools due to the results of a forming simulation. *INCLUDE_COMPENSATION_CURRENT_TOOLS and *INCLUDE_COMPENSATION_BEFORE_SPRINGBACK are used to include the rigid tools and formed sheet blank files, respectively. The first file contains a set of upper and lower tools with the original gap. This gap is then modified to the formed blank thickness from a forming simulation provided in the second file, a dynain format file. The forming results from the dynain file must include the thicknesses of the formed sheet blank.

2. **Orientation of surfaces.** For each tool, the surface normals must be consistent and must point towards the blank.

3. **Position.** The rigid tools and formed blank must all be in their punch home positions.

4. **Wrinkles.** To capture detailed tool-gap changes due to wrinkles, the rigid-body mesh size must be of the same order as the wrinkle wavelength.

Example:

A keyword example is provided below. The initial blank thickness is 1.2 mm, the original tool is provided in rigid.inc, and the file drawn.dynain contains the formed blank with thickness information.

```
*KEYWORD
*INTERFACE_THICKNESS_CHANGE_COMPENSATION
$     IFLG      THK0
         1       1.2
*INCLUDE_COMPENSATION_CURRENT_TOOLS
rigid.inc
*INCLUDE_COMPENSATION_BEFORE_SPRINGBACK
drawn.dynain
```

*INTERFACE_WELDLINE_DEVELOPMENT

Purpose: This keyword causes LS-DYNA to run a weld line development calculation instead of a finite element calculation. The input for this feature consists of (1) the formed blank from a completed metal forming simulation, (2) the corresponding initial blank, and (3) depending on which on desired whether the desired weld curve is on the formed or initial blank, the weld curve on the initial blank or on the final blank, respectively. Outputs include predicted final weld curve if input is the initial weld curve, or initial weld curve if input is the final welding curve; also nodes of any element edges that intersect the weld curve on the initial blank will be output to a file name affectednd_i.ibo and on the final blank to a file name affectednd_f.ibo.

Three additional keywords must be used together (and exclusively) with this keyword. They are: *INCLUDE_WD_INITIAL_BLANK, *INCLUDE_WD_FINAL_PART, and *INCLUDE_WD_WELDING_CURVE. See Remark 1 for a discussion of input order.

> **NOTE:** When this card is present, LS-DYNA *does not* proceed to the finite element simulation.

Development Parameter Card.

Card 1	1	2	3	4	5	6	7	8
Variable	IOPTION							
Type	I							
Default	1							

VARIABLE	DESCRIPTION
IOPTION	Welding curve development options: EQ.1: Calculate initial weld curve from final (given) weld curve, with output file name weldline.ibo, which will be on the initial blank mesh. EQ.-1: Calculate final weld curve from initial weld curve, with output file name weldline_f.ibo, which will be on the formed blank mesh.

Remarks:

1. **Input Order.** The input order of the keywords for a weld line development calculation is sensitive and must be in the order that follows:

```
*Keyword
*INTERFACE_WELDLINE_DEVELOPMENT
$  IOPTION
         1
*INCLUDE_WD_INITIAL_PART
$     filename
      initial.k
*INCLUDE_WD_FINAL_PART
$     filename
      final.k
*INCLUDE_WD_WELDING_CURVE
$     filename
      welding_curve.k
```

2. **Purpose of Weld Line Development Calculation.** For metal forming of tailor welded blanks, an initial straight weld line could become a curve on the formed part. The amount of deviation of the formed weld curve from its initial line depends on the part shape and forming conditions. Sometimes the formed weld curve is not desirable, thus a correction to the initial weld curve is needed. This keyword allows for determining the final weld curve given the initial weld curve and vice-versa, without performing another forming simulation.

3. **Mesh Adaptivity.** A mesh with adaptivity for the initial blank and final part is supported.

4. **Final (formed) Weld Curve.** The final formed weld curve should be projected onto the final blank mesh if it does not exactly lie on the mesh surface. This can be done with LS-PrePost4.2 via the menu option *GeoTol → Project → Project*, select *Closest Projection*, select *Project to Elements*, then define the destination mesh and source curves, and hit *Apply*. Sometimes the target curve may need enough points before projection; the points may be added via menu option *Curve → Spline → Method (Respace) → by number*. To write the curve out in *DEFINE_CURVE_TRIM_3D format, use *Curve → Convert → Method (To DEFINE_CURVE_TRIM) → To Key*, then write out the keyword using *FILE → Save Keyword*.

5. **Computed Weld Curve (and IGES).** Computed weld curves are written with a *DEFINE_CURVE_TRIM_3D keyword into a file called weldline.ibo (or, weldline_f.ibo, depending on IOPTION). The format of this file follows the keyword's specification. LS-PrePost4.0 can convert the computed curve to IGES; see the procedure in the manual pages for *INTERFACE_BLANKSIZE_DEVELOPMENT. After hitting Apply, the curves will show up in the graphics window, and *File → Save as → Save Geom as* can be used to write the curves out in IGES format.

Example:

As shown in Figure 30-41, given the initial (initialblank.k) and final blank (finalblank.k) configuration and a final formed weld curve (finalweldingcurve.k), the following input calculates a new initial weld curve on the initial blank. In this case, the final weld curve is specified as straight in the drawn panel.

```
*KEYWORD
*INTERFACE_WELDLINE_DEVELOPMENT
$OPTION
1
*INITIAL_BLANK
initialblank.k
*FINAL_PART
finalblank.k
*WELDING_CURVE
finalweldingcurve.k
*END
```

The output is the initial weld curve in the file weldline.ibo, Figure 30-41. Nodes of element edges that intersect the initial weld curve are output in affectednd_i.ibo; while nodes of element edges that intersect the final formed weld curve are output in affectednd_f.ibo, Figure 30-42.

To verify the predicted weld curve, the initial blank can be re-meshed according to the curve. A draw simulation can be performed again to confirm the final weld curve as straight, Figure 30-43.

Likewise, if given an initial weld curve (initialweldingcurve.k) and a final weld curve (weldline_f.ibo) can be calculated with the keyword inputs below:

```
*KEYWORD
*INTERFACE_WELDLINE_DEVELOPMENT
$OPTION
-1
*INITIAL_BLANK
initialblank.k
*FINAL_PART
finalblank.k
*WELDING_CURVE
initialweldingcurve.k
*END
```

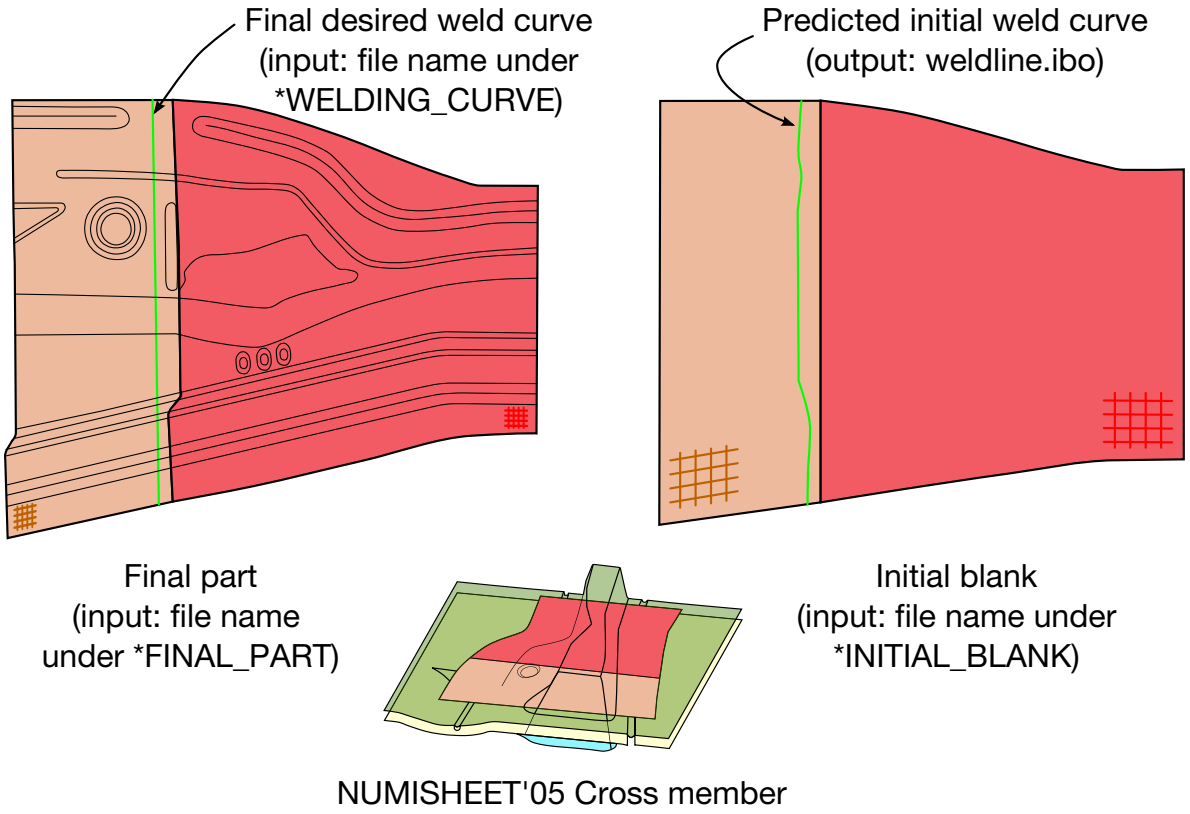


Figure 30-41. An example for a welding curve development.

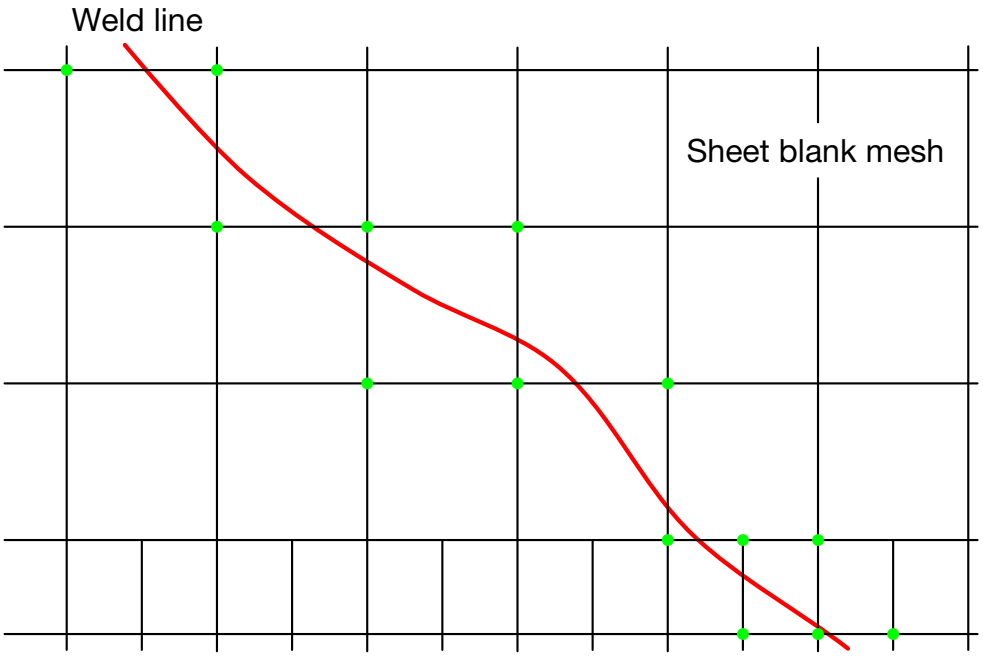


Figure 30-42. Nodes (in green) of element edges that intersect the weld curve are output in affectednd_i.ibo and affectednd_f.ibo, for initial and deformed mesh, respectively.

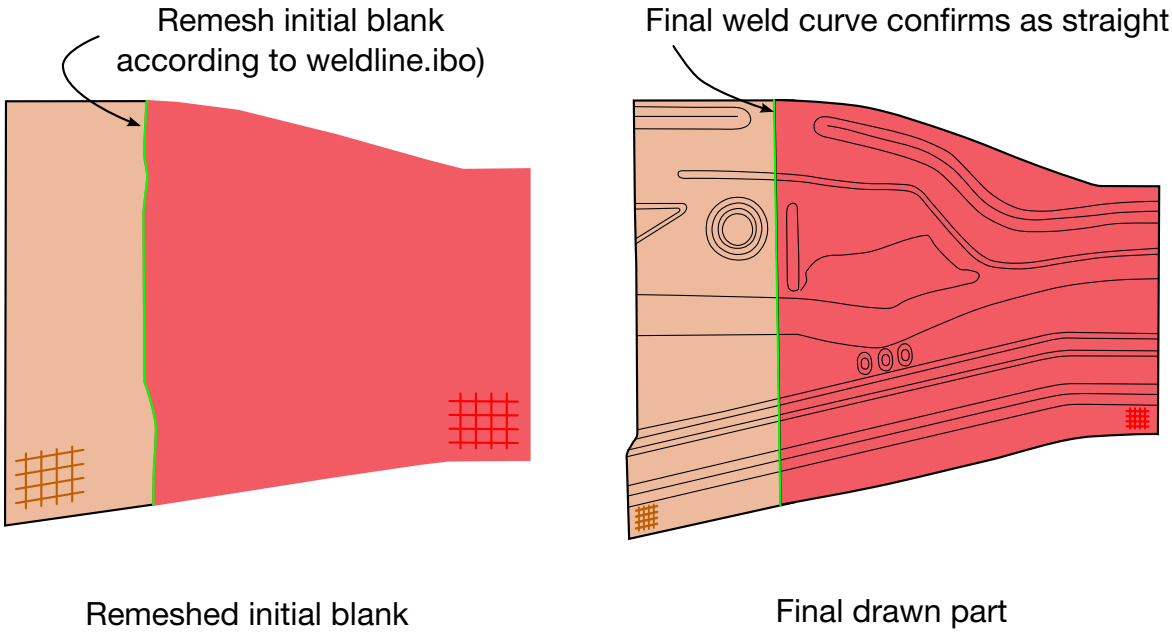


Figure 30-43. Verification run

Revision information:

1. IOPTION of "1": Revision 105189 in both double precision versions of SMP and MPP.
2. IOPTION of "-1": Revision 105727.
3. Output of nodes of element edges that intersect the weld curve: Revision 105727.
4. Later revisions may include improvements.

*ISPG

The *ISPG keywords enable creating a model for incompressible smooth particle Galerkin (ISPG) parts. This model can be used for simulating incompressible fluids and for coupling with structural parts. We created this family of keywords because these keywords cannot be in the normal LS-DYNA input deck. They can only be in a file included in the LS-DYNA input deck with *INCLUDE_ISPG. The keywords in this section are defined in alphabetical order:

*ISPG_ASCII_CPL

*ISPG_ASCII_SUMFORC

*ISPG_BOUNDARY_CONVECTION_SET

*ISPG_BOUNDARY_SYMMETRIC

*ISPG_CONTROL_ADAPTIVITY

*ISPG_CONTROL_D3ISPG

*ISPG_CONTROL_IMPLICIT

*ISPG_CONTROL_MPP

*ISPG_CONTROL_SOLUTION

*ISPG_DAMPING_SURF

*ISPG_DEFINE_MERGER_MPP

*ISPG_INITIAL_TEMPERATURE_SET

*ISPG_LOAD_GRAVITY

*ISPG_TO_ISPG_CONTACT

*ISPG_TRANS_CPL

The following keywords are additional ISPG-related keywords:

*DEFINE_ISPG_TO_SURFACE_COUPLING

*INCLUDE_ISPG

*MAT_ISPG_CARREAU

*MAT_ISPG_CROSS_CASTRO_MACOSKO

*MAT_ISPG_CROSSMODEL

*MAT_ISPG_ISO_NEWTONIAN

*MAT_THERMAL_ISPG

*SECTION_ISPG

All of the above keywords, except *DEFINE_ISPG_TO_SURFACE_COUPLING and *INCLUDE_ISPG, can only be in the ISPG input file.

Note that *SECTION_FPD and *MAT_IFPD are for an older ISPG solver. They cannot be used with these *ISPG keywords and can only be in a normal LS-DYNA input deck.

*ISPG_ASCII_CPL

Purpose: Output the contact area and reaction forces to an ASCII file for the coupling between ISPG parts and structures.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	ASCII_ID	CPL_ID	DT					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
ASCII_ID	Used for ASCII file name. For example, if ASCII_ID = 1, then the file name will be ispg_ascii000001.
CPL_ID	ID of the defined ISPG to surface coupling (see *DEFINE_ISPG_-TO_SURFACE_COUPLING)
DT	Time interval for output of files

*ISPG_ASCII_SUMFORC

Purpose: Output the reaction forces to structures from all the ISPG parts to the ASCII files ispg_sumforc_above and ispg_sumforc_below. The total reaction forces to the structure above and below the cutting plane based on COORD and NDIR are output to the respective ASCII files.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	COORD	NDIR					
Type	F	F	I					
Default	none	none	3					

VARIABLE	DESCRIPTION
DT	Time interval for the data output
COORD	Position along the direction defined by NDIR of the cutting plane for the force summation
NDIR	Normal direction of the cutting plane: EQ.1: Along the global *X*-axis EQ.2: Along the global *Y*-axis EQ.3: Along the global *Z*-axis

*ISPG_BOUNDARY_CONVECTION_SET

Purpose: Apply a convection boundary condition to ISPG parts in a coupled thermal and flow analysis. See also *ISPG_CONTROL_SOLUTION.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	FP	FPTYPE						
Type	I	I						
Default	none	none						

Card 1	1	2	3	4	5	6	7	8
Variable	HLCID	HMULT	TLCID	TMULT				
Type	I	F	I	F				
Default	0	none	0	none				

VARIABLE	DESCRIPTION
FP	Part ID for the fluid particles
FPTYPE	Type for FP: EQ.0: Part set ID EQ.1: Part ID
HLCID	Load curve ID for the curve giving the convection heat transfer coefficient, h, as a function of time. EQ.0: h is a constant defined by the value HMULT.
HMULT	Convection heat transfer coefficient, h. Ignored if HLCID > 0.
TLCID	Load curve ID for the curve giving the environment temperature,

VARIABLE	DESCRIPTION
	T_∞, as a function of time. EQ.0: T_∞ is a constant given by the value TMULT.
TMULT	Environment temperature, T_∞. Ignored if TLCID > 0.

*ISPG_BOUNDARY_SYMMETRY

Purpose: Define a symmetric boundary condition for pseudo 2D, planar 2D, and axisymmetric 2D simulations of adaptive ISPG.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	NDIR	COORD1	COORD2	ILOC				
Type	I	F	F	I				
Default	none	none	none	0				

VARIABLE	DESCRIPTION
NDIR	Normal direction of the symmetric planes used to define the symmetric boundary condition: EQ.1: Global *X*-axis EQ.2: Global *Y*-axis EQ.3: Global *Z*-axis
COORD1	Minimum coordinate value along the direction set with NDIR (ignored if ILOC = 2)
COORD2	Maximum coordinate value along the direction set with NDIR (ignored if ILOC = 1)
ILOC	Location of symmetric planes: EQ.0: Symmetric planes at both coordinates COORD1 and COORD2 EQ.1: Symmetric plane at coordinate COORD1 EQ.2: Symmetric plane at coordinate COORD2

*ISPG_CONTROL_ADAPTIVITY

Purpose: Control adaptivity parameters for ISPG.

NOTE: This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	IALIGN	IMOV	RA_SCL	RD_SCL	SSANGLE	ASANGLE	NDIV_MIN	IQCP
Type	I	I	F	F	F	F	I	I
Default	0	0	1.5	0.2	0.524	0.0873	0	0

Card 2	1	2						
Variable	HMIN	SCL_CE	IMERGE					
Type	F	F	I					
Default	0.001	1.0	0					

VARIABLE	DESCRIPTION
IALIGN	Flag to determine whether ISPG nodes automatically align with structural nodes and divisions in the structural segment determined using NDIV_MIN (see Remark 1): EQ.0.OR.EQ.1: Automatic alignment NE.0.AND.NE.1: No automatic alignment
IMOV	Enable staggering algorithm (see Remark 2): EQ.0.OR.EQ.1: Enable staggering. NE.0.AND.NE.1: Disable staggering.
RA_SCL	Scale factor for determining whether to add a new ISPG node. The adaptivity algorithm adds a node if the distance between two adjacent ISPG nodes is larger than RA_SCL $\times D_{\text{avg}}$. D_{avg} is the

VARIABLE	DESCRIPTION
	original average element edge length of an ISPG part.
RD_SCL	Scale factor for determining whether to delete an ISPG node. The adaptivity algorithm deletes a node if the distance between two adjacent ISPG nodes is smaller than RD_SCL × D_{avg}. D_{avg} is the original average element edge length of an ISPG part.
SSANGLE	Critical angle (unit rad) between adjacent structural segments for ISPG edges to stagger (see IMOV defined above).
ASANGLE	Critical angle (unit rad) of adjacent structural segments for ISPG nodes to align to structural nodes (see IALIGN defined above).
NDIV_MIN	Minimum number of divisions of the structural segment for ISPG node alignment (see IALIGN above). The number of divisions in the local *s*- (nodes 1→2 and 4→3 for a 4-node segment) and *t*-directions (nodes 2→3 and 1→4 for a 4-node segment) for a 4-node structural segment is determined by $\text{ndiv}_s = \max(\text{ndiv}_{\min}, 2L_s/D_{\text{avg}})$ and $\text{ndiv}_t = \max(\text{ndiv}_{\min}, 2L_t/D_{\text{avg}})$, where $L_s = \max(L_{12}, L_{34})$, $L_t = \max(L_{23}, L_{14})$, and D_{avg} is the original average element edge length of an ISPG part. For three-node segments, the segment's maximum length is used to determine the division number.
IQCP	Control the number of quadrature points in the ISPG element used for the detection of ISPG element penetration into structural elements: EQ.0: Use one quadrature point (center of the ISPG element) for the detection. EQ.1: Use 5 points for the detection.
HMIN	Minimum relative element height compared to the average element edge length D_{avg}. If the element's height (for the tetrahedral elements, whose 4 nodes are at the surface of the ISPG part) after adaptivity is smaller than HMIN × D_{avg}, then this element is excluded from the ISPG part. The default value of HMIN is 0.001. We recommend values of HMIN between 0.001 and 0.05.
SCL_CE	Scale factor for the element size at the regions near the contact edges (see Figure 31-1). The element size near the contact edges is SCL_CE × D_{avg} for the refinement. Here, D_{avg} is the original average element edge length for the ISPG elements. The default value of SCL_CE is 1.0. If SCL_CE is smaller than 0.333, then it is reset to

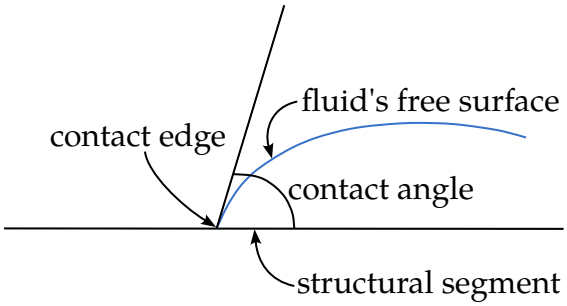


Figure 31-1. Example of contact between a structure and ISPG element. In the 3D the contact edge is the line going into the plane. In 2D, it is the point.

VARIABLE	DESCRIPTION
	0.333.
IMERGE	Flag to automatically merge parts that are sufficiently close together into one part: LE.0: Do not enable merging. GT.0: Enable merging. Note that for merging to work in MPP the parts for which merging is possible must be on the same processor. Use *ISPG_DEFINE_MERGER_MPP to distribute these parts to the same processor.

Remarks:

1. **Automatic alignment.** Automatic alignment of ISPG nodes to structural segments only occurs at structural segments under the following conditions:

   a) the angle between a structural segment and the structural segment adjacent to it is larger than ASANGLE (see the description of ASANGLE), and

   b) ISPG fluid contact surfaces cover all the nodes of the structural segment.

   The ISPG nodes align to both structural nodes and a number of locations along the structural segment (see NDIV_MIN).

2. **Staggering algorithm.** Usually, the ISPG contact edges (see Figure 31-1) move by internal forces (e.g., pressure and viscosity forces) and external forces (e.g., surface tension forces and wall adhesion forces). But at some sharp corners of the structure, it is difficult to determine the normal vector of the contact needed for calculating the wall adhesion and normal contact forces. The strong discontinuity in the structural segments causes this issue. We implemented a staggering algorithm to handle this scenario. In these regions, the ISPG edge staggers forward or backward depending on the contact angle (see *DEFINE_ISPG_TO_-

SURFACE_COUPLING). If the contact angle is larger than the advancing angle (ACA on *DEFINE_ISPG_TO_SURFACE_COUPLING), the ISPG contact edge moves forward along the structural segment. Otherwise, if the contact angle is smaller than the receding contact angle (RCA on *DEFINE_ISPG_TO_SURFACE_COUPLING), the ISPG contact edge staggers backward. The staggering occurs only at adjacent structural segments with an angle larger than SSANGLE.

*ISPG_CONTROL_D3ISPG

Purpose: Control the data output for the ISPG method into the d3ispg file for further post-processing. The d3ispg file is automatically output during ISPG simulations. *DATABASE_BINARY_D3PLOT controls the output frequency of this file. By default, d3ispg only contains pressure data. This keyword enables outputting additonal data to this file.

NOTE: This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	STRESS	STRAINR	VIS	TEMP	CURG			
Type	I	I	I	I	I			
Default	0	0	0	0	0			

VARIABLE	DESCRIPTION
STRESS	Flag to output stresses into d3ispg (see Remark 1): LE.0: No GT.0: Yes
STRAINR	Flag to output shear strain rates into d3ispg (see Remark 2): LE.0: No GT.0: Yes
VIS	Flag to output viscosity into d3ispg: LE.0: No GT.0: Yes
TEMP	Flag to output temperature into d3ispg: LE.0: No GT.0: Yes
CURG	Flag to output material curing into d3ispg: LE.0: No

VARIABLE	DESCRIPTION
	GT.0: Yes

Remarks:

1. **Stress output.** Setting STRESS greater than 0 causes the output of the normal, shear, and equivalent (σ_{eq}) stresses in the order: σ_{xx}, σ_{yy}, σ_{zz}, σ_{xy}, σ_{xz}, σ_{yz}, σ_{eq}. The equivalent stress is given by:

$$\sigma_{eq} = \sqrt{\frac{\sigma_{xx}^2}{2.0} + \frac{\sigma_{yy}^2}{2.0} + \frac{\sigma_{zz}^2}{2.0} + \sigma_{xy}^2 + \sigma_{xz}^2 + \sigma_{yz}^2}$$

2. **Strain-rate output.** Setting STRAINR greater than 0 causes the output of the normal, shear, and equivalent ($\dot{\varepsilon}_{eq}$) strain rates in the order: $\dot{\varepsilon}_{xx}$, $\dot{\varepsilon}_{yy}$, $\dot{\varepsilon}_{zz}$, $\dot{\varepsilon}_{xy}$, $\dot{\varepsilon}_{xz}$, $\dot{\varepsilon}_{yz}$, $\dot{\varepsilon}_{eq}$. The equivalent strain rate is given by:

$$\dot{\varepsilon}_{eq} = \sqrt{\frac{\dot{\varepsilon}_{xx}^2}{2.0} + \frac{\dot{\varepsilon}_{yy}^2}{2.0} + \frac{\dot{\varepsilon}_{zz}^2}{2.0} + \dot{\varepsilon}_{xy}^2 + \dot{\varepsilon}_{xz}^2 + \dot{\varepsilon}_{yz}^2}$$

*ISPG_CONTROL_IMPLICIT

Purpose: Control the implicit parameters for the ISPG method.

NOTE: This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	BETA	GAMMA	CFL	MX_SUBS	MX_ITERS			
Type	F	F	F	I	I			
Default	0.38	0.60	1.0	512	20			

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DTIMP						
Type	F	F						
Default	0.0	0.0						

VARIABLE	DESCRIPTION
BETA	β used by the Newton-Raphson iteration algorithm
GAMMA	γ used by the Newton-Raphson iteration algorithm
CFL	Courant number
MX_SUBS	Maximum subcycles of the ISPG solver for each structural time step
MX_ITERS	Maximum number of iterations for the ISPG implicit method in each sub-step
BIRTH	Starting time for the fully implicit ISPG iteration. Before BIRTH, only one-way coupling occurs, meaning no forces from the solder return to the structure. The implicit time step for ISPG parts is DTIMP instead of the structural implicit step to guarantee that the

VARIABLE	DESCRIPTION
	fluid moves with the solid boundaries. After BIRTH, two-way coupling occurs, and the ISPG solver performs a full iteration with the structural implicit step. This option is very useful for cases where the structural simulation time is very long (e.g., in seconds or minutes) while the reflow process to steady state is very short.
DTIMP	Implicit time step for ISPG parts during the simulation time before BIRTH

*ISPG_CONTROL_MPP

Purpose: Control the MPP decomposition of the ISPG fluid. Currently, this keyword is valid for the decomposition of the ISPG fluid parts into different CPU cores. During the decomposition, the nodes and elements of one fluid part are decomposed into the same CPU core to minimize the MPI communication.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable		DX	DY	DZ				
Type		F	F	F				
Default		0.0	0.0	0.0				

VARIABLE	DESCRIPTION
DX, DY, DZ	Size of the box in the global x, y, and z directions. The box is used to detect the adjacent structural segments used to define the coupling between the structure and ISPG fluids. If zero is given, $10.0 \times D_{avg,max}$ is used. $D_{avg,max}$ is the maximum value of the original averaged nodal distance for all ISPG parts.

*ISPG_CONTROL_SOLUTION

Purpose: Set the analysis solution procedure for ISPG if performing a combined thermal and flow analysis.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	SOLN							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
SOLN	Analysis solution procedure: EQ.0: Flow analysis only EQ.1: Combined flow and thermal analysis

*ISPG_DAMPING_SURF

Purpose: Define Rayleigh damping related to the fluid surface tension on all the ISPG parts. Currently, this feature supports only the 2D ISPG formulation.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	DAMP							
Type	F							
Default	none							

VARIABLE	DESCRIPTION
DAMP	Rayleigh damping value, related to the stiffness of the surface tension term

*ISPG_DEFINE_MERGER_MPP

Purpose: Associate ISPG parts with a merging group so that these parts will be distributed to the same processor to facilitate merging them into one part for solder reflow simulations in MPP. For merging to occur IMERGE must be greater than 0 on *ISPG_CONTROL_ADAPTIVITY. This keyword is needed because sufficiently close parts may be decomposed onto different processors. If that occurs, they will not be merged.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Include as many cards as desired. The next keyword ("*") card terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	MRGID	PID1	PID2	PID3	PID4	PID5	PID6	PID7
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
MRGID	Merging ID. All parts on this card are put into a group with this merging ID. The parts in the merging group are distributed to the same processor to make merging possible.
PID*i*	ISPG fluid part ID

*ISPG_INITIAL_TEMPERATURE_SET

Purpose: Define initial nodal point temperatures of ISPG parts in a coupled thermal and flow analysis. See also *ISPG_CONTROL_SOLUTION.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	FP	FPTYPE	TEMP					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
FP	Part or part set ID of the fluid particles
FPTYPE	Type for FP: EQ.0: Part set ID EQ.1: Part ID
TEMP	Temperature of the fluid particles

*ISPG_LOAD_GRAVITY

Purpose: Define the gravity load applied to all the ISPG parts.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	IDIR	GRAV						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
IDIR	Global axis direction of the gravity load: EQ.1: Global *X*-axis EQ.2: Global *Y*-axis EQ.3: Global *Z*-axis
GRAV	Value of the gravity load

*ISPG_TO_ISPG_CONTACT

Purpose: Specify sets of ISPG parts where the parts in each set interact with each other through contact during the solder reflow simulation process. Only two ISPG parts are supported for each contact set currently.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Include as many cards as desired. Each card specifies the parts in a single contact interface. The next keyword ("*") card terminates this input.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	PID1	PID2					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
CID	Contact ID
PID*i*	ISPG fluid part IDs for the parts in contact

Remarks:

1. **Contact in MPP.** ISPG contact requires the parts in a contact interface be on the same processor. To do this, use *ISPG_DEFINE_MERGER_MPP.

*ISPG_TRANS_CPL

Purpose: Define strong coupling between an ISPG fluid and rigid bodies. Currently, this keyword is only supported for models that include a single ISPG part. If there are multiple ISPG fluid parts in the model, they need to be merged into one part. If this keyword is not included in the model, a weak coupling between the ISPG fluid and rigid bodies is automatically imposed.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG. It can only be used once.

Card 1	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	non

Card 2	1	2	3	4	5	6	7	8
Variable	DOFX	DOFY	DOFZ	DOFRX	DOFRY	DOFRZ	BIRTH	THK_MIN
Type	I	I	I	I	I	I	F	F
Default	0	0	0	0	0	0	0.0	0.0

VARIABLE	DESCRIPTION
PID*i*	Part IDs for the rigid bodies that are coupled to the ISPG fluid
DOFX	Flag to specify if the translational X degree of freedom is fixed for the rigid body motion: EQ.0: Not fixed EQ.1: Fixed
DOFY	Flag to specify if the translational Y degree of freedom is fixed for the rigid body motion:

VARIABLE	DESCRIPTION
	EQ.0: Not fixed EQ.1: Fixed
DOFZ	Flag to specify if the translational *Z* degree of freedom is fixed for the rigid body motion: EQ.0: Not fixed EQ.1: Fixed
DOFRX	Flag to specify if the rotational *X* degree of freedom is fixed for the rigid body motion: EQ.0: Not fixed EQ.1: Fixed
DOFRY	Flag to specify if the rotational *Y* degree of freedom is fixed for the rigid body motion: EQ.0: Not fixed EQ.1: Fixed
DOFRZ	Flag to specify if the rotational *Z* degree of freedom is fixed for the rigid body motion: EQ.0: Not fixed EQ.1: Fixed
BIRTH	Birth time used to activate the coupling algorithm. After BIRTH, the general rigid motion defined via *BOUNDARY_PRESCRIBED_MOTION_RIGID and *BOUNDARY_SPC_*OPTION* is deactivated. Do not use the DEATH option in the keyword *BOUNDARY_PRESCRIBED_MOTION_RIGID because it can lead to unpredictable behavior.
THK_MIN	Variable used to define the minimum thickness of the fluid under the compression of rigid bodies. If the thickness of the adhesive reaches THK_MIN anywhere in the adhesive due to compression from the rigid parts, the motion of the rigid part is stopped. This variable can be used to avoid the penetration among rigid parts that are in contact with the ISPG fluid. If THK_MIN < 1.0×10^{-20}, the penetration detection algorithm is deactivated.

Remarks:

1. **Motion of the rigid bodies.** After BIRTH, the motion of the rigid bodies is controlled by applied loads (*LOAD_NODE_*OPTION* and *LOAD_SEGMENT_{*OPTION*}) with restrictions in the degrees of freedom specified with this keyword. Motion prescribed with *BOUNDARY_PRESCRIBED_MOTION_RIGID and *BOUNDARY_SPC_*OPTION* is deactivated at time BIRTH. In general, we do not recommend defining the rigid body motion for the rigid parts coupled with ISPG fluid using *MAT_RIGID.

2. **Output files.** When using this keyword in a deck with *ISPG_ASCII_SUMFORC, ispg_sumtorc_above and ispg_sumtorc_below are output in addition to ispg_sumforc_above and ispg_sumforc_below. These additional files provide the fluid's moments on the structures.

*KEYWORD

***KEYWORD**_*{OPTION} {memory} {memory2 = j} {NCPU = n}*

Available options include:

<BLANK>

ID

JOBID

Purpose: The keyword, *KEYWORD, flags LS-DYNA that the input deck is a keyword deck rather than the structured format, which has a strictly defined format. This must be the first card in the input file. Alternatively, by typing "keyword" on the execution line, keyword input formats are assumed and this beginning "*KEYWORD" line is not required.

There are 3 optional parameters that can be specified on the *KEYWORD line. If a number *{memory}* is specified, it defines the memory size in units of words to be allocated. For MPP, if the parameter {memory2 = *j*} is also given, *memory* defines the memory allocation in words for the first core and *j* defines the memory allocation in words for each of the remaining cores. Note that if the memory size is specified on the execution line, it will override the memory size specified on the *KEYWORD line.

If the parameter {NCPU = *n*} is specified it defines the number Shared Memory Parallel (SMP) threads of "*n*" to be used for each processor during the analysis. This applies to both SMP and HYBRID versions of LS-DYNA.

For the Distributed Memory Version (MPP), the number of CPUs is defined with the "mpirun –np *m*" command.

HYBRID is combination of SMP and MPP therefore the total number of cores used in this example will be *n x m* cores. Defining the number of CPUs on the execution line overrides what is specified on the *KEYWORD line and both override the number of CPUs specified by *CONTROL_PARALLEL. An example of the {memory} and {NCPU = *n*} options would be as follows:

```
*KEYWORD 120m memory2=10m NCPU=2
```

This *KEYWORD command is requesting 120 million words of memory, 10 million words of memory2 and 2 CPUs to be used for the analysis with the consistency flag (see CONST in *CONTROL_PARALLEL) turned off. To run with the consistency flag turned on (recommended), set NCPU to a negative value, e.g., "NCPU = -2" runs with 2 CPUs with the consistency flag turned on.

The ID and JOBID command line options are available to add a prefix to all output and scratch file names, i.e., not the input filenames. This allows multiple simulations in a directory since a different prefix prevents files from being overwritten. If the ID option is used, the prefix is constructed of three user specified strings separated by "_f" characters.

ID Card. Additional Card if the ID option is active.

Card 1	1	2	3	4	5	6	7	8
Variable	PROJECT		NUM		STAGE			
Type	A		A		A			
Default	none		none		none			

VARIABLE	DESCRIPTION
PROJECT	First part of the output file name prefix.
NUM	Second part of the output file name prefix.
STAGE	Third part of the output file name prefix.

By using the ID option of *KEYWORD, an output file name prefix may be specified as a combination of the variables PROJECT, NUM and STAGE as defined on Card 1 above. For example, if these variables were set literally to "PROJECT", "NUM", and "STAGE", the first d3plot would be named:

```
PROJECT_NUM_STAGE.d3plot
```

Alternatively, an output file name prefix can be assigned by including "jobid=" on the execution line. For example,

lsdyna i=**input.k** jobid=**PROJECT_NUM_STAGE**

A third way to define an output file name prefix is by using the JOBID option of the *KEYWORD command, in which case Card 1 is defined as shown below and the variable JBID acts as the output prefix.

JOBID Card. Additional card if the JOBID option is active.

Card 1	1	2	3	4	5	6	7	8
Variable	JBID							
Type	A							
Default	none							

*LOAD

The keyword *LOAD provides a way of defining applied forces. The keyword control cards in this section are defined in alphabetical order:

*LOAD_ACOUSTIC_SOURCE

*LOAD_ALE_CONVECTION

LOAD_BEAM_*OPTION

*LOAD_BLAST

*LOAD_BLAST_CLEARING

*LOAD_BLAST_ENHANCED

*LOAD_BLAST_SEGMENT

*LOAD_BLAST_SEGMENT_SET

LOAD_BODY_*OPTION

LOAD_BODY_GENERALIZED_*OPTION

*LOAD_BODY_POROUS

*LOAD_BRODE

*LOAD_DENSITY_DEPTH

LOAD_EDGE_UVW_*{OPTION}

*LOAD_ERODING_PART_SET

*LOAD_EXPANSION_PRESSURE

*LOAD_EXTERNAL_VARIABLE

LOAD_FACE_UVW_*{OPTION}

LOAD_FACE_XYZ_*{OPTION}

*LOAD_GRAVITY_PART

*LOAD_HEAT_CONTROLLER

*LOAD_HEAT_EXOTHERMIC_REACTION

LOAD_HEAT_GENERATION_*OPTION

*LOAD_HEAT_LASER_ABSORPTION

*LOAD_MASK

*LOAD_MOTION_NODE

*LOAD_MOVING_PRESSURE

LOAD_NODE_*OPTION

*LOAD_NURBS_SHELL

LOAD_POINT_UVW_*{OPTION}

*LOAD_PYRO_ACTUATOR

*LOAD_PZE

*LOAD_REMOVE_PART

*LOAD_RIGID_BODY

*LOAD_SEGMENT

*LOAD_SEGMENT_ALE_INI

*LOAD_SEGMENT_CONTACT_MASK

*LOAD_SEGMENT_FILE

*LOAD_SEGMENT_FSILINK

*LOAD_SEGMENT_NONUNIFORM

*LOAD_SEGMENT_SET

*LOAD_SEGMENT_SET_ANGLE

*LOAD_SEGMENT_SET_NONUNIFORM

*LOAD_SEISMIC_FREE_FIELD

LOAD_SEISMIC_SSI_*OPTION1_{OPTION2}

LOAD_SEISMIC_SSI_AUX_*{OPTION}

LOAD_SHELL_*OPTION1_{OPTION2}

*LOAD_SPCFORC

*LOAD_SSA

*LOAD_STEADY_STATE_ROLLING

*LOAD_STIFFEN_PART

*LOAD_SUPERPLASTIC_FORMING

LOAD_SURFACE_STRESS_*{OPTION}

*LOAD_THERMAL_BINOUT

*LOAD_THERMAL_CONSTANT

*LOAD_THERMAL_CONSTANT_ELEMENT

*LOAD_THERMAL_CONSTANT_NODE

*LOAD_THERMAL_D3PLOT

*LOAD_THERMAL_LOAD_CURVE

*LOAD_THERMAL_RSW

*LOAD_THERMAL_TOPAZ

*LOAD_THERMAL_VARIABLE

*LOAD_THERMAL_VARIABLE_BEAM

LOAD_THERMAL_VARIABLE_ELEMENT_*OPTION

*LOAD_THERMAL_VARIABLE_NODE

*LOAD_THERMAL_VARIABLE_SHELL

*LOAD_VOLUME_LOSS

*LOAD_ACOUSTIC_SOURCE

Purpose: Specify acoustic loading sources for *CONTROL_IMPLICIT_SSD_DIRECT and *CONTROL_ACOUSTIC_SPECTRAL analyses.

Card 1	1	2	3	4	5	6	7	8
Variable	NID/SSID	SRCTYP	LCID	DATA1	DATA2	DATA3	DATA4	DATA5
Type	I	I	I	F	F	F	F	F
Default	none	↓	0	1.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
NID	Node ID of the acoustic point source for SRCTYP = 1 and 5
SSID	Segment set ID of structural faces on the external fluid-structure boundary exposed to the acoustic wave source for SRCTYP = 11 and 12
SRCTYP	Acoustic source type: EQ.1: Harmonic nodal point source. DATA1, DATA2, and LCID define the harmonic nodal source strength $\dot{Q}$, the phase angle of the nodal source strength and frequency variation for a point source at node NID. EQ.5: Transient nodal point source. DATA1 and LCID define the transient nodal source strength ($\dot{Q}$) and temporal variation for a point source at node NID. EQ.11: Harmonic plane wave: DATA1 and LCID define the magnitude and frequency variation for a harmonic plane wave with direction cosines given in DATA2, DATA3, and DATA4. SSID is the segment set ID of the external structural (coupled) surface. EQ.12: Harmonic spherical wave. DATA1, DATA5, and LCID define the magnitude, reference radius and frequency variation for a harmonic spherical wave centered at coordinates x_0, y_0, and z_0 specified with DATA2, DATA3, and DATA4. SSID is the segment set ID of the external structural (coupled) surface.

VARIABLE	DESCRIPTION
LCID	Load curve ID of curve specifying the variation of the load with frequency or time. If LCID is undefined, then the loading is constant.
DATA1	SRCTYP.EQ.1: Magnitude of the harmonic nodal source strength, $\dot{Q}$ SRCTYP.EQ.5: Magnitude of the transient nodal source strength, $\dot{Q}$ SRCTYP.EQ.11: Pressure of the harmonic plane wave, P_0 SRCTYP.EQ.12: Pressure of the harmonic spherical wave, P_0, at r_0
DATA2	SRCTYP.EQ.1: Phase angle of the nodal source strength in radians SRCTYP.EQ.11: Direction cosine, α SRCTYP.EQ.12: x coordinate of center of spherical wave, x_0
DATA3	SRCTYP.EQ.11: Direction cosine, β SRCTYP.EQ.12: y coordinate of center of spherical wave, y_0
DATA4	SRCTYP.EQ.11: Direction cosine, γ SRCTYP.EQ.12: z coordinate of center of spherical wave, z_0
DATA5	For SRCTYP = 12, reference radius, r_0, where the pressure equals p_0

Remarks:

1. **Source strength.** $\dot{Q}$ has dimensions of area times acceleration. In steady state analysis, it is related to Q_0 through

$$\dot{Q} = \omega Q_0 e^{i\left(\omega t + \frac{\pi}{2}\right)}$$

2. **External, incident wave solutions.** With SRCTYP = 11 or 12, the SSD solution in the acoustic domain is for the scattered pressure.

3. **Incident pressure and particle velocity.** For direct, steady state vibration acoustic scattering invoked with *CONTROL_IMPLICIT_SSD_DIRECT, the incident pressure and particle velocity applied depend on the source type.

a) For plane waves (SRCTYP = 11), they are given by:

$$P_{\text{inc}} = P_0 e^{-ik(\alpha x+\beta y+\gamma z)}$$
$$U_{\text{inc}} = \frac{P_0}{\rho_0 c_0} e^{-ik(\alpha x+\beta y+\gamma z)}$$

b) For spherical waves (SRCTYP = 12), they are given by:

$$P_{\text{inc}} = \frac{P_0 r_0}{r} e^{-ik(r-r_0)}$$
$$U_{\text{inc}} = \frac{P_0}{\rho_0 c_0}\left(\frac{r_0}{r}\right)\left(\frac{kr-i}{kr}\right) e^{-ik(r-r_0)}$$

*LOAD_ALE_CONVECTION_{OPTION}

Purpose: Define the convection thermal energy transfer from a hot ALE fluid to the surrounding Lagrangian structure (Remark 1). It is associated with a corresponding coupling card defining the interaction between the ALE fluid and the Lagrangian structure. It is only used when thermal energy transfer from the ALE fluid to the surrounding Lagrangian structure is significant. This is designed specifically for airbag deployment applications where the heat transfer from the inflator gas to the inflator compartment can significantly affect the inflation potential of the inflator.

Available options include:

<BLANK>

ID

To define an ID number for each convection heat transfer computation in an optional card preceding all other cards for this command. This ID number can be used to output the part temperature and temperature change as functions of time in the *DATABASE_FSI card. To do this, set CONVID in *DATABASE_FSI equal to this ID.

ID Card. Additional card for ID keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						
Default	none	none						

Convection Card. Include as many cards as necessary. This input terminates at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	LAGPID	LAGT	LAGCP	H	LAGMAS			
Type	I	F	F	F	F			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
ID	ID number for each convection heat transfer computation
TITLE	A description of this convection heat transfer
LAGPID	Lagrangian structure PID from a corresponding coupling card which receives the thermal energy in the convection heat transfer
LAGT	Initial temperature of this Lagrangian structure part
LAGCP	Constant-pressure heat capacity of this Lagrangian structure part. It has a per-mass unit (for example, J/[kg*K]).
H	Convection heat transfer coefficient on this Lagrangian structure part surface. It is the amount of energy (J) transferred per unit area, per time, and per temperature difference. For example, its units may be J/[m^2*s*K].
LAGMAS	The mass of the Lagrangian structure part receiving the thermal energy. This is in absolute mass unit (for example, kg).

Remarks:

1. **Applications.** The only application of this card so far has been for the transfer of thermal energy from the ALE hot inflator gas to the surrounding Lagrangian structure (inflator canister and airbag-containing compartment) in an airbag deployment model.

2. **Model.** The heat transferred is taken out of the inflator gas thermal energy thus reducing its inflating potential. This is not a precise heat transfer modeling attempt. It is simply one mechanism for taking out excessive energy from the inflating potential of the hot inflator gas.

   The heat transfer formulation may roughly be represented as the following equation. Some representative units are shown just for clarity.

$$[\dot{Q}] = [\mathrm{H} \times A \times \Delta T] = \left(\frac{[\mathrm{E}]}{[L]^2[\mathrm{t}][\mathrm{T}]}\right) \times [\mathrm{L}]^2 \times [\mathrm{T}] = [\mathrm{Power}]$$

$$[\dot{Q}] = \left[\dot{M} C_p \left(T_{\mathrm{Lag\ New}} - T_{\mathrm{Lag\ Orig}}\right)\right] = \left(\frac{[\mathrm{M}]}{[t]}\right) \times \left(\frac{[\mathrm{E}]}{[\mathrm{M}][\mathrm{T}]}\right) \times [\mathrm{T}] = \frac{[\mathrm{E}]}{[\mathrm{t}]}$$

LOAD_BEAM_*OPTION

Available options include:

ELEMENT

SET

Purpose: Apply the distributed traction load along any local axis of a beam or a set of beams for non-discrete beams. For discrete beams, the load is evenly split between the two nodes (see Remark 2). The local axes are defined in Figure 33-1; see also *ELEMENT_BEAM.

Beam Cards. Include as many as necessary. This input stops at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	EID/ESID	DAL	LCID	SF				
Type	I	I	I	F				
Default	none	none	none	1.				

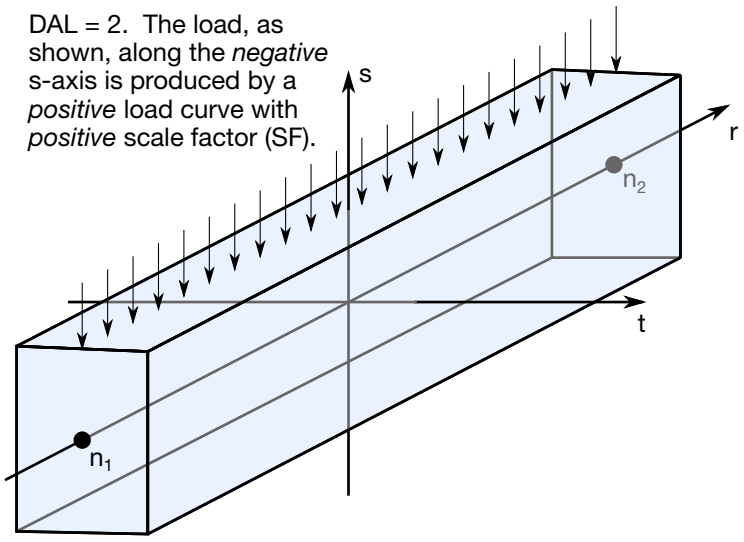


Figure 33-1. Applied traction loads are given in force per unit length. The *s* and *t*-directions are defined on the *ELEMENT_BEAM keyword.

VARIABLE	DESCRIPTION
EID/ESID	Beam ID (EID) or beam set ID (ESID); see *ELEMENT_BEAM or *SET_BEAM
DAL	Direction of applied load: EQ.1: Parallel to *r*-axis of beam, EQ.2: Parallel to *s*-axis of beam, EQ.3: Parallel to *t*-axis of beam, EQ.4: Parallel to global *X*-axis, EQ.5: Parallel to global *Y*-axis, EQ.6: Parallel to global *Z*-axis.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). See Remark 1.
SF	Load curve scale factor. This is for a simple modification of the function values of the load curve.

Remark:

1. **LCID function arguments.** The function defined by LCID has 7 arguments: time, the 3 current coordinates, and the 3 reference coordinates. For example, the function (see *DEFINE_FUNCTION),

   ```
   f(t,x,y,z,x0,y0,z0)= -10.*sqrt ( (x-x0)*(x-x0)+(y-y0)*(y-y0)+(z-z0)*(z-z0) ),
   ```

   applies a force proportional to the distance from the initial coordinates. Note that the current coordinates and reference coordinates are the average values between the two nodes.

2. **Discrete beams.** For discrete beams, the value of the load curve or function is assumed to be a force instead of a traction value. This force is then divided by two and applied to each of the nodes of the discrete beam. Note that the length of the discrete beam is assumed to be 1.0 and the local coordinate system is updated according to SCOOR.

*LOAD_BLAST

Purpose: Define an air blast function for the application of pressure loads from the detonation of conventional explosives. The implementation is based on a report by Randers-Pehrson and Bannister [1997] where it is mentioned that this model is adequate for use in engineering studies of vehicle responses due to the blast from land mines. This option determines the pressure values when used in conjunction with the keywords: *LOAD_SEGMENT, *LOAD_SEGMENT_SET, or *LOAD_SHELL.

Card 1	1	2	3	4	5	6	7	8
Variable	WGT	XBO	YBO	ZBO	TBO	IUNIT	ISURF	
Type	F	F	F	F	F	I	I	
Default	none	0.0	0.0	0.0	0.0	2	2	

Card 2	1	2	3	4	5	6	7	8
Variable	CFM	CFL	CFT	CFP	DEATH			
Type	F	F	F	F	F			
Default	0.0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
WGT	Equivalent mass of TNT.
XBO	x-coordinate of point of explosion.
YBO	y-coordinate of point of explosion.
ZBO	z-coordinate of point of explosion.
TBO	Time-zero of explosion.
IUNIT	Unit conversion flag. EQ.1: feet, pound-mass, seconds, psi EQ.2: meters, kilograms, seconds, pascals (default)

VARIABLE	DESCRIPTION
	EQ.3: inch, dozens of slugs, seconds, psi EQ.4: centimeters, grams, microseconds, megabars EQ.5: user conversions will be supplied (see Card 2)
ISURF	Type of burst. See Remark 2. EQ.1: surface burst. The blast is located on or very near the ground, which is modeled as a surface. See Remark 5. EQ.2: air burst. The blast is spherical (default).
CFM	Conversion factor: pounds per LS-DYNA mass unit.
CFL	Conversion factor: feet per LS-DYNA length units.
CFT	Conversion factor: milliseconds per LS-DYNA time unit.
CFP	Conversion factor: psi per LS-DYNA pressure unit
DEATH	Death time. Blast pressures are deactivated at this time.

Remarks:

1. **Load Curves.** A minimum of two load curves, even if unreferenced, must be present in the model. Even though this keyword does not use load curves, unless there are at least two *DEFINE_CURVE cards LS-DYNA will error terminate with *LOAD_BLAST. Note that one of the enhancements of *LOAD_BLAST_ENHANCED is that it does not require two (place holding) load curves.

2. **Orientation.** The target's segments must be consistently oriented so that the normal vector of the portion of the target having a direct line of sight to the blast points roughly towards the blast. For example, segments defining a structure being blasted from the outside should have outwardly oriented normal vectors, whereas a cavity containing a blast should have inwardly facing normal vectors.

3. **Equivalent Mass of TNT.** Several methods can be used to approximate the equivalent mass of TNT for a given explosive. The simplest involves scaling the mass by the ratio of the Chapman-Jouguet detonation velocities given the by:

$$M_{\mathrm{TNT}} = M_{\mathrm{e}} \frac{v_{\mathrm{e}}^2}{v_{\mathrm{TNT}}^2}$$

where M_{TNT} is the equivalent TNT mass and v_{TNT} is the Chapman-Jouguet (CJ) detonation velocity of TNT. M_{e} and v_{e} are, respectively, the mass and CJ velocity

of the explosive under consideration. LS-DYNA takes the density of "Standard" TNT to be 1.57 $\frac{\text{g}}{\text{cm}^3}$ and v_{TNT} to be 0.693 $\frac{\text{cm}}{\mu\text{s}}$.

4. **Limits of Model's Validity.** Define a scaled distance, Z, as follows:

$$Z = \frac{R}{M_{\text{e}}^{1/3}}$$

where R is the distance from the charge center to the target and M_{e} is the equivalent TNT mass (see Remark 3). The scaled distance determines the range of validity for these models. The spherical airburst model is valid for Z ranging from 0.37 ft/lbm$^{1/3}$ (0.147 m/kg$^{1/3}$) out to 100 ft/lb-m$^{1/3}$ (40 m/kg$^{1/3}$). The hemispherical surface burst is valid for Z ranging from 0.45 ft/lbm$^{1/3}$ (0.178 m/kg$^{1/3}$) out to 100 ft/lbm$^{1/3}$ (40 m/kg$^{1/3}$).

5. **Surface Burst Model.** The surface blast model is ideal for when a charge is located on or very near the ground. In this case the initial blast wave is immediately reflected and reinforced by the nearly rigid ground (the model accounts for yielding) to produce a hemispherical reflected wave from the point of the burst. This reflected wave merges with the initial incident wave producing overpressures which are greater than those produced by the initial wave alone. Target points equidistant from the burst point are loaded identically with the surface burst option.

6. **Single Charge Limitation.** This feature cannot model multiple blasts in one simulation. Use *LOAD_BLAST_ENHANCED for multiple charges.

7. **2D Analysis.** This feature cannot be used for 2D analysis. Instead, use *LOAD_BLAST_ENHANCED.

*LOAD_BLAST_CLEARING

Purpose: Define a surface to be cleared by a rarefaction wave that arises from the diffraction of a blast wave around the free edges of a target face. This feature works only with *LOAD_BLAST_ENHANCED. The implementation is based on a report by Hudson [1955].

Card 1	1	2	3	4	5	6	7	8
Variable	BID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
BID	Blast source ID (see *LOAD_BLAST_ENHANCED)
N1 - N4	Node IDs of the corners of the surface

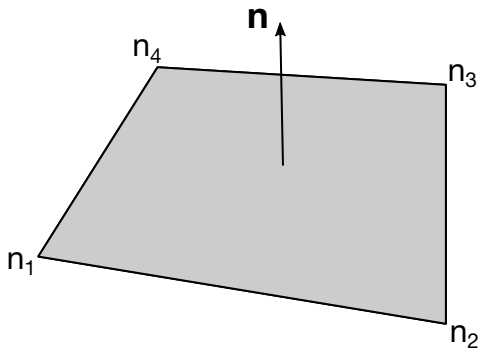


Figure 33-2. Nodal numbering for a cleared surface. Normal vector, **n**, roughly points toward the blast origin.

Remarks:

1. **Orientation.** The normal vector for the clear surface must roughly point toward the blast origin. The node numbering and normal convention are specified in Figure 33-2.

*LOAD_BLAST_ENHANCED

Purpose: Define an air blast function for the application of pressure loads due the detonation of a conventional explosive. While similar to *LOAD_BLAST, this feature includes enhancements for treating ground-reflected waves, moving warheads and multiple blast sources. The loads are applied to facets defined with the keyword *LOAD_BLAST_SEGMENT. A database containing blast pressure history is also available (see *DATABASE_BINARY_BLSTFOR).

Card Summary:

Card Sets. Include as many sets of the following cards as necessary. This input terminates at the next keyword ("*") card.

Card 1. This card is required.

BID	M	XBO	YBO	ZBO	TBO	UNIT	BLAST

Card 2. This card is required.

CFM	CFL	CFT	CFP	NIDBO	DEATH	NEGPHS	

Card 3a. This card is included if BLAST = 3.

VEL	TEMP	RATIO	VID				

Card 3b. This card is included if BLAST = 4.

GNID	GVID				FLOOR		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	BID	M	XBO	YBO	ZBO	TBO	UNIT	BLAST
Type	I	F	F	F	F	F	I	I
Default	none	0.0	0.0	0.0	0.0	0.0	2	2

VARIABLE	DESCRIPTION
BID	Blast ID. A unique number must be defined for each blast source (charge). Multiple charges may be defined, however, interaction of the waves in air is not considered. LT.-2: Blast ID that can be referenced by LCID in *LOAD_ERODING_PART_SET, allowing the pressure from the blast to be used by *LOAD_ERODING_PART_SET.
M	Equivalent mass of TNT (see Remark 1).
XBO	x-coordinate of charge center.
YBO	y-coordinate of charge center.
ZBO	z-coordinate of charge center.
TBO	Time of detonation. See Remark 3.
UNIT	Unit conversion flag. See Remark 4. EQ.1: pound-mass, foot, second, psi EQ.2: kilogram, meter, second, Pascal (default) EQ.3: dozen slugs (i.e., lbf-s2/in), inch, second, psi EQ.4: centimeters, grams, microseconds, Megabars EQ.5: user conversions will be supplied (see Card 2) EQ.6: kilogram, millimeter, millisecond, GPa EQ.7: metric ton, millimeter, second, MPa EQ.8: gram, millimeter, millisecond, MPa
BLAST	Type of blast source (see Remark 5). EQ.1: Hemispherical surface burst – charge is located on or very near the ground surface (see Remark 7) EQ.2: Spherical air burst (default) – no amplification of the initial shock wave due to interaction with the ground surface EQ.3: Air burst – moving non-spherical warhead EQ.4: Air burst with ground reflection – initial shock wave impinges on the ground surface and is reinforced by the reflected wave to produce a Mach front. See Remark 8.

Card 2	1	2	3	4	5	6	7	8
Variable	CFM	CFL	CFT	CFP	NIDBO	DEATH	NEGPHS	
Type	F	F	F	F	I	F	I	
Default	0.0	0.0	0.0	0.0	none	10^{20}	0	

VARIABLE	DESCRIPTION
CFM	Conversion factor - pounds per LS-DYNA mass unit.
CFL	Conversion factor - feet per LS-DYNA length units.
CFT	Conversion factor - milliseconds per LS-DYNA time unit.
CFP	Conversion factor - psi per LS-DYNA pressure unit.
NIDBO	Optional node ID representing the charge center. If nonzero, XBO, YBO, and ZBO are ignored.
DEATH	Death time. Blast pressures are deactivated at this time.
NEGPHS	Treatment of negative phase pressure and positive phase duration (see Remark 9): EQ.0: Negative phase dictated by the Friedlander equation EQ.1: Negative phase pressure ignored EQ.10: Positive phase duration in closer agreement with ConWep results (not recommended). EQ.11: Positive phase duration in closer agreement with ConWep results and negative phase pressure ignored (recommended for comparisons with ConWep).

Moving Non-Spherical Warhead Card. Additional Card for BLAST = 3.

Card 3a	1	2	3	4	5	6	7	8
Variable	VEL	TEMP	RATIO	VID				
Type	F	F	F	F				
Default	0.0	70.0	1.0	none				

VARIABLE	DESCRIPTION
VEL	Speed of warhead
TEMP	Ambient air temperature, Fahrenheit
RATIO	Aspect ratio of the non- spheroidal blast front. This is the longitudinal axis radius divided by the lateral axis radius. Shaped charge and EFP warheads typically have significant lateral blast resembling an oblate spheroid with RATIO < 1. Cylindrically cased explosives produce more blast in the longitudinal direction, so RATIO > 1, rendering a prolate spheroid blast front, is more appropriate.
VID	Vector ID representing the longitudinal axis of the warhead (see *DEFINE_VECTOR). This vector is parallel to the velocity vector when a non-zero velocity VEL is defined.

Spherical Air Burst with Ground Reflection Card. Additional card for BLAST = 4.

Card 3b	1	2	3	4	5	6	7	8
Variable	GNID	GVID				FLOOR		
Type	I	I				I		
Default	none	none				0		

VARIABLE	DESCRIPTION
GNID	ID of node residing on the ground surface

VARIABLE	DESCRIPTION
GVID	ID of vector representing the vertically upward direction, that is, normal to the ground surface (see *DEFINE_VECTOR).
FLOOR	Treatment subterranean segments. EQ.0: Subterranean segments are not affected by the blast. EQ.1: Blast pressure is applied to segments residing below the ground plane. The pressure is computed for the ground plane location regardless of the segment's depth below the plane.

Remarks:

1. **Equivalent Mass of TNT.** Several methods can be used to approximate the equivalent mass of TNT for a given explosive. The simplest involves scaling the mass by the ratio of the Chapman-Jouguet detonation velocities given by:

$$M_{\text{TNT}} = M_e \frac{v_e^2}{v_{\text{TNT}}^2}$$

where M_{TNT} is the equivalent TNT mass and v_{TNT} is the Chapman-Jouguet (CJ) detonation velocity of TNT. M_e and v_e are, respectively, the mass and CJ velocity of the explosive under consideration. LS-DYNA takes the density of "Standard" TNT to be $1.57\,\text{g/cm}^3$ and v_{TNT} to be $0.693\,\text{cm}/\mu\text{s}$.

2. **Orientation.** The target's segments must be consistently oriented so that the normal vector of the portion of the target having a direct line of sight to the blast points roughly towards the blast, unless pressure is to be applied to the leeward side of a structure. For example, segments defining a structure being blasted from the outside should have outwardly oriented normal vectors, whereas a cavity containing a blast should have inwardly facing normal vectors. The angle of incidence is zero when the segment normal points directly at the charge. Only incident pressure is applied to a segment when the angle of incidence is greater than 90 degrees.

3. **Detonation Time.** The blast time offset TBO can be used to adjust the detonation time of the charge relative to the start time of the LS-DYNA simulation. The detonation time is delayed when TBO is positive. More commonly, TBO is set negative so that the detonation occurs before time-zero of the LS-DYNA calculation. Time is, therefore, not wasted while "waiting" for the blast wave to reach the structure. The following message, written to the messag and d3hsp files as well as the screen, is useful for setting TBO:

```
Blast wave reaches structure at 2.7832E-01 milliseconds
```

For example, one might run LS-DYNA for one integration cycle and record the arrival time listed in the message above. Then TBO is set to a negative number slightly smaller in magnitude than the reported arrival time, for example TBO = -0.275 milliseconds. For this case, the blast wave would reach the structure shortly after the start of the simulation.

4. **Units.** Computation of blast pressure relies on an underlying method which uses base units of lbm-foot-millisecond-psi; note that this internal unit system is inconsistent. Calculations require that the system of units in which the LS-DYNA model is constructed must be converted to this internal set of units. Predefined and user-defined unit conversion factors are available (see the parameter UNIT) and these unit conversion factors are echoed back in the d3hsp file. Below is an example of user-defined (UNIT = 5) conversion factors for the gm-mm-millisecond-MPa unit system.

$$1 = \left[\frac{\text{CFM} \times \text{lb}}{\text{LS-DYNA mass unit}}\right] = \left[\underbrace{2.2 \times 10^{-3}}_{=\text{CFM}} \frac{\text{lbm}}{\text{gm}}\right]$$
$$1 = \left[\frac{\text{CFL} \times \text{ft}}{\text{LS-DYNA length unit}}\right] = \left[3.28 \times 10^{-3} \frac{\text{ft}}{\text{mm}}\right]$$
$$1 = \left[\frac{\text{CFT} \times \text{ms}}{\text{LS-DYNA time unit}}\right] = \left[1.0 \frac{\text{ms}}{\text{ms}}\right]$$
$$1 = \left[\frac{\text{CFP} \times \text{psi}}{\text{LS-DYNA pressure unit}}\right] = \left[145.0 \frac{\text{psi}}{\text{MPa}}\right]$$

5. **Limits of Model's Validity.** Define a scaled distance, Z, as follows:

$$Z = \frac{R}{M_{\text{TNT}}^{1/3}},$$

where R is the distance from the charge center to the target and M_{TNT} is the equivalent TNT mass (see Remark 1). The scaled distance determines the range of validity for these models. The spherical airburst model is valid for Z ranging from 0.37 ft/lbm$^{1/3}$ (0.147 m/kg$^{1/3}$) out to 100 ft/lbm$^{1/3}$ (40 m/kg$^{1/3}$). The hemispherical surface burst is valid for Z ranging from 0.45 ft/lbm$^{1/3}$ (0.178 m/kg$^{1/3}$) out to 100 ft/lbm$^{1/3}$ (40 m/kg$^{1/3}$).

For the spherical air burst with ground reflection (BLAST = 4), the same scaled distance can be used to determine validity, except R is now the height of the charge center above the ground. This model is valid for Z ranging from 1.0 ft/lbm$^{1/3}$ (0.397 m/kg$^{1/3}$) out to 7.0 ft/lbm$^{1/3}$ (2.78 m/kg$^{1/3}$)

6. **Axisymmetric Analyses.** Blast loads can be used in 2D axisymmetric analyses. Repeat the second node for the third and fourth nodes of the segment definition in *LOAD_BLAST_SEGMENT and *LOAD_BLAST_SEGMENT_SET.

7. **Surface Burst Model.** The surface blast model is ideal for when a charge is located on or very near the ground. In this case the initial blast wave is

immediately reflected and reinforced by the nearly rigid ground (the model accounts for yielding) to produce a hemispherical reflected wave from the point of the burst. This reflected wave merges with the initial incident wave producing overpressures which are greater than those produced by the initial wave alone. Target points equidistant from the burst point are loaded identically with the surface burst option.

8. **BLSTFOR Components.** The BLSTFOR file (see *DATABASE_BINARY_BLSTFOR) contains blast pressures, blast wind velocity, air density, and wave index. The "wave index", relevant only for BLAST = 4, denotes the nature of the blast pressure applied to the segment, such that,

   EQ.-1: below the ground plane

   EQ.0: no blast pressure

   EQ.1: primary incident wave

   EQ.2: detached ground-reflected wave

   EQ.3: Mach stem region

9. **ConWep.** ConWep is neither embedded in nor coupled with LS-DYNA. The *LOAD_BLAST_ENHANCED feature is based on the work of Randers-Pehrson and Bannister. NEGPHS = 11 gives positive phase duration in close agreement with ConWep, and it zeroes out the negative phase pressure, as in ConWep. NEGPHS = 10 should not be used for analysis because it presents a significant negative phase pressure; it is included only to illustrate this unwanted effect. *LOAD_BLAST_ENHANCED combines incident and reflected pressure according to the work of Randers-Pehrson and Bannister while ConWep computes reflected pressure in a different manner.

10. **Internally Used Values for Ambient Air Pressure and Density.** The following values for ambient air pressure and density are used internally:

$$p = 101.325 \text{ kPa}$$
$$\rho = 1.225 \text{ kg/m}^3$$

These values come into play only when the blast load is coupled to ALE air (see ALEPID in LOAD_BLAST_SEGMENT). Accordingly, the initial density of the air should be set (in *MAT) as well as pressure (through *EOS). Those values should reflect the unit system selected with UNIT in *LOAD_BLAST_ENHANCED.

*LOAD_BLAST_SEGMENT

Purpose: Apply blast pressure loading over a triangular or quadrilateral segment for 3D geometry or line segment for 2D geometry (see *LOAD_BLAST_ENHANCED).

Segment Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	BID	N1	N2	N3	N4	ALEPID	SFNRB	SCALEP
Type	I	I	I	I	I	I	F	F
Default	none	none	none	none	none	none	0.	1.

VARIABLE	DESCRIPTION
BID	Blast source ID (see *LOAD_BLAST_ENHANCED).
N1	Node ID.
N2	Node ID.
N3	Node ID. For line segments on two-dimensional geometries set N3 = N2.
N4	Node ID. For line segments on two-dimensional geometries set N4 = N3 = N2 or for triangular segments in three diemensions set N4 = N3.
ALEPID	Part ID of ALE ambient part underlying this segment to be loaded by this blast (see *PART and *SECTION_SOLID, AET = 5). This applies only when the blast load is coupled to an ALE air domain.
SFNRB	Scale factor for the ambient element non-reflecting boundary condition. Shocks waves reflected back to the ambient elements can be attenuated with this feature. A value of 1.0 works well for most situations. The feature is disabled when a value of zero is specified
SCALEP	Pressure scale factor.

*LOAD_BLAST_SEGMENT_SET

Purpose: Apply blast pressure loading over each segment in a segment set (see *LOAD_-BLAST_ENHANCED).

Segment Set Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	BID	SSID	ALEPID	SFNRB	SCALEP			
Type	I	I	I	F	F			
Default	none	none	↓	0.	1.			

VARIABLE	DESCRIPTION
BID	Blast source ID (see *LOAD_BLAST_ENHANCED).
SSID	Segment set ID (see *SET_SEGMENT).
ALEPID	Part ID of ALE ambient part underlying this segment to be loaded by this blast (see *PART and *SECTION_SOLID, AET = 5). This applies *only* when the blast load is coupled to an ALE air domain.
SFNRB	Scale factor for the ambient element non-reflecting boundary condition. Shocks waves reflected back to the ambient elements can be attenuated with this feature. A value of 1.0 works well for most situations.
SCALEP	Pressure scale factor.

Remarks:

1. **Triangular Segments.** Triangular segments are defined by setting N4 = N3.

2. **Line Segments.** Line segments for two-dimensional geometries are defined by setting N4 = N3 = N2.

***LOAD_BODY**_*OPTION*

Available options include for base accelerations:

X

Y

Z

for angular velocities:

RX

RY

RZ

for loading in any direction, specified by vector components (see About Keyword Option VECTOR):

VECTOR

and to specify a part set:

PARTS

Purpose: Define body force loads due to a prescribed base acceleration or angular velocity using global axes directions. This option applies nodal forces only: it cannot be used to prescribe translational or rotational motion. By default, these body forces do not take into account non-physical mass added via mass scaling, see DT2MS on *CONTROL_TIMESTEP. However, if in addition EMSCL > 0 on *CONTROL_TIMESTEP, then EMSCL ×100% of the added mass contributes to the gravity load.

NOTE: This data applies to all nodes in the complete problem unless a part subset is specified via the *LOAD_BODY_PARTS keyword.

If a part subset with *LOAD_BODY_PARTS, then all nodal points belonging to the subset will have body forces applied.

NOTE: Only one *LOAD_BODY_PARTS card is permitted per deck. To specify, for instance, one body load on one part and another body load on another part use *LOAD_BODY_GENERALIZED instead.

Card Summary:

Card 1a.1. This card is included if and only if the keyword option is X, Y, Z, RX, RY, RZ, or VECTOR.

LCID	SF	LCIDDR	XC	YC	ZC	CID	

Card 1a.2. This card is included if and only if the VECTOR keyword option is used.

V1	V2	V3					

Card 1b. This card is included if and only if the PARTS keyword option is used.

PSID							

Data Cards:

For options X, Y, Z, RX, RY, RZ and VECTOR.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	LCID	SF	LCIDDR	XC	YC	ZC	CID	
Type	I	F	I	F	F	F	I	
Default	none	1.	↓	0.	0.	0.	global	

VARIABLE	DESCRIPTION
LCID	Load curve ID specifying loading, see *DEFINE_CURVE. See Remark 1.
SF	Load curve scale factor. Applies to both LCID and LCIDDR.
LCIDDR	Load curve ID for dynamic relaxation phase (optional). This is needed when dynamic relaxation is defined and a different load curve to LCID is required during the dynamic relaxation phase. Note if LCID is undefined, then no body load will be applied during dynamic relaxation regardless of the value LCIDDR. See *CONTROL_DYNAMIC_RELAXATION. See Remark 1.
XC	x-center of rotation, define for angular velocities.
YC	y-center of rotation, define for angular velocities.

VARIABLE	DESCRIPTION
ZC	z-center of rotation, define for angular velocities.
CID	Coordinate system ID to define acceleration in local coordinate system. The accelerations (LCID) are with respect to CID. EQ.0: Global

For option VECTOR.

Card 1a.2	1	2	3	4	5	6	7	8
Variable	V1	V2	V3					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
V1, V2, V3	Components of vector **V**

For option PARTS.

Card 1b	1	2	3	4	5	6	7	8
Variable	PSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PSID	Part set ID

Remarks:

1. **Base Accelerations.** Translational base accelerations allow body force loads to be imposed on a structure. Conceptually, base acceleration may be thought of as accelerating the coordinate system in the direction specified, and, thus, the

inertial loads acting on the model are of opposite sign. For example, if a cylinder were fixed to the *yz*-plane and extended in the positive *x*-direction, then a positive *x*-direction base acceleration would tend to shorten the cylinder, i.e., create forces acting in the negative *x*-direction.

Base accelerations are frequently used to impose gravitational loads during dynamic relaxation to initialize the stresses and displacements. During the analysis, in this latter case, the body forces loads are held constant to simulate gravitational loads. When imposing loads during dynamic relaxation, the load curve should slowly ramp up to avoid the excitation of a high frequency response.

2. **Angular Velocity.** Body force loads due to the angular velocity about an axis are calculated with respect to the deformed configuration and act radially outward from the axis of rotation. Torsional effects which arise from changes in angular velocity are neglected with this option. The angular velocity is assumed to have the units of radians per unit time.

3. **Body Force Density.** The body force density is given at a point **P** of the body by:

$$\mathbf{b} = \rho[\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})] \ ,$$

where ρ is the mass density, $\boldsymbol{\omega}$ is the angular velocity vector, and **r** is a position vector from the origin to point **P**. Although the angular velocity may vary with time, the effects of angular acceleration are not included.

4. **Transient Deformation.** Angular velocities are useful for studying transient deformation of spinning three-dimensional objects. Typical applications have included stress initialization during dynamic relaxation where the initial rotational velocities are assigned at the completion of the initialization, and this option ceases to be active.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *LOAD_BODY_Z
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Add gravity such that it acts in the negative Z-direction.
$  Use units of mm/ms2.  Since gravity is constant, the load
$  curve is set as a constant equal to 1.  If the simulation
$  is to exceed 1000 ms, then the load curve needs to be
$  extended.
$
$$$  Note: Positive body load acts in the negative direction.
$
*LOAD_BODY_Z
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
```

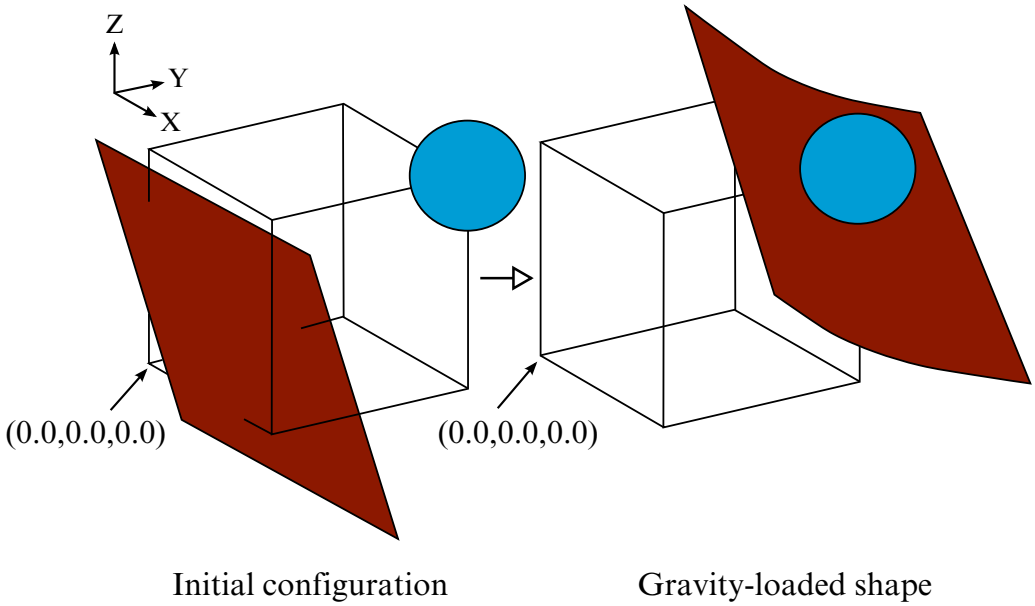


Figure 33-3. A validation example for option **VECTOR**.

```
$     lcid        sf    lciddr        xc        yc        zc
         5   0.00981
$
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
         5
$
$           abscissa            ordinate
               0.00               1.000
            1000.00               1.000
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

About Keyword Option VECTOR:

The vector **V** defines the direction of the body force. Body forces act in the negative direction of the vector **V**.

In the example shown in Figure 33-3, a rectangular sheet metal blank is loaded with gravity which causes it to be deformed by a fixed rigid ball. Given the global coordinate system shown, if the part set ID of the blank is 1, the keywords responsible for specifying the body force (in units of mm, second, tonne and Newton) in positive direction of $(1.0, 1.0, 1.0)$ will be as follows,

```
*LOAD_BODY_PARTS
1
*LOAD_BODY_VECTOR
101, 9810.0
-1.0, -1.0, -1.0
*DEFINE_CURVE
101
0.0, 1.0
10.0, 1.0
```

It is note that straight lines represent a cube with each edge length of 500.0 mm.

Revision Information:

The VECTOR keyword option is available in LS-DYNA R5 Revision 59290 and later releases.

LOAD_BODY_GENERALIZED_*OPTION

Available options include:

SET_NODE

SET_PART

Purpose: Define body force loads due to a prescribed base acceleration or a prescribed angular motion over a subset of the complete problem. The subset is defined by using nodes or parts. Warning: Nodes, which belong to rigid bodies, should not be specified. Rigid bodies must be included within the part sets definitions.

The body forces defined using this command do not take into account non-physical mass added via mass scaling; see *CONTROL_TIMESTEP.

Card Sets. Include as many sets of Cards 1 and 2 as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	N1/SID	N2	LCID	DRLCID	XC	YC	ZC	
Type	I	I	I	I	F	F	F	
Default	none	none	none	0	0.	0.	0.	

Card 2	1	2	3	4	5	6	7	8
Variable	AX	AY	AZ	OMX	OMY	OMZ	CID	ANGTYP
Type	F	F	F	F	F	F	I	A
Default	0.	0.	0.	0.	0.	0.	global	CENT
Remarks	1, 2	1, 2	1, 2	3, 4, 5	3, 4, 5	3, 4, 5		3

VARIABLE	DESCRIPTION
N1/SID	Beginning node ID for body force load or the node or part set ID.

VARIABLE	DESCRIPTION
N2	Ending node ID for body force load. Set to zero if a set ID is defined.
LCID	Load curve ID; see *DEFINE_CURVE.
DRLCID	Load curve ID for dynamic relaxation phase. Only necessary if dynamic relaxation is defined. See *CONTROL_DYNAMIC_RELAXATION.
XC	x-center of rotation. Define only for angular motion.
YC	y-center of rotation. Define only for angular motion.
ZC	z-center of rotation. Define only for angular motion.
AX	Scale factor for acceleration in x-direction
AY	Scale factor for acceleration in y-direction
AZ	Scale factor for acceleration in z-direction
OMX	Scale factor for x-angular velocity or acceleration
OMY	Scale factor for y-angular velocity or acceleration
OMZ	Scale factor for z-angular velocity or acceleration
CID	Coordinate system ID to define acceleration in the local coordinate system. The coordinate (XC, YC, ZC) is defined with respect to the local coordinate system if CID is nonzero. The accelerations, LCID and their scale factors are with respect to CID. EQ.0: global
ANGTYP	Type of body loads due to angular motion: EQ.CENT: body load from centrifugal acceleration, $\rho[\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})]$. EQ.CORI: body load from Coriolis-type acceleration, $2\rho(\boldsymbol{\omega} \times \mathbf{v})$. EQ.ROTA: body load from rotational acceleration, $\rho(\boldsymbol{\alpha} \times \mathbf{r})$,

VARIABLE	DESCRIPTION
	where $\boldsymbol{\omega}$ is the angular velocity, $\boldsymbol{\alpha}$ is the angular acceleration, **r** is the position vector relative to center of rotation and **v** is the velocity vector

Remarks:

1. **Base Accelerations.** Translational base accelerations allow body forces loads to be imposed on a structure. Conceptually, base acceleration may be thought of as accelerating the coordinate system in the direction specified, and, thus, the inertial loads acting on the model are of opposite sign. For example, if a cylinder were fixed to the *yz*-plane and extended in the positive *x*-direction, then a positive *x*-direction base acceleration would tend to shorten the cylinder, that is, create forces acting in the negative *x*-direction.

2. **Gravitational Loads.** Base accelerations are frequently used to impose gravitational loads during dynamic relaxation to initialize the stresses and displacements. During the analysis, in this latter case, the body forces loads are held constant to simulate gravitational loads. When imposing loads during dynamic relaxation, the load curve should slowly ramp up to avoid the excitation of a high frequency response.

3. **Angular Motion.** Body force loads due to the angular motion about an axis are calculated with respect to the deformed configuration. When ANGYP = CENT or CORI, torsional effects which arise from changes in angular velocity are neglected. Such torsional effects can be taken into account by setting ANGTYP = ROTA. The angular velocity is assumed to have the units of radians per unit time, accordingly angular acceleration has the units of radians/time^2.

4. **Body Force Density.** With ANGTYP = CENT, the body force density at a point **P** of the body (similar to *LOAD_BODY) is given by:

$$\mathbf{b} = \rho[\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})] \ ,$$

   where ρ is the mass density, $\boldsymbol{\omega}$ is the angular velocity vector, and **r** is a position vector from the origin to point **P**.

   One of the enhancements in *LOAD_BODY_GENERALIZED, compared to *LOAD_BODY, is that it provides additional options ANGTYP = CORI and ANGTYP = ROTA to include Coriolis and Euler forces, respectively.

5. **Transient Deformation.** Angular velocities are useful for studying transient deformation of spinning three-dimensional objects. Typical applications have included stress initialization during dynamic relaxation where the initial

rotational velocities are assigned at the completion of the initialization, and this option ceases to be active.

*LOAD_BODY_POROUS

Purpose: Define the effects of porosity on the flow with body-force-like loads applied to the ALE element nodes. Ergun porous flow assumptions are used. This only applies to non-deformable (constant-porosity), fully saturated porous media. This model only works with a non-zero and constant viscosity fluid defined using either *MAT_NULL or *MAT_ALE_VISCOUS card.

Card Sets. Include as many sets of Cards 1 and 2 as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SIDTYP	AX	AY	AZ	BX	BY	BZ
Type	I	I	F	F	F	F	F	F
Default	none	0	0.0	0.0	0.0	0.0	0.0	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	AOPT							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
SID	Set ID of the ALE fluid part subjected to porous flow condition.
SIDTYP	Set ID type of the SID above: EQ.0: Part set ID (default) EQ.1: Part ID EQ.2: Element set ID (*SET_BEAM for 1D ALE, *SET_SHELL for 2D ALE, and *SET_SOLID for 3D ALE)
AX, AY, AZ	Viscous coefficients for viscous terms in global x, y, and z directions (see Remark 1). If $A_x \neq 0.0$ and $A_y = A_z = 0.0$, then an isotropic

VARIABLE	DESCRIPTION
	viscous permeability condition is assumed for the porous medium.
BX, BY, BZ	Inertial coefficients for inertia terms in global x, y, and z directions (see Remarks 1 and 2). If $B_x \neq 0.0$, and $B_y = B_z = 0.0$, then an isotropic inertial permeability condition is assumed for the porous medium.
AOPT	Material axis option: EQ.0: Inactive EQ.1: The forces are applied in a local system attached to the ALE solid (see CTYPE = 12 and DIREC = 1 in *CONSTRAINED_LAGRANGE_IN_SOLID).

Remarks:

1. **Ergun porous flow model.** Consider the basic general Ergun equation for porous flow in one direction:

$$\frac{\Delta P}{\Delta L} = \frac{\mu}{k_1} V_s + \frac{\rho}{k_2} {V_s}^2 \ ,$$

where

$$\rho = \text{Fluid Density}$$
$$\mu = \text{Fluid dynamic vicosity}$$
$$V_s = \frac{4Q}{\pi D^2} = \text{Superficial fluid velocity}$$
$$Q = \text{Overall volume flow rate } \left(\frac{\text{m}^3}{\text{s}}\right)$$
$$D = \text{Porous channel characteristic width (perpendicular to } \Delta\text{L)}$$
$$k_1 = \frac{\varepsilon^3 {d_p}^2}{150(1-\varepsilon)^2} = \text{Permeability parameter}$$
$$k_2 = \frac{\varepsilon^3 d_p}{1.75(1-\varepsilon)} = \text{Passability parameter}$$
$$\varepsilon = \text{Porosity} = \frac{\text{total pore volume}}{\text{total media volume}}$$
$$d_p = \text{Effective particle diameter}$$

The above equation can be generalized into 3 dimensional flows where each component may be written as

$$-\frac{dP}{dx_i} = A_i \mu V_i + B_i \rho |V_i| V_i$$

where $i = 1,2,3$ refers to the global coordinate directions (no summation intended for repeated indices), μ is the constant dynamic viscosity, ρ is the fluid density, V_i is the fluid velocity components, A_i is analogous to k_1 above, and B_i is analogous to k_2 above. A matrix version can be defined by ALE elements with *DEFINE_POROUS_ALE.

2. **Inertia coefficients.** If $B_i = 0$, the equation is reduced to simple Darcy Law for porous flow (may be good for sand-like flow). For coarse grain (rocks) media, the inertia term will be important, so you need to input these coefficients.

*LOAD_BRODE

Purpose: Define Brode function for application of pressure loads due to explosion; see Brode [1970]. See also *LOAD_SEGMENT, *LOAD_SEGMENT_SET, or *LOAD_SHELL.

Card 1	1	2	3	4	5	6	7	8
Variable	YLD	BHT	XBO	YBO	ZBO	TBO	TALC	SFLC
Type	F	F	F	F	F	F	I	I
Default	0.0	0.0	0.0	0.0	0.0	0.0	0	0

Card 2	1	2	3	4	5	6	7	8
Variable	CFL	CFT	CFP					
Type	F	F	F					
Default	0.0	0.0	0.0					

VARIABLE	DESCRIPTION
YLD	Yield, *W*, (Kt, equivalent tons of TNT). This parameter is constant yield unless SFLC is defined. In that case it becomes a parameter for calculating the variable yield where it will be referenced as YLD. See Remark 1.
BHT	Height of burst.
XBO	*x*-coordinates of Brode origin.
YBO	*y*-coordinates of Brode origin.
ZBO	*z*-coordinates of Brode origin.
TBO	Time offset of Brode origin.
TALC	Load curve number giving time of arrival as a function of range relative to Brode origin (space, time). See *DEFINE_CURVE and Remark 1.

VARIABLE	DESCRIPTION
SFLC	Load curve number giving yield scaling, W_s, as a function of scaled time. Scaled time is time relative to the Brode origin, t_B, divided by $\text{YLD}^{1/3}$, that is, $t_B/\text{YLD}^{1/3}$. See *DEFINE_CURVE and Remark 1.
CFL	Conversion factor - kft to LS-DYNA length units.
CFT	Conversion factor - milliseconds to LS-DYNA time units.
CFP	Conversion factor - psi to LS-DYNA pressure units.

Remarks:

1. **Yield Parameter.** If the curves, TALC and SFLC, are defined, a variable yield as opposed to a constant yield is assumed. *Both* load curves must be specified for the variable yield option. If this option is used, the shock time of arrival is found from the time of arrival curve. The yield used in the Brode formulas is computed by taking the yield scaling value, W_s, at the current scaled time ($t_B/\text{YLD}^{1/3}$) and multiplying W_s by the constant yield input in field YLD, that is,

$$W\Big|_{t_B} = W_s\Big|_{t_B/\text{YLD}^{1/3}} \times \text{YLD} \ .$$

*LOAD_DENSITY_DEPTH

Purpose: Define density versus depth for gravity loading. This option has been occasionally used for analyzing underground and submerged structures where the gravitational preload is important. The purpose of this option is to initialize the hydrostatic pressure field at the integration points in the element. See also GRAV in *PART.

This card should be only defined once in the input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	GC	DIR	LCID				
Type	I	F	I	I				
Default	0	0.0	1	none				
Remarks	1,2			3				

VARIABLE	DESCRIPTION
PSID	Part set ID, see *SET_PART. If a PSID of zero is defined, all parts are initialized.
GC	Gravitational acceleration value
DIR	Direction of loading: EQ.1: Global x EQ.2: Global y EQ.3: Global z
LCID	Load curve ID defining density as a function of depth; see *DEFINE_CURVE.

Remarks:

1. **Hydrostatic pressure.** Density as a function of depth curves are used to initialize hydrostatic pressure due to gravity acting on an overburden material. The hydrostatic pressure acting at a material point at depth, d, is given by:

$$p = -\int_{d}^{d_{\text{surface}}} \rho(z) g dz \ ,$$

where p is pressure, d_{surface}, is the depth of the surface of the material to be initialized (usually zero), $\rho(z)$ is the mass density at depth z, and g is the acceleration of gravity. This integral is evaluated for each integration point. Depth may be measured along any of the global coordinate axes, and the sign convention of the global coordinate system should be respected. The sign convention of gravity also follows that of the global coordinate system. For example, if the positive z-axis points "up", then gravitational acceleration should be input as a negative number.

2. **Limitations.** For this option there is a limit of 12 parts that can be defined by PSID, unless all parts are initialized.

3. **Depth LCID.** Depth is the ordinate of the curve and is input as a descending x, y, or z coordinate value. Density is the abscissa of the curve and must vary (increase) with depth, that is, an infinite slope is not allowed.

***LOAD_EDGE_UVW**_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Apply a uniformly distributed load on a parametric edge or a set of parametric edges.

Edge/Edge Set Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EUID	DOF	LCID	SF				
Type	I	I	I	F				
Default	none	none	none	1.				

VARIABLE	DESCRIPTION
EUID	Parametric edge ID or parametric edge set ID when using the SET keyword option
DOF	Applicable load direction: EQ.1: x-direction of load action EQ.2: y-direction of load action EQ.3: z-direction of load action EQ.5: Moment about the x-axis EQ.6: Moment about the y-axis axis EQ.7: Moment about the z-axis axis
LCID	Load curve ID (see *DEFINE_CURVE) giving the load as a function of time
SF	Load curve scale factor

*LOAD_ERODING_PART_SET

Purpose: Apply a pressure load to the exposed surface composed of solid elements that may erode.

Card Sets. Include as many sets of Cards 1 and 2 as necessary. This input terminates at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	ID	LCID	SF	AT	PSID	BOXID	MEM	ALPHA
Type	I	I	F	F	I	I	I	F
Default	none	none	1.0	0.0	none	0	50	80

Card 2	1	2	3	4	5	6	7	8
Variable	IFLAG	X	Y	Z	BETA			
Type	I	F	F	F	F			
Default	0	0.0	0.0	0.0	90			

VARIABLE	DESCRIPTION
ID	ID number.
LCID	Load curve ID defining pressure as a function of time; see *DEFINE_CURVE. LT.0: Pressure determined from certain *LOAD keywords. See Remark 1.
SF	Scale factor.
AT	Arrival time.
PSID	Part set ID; see *SET_PART.

VARIABLE	DESCRIPTION
BOXID	Box ID; see *DEFINE_BOX. Any segment that would otherwise be loaded but whose centroid falls outside of this box is not loaded.
MEM	Extra memory, in percent, to be allocated above the initial memory for storing the new load segments exposed by the erosion.
ALPHA	The maximum angle (in degrees) permitted between the normal of a segment at its centroid and the average normal at its nodes. This angle is used to eliminate interior segments.
IFLAG	Flag for choosing a subset of the exposed surface that is oriented towards a blast or other loading source. The vector from the center of the element to the source location must be within an angle of BETA of the surface normal. If IFLAG > 0, then the subset is chosen, otherwise if IFLAG = 0, the entire surface is loaded.
X, Y, Z	Optional source location.
BETA	Maximum permitted angle (in degrees) between the surface normal and the vector to the source. The exposed segment is not loaded if the calculated angle is greater than BETA.

Remarks:

1. **Special Values Input for LCID.** If LCID is input as:

   a) -1, then the Brode function is used to determine the pressure for the segments; see *LOAD_BRODE.

   b) -2, then an empirical air blast function is used to determine the pressure for the segments; see *LOAD_BLAST.

   c) An integer < -2, then LCID references BID from *LOAD_BLAST_ENHANCED and the air blast function specified with the referenced *LOAD_BLAST_ENHANCED instantiation is used to determine the pressure for the segments.

2. **Load Curve Multipliers.** The load curve multipliers may be used to increase or decrease the pressure. The time value is not scaled.

3. **Activation Time.** The activation time, AT, is the time during the solution that the pressure begins to act. Until this time, the pressure is ignored. The function value of the load curves will be evaluated at the offset time given by the difference of the solution time and AT, that is solution time − AT.

4. **Requirements.** For proper evolution of the loaded surface, it is a requirement that DTMIN in *CONTROL_TERMINATION be greater than zero and ERODE in *CONTROL_TIMESTEP be set to 1.

*LOAD_EXPANSION_PRESSURE_{OPTION}

To define an ID for the pressure loading, the following option is available:

ID

Purpose: Apply a uniform pressure to a section of a chamber that can vary in size due to a moving edge. For instance, this keyword can be used to apply pressure to a cylinder on one side of a moving piston. See Figure 33-4.

ID Card. Additional card needed for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	Heading						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	LCID	SF	AT				
Type	I	I	F	F				
Default	none	none	1.0	0.0				

Card 2	1	2	3	4	5	6	7	8
Variable	NSID	XN	YN	ZN				
Type	I	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
ID	Loading ID
Heading	Description of the loading

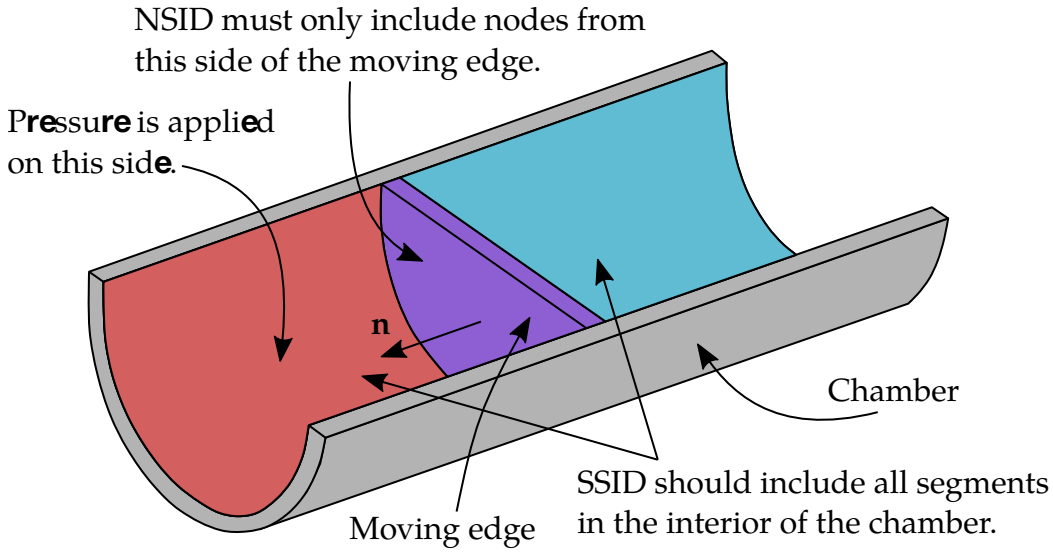


Figure 33-4. Schematic view of half a chamber (cut for clarity). Pressure is applied through this keyword to the red region but not the blue region of the chamber. The normal of the moving edge (in purple) must point toward the part of the chamber where the pressure is applied.

VARIABLE	DESCRIPTION
SSID	Segment set ID which specifies the interior of the chamber. As the edge moves, the pressure is applied or could be applied to these segments.
LCID	Load curve ID that defines the pressure as a function of time
SF	Load curve scale factor
AT	Activation time which is the time at which the pressure begins to be applied. Before this time, pressure will not be applied to the chamber. See Remark 2.
NSID	Node set ID that defines the moving edge/plane of the dynamic chamber. Note that this node set must include at least 3 nodes to define a plane. See Remark 1.
XN	*X* component of the initial outward normal of the moving plane/edge. See Remark 1.
YN	*Y* component of the initial outward normal of the moving plane/edge
ZN	*Z* component of the initial outward normal of the moving plane/edge

Remarks:

1. **Moving plane/edge.** NSID defines the position of the moving plane/edge. It must include enough nodes (at least 3) on the side facing where the pressure is applied to define a plane. Note that the moving plane/edge is not loaded with this keyword. It should be loaded with the same load curve, LCID, as this keyword using *LOAD_SEGMENT_SET.

   The reference normal direction of the plane/edge defined by XN, YN, and ZN should point toward the part of the chamber where the pressure is applied. At each time step, the normal direction of the edge is found by applying the right-hand rule to the node set in the order of the nodes in the set. Note that for a non-flat edge, the normal can be an average or estimate.

2. **Activation time.** The activation time, AT, is the time during the solution that the pressure begins to act. Until this time, the pressure is ignored. The function value of the load curves will be evaluated at the offset time given by the difference of the solution time and AT, that is, solution time − AT.

*LOAD_EXTERNAL_VARIABLE

Purpose: Define the nodal values of an external variable α with a user-defined ID. The value can be given as a function of time for individual nodes or node sets or read from the temperature field from a previous simulation. For nodes that are not explicitly accounted for by the keyword definition, default values can be set. If this keyword is used more than once, each external variable must have a unique ID. The external variables can be used to influence the material in certain structural and thermal materials. They can also be used as an electromagnetic property for influencing shorts with *EM_RANDLES_SHORT. Additionally, a volumetric expansion can be defined based on the distribution of an external variable using *MAT_ADD_EXTVAR_EXPANSION.

Card Summary:

Card 1. This card is required.

VID	DBAS	DSCA	DLCID	NMP	NTMP	NEMP	

Card 2. Include NMP instantiations of this card.

IMP	PID	PTYP					

Card 3. Include NTMP instantiations of this card.

ITMP	TPID	TPTYP					

Card 4. Include NEMP instantiations of this card.

IEMP	EID	ETYP					

Card 5. Include as many instantiations of this card or of this card in a set with Card 5.1 if IDTYP = 3 as desired. This input ends at the next keyword ("*") card.

ID	IDTYP	BAS	SCA	LCID			

Card 5.1. Include this card in a set of Card 5 and this card if IDTYP = 3.

FILENAME

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	VID	DBAS	DSCA	DLCID	NMP	NTMP	NEMP	
Type	I	F	F	I	I	I	I	
Default	1	0.0	1.0	none	0	0	0	

VARIABLE	DESCRIPTION
VID	Variable ID. If this keyword contains more than one *LOAD_EXTERNAL_VARIABLE instantiation, each variable must have a unique ID.
DBAS	Base value $\tilde{\alpha}_B$. It is the default used for nodes that are not specified in a node or node set card (Card 2). See Remark 4.
DSCA	Scaled value $\tilde{\alpha}_S$. It is the default used for nodes that are not specified in a node or node set card (Card 2). See Remark 4.
DLCID	Load curve ID defining a scaling $\tilde{f}(t)$ as function of time. This is the default for all nodes not specified in a node or node set card (Card 2). See Remark 4.
NMP	Number of material properties to be defined
NTMP	Number of thermal material properties to be defined
NEMP	Number of electromagnetic properties to be defined

Material Properties Card. Include NMP cards.

Card 2	1	2	3	4	5	6	7	8
Variable	IMP	PID	PTYP					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
IMP	Index of the material property. See Remark 5.
PID	Part or part set ID. For the parts referenced, the material property associated with material property index IMP will be modified by the state of the external variable. See Remark 5.
PTYPE	ID type of PID: EQ.0: Part ID EQ.1: Part set ID

Thermal Material Properties Card. Include NTMP cards.

Card 3	1	2	3	4	5	6	7	8
Variable	ITMP	TPID	TPTYP					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
ITMP	Index of the thermal material property. See Remark 6.
TPID	Part or part set ID. For parts referenced here, the thermal material property associated with material property index ITMP will be modified by the state of the external variable. See Remark 6.
TPTYPE	ID type of TPID: EQ.0: Part ID EQ.1: Part set ID

EM Properties Card. Include NEMP cards.

Card 4	1	2	3	4	5	6	7	8
Variable	IEMP	EID	ETYP					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
IEMP	Index of the electromagnetic property. See Remark 7.
EID	Function ID. For entities referenced here, the electromagnetic property associated with property index IEMP will be modified by the state of the external variable. See Remark 7.
ETYPE	ID type of EID: EQ.2: Function ID

Node/Node Set/LSDA Cards. Include as many cards or card sets of this card and Card 5.1 for the LSDA option (IDTYP = 3) in the following format as desired. This input ends at the next keyword ("*") card. See Remark 3.

Card 5	1	2	3	4	5	6	7	8
Variable	ID	IDTYP	BAS	SCA	LCID			
Type	I	I	F	F	I			
Default	none	1	0.	1.0	none			

Additional LSDA file cards for IDTYP=3. Additional input card needed in a set with Card 5 if IDTYP = 3 (LSDA option) in Card 5.

Card 5.1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

VARIABLE	DESCRIPTION
ID	Node or node set ID. Ignored for IDTYP = 3.
IDTYP	ID type (see Remark 3): EQ.1: Node ID. See Remark 1. EQ.2: Node set ID. See Remark 1. EQ.3: LSDA option (rest of this Card 5 is ignored in this case). See Remark 2.
BAS	Base value α_B used for nodes defined by ID and IDTYP. See Remark 1. Ignored for IDTYP = 3.
SCA	Scaling parameter α_S used for nodes defined by ID and IDTYP. See Remark 1. Ignored for IDTYP = 3.
LCID	Load curve ID defining a scaling $f(t)$ as a function of time used for nodes defined by ID and IDTYP. See Remark 1. Ignored for IDTYP = 3.
FILENAME	Name of an LSDA file that contains temperature information that is interpreted as the value of the external variable. See Remark 2.

Remarks:

1. **Tabular input data for specific nodes.** For nodes referenced by their node ID (IDTYP = 1) or by a node set ID (IDTYP = 2), the value α of the external variable VID can be defined as a function of time. Using tabulated data (see *DEFINE_CURVE), α is defined as

$$\alpha = \alpha_B + \alpha_S \times f(t)$$

where $f(t)$ is the current value of the load curve LCID, α_S is the scaling parameter SCA, and α_B is the base value BAS. Note that these parameters are defined individually for each node or node set.

2. **LSDA option.** With IDTYP = 3, LS-DYNA interprets the temperature results from a previous simulation as a nodal distribution of an external variable varying in time. Here, a binary LSDA file must be specified as input, which should provide the same information as the TPRINT section in the binary database binout.

   The algorithm then loops over the nodes in the LSDA file, reads the respective temperature values, and uses that value as external variable data if a node with the same user ID is found in the current model.

3. **Combining input options.** It is possible to define the external variable for an arbitrary number of nodes, node sets, and LSDA files by repetitive input of Card 5 or Card 5 and Card 5.1. Any combination is possible. If a node is included more than once, such as in two different sets, the input order is important because data from the last entry overwrites previous data for that node.

4. **Tabular input for default values.** For nodes not included in Card 5 or the set of Cards 5 and 5.1, default values can be defined. The treatment is very similar to that of specified nodes as discussed in Remark 1. Tabular data can be used to define the default value, $\tilde{\alpha}$, of the external variable as:

   $$\tilde{\alpha} = \tilde{\alpha}_B + \tilde{\alpha}_S \times \tilde{f}(t).$$

   Here, load curve DLCID defines the function $\tilde{f}(t)$, parameter DSCA the scaling factor $\tilde{\alpha}_S$, and parameter DBAS the base value $\tilde{\alpha}_B$.

5. **Defining structural material properties.** For all elements in a given part or part set, the external variable can be used to locally modify one or more material properties. The material property index IMP defines which material property the external variable field influences. The following structural material models currently support the definition of certain properties via external variables: *MAT_106, *MAT_251, and *MAT_254. Tables referencing the material indexes and the material properties can be found in the respective material sections of the Material Models manual

6. **Defining thermal material properties.** The external variable can be used to locally modify one or more thermal material properties for a part or part set. The material property index ITMP defines which material property the external variable field influences. Currently, *MAT_T08 and *MAT_T10 support external variable modifications. The meanings of the material indexes are shown in tables in the respective material sections of the Material Models manual.

7. **Defining electromagnetic properties.** The external variable can be used to locally influence the behavior of the electromagnetic solver. Currently, only the keyword *EM_RANDLES_SHORT supports external variable modifications to trigger a short circuit. To do this, the external variable is used as a function

argument, `float extvar`, in the *DEFINE_FUNCTION referenced by both FUNCTID on *EM_RANDLES_SHORT and EID on this keyword. Because only function IDs are allowed, EITYP must be set to 2. Currently, IEMP must be set to 1.

8. **Example input.** In the following example, it is assumed that external variable 2 is defined for two nodes directly, for two node sets, and by two temperature result files. The external variable follows load curves 101 and 102 for nodes in node sets 11 and 12, respectively. Temperature data from files res1.binout and res2.binout are used. Finally, for nodes 1 and 3, the same load curve 100 is used but with an offset of 2.5 for node 3. For all other nodes, load curve 105 gives the default value.

   The external variable distribution influences the structural material properties with indexes 3 and 5 for part 1 and the thermal material property with index 1 for part set 11.

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
-8
*LOAD_EXTERNAL_VARIABLE
$      VID      DBAS      DSCA     DLCID       NMP      TNMP
         2       0.0       1.0       105         2         1
$      IMP       PID      PTYP
         3         1         0
$      IMP       PID      PTYP
         5         1         0
$     ITMP      TPID     TPTYP
         1        11         1
$       ID     IDTYP       BAS       SCA      LCID
        11         2       0.0       1.0       101
$       ID     IDTYP       BAS       SCA      LCID
        12         2       0.0       1.0       102
$       ID     IDTYP       BAS       SCA      LCID
                   3
$ FILENAME
res1.binout
$       ID     IDTYP       BAS       SCA      LCID
                   3
$ FILENAME
res2.binout
$       ID     IDTYP       BAS       SCA      LCID
         1         1       0.0       1.0       100
$       ID     IDTYP       BAS       SCA      LCID
         3         1       2.5       1.0       100
```

***LOAD_FACE_UVW**_*{OPTION}*

Available options include:

SET

Purpose: Apply a uniform pressure load on a parametric face or a set of parametric faces.

Face/Face Set Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	FID	LCID	SF	AT				
Type	I	I	F	F				
Default	none	none	1.0	0.0				

VARIABLE	DESCRIPTION
FID	Parametric face ID or parametric face set ID for the SET keyword option; see *IGA_FACE_UVW and *SET_IGA_FACE_UVW (see Remark 1)
LCID	Load curve ID (see *DEFINE_CURVE). The load curve must provide pressure as a function of time.
SF	Load curve scale factor
AT	Arrival or birth time for pressure

Remarks:

1. **Parametric Control Points.** The coordinates of the control points in *IGA_2D_NURBS_UVW referenced in *IGA_FACE_UVW are defined in the parametric space of *IGA_3D_NURBS_XYZ. The parametric plane defined by this *IGA_2D_NURBS_UVW must be parallel to one of the parametric *r*-*s*, *s*-*t*, or *t*-*r* planes (*u*, *v*, or *w* must be constant). General cases are currently not supported.

*LOAD_FACE_XYZ_{OPTION}

Available options include:

SET

Purpose: Apply a uniform pressure load on a physical face or a set of physical faces.

Face/Face Set Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	FXYZID	LCID	SF	AT				
Type	I	I	F	F				
Default	none	none	1.0	0.0				

VARIABLE	DESCRIPTION
FXYZID	Physical face ID or physical face set ID for the SET keyword option; see *IGA_FACE_XYZ and *SET_IGA_FACE_XYZ (see Remark 2)
LCID	Load curve ID (see *DEFINE_CURVE). The load curve must provide pressure as a function of time.
SF	Load curve scale factor
AT	Arrival or birth time for pressure

Remarks:

2. **Pressure Load on Partial Areas.** When an instantiation of *IGA_SHELL references this FXYZID, the pressure load will be applied on all elements of this physical face (full area). If you want to apply a pressure load on a certain area of this physical face, you need to define another *IGA_FACE_XYZ for this area and reference this face on *LOAD_FACE_XYZ. This additional *IGA_FACE_XYZ references the same *IGA_2D_NURBS_XYZ of the physical face. It also has its own *IGA_1D_BREP that selects the partial area on the current physical face. No additional *IGA_SHELL is needed since the shell elements have been defined by the current *IGA_SHELL.

***LOAD_GRAVITY_PART**_*{OPTION}*

Available options are:

<BLANK>

SET

Purpose: Define gravity for individual parts. This feature is intended for use with *LOAD_STIFFEN_PART to simulate staged construction. This keyword is available for solids, shells and thick shells as well as beam element types 1, 2, 3, 4, 5, 6, 9, 11, 12 and 13.

Part Cards. Include this card as many times as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	DOF	LC	ACCEL	LCDR	STGA	STGR	
Type	I	I	I	F	I	I	I	
Default	none	none	1.0	0	none	0	0	

VARIABLE	DESCRIPTION
PID/PSID	Part ID (or Part Set ID for the SET option) for application of gravity load
DOF	Direction: enter 1, 2 or 3 for x, y or z
LC	Load curve defining factor as a function of time (or zero if STGA, STGR are defined). See Remark 1.
ACCEL	Acceleration (will be multiplied by factor from curve)
LCDR	Load curve defining as a function of time during dynamic relaxation
STGA	Construction stage at which part is added (optional)
STGR	Construction stage at which part is removed (optional)

Remarks:

1. **Defining Gravity Load.** There are 3 options for defining how the gravity load on a part varies with time.

   a) Curve LC gives factor as a function of time. This overrides the other methods if LC is non-zero. A constant factor of 1.0 is assumed if LC is not specified.

   b) STGA, STGR refer to stages at which part is added and removed – the stages are defined in *DEFINE_CONSTRUCTION_STAGES. If STGA is zero, the gravity load starts at time zero. If not, it ramps up from the small factor FACT (on *CONTROL_STAGED_CONSTRUCTION) up to full value over the ramp time ATR at the start of stage STGA (see *DEFINE_CONSTRUCTION_STAGES). If STGR is zero, the gravity load continues until the end of the analysis. If not, it ramps down from full value to FACT over the ramp time ATR at the start of stage STGR.

   c) *DEFINE_STAGED_CONSTRUCTION_PART can be used instead of *LOAD_GRAVITY_PART to define this loading. During initialization, a *LOAD_GRAVITY_PART card will be created and the effect is the same as using the STGA, STGR method described above; ACCEL is then taken from *CONTROL_STAGED_CONSTRUCTION.

2. **Mass Loading.** This feature calculates the loading from the mass of the elements of the referenced parts only (density × volume). This mass does not include any attached lumped mass elements. Only solid, beam, shell and thick shell elements can be loaded.

*LOAD_HEAT_CONTROLLER

Purpose: Used to define a thermostat control function. The thermostat controls the heat generation within a material by monitoring a remote nodal temperature. Control can be specified as on-off, proportional, integral, or proportional with integral.

Sensor Node Cards. Include up to 20 cards. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NODE	PID	LOAD	TSET	TYPE	GP	GI	
Type	I	I	F	F	I	F	F	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
NODE	Sensor is located at this node number.
PID	Part ID assigned to the elements modeling the heater or cooler being controlled.
LOAD	Heater output q_0. [typical units: W/m^3]
TSET	Controller set point temperature at the location identified by NODE.
TYPE	Type of control function. EQ.1: on-off EQ.2: proportional + integral
GP	Proportional gain.
GI	Integral gain.

Remarks:

The thermostat control function is

$$\dot{q}''' = \dot{q}_0''' \left[G_P (T_{\text{set}} - T_{\text{node}}) + G_I \int_{t=0}^{t} (T_{\text{set}} - T_{\text{node}})\, dt \right]$$

*LOAD_HEAT_EXOTHERMIC_REACTION

Purpose: Define solid element sets with heat generation coming from battery abuse thermal models. This keyword is intended for thermal runaway studies.

Card Summary:

Card 1. This card is required.

HSID	STYPE	ESID	BT	DT	TMIN	TMAX	TOFF

Card 2a. This card is included if STYPE = 0 or 1.

CSEI0	ASEI	EASEI	MSEI	HSEI	WC		RU

Card 2b. This card is included if STYPE = 2.

ALPHA0	ASEI	EASEI	MSEI	HSEI	WC	P	RU

Card 3a. This card is included if STYPE = 0 or 1.

CNE0	ANE	EANE	MNE	HNE	WCNE	TSEI0	TSEIR

Card 3b. This card is included if STYPE = 2.

N							

Card 4. This card is included if STYPE = 0 or 1.

ALPHA0	APE	EAPE	MPEP1	HPE	WPE	MPEP2	

Card 5. This card is included if STYPE = 0 or 1.

CE0	AE	EAE	ME	HE	WE		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	HSID	STYPE	NSID	BT	DT	TMIN	TMAX	TOFF
Type	I	I	I	F	F	F	F	F
Default	none	none	none.	0.	10^{16}	0.	10^{16}	0.

VARIABLE	DESCRIPTION
HSID	Heat source ID
STYPE	Heat source model type: EQ.0.or.1: Heat source defined by NREL's 4-Equation model. See Remark 1. EQ.2: Heat source defined by 1-Equation model. See Remark 2.
ESID	Solid Element set ID
BT/DT	Birth and death times for application of heat source term
TMIN/TMAX	Minimum and maximum temperature before heat source activation is triggered
TOFF	Option offset for temperature used in heat source calculation

Solid Electrolyte Interface (SEI) decomposition reaction. This card is included if STYPE = 0 or 1. See Remark 1.

Card 2a	1	2	3	4	5	6	7	8
Variable	CSEI0	ASEI	EASEI	MSEI	HSEI	WC		RU
Type	F	F	F	F	F	F		F
Default	0.	0.	0.	0.	0.	0.		8.314

VARIABLE	DESCRIPTION
CSEI0	Initial concentration of Solid Electrolyte Interphase (SEI)
ASEI	SEI decomposition frequency factor, A_{sei}
EASEI	SEI decomposition activation energy, $E_{a,\text{sei}}$
MSEI	Reaction order for c_{sei}, m_{sei}
HSEI	SEI decomposition heat release, H_{sei}
WC	Specific carbon content in jellyroll, W_c

VARIABLE	DESCRIPTION
RU	Reaction constant. R_u

1-Equation Model Card. This card is included if STYPE = 2. See Remark 2.

Card 2b	1	2	3	4	5	6	7	8
Variable	ALPHA0	ASEI	EASEI	MSEI	HSEI	WC	P	RU
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	8.314

VARIABLE	DESCRIPTION
ALPHA0	Initial concentration of α
ASEI	SEI decomposition frequency factor, A_{sei}
EASEI	SEI decomposition activation energy, $E_{a,\text{sei}}$
MSEI	Reaction order for c_{sei}, m_{sei}
HSEI	SEI decomposition heat release, H_{sei}
WC	Specific carbon content in jellyroll, W_c
P	Reaction order of $\log(1-\alpha)$
RU	Reaction constant, R_u

Negative solvent reaction. This card is included if STYPE = 0 or 1. See Remark 1.

Card 3a	1	2	3	4	5	6	7	8
Variable	CNE0/N	ANE	EANE	MNE	HNE	WCNE	TSEI0	TSEIR
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
CNE0	Initial concentration value of NE
ANE	Negative solvent frequency factor, A_{ne}
EANE	Negative solvent activation energy, $E_{a,ne}$
MNE	Reaction order forc_{neg}, $m_{ne,n}$
HNE	Negative solvent heat release, H_{ne}
WCNE	Specific carbon content in jellyroll, W_{cne}
TSEI0	Initial value of t_{sei}
TSEIR	Reference t_{sei} value, $t_{sei,ref}$

Second 1-Equation Model Card. This card is included if STYPE = 2. See Remark 2.

Card 3b	1	2	3	4	5	6	7	8
Variable	N							
Type	F							
Default	0.							

VARIABLE	DESCRIPTION
N	$(1-\alpha)$ reaction order, n

Positive Solvent Reaction Card. This card is included if STYPE = 0 or 1. See Remark 1.

Card 4	1	2	3	4	5	6	7	8
Variable	ALPHA0	APE	EAPE	MPEP1	HPE	WPE	MPEP2	
Type	F	F	F	F	F	F	F	
Default	0.	0.	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
ALPHA0	Initial value of α
APE	Positive solvent frequency factor, A_{pe}
EAPE	Positive solvent activation energy, $E_{a,\text{pe}}$
MPEP1	Reaction order for α, $m_{\text{pe,p1}}$
HPE	Positive solvent heat release, H_{pe}
WPE	Specific positive active content, W_p
MPEP2	Reaction order for $(1-\alpha)$, $m_{\text{pe,p2}}$

Electrolyte Decomposition Reaction Card. This card is included if STYPE = 0 or 1. See Remark 1.

Card 5	1	2	3	4	5	6	7	8
Variable	CE0	AE	EAE	ME	HE	WE		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
CE0	Initial concentration value of CE (c_e)
AE	Electrolyte decomposition frequency factor, A_e
EAE	Electrolyte activation energy, $E_{a,e}$
ME	Reaction order for CE, m_e
HE	Electrolyte decomposition heat release, H_e
WE	Specific electrolyte content, W_e

Remarks:

1. **NREL's 4-equation model.** When this model is selected, the total heat generation can be modelled as four exothermic reactions:

$$S_{\text{abuse_chem}} = S_{\text{sei}} + S_{\text{ne}} + S_{\text{pe}} + S_{\text{ele}}$$

These reactions are the solid electrolyte interface (SEI) decomposition reaction, the negative solvent reaction, the positive solvent reaction, and the electrolyte decomposition reaction. The models for these reactions are described below. See Hatchard et al 2001 and Kim et al 2007 for details about this model. The equations here are taken from Kim et al 2007.

a) SEI decomposition reaction:

$$\begin{aligned} R_{\text{sei}}(T, c_{\text{sei}}) &= A_{\text{sei}} \times \exp\left(-\frac{E_{a,\text{sei}}}{R_u T}\right) \times c_{\text{sei}}^{m_{\text{sei}}} \\ S_{\text{sei}} &= H_{\text{sei}} \times W_c \times R_{\text{sei}} \\ \frac{dc_{\text{sei}}}{dt} &= -R_{\text{sei}} \end{aligned}$$

Here A_{sei}, $E_{a,\text{sei}}$, m_{sei}, H_{sei}, and W_c are model constants input on Card 2.

b) Negative solvent reaction:

$$\begin{aligned} R_{\text{ne}}(T, c_{\text{neg}}, t_{\text{sei}}) &= A_{\text{ne}} \times \exp\left(-\frac{t_{\text{sei}}}{t_{\text{sei,ref}}}\right) \times c_{\text{neg}}^{m_{\text{ne},n}} \times \exp\left(-\frac{E_{a,\text{ne}}}{R_u T}\right) \\ S_{\text{ne}} &= H_{\text{ne}} \times W_{\text{cne}} \times R_{\text{ne}} \\ \frac{dc_{\text{neg}}}{dt} &= -R_{\text{ne}} \\ \frac{dt_{\text{sei}}}{dt} &= R_{\text{ne}} \end{aligned}$$

Here A_{ne}, $m_{\text{ne,n}}$, $E_{a,\text{ne}}$, H_{ne}, W_{cne}, and $t_{\text{sei,ref}}$ are model constants input on Cards 2 and 3.

c) Positive solvent reaction:

$$\begin{aligned} R_{\text{pe}}(t, \alpha) &= A_{\text{pe}} \times \alpha^{m_{\text{pe,p1}}} \times (1-\alpha)^{m_{\text{pe,p2}}} \times \exp\left(-\frac{E_{a,\text{pe}}}{R_u T}\right) \\ S_{\text{pe}} &= H_{\text{pe}} \times W_p \times R_{\text{pe}} \\ \frac{d\alpha}{dt} &= R_{\text{pe}} \end{aligned}$$

Here A_{pe}, $m_{\text{pe,p1}}$, $m_{\text{pe,p2}}$, $E_{a,\text{pe}}$, H_{pe}, and W_p are model constants

d) Electrolyte decomposition reaction:

$$R_e(T, c_e) = A_e \times \exp\left(-\frac{E_{a,e}}{R_u T}\right) \times c_e^{m_e}$$
$$S_{\text{ele}} = H_e \times W_e \times R_e$$
$$\frac{dc_e}{dt} = -R_e$$

Here A_e, $E_{a,e}$, m_e, H_e, and W_e are model constants

2. **1-equation reaction model.** Total heat generation due to thermal abuse can be modeled as one exothermic reaction (MacNeil and Dahn 2001):

$$R(T, \alpha) = A_{\text{sei}} \times \alpha^{m_{\text{sei}}} \times (1-\alpha)^n \times (-\ln(1-\alpha))^p \times \exp\left(-\frac{E_{a,\text{sei}}}{R_u T}\right)$$
$$S = H_{\text{sei}} \times W_c \times R$$
$$\frac{d\alpha}{dt} = R$$

Here A_{sei}, m_{sei}, n, p, $E_{a,\text{sei}}$, R_u, H_{sei}, and W_c are input parameters.

LOAD_HEAT_GENERATION_*OPTION

Available options include:

SOLID

SET_SOLID

SHELL

SET_SHELL

Purpose: Define elements or element sets with heat generation.

Generation Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	LCID	MULT	WBLCID	CBLCID	TBLCID		
Type	I	I	F	I	I	I		
Default	none	none	0.	0	0	0		

VARIABLE	DESCRIPTION
SID	Element ID or element set ID, see *ELEMENT_SOLID, *SET_SOLID, *ELEMENT_SHELL, and *SET_SHELL, respectively.
LCID	Volumetric heat generation rate, $\dot{q}'''$, specification. SI units are W/m^3. This parameter can reference a load curve ID (see *DEFINE_CURVE) or a function ID (see *DEFINE_FUNCTION and Remark 1). When the reference is to a curve, LCID has the following interpretation: GT.0: $\dot{q}'''$ is defined by a curve consisting of (time, $\dot{q}'''$) data pairs. EQ.0: $\dot{q}'''$ is a constant defined by the value MULT. LT.0: $\dot{q}'''$ is defined by a curve consisting of (temperature, $\dot{q}'''$) data pairs. Enter |-LCID| on the DEFINE_CURVE keyword.
MULT	Volumetric heat generation, $\dot{q}'''$, curve multiplier.

VARIABLE	DESCRIPTION
WBLCID	Load curve ID defining the blood perfusion rate [e.g., kg/m^3 sec] as a function of time.
CBLCID	Load curve ID defining the blood specific heat [e.g., J/kg C] as a function of the blood temperature.
TBLCID	Load curve ID defining the blood temperature [e.g., C] as a function of time.

Remarks:

1. **Volumetric Heat Generation Rate Function.** If LCID references a *DEFINE_FUNCTION, the volumetric heat generation rate can be a function of element centroid coordinates, element centroid velocity components, the element integration point temperature, and solution time, that is, "f(x, y, z, vx, vy, vz, temp, time)."

2. **Blood Heat Transfer to Tissue Rate.** Rate of heat transfer from blood to tissue is $W_b C_b (T_b - T)$ [units: J/m^3 sec].

*LOAD_HEAT_LASER_ABSORPTION

Purpose: Define a moving heat source on a block of translucent material. Heat is applied as a volumetric heat source in multiple layers of the structure, and erosion of the material can be accounted for. A nodal path and a prescribed velocity on this path describe the motion of the source. This keyword is applicable in coupled thermal-structural and thermal-only simulations.

Card Summary:

Card 1. This card is required.

NSID1	SPD1	NSID2	SPD2	RELVEL	TX	TY	TZ

Card 2. This card is required.

IFORM	P1	P2	P3	P4	P5	P6	P7

Card 3. This card is required.

NLAY	LCROT	LCLAT	LCTIM	Q			

Card 3.1. Include NLAY instantiations of this card.

ID	IDTYP	ABSORP	LCABS				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	NSID1	SPD1	NSID2	SPD2	RELVEL	TX	TY	TZ
Type	I	I	I	F	I	F	F	F
Default	none	none	none	0.	0	0.	0.	0.

VARIABLE	DESCRIPTION
NSID1	Node set defining the path of the laser heat source. The source travels along the path at speed SPD1. The nodes are traversed in the order of the node set. See Remark 1.
SPD1	Speed of the heat source on the trajectory:

VARIABLE	DESCRIPTION
	GT.0.0: Constant speed LT.0.0: |SPD1| is a load curve ID defining weld speed as a function of time.
NSID2	Node or segment set containing information about the laser orientation: GT.0: NSID2 together with SPD2 define a curve in the same way that NSID1 and SPD1 define a curve. The aiming direction is taken to be the vector pointing from the current position along NSID2 (for example, your hand holding the torch) to the current position on NSID1 (the weld source). EQ.0: The beam aiming direction is (TX, TY, TZ).
SPD2	Speed of reference point in NSID2 (ignored unless NSID2 > 0) GT.0.0: Constant speed LT.0.0: |SPD2| is a load curve ID defining weld speed as a function of time.
RELVEL	Defines if SPD1 and SPD2 are relative or absolute speeds in coupled simulations: EQ.0: Absolute speeds EQ.1: Relative speeds with respect to the underlying structure
TX, TY, TZ	Laser beam vector in global coordinates (NSID2 = 0 only)

Card 2	1	2	3	4	5	6	7	8
Variable	IFORM	P1	P2	P3	P4	P5	P6	P7
Type	I	R	R	R	R	R	R	R
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
IFORM	Geometry description for energy rate density distribution (see Remark 3):

VARIABLE	DESCRIPTION
	EQ.1: Double elliptic with constant density
	EQ.2: Double elliptic with Gaussian distribution
P*i*	Parameters defining flux geometry, depending on field IFORM. See Remark 3 for details.

Card 3	1	2	3	4	5	6	7	8
Variable	NLAY	LCROT	LCLAT	LCTIM	Q			
Type	I	I	I	I	I			
Default	none	none	none	0	0			

VARIABLE	DESCRIPTION
NLAY	Number of material layers (see Remark 4)
LCROT	Load curve defining the rotation (angle in degrees) of the heat source around the trajectory as a function of time. See Remark 2.
LCLAT	Load curve giving the lateral offset of the heat source as a function of time. See Remark 2.
LCTIM	Load curve ID giving a flux energy input rate multiplier, $q_1(t)$, as a function of time; see Remark 3.
	EQ.0: Use constant multiplier value $q_1(t) = 1.0$.
Q	Base energy input rate Q_b [energy/time].

Layer definition. Add NLAY cards.

Card 3.1	1	2	3	4	5	6	7	8
Variable	ID	IDTYP	ABSOR	LCABS				
Type	I	I	F	I				
Default	none	0	0.	none				

VARIABLE	DESCRIPTION
ID	Set or part ID defining the material layer
IDTYP	Type for ID: EQ.0: Part ID EQ.1: Part set ID EQ.2: Solid element set ID
ABSOR	Absorption factor
LCABS	Load curve ID defining the absorption factor as a function of temperature T

Remarks:

1. **Heat source motion.** This card can be applied to coupled thermal-structure and thermal-only simulations. The source motion can be defined independently from the motion of nodes. This keyword applies to both solid and thermal thick shells.

   If NSID1 consists of workpiece nodes, the heat source will follow the workpiece's motion. By setting field RELVEL to 1, the velocity of the heat source can be defined relative to the motion of the nodes in NSID1 (the workpiece).

2. **Heat source direction (tilting the laser).** There is a natural local coordinate system (**r**, **s**, **t**) associated with the motion of the heat source. The relative velocity vector on the trajectory of the heat source defines the "forward"" direction **r**, so that material points that are approaching the heat source are in "front" of the beam. The heat source aiming direction, denoted by **t**, determines the direction of the volumetric heat source depth. The coordinate direction **s** is normal to the plane containing both the relative velocity and the aiming direction; note that

$\mathbf{s} = \mathbf{t} \times \mathbf{r}$ to have a right-handed coordinate system. It determines the direction of the volumetric heat source width.

The position and aiming direction of the heat source can be adjusted by rotating and translating the local coordinate system. To do this, the system is first rotated around the vector **r** by a value given by the load curve LCROT, resulting in a new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. Then, the system is translated in direction $\mathbf{s}'$ using LCLAT, respectively.

3. **Energy rate surface distributions.** Several different heat source geometries, described below, can be defined with this keyword. The local coordinate system needed for the description of the geometry is discussed in Remark 2.

For IFORM = 1 heat is generated in an elliptical region centered at the heat source with a constant energy rate surface density. The half-width a of the ellipse in the $\mathbf{s}'$-direction is given by input parameter P1. The half-widths of the ellipse in the **r**-direction are different in front and behind the beam. The half-width in front of the beam is denoted by b_f (input parameter P2) and behind the beam is referred to as b_r (input parameter P3). Consequently, the flux value at a point p in the ellipse is given as:

$$q = \frac{F}{\pi a b},$$

where:

$$F = \begin{cases} F_f \text{ if point } p \text{ is in front of beam} \\ F_r \text{ if point } p \text{ is behind beam} \end{cases}$$
$$b = \begin{cases} b_f \text{ if point } p \text{ is in front of beam} \\ b_r \text{ if point } p \text{ is behind beam} \end{cases}$$

It is expected that the sum of the weighting factors, F_f and F_r (defined as input parameters P4 and P5), equals 2.

The normalized surface density q for IFORM = 2 is assumed to be controlled by an exponential decay from the center of the ellipse to the boundary. The geometry of the source and its input correspond to IFORM = 1, but the energy rate surface density is here given as:

$$q = \frac{nF}{\pi a b} \exp\left(\frac{-nx^2}{a^2}\right) \exp\left(\frac{-ny^2}{b^2}\right)$$

with the same assumptions for F and b as above and with (x, y) being the first two coordinates of point p with respect to the new local coordinate system $(\mathbf{r}, \mathbf{s}', \mathbf{t}')$. The additional parameter n is defined by input parameter P6.

IFORM	1	2
P1	a	a
P2	b_f	b_f
P3	b_r	b_r
P4	F_f	F_f
P5	F_r	F_r
P6		n
P7		

4. **Layer-wise energy absorption.** To model energy absorption in a translucent material, this keyword uses the Beer-Lambert law. The intensity I of light traveling through an absorbing medium with absorption coefficient $\kappa(z)$ satisfies the differential equation:

$$\frac{dI(z)}{dz} = -\kappa(z)I(z) \quad .$$

Here, the coordinate z is aligned with the laser direction and represents the depth of light penetration into the material. The intensity I_0 of the laser before entering the material ($z = 0$) is given by the surface integral of the laser energy rate distribution q discussed in Remark 3.

The absorbing structure is discretized into NLAY material layers along the coordinate z. This user-defined discretization, together with a second-order integration scheme, is used to numerically solve the differential equation above. For each layer, the absorption coefficient κ may be either constant or temperature-dependent.

The order of the input determines the sequence in which the laser penetrates the material layers. Using the incoming intensity for a layer and that layer's absorption properties, the model computes the absorbed heat and the remaining transmitted intensity. That transmitted intensity then becomes the incoming intensity for the next layer. Eroded elements do not contribute to energy absorption in their material layer.

*LOAD_MASK

Purpose: Apply a distributed pressure load over a three-dimensional shell part. The pressure is applied to a subset of elements that are within a fixed global box and lie either outside or inside of a closed curve in space which is projected onto the surface.

Card Sets. Include as many sets of Cards 1 and 2 as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCID	VID1	OFF	BOXID	LCIDM	VID2	INOUT
Type	I	I	I	F	I	I	I	I
Default	none	none	0	0.	↓	0	none	0

Card 2	1	2	3	4	5	6	7	8
Variable	ICYCLE							
Type	I							
Default	200							

VARIABLE	DESCRIPTION
PID	Part ID (PID). This part must consist of three-dimensional shell elements. To use this option with solid elements the surface of the solid elements must be covered with null shells. See *MAT_NULL.
LCID	Curve ID defining the pressure time history, see *DEFINE_-CURVE.
VID1	Vector ID normal to the surface on which the applied pressure acts. Positive pressure acts in a direction that is in the opposite direction. This vector may be used if the surface on which the pressure acts is relatively flat. EQ.0: The pressure load depends on the orientation of the shell elements as shown in Figure 33-12.

VARIABLE	DESCRIPTION
OFF	Pressure loads will be discontinued if $\lvert \text{VID1} \cdot \mathbf{n}_{\text{shell}} \rvert < \text{OFF}$, where $\mathbf{n}_{\text{shell}}$ is the normal vector to the shell element.
BOXID	Only elements inside the box with part ID, PID, are considered. If no ID is given, all elements of PID, are included. When the active list of elements is updated, elements outside the box will no longer have pressure applied, that is, the current configuration is always used.
LCIDM	Curve ID defining the mask. This curve defines (x,y) pairs of points in a local coordinate system defined by the vector ID, VID2. Generally, the curve should form a closed loop, that is, the first point is identical to the last point, and the curve should be flagged as a DATTYP = 1 curve in the *DEFINE_CURVE section. If no curve ID is given, all elements of PID are included with the exception of those deleted by the box. The mask works like the trimming option; see DEFINE_CURVE_TRIM and Figure 17-27.
VID2	Vector ID used to project the masking curve onto the surface of part ID, PID. The origin of this vector determines the origin of the local system that the coordinates of the PID are transformed into prior to determining the pressure distribution in the local system. This vector must be defined if LCIDM is nonzero. See Figure 17-27.
INOUT	Flag for applying pressure to elements inside or outside of projected curve: EQ.0: Elements whose center falls inside the projected curve are considered. EQ.1: Elements whose center falls outside the projected curve are considered.
ICYCLE	Number of time steps between updating the list of active elements. The list update can be quite expensive and should be done at a reasonable interval. The default is not appropriate for all problems.

Remarks:

1. **Verifying the load application.** A shell set containing the shell elements included in the mask in the reference configuration is printed to the d3hsp file. This set can be used to verify the load application.

*LOAD_MOTION_NODE

Purpose: Apply a concentrated nodal force or moment to a node based on the motion of another node.

Node Cards. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	NODE1	DOF1	LCID	SF	CID1	NODE2	DOF2	CID2
Type	I	I	I	F	I	I	I	I
Default	none	none	none	1.	global	none	none	global

VARIABLE	DESCRIPTION
NODE1	Node ID for the concentrated force.
DOF1	Applicable degrees-of-freedom: EQ.1: x-direction of load action, EQ.2: y-direction of load action, EQ.3: z-direction of load action, EQ.4: Moment about the x-axis, EQ.5: Moment about the y-axis, EQ.6: Moment about the z-axis.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). The applied force is a function of the applicable degree-of-freedom of NODE2.
SF	Load curve scale factor.
CID1	Coordinate system ID (optional). See Remark 1.
NODE2	Node ID for calculating the force.
DOF2	Applicable degrees-of-freedom: EQ.1: x-coordinate EQ.2: y-coordinate,

VARIABLE	DESCRIPTION
	EQ.3: z-coordinate, EQ.4: x-translational displacement, EQ.5: y-translational displacement, EQ.6: z-translational displacement, EQ.7: Rotational displacement about the x-axis, EQ.8: Rotational displacement about the y-axis, EQ.9: Rotational displacement about the z-axis. EQ.10: x-translational velocity, EQ.11: y-translational velocity, EQ.12: z-translational velocity, EQ.13: Rotational velocity about the x-axis, EQ.14: Rotational velocity about the y-axis, EQ.15: Rotational velocity about the z-axis.
CID2	Coordinate system ID (optional); see Remark 1.

Remarks:

1. **Coordinate system.** The global coordinate system is the default. The local coordinate system IDs are defined in the *DEFINE_COORDINATE_SYSTEM section.

*LOAD_MOVING_PRESSURE

Purpose: Apply moving pressure loads to a surface. The pressure loads approximate a jet of high velocity fluid impinging on the surface. Multiple surfaces may be defined each acted on by a set of nozzles.

Card 1	1	2	3	4	5	6	7	8
Variable	LOADID							
Type	I							
Default	none							

Nozzle Cards. Define the following cards for each nozzle. Include as many cards as desired. This input ends at the first card with the second field (NODE2) ≤ 3.

Card 2	1	2	3	4	5	6	7	8
Variable	NODE1	NODE2	LCID	CUTOFF	LCIDT	LCIDD	IDIR	LSFLG
Type	I	I	I	F	I	I	I	I
Default	none	none	none	none	0	0	0	0

The following card defines the surface where the nozzles act.

Card 3	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	NIP					
Type	I	I	I					
Default	none	none	3x3					

VARIABLE	DESCRIPTION
LOADID	Loading ID
NODE1	Node located at the origin of the nozzle

VARIABLE	DESCRIPTION
NODE2	Node located at the head of the nozzle
LCID	Load curve or function ID (see *DEFINE_FUNCTION) defining pressure as a function of radial distance from the center of the jet
CUTOFF	Outer radius of jet. The pressure acting outside this radius is set to zero.
LCIDT	Load curve or function ID (see *DEFINE_FUNCTION), which scales the pressure as a function of time. If a load curve isn't specified, the scale factor defaults to 1.0.
LCIDD	Load curve or function ID (see *DEFINE_FUNCTION), which scales the pressure as a function of distance from the nozzle. If a load curve isn't specified, the scale factor defaults to 1.0.
IDIR	Direction of the pressure applied to the segments (see Remark 1): EQ.0: The normal direction of the segments EQ.1: The direction from the nozzle (NODE1) to the segments EQ.2: The direction from NODE1 to NODE2 EQ.3: Pressure is in the direction from NODE1 to NODE2 but only the normal component is applied on the segments.
LSFLG	Line-of-sight flag: EQ.0: See Remark 2. EQ.1: Pressure is applied on the first-hit segments from the nozzle.
ID	Segment set ID, shell element set ID, shell part set ID, or shell part ID. See IDTYPE below.
IDTYPE	Value that determines the meaning of variable ID. If segments to be loaded are located on solid elements or on thick shell elements, IDTYPE must be set to 0. EQ.0: ID is a segment set ID. EQ.1: ID is a shell set ID. EQ.2: ID is a shell part set ID. EQ.3: ID is a shell part ID.

VARIABLE	DESCRIPTION
NIP	Number of integration points in segment used to compute pressure loads

Remarks:

1. **Pressure direction.** Pressure directions for different cases of IDIR are illustrated in Figure 33-5. The magnitude of the pressure, p, is determined by the curves given in LCID, LCIDT, and LCIDD.

2. **Line of sight.** When LSFLG = 0, this feature is turned off, and the originally implemented algorithm of *LOAD_MOVING_PRESSURE is performed. With this algorithm, pressure is applied differently depending on if MPP or SMP is used. In SMP, it works well for non-disjoint and single-layer surfaces. Examples of applying pressure in SMP are demonstrated in Figure 33-6. For continuous surfaces, the pressure is applied to all segments inside the cylinder defined by the CUTOFF radius. However, with disjoint surfaces, the pressure is only applied to the surface which is cut by the line through NODE1 and NODE2. For the cases with multiple layers, the algorithm does not work consistently and may apply the pressure to the bottom layer as illustrated in Figure 33-6. In MPP, pressure will be applied to all segments inside the cylinder (see Figure 33-7). Note that, in the multiple-layer case, the pressure is applied on all layers.

   When LSFLG = 1, the line-of-sight feature is turned on and the pressure is applied to the first-hit segments (from the nozzle). LS-DYNA will check all the potential segments inside the cylinder to find these segments. Note that this cylinder is defined by the CUTOFF radius, so specifying this radius by a value larger than the effective radius defined by the curve in LCID will unnecessarily increase the computational time. For example, if the curve in LCID is in the range of 0 to 100 with non-zero values only from 0 to 5, the CUTOFF radius should be 5. With this feature, the pressure is applied in the same way in both SMP and MPP (see Figure 33-8).

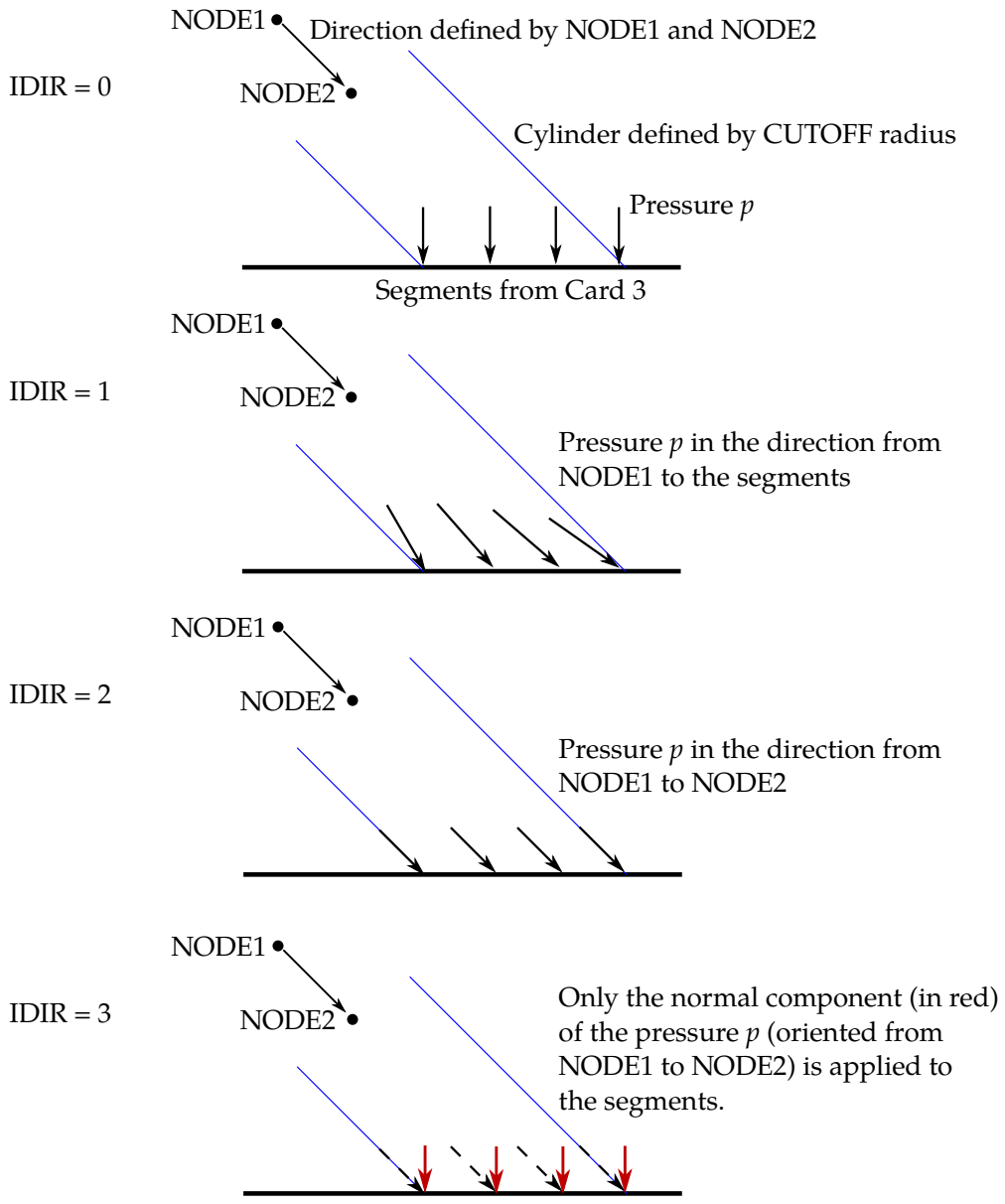


Figure 33-5. Pressure directions for different values of IDIR.

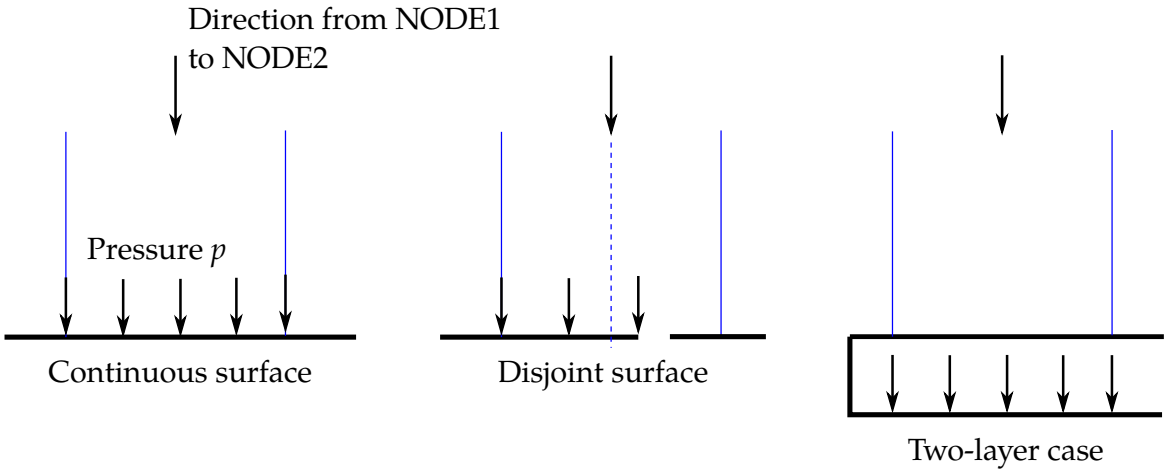


Figure 33-6. Applying pressure in SMP when LSFLG = 0

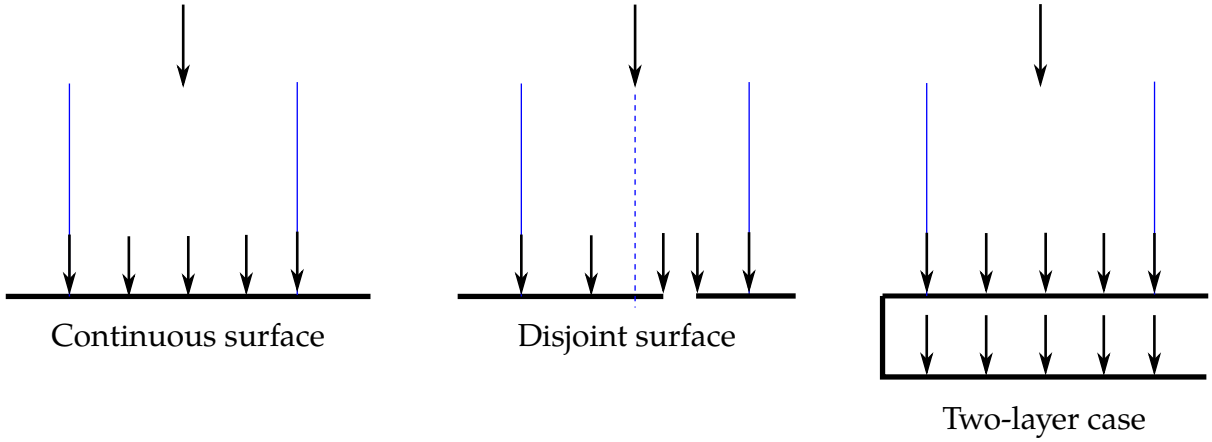


Figure 33-7. Applying pressure in MPP when LSFLG = 0

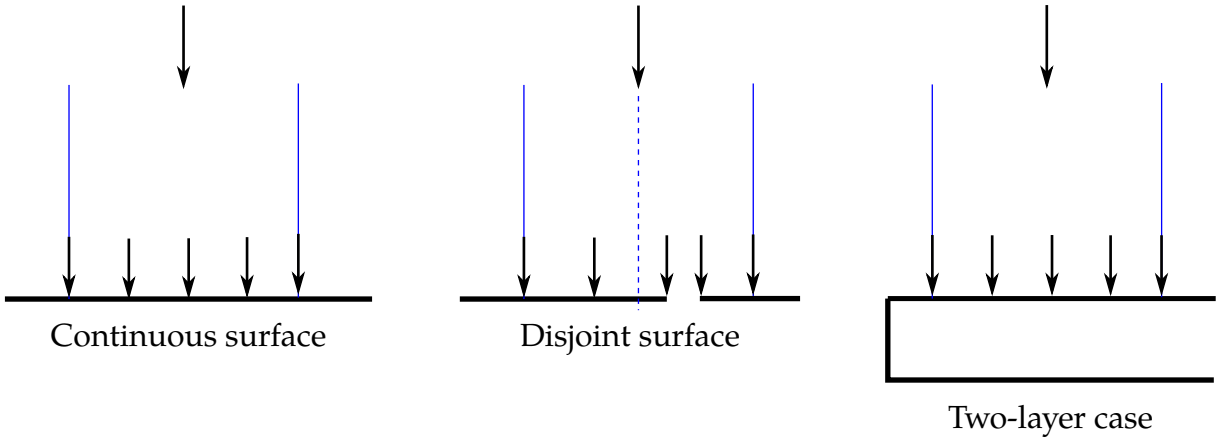


Figure 33-8. Applying pressure in SMP/MPP when LSFLG = 1

***LOAD_NODE**_OPTION

Available options include:

POINT

SET

SET_ONCE

For SET_ONCE, the load function (LCID) is only evaluated once. The value is stored and applied to the rest of the run.

Purpose: Apply a concentrated nodal force to a node or each node in a set of nodes.

Node/Node Set Cards. Include as many of this card for the SET and POINT keyword options or sets of this card with the next card for the SET_ONCE keyword option. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID/NSID	DOF	LCID	SF	CID	M1	M2	M3
Type	I	I	I	F	I	I	I	I
Default	none	none	none	1.	global	↓	↓	↓

VARIABLE	DESCRIPTION
NID/NSID	Node ID (POINT keyword option) or nodal set ID (SET or SET_ONCE keyword option); see *SET_NODE.
DOF	Applicable degrees of freedom: EQ.1: x-direction of load action. For a cylindrical or spherical system, this is the direction of the local radial degree of freedom r. EQ.2: y-direction of load action. For a cylindrical system, this is the direction of the local circumferential degree of freedom φ, and for a spherical system, it is the direction of the local latitude degree of freedom θ. EQ.3: z-direction of load action. For a cylindrical system, this is the direction of the local z-translational degree of freedom, and for a spherical system, it is the direction of the local

VARIABLE	DESCRIPTION
	longitude degree of freedom φ.
	EQ.4: Follower force (see Remark 2),
	EQ.5: Moment about the x-axis (see Remark 4). For a cylindrical or spherical system, this is the direction of the local radial degree of freedom r.
	EQ.6: Moment about the y-axis axis (see Remark 4). For a cylindrical system, this is the direction of the local circumferential degree of freedom φ, and for a spherical system, it is the direction of the local latitude degree of freedom θ.
	EQ.7: Moment about the z-axis axis (see Remark 4). For a cylindrical system, this is the direction of the local z-translational degree of freedom, and for a spherical system, it is the direction of the local longitude degree of freedom φ.
	EQ.8: Follower moment (see Remarks 2 and 4).
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION and Remark 5 below). For SET_ONCE, LCID must refer to a function ID. In this case, the function is only evaluated once. The value is stored and applied for the rest of the run.
SF	Load curve scale factor. For SET_ONCE, this field is ignored.
CID	Coordinate system ID (optional); see Remark 1.
M1	Node 1 ID. Only necessary if DOF = 4 or 8; see Remark 2 below.
M2	Node 2 ID. Only necessary if DOF = 4 or 8; see Remark 2 below.
M3	Node 3 ID. Only necessary if DOF = 4 or 8; see Remark 2 below.

SET_ONCE Scale Factor Card. Include this card in a set with the previous card for the SET_ONCE keyword option. Include as many sets as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	LCIDSF							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
LCIDSF	Load curve ID giving a scale factor as a function of time, which scales the stored value

Remarks:

1. **Coordinate systems.** The global coordinate system is the default. If CID is nonzero, *DEFINE_COORDINATE_*OPTION* specifies the local coordinate system. For local Cartesian coordinate systems the direction is not updated in time and thus the direction of the load is static unless you are using *DEFINE_COORDINATE_NODES with FLAG = 1. In that case, the local Cartesian coordinate system and thus the direction of the applied load is updated with time. An alternative way to update the direction of the load is to use the follower forces option (see next remark).

   For cylindrical and spherical systems, the system is updated by construction. Note that the unit of the load is force (*N* if SI-units are used) in translational directions and moment (*Nm* if SI-units are used) in rotational directions; thus, the load is *not* energy conjugate to the degrees of freedom used for these systems. Only *DEFINE_COORDINATE_NODES and *DEFINE_COORDINATE_SYSTEM are supported for specifying cylindrical and spherical coordinate systems. Cylindrical coordinates and spherical coordinates are constructed from the local Cartesian coordinate system determined from these keywords if requested through TYPE. See Figure 17-10 on *DEFINE_COORDINATE_NODES.

2. **Follower forces.** The current positions of nodes M1, M2, and M3 are used to control the direction of a follower force. A positive follower force acts normal to the plane defined by these nodes, and a positive follower moment puts a counterclockwise torque about the *t*-axis. These actions are depicted in Figure 33-9.

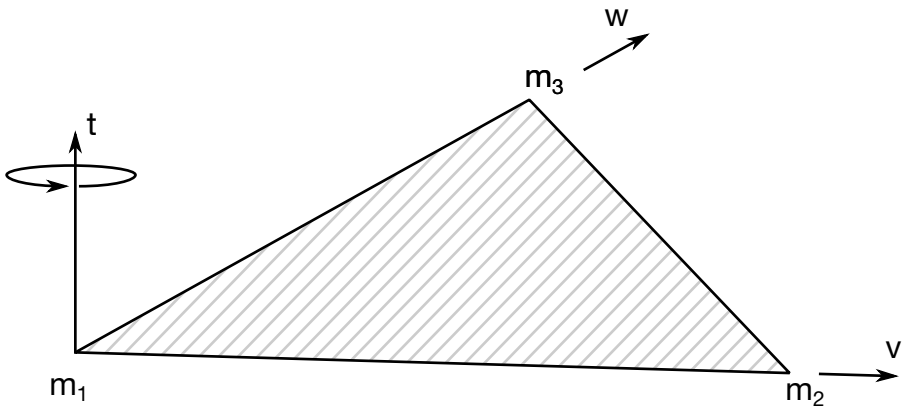


Figure 33-9. Nodes M1, M2, and M3 define a plane. A positive follower force acts in the positive *t*-direction of that plane, that is, along the normal vector of the plane. A positive follower moment puts a counterclockwise torque about the normal vector. The normal vector is found by the cross product $\mathbf{t} = \mathbf{v} \times \mathbf{w}$ where **v** and **w** are vectors as shown.

An alternative way to define the force direction is by setting M3 to any non-positive value, in which case the follower force is in the M1 to M2 direction.

3. **Axisymmetric elements with area and volume weighting.** For shell formulations 14 and 15, the axisymmetric solid elements with area and volume weighting, respectively, the specified nodal load is per unit length (type14) and per radian (type 15).

4. **Moments.** Moments can only be applied to nodes that have rotational degrees of freedom. Element type and formulation determine the degrees of freedom for a node. For example, the nodes of solid formulation 1 have only 3 translational degrees of freedom and no rotational degrees of freedom.

5. ***DEFINE_FUNCTION for LCID.** The function defined by LCID has 8 arguments: time (float), the 3 current coordinates (float), the 3 reference coordinates (float) and the number of nodes in the node set (int). A function that applies a force proportional to the distance from the initial coordinates would be:

```
f(t,x,y,z,x0,y0,z0,nnodes)= -10.*sqrt ( (x-x0)*(x-x0)+(y-y0)*(y-y0)+(z-z0)*(z-z0) )
```

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *LOAD_NODE_SET
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A cantilever beam (made from shells) is loaded on the two end nodes
$  (nodes 21 & 22).  The load is applied in the y-direction (dof=2).
$  Load curve number 1 defines the load, but is scaled by sf=0.5 in the
```

```
$  *LOAD_NODE_SET definition.
$
*LOAD_NODE_SET
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid       dof      lcid        sf       cid        m1        m2        m3
        14         2         1       0.5
$
$
*SET_NODE_LIST
$      sid
        14
$
$     nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
        21        22
$
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
         1
$
$           abscissa            ordinate
                 0.0                 0.0
                10.0               100.0
                20.0                 0.0
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*LOAD_NURBS_SHELL_{OPTION}

The following option is available to define an ID for the load applied to a NURBS shell patch:

ID

If the ID is defined, an additional card is required.

Purpose: Apply a load over a NURBS shell patch.

Card Summary:

Card ID. This card is included if and only if the ID option is used.

ID	HEADING

Card 1. This card is required.

PTID	LCID	SF	AT	DT	LTYPE	REGDEF	

Card 1.1. This card is included if and only if LTYPE = "TRACT".

CID	V1	V2	V3				

Card 2a. This card included if and only if LTYPE = "PRESS" or "TRACT" and REGDEF = "RS". Define as many cards as needed.

RMIN	SMIN	RMAX	SMAX				

Card 2b. This card is included if and only if LTYPE = "PRESS" or "TRACT" and REGDEF = "NBEW". Define as many cards as needed.

NE1	NE2	NE3	NE4	NE5	NE6	NE7	NE8

Card 2c. This card is included if and only if LTYPE = "PRESS" or "TRACT" and REGDEF = "NBEP". Define as many cards as needed.

NT1	NT2	NT3	NT4	NTE			

Card 2d. This card is included if and only if LTYPE = "CRV," "CRVS," "CRVT," or "CRVN" and REGDEF = "RS". Define as many cards as needed.

R1	S1	R2	S2				

Card 2e. This card is included if and only if LTYPE = "CRV," "CRVS," "CRVT," or "CRVN" and REGDEF = "NBEP". Define as many cards as needed.

NC1	NC2	NCE					

Data Cards:

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Loading ID
HEADING	A description of the loading.

Card 1	1	2	3	4	5	6	7	8
Variable	PTID	LCID	SF	AT	DT	LTYPE	REGDEF	
Type	I	I	F	F	F	A10	A10	
Default	none	none	1.	0.	10^{16}	PRESS	RS	

VARIABLE	DESCRIPTION
SSID	NURBS shell patch ID; see *ELEMENT_SHELL_NURBS_PATCH.
LCID	Load curve ID
SF	Load curve scale factor
AT	Arrival/birth time for load.
DT	Death time for load.

VARIABLE	DESCRIPTION
LTYPE	Load type. Default = "PRESS".
	EQ.CRV: Loading is applied on a curve on a NURBS patch, along (V1,V2,V3) of CID coordinate system. The loading dimension is force per unit length along the curve.
	EQ.CRVS: Loading, force per unit length, is applied on a curve on the surface of a NURBS patch, along the local shear direction, CS direction, in Figure 33-10.
	EQ.CRVT: Loading, force per unit length, is applied on a curve on the surface of a NURBS patch, along the local transverse direction, CT direction, in Figure 33-10.
	EQ.CRVN: Loading, force per unit length, is applied on a curve on the surface of a NURBS patch, along the local normal direction, CN direction, in Figure 33-10.
	EQ.PRESS: Surface traction is applied on a region on a NURBS patch, along the opposite direction to the NURBS patch surface normal. The loading unit is force per unit area of the region.
	EQ.TRACT: Surface traction, force per unit area, is applied on a region on a NURBS patch, along (V1,V2,V3) of the coordinate system CID.
REGDEF	The method of defining the region of a NURBS shell patch on which the loading is applied. Default = "RS".
	When LTYPE = "PRESS" or "TRACT":
	EQ.RS: The extreme values of the univariate knot vector in local *r* and *s*-directions are used to define the traction application region; see RK*i* and SK*i* of *ELEMENT_SHELL_NURBS_PATCH. When not defined, the whole NURBS shell patch is subject to the traction.
	EQ.NBEW: The traction loading is applied to the whole NURBS-element which is identified by an interior node on the surface of the NURBS-element, such as NE1 through NE4 in Figure 33-10.
	EQ.NBEP: The traction loading is applied to part of a NURBS-element which is identified by four nodes on the surface of the NURBS-element, such as NT1 through NT4 in Figure 33-10.
	When LTYPE = "CRV," "CRVS," "CRVT," or "CRVN":

VARIABLE	DESCRIPTION
	EQ.RS: The values of (RK, SK) of the starting point and the ending point of a curve. NC1 and NC2 in Figure 33-10, on the surface of the NURBs shell patch are used to define the curve loading application region.
	EQ.NBEP: The curve loading is applied to a curve on a NURBS-element which is identified by two nodes on the surface of the NURBS-element, NC1 and NC2 in Figure 33-10.

"TRACT" Coordinate Card. Include this card if LTYPE = "TRACT." This card defines the traction load's coordinate system and loading direction.

Card 1.1	1	2	3	4	5	6	7	8
Variable	CID	V1	V2	V3				
Type	I	F	F	F				
Default	0	none	none	none				

VARIABLE	DESCRIPTION
CID	Coordinate system ID, only applicable when LTYPE = "TRACT".
V1, V2, V3	Vector direction cosines defining the direction of the traction loading.

Region definition card for surface traction loading using when LTYPE = "PRESS" or "TRACT" and REGDEF = "RS". Define as many cards as needed.

Card 2a	1	2	3	4	5	6	7	8
Variable	RMIN	SMIN	RMAX	SMAX				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
RMIN, SMIN, RMAX, SMAX	The minimum and maximum values of the univariate knot vector in local *r* and *s*-directions of the area on which the pressure loading is applied. See *ELEMENT_SHELL_NURBS_PATCH for details.

Region definition card when LTYPE = "PRESS" or LTYPE = "TRACT" and REGDEF = "NBEW". Define as many cards as needed.

Card 2b	1	2	3	4	5	6	7	8
Variable	NE1	NE2	NE3	NE4	NE5	NE6	NE7	NE8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
NE*i*	An interior node on the surface of the NURBS element on which the traction loading is applied.

Region definition card when LTYPE = "PRESS" or LTYPE = "TRACT" and REGDEF = "NBEP". Define as many cards as needed.

Card 2c	1	2	3	4	5	6	7	8
Variable	NT1	NT2	NT3	NT4	NTE			
Type	I	I	I	I	I			
Default	none	none	none	none	see below			

VARIABLE	DESCRIPTION
NT*i*	Nodes defining an area on the surface of a NURBS element where loading is applied. Using nodes shared with other NURBS elements, such as node NB in Figure 33-10, as NT*i* is not recommended.
NTE	Optional node used to identify the NURBS element on which the load application area defined by the NT*i*'s is located. NTE is *only*

VARIABLE	DESCRIPTION
	needed when *all* NT*i*'s are shared with other NURBS elements. If not defined, node NT1 has to be owned exclusively by the NURBS element containing the load application region.

Region definition card for curve loading when LTYPE = "CRV", "CRVS", "CRVT", or "CRVN" and REGDEF = "RS". Define as many cards as needed.

Card 2d	1	2	3	4	5	6	7	8
Variable	R1	S1	R2	S2				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
R1, S1, R2, S2	The univariate knot vector in local *r* and *s*-directions of the starting and ending points of the curve on which the curve loading is applied. An example would be the knot vector for nodes NC1 and NC2 as shown in Figure 33-10.

Region definition card for curve loading when LTYPE = "CRV", "CRVS", "CRVT", or "CRVN" and REGDEF = "NBEP". Define as many cards as needed.

Card 2e	1	2	3	4	5	6	7	8
Variable	NC1	NC2	NCE					
Type	I	I	I					
Default	none	none	see below					

VARIABLE	DESCRIPTION
NC*i*	Nodes defining a curve on the surface of a NURBS element where loading is applied. Using nodes shared with other NURBS elements, such as node NB in Figure 33-10, as NC*i* is not recommended.

VARIABLE	DESCRIPTION
NCE	Optional node used to identify the NURBS element on which the load application curve defined by the NT*i*s is located. NCE is *only* needed when *both* NC*i*s are shared with other NURBS elements. If not defined, node NC1 has to be owned exclusively by the NURBS element containing the load application curve.

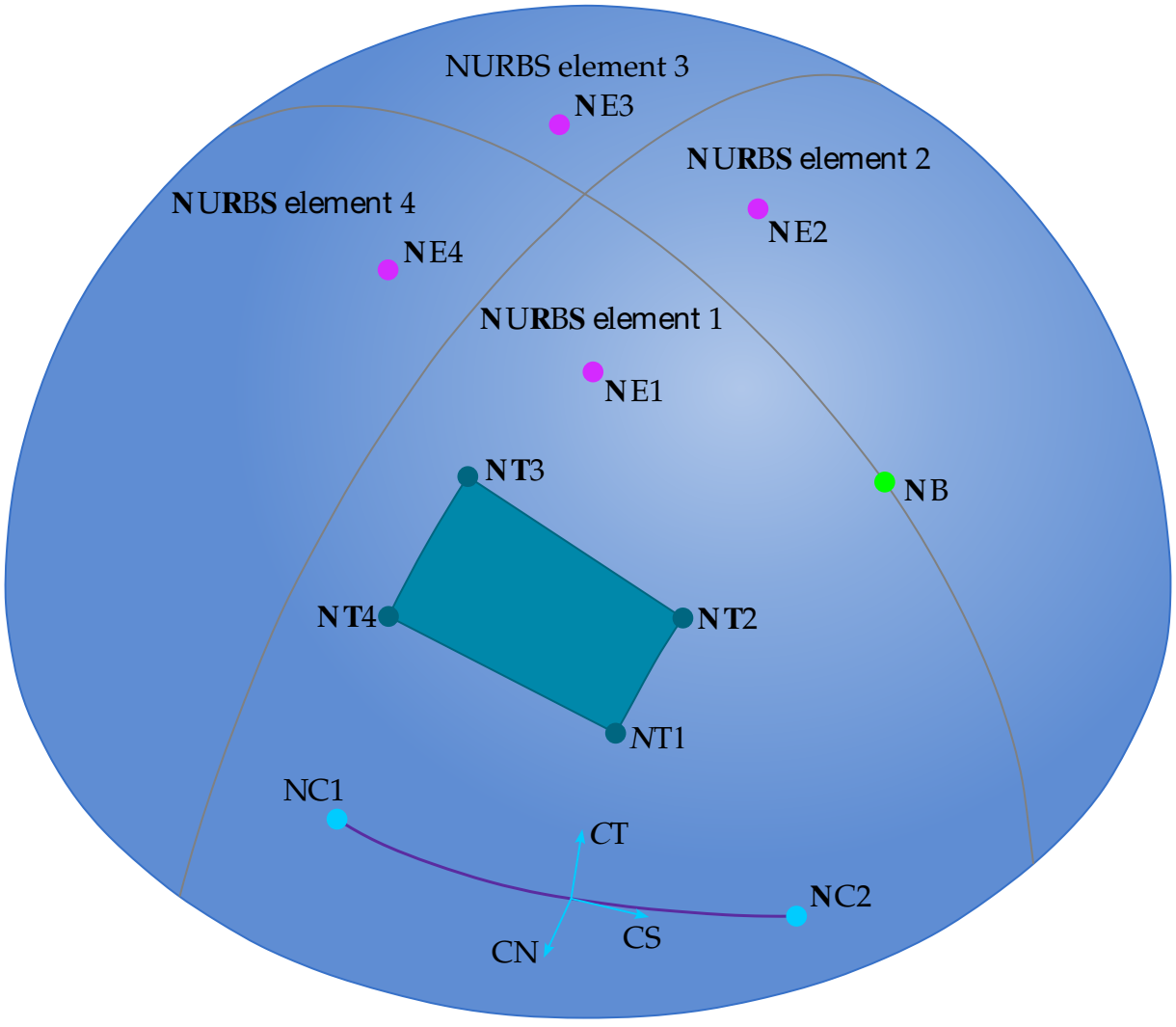


Figure 33-10. NURBS shell patch with four NURBS shell elements.

***LOAD_POINT_UVW**_*{OPTION}*

Available options include:

SET

Purpose: Apply a concentrated force to a parametric point or each parametric point in a set of parametric points.

Point/Point Set Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	DOF	LCID	SF				
Type	I	I	I	F				
Default	none	none	none	1.				

VARIABLE	DESCRIPTION
PID	Parametric point ID or parametric point set ID for the SET keyword option; see *IGA_POINT_UVW and *SET_IGA_POINT_UVW
DOF	Applicable degrees-of-freedom: EQ.1: x-direction of load action, EQ.2: y-direction of load action, EQ.3: z-direction of load action, EQ.5: Moment about the x-axis, EQ.6: Moment about the y-axis axis, EQ.7: Moment about the z-axis axis.
LCID	Load curve ID (see *DEFINE_CURVE)
SF	Load curve scale factor

*LOAD_PYRO_ACTUATOR

Purpose: Calculate forces from a pyrotechnic actuator between nodes/segments. The main application in mind is active hood lifters for pedestrian protection. The keyword models the expansion/contraction of the gas chamber in a piston actuator using a mass flow curve, the chamber cross section area, and cylinder length determined by the nodes/segments. It uses the same gas law as *AIRBAG_SIMPLE_AIRBAG_MODEL but does not require modeling the chamber/piston. You can access chamber solution data by including *DATABASE_PYRO in the input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	ID1	ID2	CSA	VOL	PRS	DENS	ATIME
Type	I	I	I	F	F	F	F	F

Card 2	1	2	3	4	5	6	7	8
Variable	MCID	CV	CP	TEMP				
Type	F	F	F					

VARIABLE	DESCRIPTION
ID	Unique ID for actuator
ID1	Node or segment set where forces act on one side of the chamber (see Remarks): GT.0: Node ID LT.0: |ID1| is a segment set ID.
ID2	Node or segment set where forces act on the other side of the chamber (see Remarks): GT.0: Node ID LT.0: |ID2| is a segment set ID.
CSA	Chamber cross section area
VOL	GT.0: Initial chamber volume

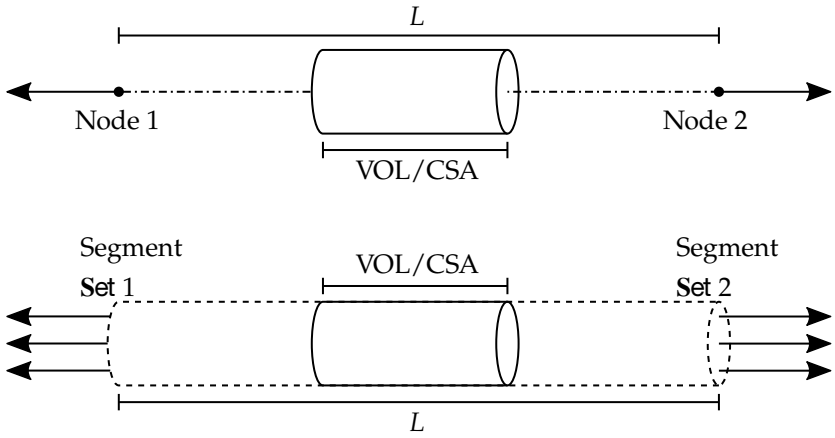


Figure 33-11. Model of a chamber between nodes or segments. For VOL > 0.0, the initial chamber length is VOL/CSA which may be shorter or longer than the node/set distance *L*.

VARIABLE	DESCRIPTION
	EQ.0: Initial chamber volume given by distance between ID1 and ID2. See Remarks and Figure 33-11.
PRS	Ambient pressure
DENS	Ambient gas density
ATIME	Activation time
MCID	Mass flow curve ID (mass flow as function of time)
CV	Specific heat capacity at constant pressure
CP	Specific heat capacity at constant volume
TEMP	Gas generator temperature

Remarks:

This feature calculates axial forces resulting from pressure in a (cylindric) chamber of varying length. ID1 and ID2 can be segment sets or single nodes upon which the forces act. Before the activation time, ATIME,

$$p(t) = \text{PRS} \ .$$

Starting at the activation time, LS-DYNA solves the following pressure ODE:

$$\dot{p}(t) = \frac{\gamma}{V(t)}\left(R \times \text{TEMP} \times \text{MCID}(t) - p(t)\dot{V}(t)\right) \ .$$

where

$$R = \mathrm{CP} - \mathrm{CV}, \qquad \gamma = \frac{\mathrm{CP}}{\mathrm{CV}} \ .$$

The chamber volume is given by

$$V(t) = \begin{cases} \big(L(t) - L(\mathrm{ATIME})\big) \times \mathrm{CSA} + \mathrm{VOL}, & \mathrm{VOL} > 0, \\ L(t) \times \mathrm{CSA}, & \mathrm{VOL} = 0, \end{cases}$$

where the length, $L(t)$, is the current distance between the nodes/sets ID1 and ID2. Note that for VOL > 0.0, the actual chamber length may be shorter or longer than the distance between ID1 and ID2. For VOL $= 0$, ID1 and ID2 mark the endpoints of the chamber. If ID1 and ID2 are nodes, the initial length must be greater than 0 to calculate the direction of the forces on the nodes. We assume the segment sets are planar, parallel, and form the two ends of a chamber. The forces are applied in the direction of the segment normal vectors.

*LOAD_PZE

Purpose: Apply a concentrated or distributed electric charge over a set of segments or nodes with piezoelectric properties.

Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SETID	LCID	SF	SETYP				
Type	I	I	F	A				
Default	none	0	none	NSET				

VARIABLE	DESCRIPTION
SETID	Set ID; see *SET_SEGMENT or *SET_NODE.
LCID	Load curve gives concentrated charge (in electric charge units) or distributed electric charge (in unit of electric charge per unit area) vs. time.
SF	Scale factor on curve or constant electric charge if LCID = 0.
SETYP	Type of SETID EQ.NSET: SETID is a node set. EQ.SEGSET: SETID is a segment set.

***LOAD_REMOVE_PART**_*{OPTION}*

Available options include:

<BLANK>

SET

Purpose: Delete the elements of a part in a staged construction simulation. Shock effects are prevented by gradually reducing the stresses prior to deletion. This feature is available for solid, shell, thick shell and beam elements.

Part Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	TIME0	TIME1	STGR				
Type	I	F	F	I				
Default	none	0.	0.	0				

VARIABLE	DESCRIPTION
PID	Part ID (or Part Set ID for the SET keyword option) for deletion
TIME0	Time at which stress reduction starts
TIME1	Time at which stresses become zero and elements are deleted
STGR	Construction stage at which the part is removed (optional)

Remarks:

There are 3 methods to define the part removal time:

1. TIME0 and TIME1 override all the other methods if non-zero.

2. STGR refers to the stage at which the part is removed; the stages are defined in *DEFINE_CONSTRUCTION_STAGES. This method is equivalent to setting TIME0 and TIME1 equal to the start and end of the ramp time at the beginning of stage STGR.

3. *DEFINE_STAGED_CONSTRUCTION_PART can be used instead of *LOAD_-REMOVE_PART to achieve the same effect.

*LOAD_RIGID_BODY

Purpose: Apply a concentrated nodal force to a rigid body. The force is applied at the center of mass or a moment is applied around a global axis. As an option, local axes can be defined for force or moment directions.

Rigid Body Cards. Include as many Rigid Body Cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	DOF	LCID	SF	CID	M1	M2	M3
Type	I	I	I	F	I	I	I	I
Default	none	none	none	1.	global	0	0	0

VARIABLE	DESCRIPTION
PID	Part ID of the rigid body; see *PART_*OPTION*.
DOF	Applicable degrees-of-freedom: EQ.1: *x*-direction of load action, EQ.2: *y*-direction of load action, EQ.3: *z*-direction of load action, EQ.4: follower force; see Remark 2, EQ.5: moment about the *x*-axis, EQ.6: moment about the *y*-axis, EQ.7: moment about the *z*-axis. EQ.8: follower moment; see Remark 2.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION and Remark 3). GT.0: force as a function of time. LT.0: force as a function of the absolute value of the rigid body displacement. This option only applies to load curves.
SF	Load curve scale factor

VARIABLE	DESCRIPTION
CID	Coordinate system ID. See Remark 1.
M1	Node 1 ID. Only necessary if DOF = 4 or 8; see Remark 2.
M2	Node 2 ID. Only necessary if DOF = 4 or 8; see Remark 2.
M3	Node 3 ID. Only necessary if DOF = 4 or 8; see Remark 2.

Remarks:

1. **Coordinate System.** The global coordinate system is the default. The local coordinate system ID's are defined in the *DEFINE_COORDINATE_SYSTEM section. This local axis is fixed in inertial space, i.e., it does not move with the rigid body.

2. **Follower Force and Moments.** Nodes M_1, M_2, and M_3 must be defined for a follower force or moment. The follower force acts normal to the plane defined by these nodes as depicted in Figure 33-9. The positive *t*-direction is found by the cross product $\mathbf{t} = \mathbf{v} \times \mathbf{w}$ where **v** and **w** are vectors as shown. The follower force is applied at the center of mass. A positive follower moment puts a counterclockwise torque about the *t*-axis. An alternative way to define the force direction is by setting M_3 to any non-positive value, in which case the follower force is in the M_1 to M_2 direction.

3. **Force Function.** When LCID defines a function, the function has seven arguments: time, the 3 current coordinates for the center of mass, and the 3 reference coordinates. A function that applies a force proportional to the distance from the initial coordinates would be

```
f(t,x,y,z,x0,y0,z0)= -10.*sqrt ( (x-x0)*(x-x0)+(y-y0)*(y-y0)+(z-z0)*(z-z0) ).
```

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *LOAD_RIGID_BODY
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  From a sheet metal forming example.  A blank is hit by a punch, a binder is
$  used to hold the blank on its sides.  The rigid holder (part 27) is held
$  against the blank using a load applied to the cg of the holder.
$
$  The direction of the load is in the y-direction (dof=2) but is scaled
$  by sf = -1 so that the load is in the correct direction.  The load
$  is defined by load curve 12.
$
```

```
*LOAD_RIGID_BODY
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid       dof      lcid        sf       cid        m1        m2        m3
        27         2        12      -1.0
$
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
        12
$
$           abscissa            ordinate
           0.000E+00           8.000E-05
           1.000E+04           8.000E-05
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*LOAD_SEGMENT_{OPTION}

To define an ID for the segment loading, the following option is available:

ID

If the ID is defined, an additional card is required.

Purpose: Apply the distributed pressure load over one triangular or quadrilateral segment defined by four, six, or eight nodes, or in the case of two-dimensional geometries, over one two-noded line segment. The pressure and node numbering convention are specified in Figure 33-12. To apply a uniform pressure, see Remark 5.

ID Card. Additional card for the ID keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 2	1	2	3	4	5	6	7	8
Variable	LCID	SF	AT	N1	N2	N3	N4	N5
Type	I	F	F	I	I	I	I	I
Default	none	1.	0.	none	none	none	none	none

Additional card for when N5 ≠ 0.

Card 3	1	2	3	4	5	6	7	8
Variable	N6	N7	N8					
Type	I	I	I					
Default	none	none	none					

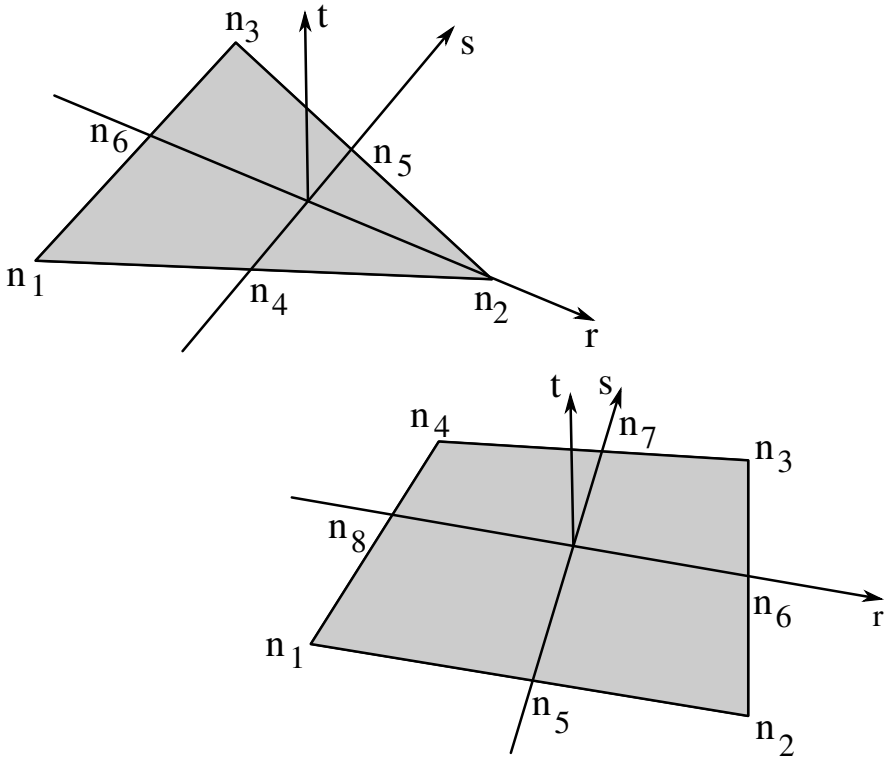


Figure 33-12. Nodal numbering for pressure segments in three-dimensional geometries. Positive pressure acts in the negative **t**-direction.

VARIABLE	DESCRIPTION
ID	Loading ID
HEADING	A description of the loading.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). For a load curve ID, the load curve gives the pressure as a function of time. For a function ID, the function for pressure has 10 arguments: time, the 3 current coordinates, the 3 reference coordinates, and the 3 velocities; that is, f(t,x,y,z,x0,y0,z0,vx,vy,vz). See Remark 1.
SF	Load curve scale factor. See Remark 2.
AT	Arrival time for pressure or birth time of pressure. See Remark 3.
N1	Node ID
N2	Node ID
N3	Node ID. See Remark 4.

VARIABLE	DESCRIPTION
N4	Node ID. See Remark 4.
N5	Mid-side node ID, if applicable. See Figure 33-12.
N6	Mid-side node ID, if applicable. See Figure 33-12.
N7	Mid-side node ID, if applicable. See Figure 33-12.
N8	Mid-side node ID, if applicable. See Figure 33-12.

Remarks:

1. **Empirical functions for pressure.** If LCID is input as -1, then the Brode function is used to determine the pressure for the segments; see *LOAD_BRODE. If LCID is input as -2, then an empirical airblast function is used to determine the pressure for the segments; see *LOAD_BLAST.

2. **Load curve multipliers.** The load curve multipliers may be used to increase or decrease the pressure. The time value is not scaled.

3. **Activation time.** The activation time, AT, is the time during the solution that the pressure begins to act. Until this time, the pressure is ignored. The function value of the load curves will be evaluated at the offset time given by the difference of the solution time and AT, that is, solution time − AT.

4. **Triangular segments and two-dimensional geometries.** Triangular segments without mid-side nodes are defined by setting N4 = N3. Segments for two-dimensional geometries are defined by two nodes, N1 and N2. Leave N3 and N4 as zero or set both equal to N2. A positive pressure acts on the segment in the $\hat{\mathbf{z}} \times (\mathrm{N1} - \mathrm{N2})$ direction where $\hat{\mathbf{z}}$ is the unit vector in the z-direction and $(\mathrm{N1} - \mathrm{N2})$ is the vector from N1 to N2.

5. **Uniform pressure.** To apply a uniform pressure to type 17 tetrahedral elements, 4 triangular segments should be defined for each loaded face of an element. Three of the segments should each have one corner node and the two adjacent mid-side nodes. The 4th segment should be made from the 3 mid-side nodes.

   To apply a uniform pressure to type 16 tetrahedral elements or type 24 triangular shell elements, one 6-node segment may be defined for each loaded face. However, this segment loading feature will accept and properly treat "any" other segment definition. In other words, if the preprocessor happens to create *one* 3-node segment corresponding to local node numbering convention {1,2,3,3} in Figure 33-12, or permutations thereof, then this will internally be detected as

a 6-node segment, and the nodal forces will be consistently distributed over the 6 involved nodes. Likewise, if four 3-noded segments corresponding to local node numbering conventions {1,4,6,6}, {2,5,4,4}, {3,6,5,5} and {4,5,6,6}, or permutations thereof, are created, then these will be detected to belong to the same 6-noded segment and again result in the correct nodal forces. The difference between the two latter approaches is that using four "sub-segments" will provide four different normal directions **t** instead of just one, and for curved faces this will provide a more accurate representation of the pressure load.

To apply a uniform pressure to type 23 hexahedral elements or type 23 quadrilateral shell elements, one 8-node segment may be defined for each loaded face. However, this segment loading feature will accept and properly treat a segment excluding the 4 mid-side nodes. In other words, if the preprocessor happens to create just one 4-noded segment corresponding to local node numbering convention {1,2,3,4} in Figure 33-12, or permutations thereof, then this will internally be detected as an 8-node segment, and the nodal forces will be consistently distributed over the 8 involved nodes.

To apply a uniform pressure to type 24 hexahedral elements, either one 4-node segment corresponding to the numbering convention {1,2,3,4} in Figure 33-12, or permutations thereof, may be defined for each loaded face. But this segment loading feature will also accept and properly treat *four* 4 node segments corresponding to {1,5,9,8}, {2,6,9,5}, {3,7,9,6} and {4,8,9,7}, where local node number 9 is located in the center of the face. The difference between these two approaches is that using four "sub-segments" will provide four different normal directions **t** instead of just one, and for curved faces this will provide a more accurate representation of the pressure load.

6. **Example.** The partial input deck below applies pressure to a solid block, using this keyword. The function defined by LCID has 10 arguments: time, the 3 current coordinates, the 3 reference coordinates, and the 3 velocities. A function that applies a pressure proportional to the distance from the initial coordinates with damping in the *x*-direction would be, for example,

```
f(t,x,y,z,x0,y0,z0,vx,vy,vz)= 10.*sqrt ( (x-x0)*(x-x0)+(y-y0)*(y-y0)+(z-z0)*(z-z0) )+2.*vx
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *LOAD_SEGMENT
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A block of solid elements is pressed down onto a plane as it moves along
$  that plane.  This pressure is defined using the *LOAD_SEGMENT keyword.
$
$  The pressure is applied to the top of the block.  This top is defined
$  by the faces on top of the appropriate solid elements.  The faces are
$  defined with segments.  For example, nodes 97, 106, 107 & 98 define
$  a top face on one of the solids (and thus, one of the faces to apply the
```

```
$  pressure too).  This "face" is referred to as a single segment.
$
$  The load is defined with load curve number 1.  The curve starts at zero,
$  ramps to 100 in 0.01 time units and then remains constant.  However,
$  the curve is then scaled by sclo = 2.5.  Thus, raising the load to 250.
$  Note that the load is NOT scaled in the *LOAD_SEGMENT keyword, but
$  could be using the sf variable.
$
*LOAD_SEGMENT
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     lcid        sf        at        n1        n2        n3        n4
         1      1.00       0.0        97       106       107        98
         1      1.00       0.0       106       115       116       107
         1      1.00       0.0        98       107       108        99
         1      1.00       0.0       107       116       117       108
$
$
*DEFINE_CURVE
$
$     lcid      sidr      scla      sclo      offa      offo
         1         0       0.0       2.5
$
$           abscissa            ordinate
              0.000                 0.0
              0.010               100.0
              0.020               100.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*LOAD_SEGMENT_ALE_INI

Purpose: Apply the initial pressures of boundary ALE elements to their faces that lack neighbors. This keyword has no data cards.

*LOAD_SEGMENT_CONTACT_MASK

Purpose: Mask the pressure from a *LOAD_SEGMENT_SET when the pressure segments are in contact with another material. For non-Mortar contact, this keyword is currently only supported in the MPP version, while Mortar contact is supported in all versions. Note that the Heading Card is required.

Heading Card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Load Set Card.

Card 2	1	2	3	4	5	6	7	8
Variable	LSID	P1	P2	CID1	CID2	CID3	CID4	CID5
Type	I	F	F	I	I	I	I	I

Optional Cards. Include as many cards as desired. This data ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	CID6	CID7	CID8	CID9	CID10	CID11N	CID12	CID13
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
LSID	Load set ID to mask, which must match a *LOAD_SEGMENT_SET. See Remark 2.
P1	Lower pressure limit. When the surface pressure due to contact is below P1, no masking is done and the full load defined in *LOAD_SEGMENT_SET is applied. For pressures between P1 and P2 see Remark 1.

VARIABLE	DESCRIPTION
P2	Upper pressure limit. When the surface pressure due to contact is above P2, no load is applied due to the *LOAD_SEGMENT_SET. For pressures between P1 and P2 see Remark 1.
CID*n*	The IDs of contacts that can mask the pressure loads. The specified contact definitions must all be of the same type. Furthermore, only non-automatic SURFACE_TO_SURFACE (two way) and AUTOMATIC_SURFACE_TO_SURFACE_TIEBREAK contacts are supported. For TIEBREAK contacts, pressure is masked until the tie fails. Once the tie fails, the full pressure will be applied for the remainder of the simulation. The values P1 and P2 are ignored. For other contact types, the contact forces, along with the nodal contact surface areas, are used to compute the contact pressure at each node to determine any masking effect.

Remarks:

1. **Intermediate Pressures.** If the contact pressure, p_{contact} is between P1 and P2, the pressure load is scaled by a factor of

$$f = \frac{\text{P2} - p_{\text{contact}}}{P2 - P1}.$$

   P1 may be set equal to P2 if desired.

2. **LSID.** The LSID values must be unique. Having two instances of this referencing the same*LOAD_SEGMENT_SET *is not supported*. However, a contact ID *may* appear in two different instances of this keyword.

3. **IDOF on *SECTION_SHELL.** When IDOF = 3 on *SECTION_SHELL, the shell thickness strain is governed by the stress from contact and pressure loads. If IDOF = 3 *and* the masking contact is of Mortar type, the masking will not apply for the thickness strain. Other contact types do not have this limitation.

*LOAD_SEGMENT_FILE

Purpose: To define *time-varying* distributed pressure loads over triangular or quadrilateral segments defined by four, six, or eight nodes using *a binary file*.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

Card 2 is required but may be left blank.

Card 2	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
FILENAME	Filename of binary database containing segment pressures as a function of time. There are three sections in this database.
LCID	Optional load curve ID defining segment pressure scale factor as a function of time.

Remarks:

Each database is assumed to be written with a block size equal to a multiple of 512 words, with each block containing 1 or more states. If the data does not complete the last block, it is padded with zeros. The code will open and read the next family file if it detects a phony word or EOF. If the IO returns a zero length read, the code assumes end of IO and stops reading.

Linear interpolation is used if the IO can find the current time between two states. Therefore, the given database can have more states than simulation time steps.

Linear extrapolation will only be used when the IO reaches the end of the database. The last two states will be used and the warning message "MSG_SOL+1323" will be issued to the d3hsp and messag files.

Section	Words	Description
Control Section 64 words	10	Title
	1	NSEG: The absolute value, |NSEG|, specifies the number of segments contained within the file. If NSEG < 0, then the file contains mid-side nodes.
Segment Data 4 words	1	N1: Node ID.
	1	N2: Node ID.
	1	N3: Node ID. Repeat N2 for two-dimensional geometries.
	1	N4: Node ID. Repeat N2 for two-dimensional geometries or repeat N3 for 3-node triangular segments.
Mid-side Nodes 4 Words. Omitted unless NSEG < 0.	1	N5: Optional mid-side node ID.
	1	N6: Optional mid-side node ID.
	1	N7: Optional mid-side node ID.
	1	N8: Optional mid-side node ID.
⋮	4 or 8	⋮
NSEGth segment data	4 or 8	Last set of segment data
"State" Section 1 + |NSEG| words	1	Time
	1	Segment Pressure
	⋮	⋮
	1	Pressure of last segment
⋮	1	⋮
	NSEG	⋮

*LOAD_SEGMENT_FSILNK

Purpose: Apply distributed pressure or force loads from a previous ALE analysis to a specified segment set in the current analysis. This capability trades some of the model's accuracy for a large reduction in model size.

> NOTE: The deck for the "previous" run must include a *DATABASE_BINARY_FSILNK or *DATABASE_BINARY_FSILNK_FULL card to activate the creation of the fsilink file. Either the *LOAD_SEGMENT_FSILNK card (this card) or the *DATABASE_BINARY_FSILNK/FSILNK_FULL card may be in an input deck, *but not both*.

Card 1	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

Card 2	1	2	3	4	5	6	7	8
Variable	NINT	LCID						
Type	I	I						
Default	none	0						

Coupling ID Cards. Read in NINT coupling IDs. Repeat this card as many times as necessary to input all NINT values.

Card 3	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
FILENAME	Filename of the interface linking file.
NINT	Number of couplings for which the previous run provides pressure data
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). The curve referred to by LCID provides a scale factor as a function of time. The pressure data that is read in from the fsilnk file is scaled according to this value.
ID*i*	These must match COUPIDs from the *CONSTRAINED_LAGRANGE_IN_SOLID card from the previous runs. These IDs specify which of the first run's couplings are propagated into this run through pressure data read from the fsilnk file.

The algorithm:

This feature provides a method for using pressure or force time history data from *CONSTRAINED_LAGRANGE_IN_SOLID Lagrangian-to-ALE couplings in one calculation as pressure or force data for the same segment in subsequent calculations. The time range covered by the subsequent calculation *must* overlap the time range of the initial calculation.

First calculation: Write out pressure or force data.

1. Add a *DATABASE_BINARY_FSILNK or *DATABASE_BINARY_FSILNK_FULL card to the first run.

2. Specify the filename for the fsilnk file by adding the following command line argument.

   ls-dyna … fsilnk=**fsi_filename** …

   Without a *DATABASE_BINARY_FSILNK or *DATABASE_BINARY_FSILNK_FULL card this command line argument will have no effect.

Subsequent calculations: Read fsilnk File.

Include this keyword, *LOAD_SEGMENT_FSILNK, and be careful to remove the *DATABASE_BINARY_FSILNK/FSILNK_FULL keyword. Specify the name of the fsilnk file from the previous run on *LOAD_SEGMENT_FSILNK's first data card.

Then, at each time step, the pressures or forces of the specified coupling IDs are set on the Lagrangian-mesh side from data in the fsilnk file. For times outside of the fsilnk file's range, the data is extrapolated.

1. If the current time is before the 1st fsilnk time, then the pressures or forces are set to 0.

2. If the current time is in the range of times in the fsilnk file, then the pressures or forces are linearly interpolated from the data at the two time states in the fsilnk file bracketing the current time.

3. If the current time is after the last fsilnk time, then the pressures or forces are set to the fsilnk pressures or forces at the last time step.

*LOAD_SEGMENT_NONUNIFORM_{OPTION}

To define an ID for the non-uniform segment loading the following option is available:

ID

If the ID option is used, an additional card is required.

Purpose: Apply a distributed load over one triangular or quadrilateral segment defined by three, four, six, or eight nodes. The loading and node numbering convention follows Figure 33-9.

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card Sets. Each segment is specified by a set of the following 3 cards. Include as many sets as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	SF	AT	DT	CID	V1	V2	V3
Type	I	F	F	F	I	F	F	F
Default	none	1.	0.	1.E+16	0	none	none	none

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Card 3	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ID	Loading ID
HEADING	A description of the loading.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). For a load curve ID the load curve must provide pressure as a function of time. For a function ID, the function is expected to have seven arguments: current time minus the birth time, the current x, y, and z coordinates, and the initial x, y, and z coordinates. LT.0: Applies to 3, 4, 6 and 8-noded segments. With this option the load becomes a follower load, meaning that the direction of the load is constant with respect to the local segment coordinate system.
SF	Load curve scale factor
AT	Arrival / birth time for the traction load.
DT	Death time for the traction load
CID	Coordinate system ID
V1, V2, V3	Vector direction cosines relative to coordinate system CID defining the direction of the traction loading. Note that for LCID < 0 this vector rotates with the geometry of the segment.
N1	Node ID
N2	Node ID
N3	Node ID. Repeat N2 for two-dimensional geometries.

VARIABLE	DESCRIPTION
N4	Node ID. Repeat N2 for two-dimensional geometries or repeat N3 for 3-node triangular segments.
N5	Optional mid-side node ID (see Figure 33-12).
N6	Optional mid-side node ID (see Figure 33-12).
N7	Optional mid-side node ID (see Figure 33-12).
N8	Optional mid-side node ID (see Figure 33-12).
P1	Scale factor at node ID, N1.
P2	Scale factor at node ID, N2.
P3	Scale factor at node ID, N3.
P4	Scale factor at node ID, N4.
P5	Scale factor at node ID, N5.
P6	Scale factor at node ID, N6.
P7	Scale factor at node ID, N7.
P8	Scale factor at node ID, N8.

*LOAD_SEGMENT_SET_{OPTION}

To define an ID for the segment loading, the following option is available:

ID

If the ID is defined, an additional card is required.

Purpose: Apply the distributed pressure load over each segment in a segment set. See *LOAD_SEGMENT for a description of the pressure sign convention and remarks on higher order segment definitions.

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Segment Set Cards. Include as many segment set cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	LCID	SF	AT				
Type	I	I	F	F				
Default	none	none	1.	0.				
Remarks		1	2	3				

VARIABLE	DESCRIPTION
SSID	Segment set ID, see *SET_SEGMENT.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). For a load curve ID, the load curve must provide pressure as a function of time. For a function ID, the function for pressure has 10 arguments: time, the 3 current coordinates, the 3 reference coordinates, and the 3 velocities; that is,

VARIABLE	DESCRIPTION
	f(t,x,y,z,x0,y0,z0,vx,vy,vz). See Remark 1.
SF	Load curve scale factor
AT	Arrival time for pressure or birth time of pressure.

Remarks:

1. **Pressure functions.** If LCID is input as -1, then the Brode function is used to determine pressure for the segment set; see *LOAD_BRODE. If LCID is input as -2, then an empirical airblast function is used to determine the pressure for the segments; see *LOAD_BLAST.

2. **Load curve multipliers.** The load curve multipliers may be used to increase or decrease the pressure. The time value is not scaled.

3. **Activation time.** The activation time, AT, is the time during the solution that the pressure begins to act. Until this time, the pressure is ignored. The function value of the load curves will be evaluated at the offset time given by the difference of the solution time and AT, that is, (solution time-AT).

*LOAD_SEGMENT_SET_ANGLE

Purpose: Apply a traction load over a segment set that is dependent on the orientation of a vector. An example application is applying a pressure to a cylinder as a function of the crank angle in an automobile engine. The pressure and node numbering convention follows Figure 33-12.

Card Sets. Include as many sets of Cards 1 and 2 as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDSS	LCID	SCALE	IOPTP	IOPTD		
Type	I	I	I	F	I	I		
Default	none	none	none	1.	0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	NA	NI				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
ID	Loading ID
IDSS	Segment set ID.
LCID	Load curve or function ID defining the traction as a function of the angle. If IOPTP, defined below, is 0, define the abscissa between 0 and 2π radians or 0 and 360 degrees depending on IOPTD.
SCALE	Scale factor on value of the load curve or function.
IOPTP	Flag for periodicity of load curve LCID:

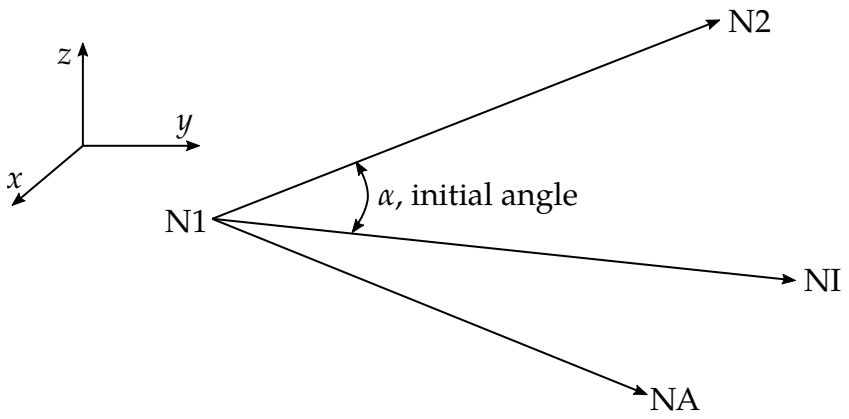


Figure 33-13. Orientation of rotating vector

VARIABLE	DESCRIPTION
	EQ.0: Requires the load curve to be defined between 0 and 2π. This is useful, for example, for modeling an engine that is running at a steady state since each rotation will experience the same loading. EQ.1: Load curve defined over the full range of angles for modeling a transient response. A different response is, therefore, permitted on the second and subsequent revolutions.
IOPTD	Flag for specifying angle units of the load curve LCID: EQ.0: Load curve or function argument is in radians. EQ.1: Load curve or function argument is in degrees.
N1	The node specifying the tail of the rotating vector. See Figure 33-13.
N2	The node specifying the head of the rotating vector. See Figure 33-13.
NA	The node specifying the head of the vector defining the axis of rotation. The node N1 specifies the tail. See Figure 33-13.
NI	The node specifying the orientation of the vector at an angle of zero. If the initial angle is zero, NI should be equal to N2. See Figure 33-13.

*LOAD_SEGMENT_SET_NONUNIFORM_{OPTION}

To define an ID for the non-uniform segment loading the following option is available:

ID

If the ID is defined, an additional card is required.

Purpose: Apply the traction load over one triangular or quadrilateral segment defined by three or four nodes. The pressure and node numbering convention follows Figure 33-12.

ID Card. Additional card for ID keyword option.

Optional	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card Sets. Include as many pairs of Cards 1 and 2 as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	LCID	SF	AT	DT	ELTYPE		
Type	I	I	F	F	F	A		
Default	none	none	1.	0.	10^{16}	↓		

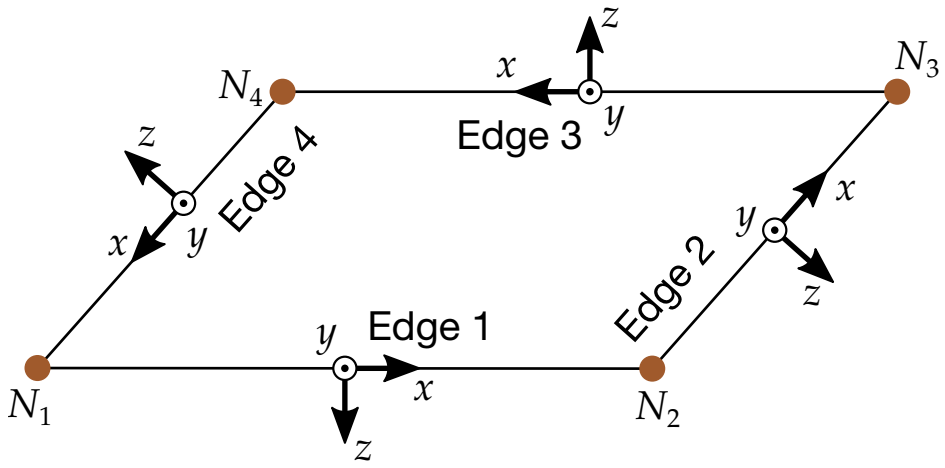


Figure 33-14. Local coordinates system for edge load.

Card 2	1	2	3	4	5	6	7	8
Variable	CID	V1	V2	V3				
Type	I	F	F	F				
Default	0	none	none	none				

VARIABLE	DESCRIPTION
ID	Loading ID
HEADING	A description of the loading.
SSID	Segment set ID.
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION). The seven arguments for the function are current time minus the birth time, the current x, y, and z coordinates, and the initial x, y, and z coordinates. LT.0: Applies to 3, 4, 6 and 8-noded segment sets. With this option the load becomes a follower load, meaning that the direction of the load is constant with respect to the local segment coordinate system. The local coordinate system for edge load, see ELTYPE, is shown in Figure 33-14.
SF	Load curve scale factor
AT	Arrival/birth time for pressure.

VARIABLE	DESCRIPTION
DT	Death time for pressure.
ELTYPE	Optional edge loading type. If left blank, pressure on the segment will be applied. EQ.EF1: Distributed force per unit length along edge 1, Figure 33-14. EQ.EF2: Distributed force per unit length along edge 2, Figure 33-14. EQ.EF3: Distributed force per unit length along edge 3, Figure 33-14. EQ.EF4: Distributed force per unit length along edge 4, Figure 33-14.
CID	Coordinate system ID
V1, V2, V3	Vector direction cosines relative to the coordinate system CID defining the direction of the traction loading. Note that for LCID < 0 this vector rotates with the geometry of the segment.

*LOAD_SEISMIC_FREE_FIELD

Purpose: Apply a seismic load consistent with representative free-field displacement and stress fields. This feature provides a non-reflecting link from the boundary of the modeled soil domain to a soil column representing the free-field site response during a seismic event. This keyword follows the "free-field boundary elements" method outlined by Løkke & Chopra in PEER Report No. 2019/02, Section 3.5.2 and Figure 3.3(b).

Card 1	1	2	3	4	5	6	7	8
Variable	LSSID	RSSID	LCD	LCS				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
LSSID	Segment set ID to which the consistent seismic load is applied (see Remarks 3, 6, and 7)
RSSID	Segment set ID attached to reference elements (soil column) representing free-field site response (see Remarks 3, 5, and, 6)
LCD	Load curve defining ρV_p (ordinate) as a function of the global Z-coordinate (abscissa). Here, ρ is density and V_p is the P-wave speed. See Remark 10.
LCS	Load curve defining ρV_s (ordinate) as a function of the global Z-coordinate (abscissa). Here, ρ is density and V_s is the shear wave speed. See Remark 10.

Remarks:

1. **Terminology and concept.** A "site" is represented by horizontal layers of soil and/or rock. Each layer may have different linear or nonlinear material properties. In the absence of any buildings or other construction, the site's response to a given earthquake loading is known as the "free-field" response. The free field is idealized as having infinite horizontal extent, and the material properties are assumed not to vary in the horizontal directions. The earthquake loading is assumed to consist of shear waves and/or compression waves that travel vertically and reflect at the ground surface. In the horizontal directions, the waves

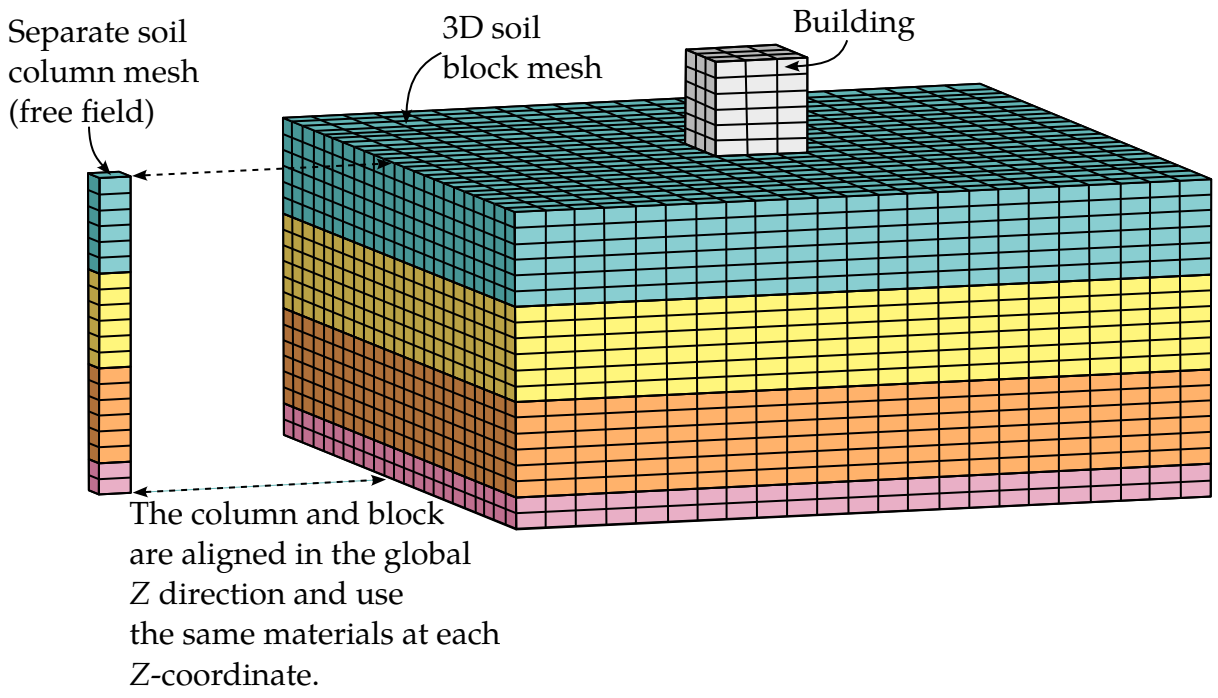


Figure 33-15. Example of how to use this feature. The column models the free-field response, which is then loaded onto the soil block. The column and block must be aligned in the global *Z*-direction with the same materials at each *Z*-coordinate. The colors indicate the different materials in the layers. The segment set LSSID to which the seismic load is applied should coat all the vertical side faces and the bottom face of the soil block. The segment set RSSID should coat one of the vertical side faces and the bottom face of the column.

are assumed to be uniform and of infinite extent. Thus, at any given depth, the free-field response is assumed not to vary with horizontal position; therefore, it can be modelled by a one-dimensional "column" of elements. However, when a building, dam, or other structure is present, the structure will be loaded by shaking transmitted from the ground, and the presence of the structure will modify the response of the soil compared to the free-field response ("Soil-Structure Interaction" or "SSI"). The effect of the structure on the site's response decreases with distance from the structure; at large distances from the structure, the site response tends towards the free-field site response.

When modeling SSI problems, you must decide how far away from the structure to place the boundaries of the soil, meaning how large a "soil block" must be included in the model. If too large, computing resources are wasted. If too small, the presence of the artificial model boundary close to the structure affects the results. The objective of *LOAD_SEISMIC_FREE_FIELD is to enable a smaller soil block while providing results that are insignificantly affected by the closeness of the boundary to the structure.

2. **Model orientation.** This keyword assumes a 3D model in which the global Z-axis points vertically upwards.

3. **Model contents.** The model should contain a block of soil (which may also have other items attached to it, such as a building) and also a separate soil column (see Figure 33-15). The column represents the free field, meaning the site in the absence of any construction. The soil column must be separate from the soil block, meaning it must not share nodes with the soil block. Both the column and the block will be analyzed simultaneously in the same model. The column and the block should be aligned in the Z-direction. It is expected that the material properties at any given Z-coordinate will be the same in the column model as in the soil block.

4. **How it works.** *LOAD_SEISMIC_FREE_FIELD effectively transfers the motion at each value of Z-coordinate from the soil column (free field) to the soil block boundaries covered by the segment set LSSID. It does this by penalizing deviations from the soil column velocity with a damping stress, in which the damping property is related to the soil material properties in the same way as a non-reflecting boundary. Thus, the boundary is a non-reflecting one that moves with the free-field motion. Additionally, stresses in the soil column elements normal to the boundary are transferred as pressure loads onto the boundary of the soil block. The interaction between the soil column and the soil block is one-way: no forces are transferred back to the soil column.

5. **Soil column.** The soil column represents the free-field site response and consists of a stack of soil elements, meaning one element wide in the horizontal dimensions and multiple elements vertically. The soil column should have the necessary boundary conditions, constraints, and loading to replicate the desired free-field site response.

6. **Segment sets.** The segment set LSSID should coat all the vertical side faces and the bottom face of the soil block. The segment set RSSID should coat one of the vertical side faces and the bottom face of the column. See Figure 33-15.

7. **Soil block.** When the segment sets are as described in Remark 6, the soil block boundary conditions and seismic loading are applied entirely by *LOAD_SEISMIC_FREE_FIELD, and no other boundary conditions or seismic loading inputs on the soil block are required. However, it is possible to use *LOAD_SEISMIC_FREE_FIELD only for a subset of the soil block boundaries (e.g., for the sides but not the bottom face), in which case the remaining boundaries require additional cards to define boundary conditions, seismic loading, etc.

8. **Gravity loading.** If gravity loading and/or a nonzero initial stress state are desired, these should be applied to both the soil column and the soil block. The soil column will require some way to react to the gravity load (for example, an

upward pressure load on the bottom face of the column equal to the weight of the column), but do not apply this same condition to the soil block. Since *LOAD_SEISMIC_FREE_FIELD applies the normal stress in the soil column element as a pressure load on the face of the soil block, in the case of the bottom boundary, this will provide a condition that is close to equilibrium under gravity. However, if a small adjustment to this pressure is found to be needed to prevent the soil block from drifting vertically during the analysis, this may be achieved by adding a separate pressure loading to the bottom face of the soil block and calibrating the pressure by trial and error.

9. **Material types.** *LOAD_SEISMIC_FREE_FIELD works with all non-rigid material types.

10. **LCD and LCS.** LCD and LCS are curves that contain Z-coordinates on the x-axis and a material property of the soil on the y-axis. You must supply these curves. They are not generated automatically.

The curve LCD gives the material property Y_{LCD}, which is density times the P-wave speed:

$$Y_{\mathrm{LCD}} = \rho V_{\mathrm{p}} = \rho\sqrt{(B + 4G/3)/\rho} \ .$$

The curve LCS provides the material property Y_{LCS}, which is density times shear wave speed:

$$Y_{\mathrm{LCS}} = \rho V_{\mathrm{s}} = \rho\sqrt{G/\rho} \ .$$

In the above, ρ is density, G is the elastic shear modulus, and B is the elastic bulk modulus.

***LOAD_SEISMIC_SSI**_*OPTION1_{OPTION2}*

Available options for *OPTION1* include:

NODE

SET

POINT

DECONV

OPTION2 allows an optional ID to be given:

ID

Purpose: Apply earthquake load due to free-field earthquake ground motion at certain locations — defined by either nodes or coordinates — on a soil-structure interface for use in earthquake soil-structure interaction analysis. The specified motions determine a set of effective forces in the soil elements adjacent to the soil-structure interface, according to the effective seismic input–domain reduction method [Bielak and Christiano (1984)].

Card Summary:

Card Sets. Include as many sets of the following cards (depending on the keyword options) as needed. This input ends at the next keyword ("*") card.

Card ID. Include this card if using the ID keyword option.

ID	HEADING

Card 1a. This card is included if the NODE or SET keyword option is used.

SSID	TYPEID	GMX	GMY	GMZ			

Card 1b. Include this card if using the POINT or DECONV keyword option.

SSID	XP	YP	ZP	GMX	GMY	GMZ

Card 2. This card is required.

SF	CID	BIRTH	DEATH	ISG	IGM	PSET	VDIR

Data Card Definitions:

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
ID	Optional ID. This ID does not need to be unique.
HEADING	An optional descriptor for the given ID.

Node and set Cards. Card 1 for keyword options NODE and SET:

Card 1a	1	2	3	4	5	6	7	8
Variable	SSID	TYPEID	GMX	GMY	GMZ			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

Point Cards. Card 1 for keyword options POINT and DECONV.

Card 1b	1	2	3	4	5	6	7	8	9	10
Variable	SSID	XP		YP		ZP		GMX	GMY	GMZ
Type	I	F		F		F		I	I	I
Default	none	0.		0.		0.		none	none	none

VARIABLE	DESCRIPTION
SSID	Soil-structure interface ID.
TYPEID	Node ID (NID in *NODE) or nodal set ID (SID in *SET_NODE).

VARIABLE	DESCRIPTION
XP	x coordinate of ground motion location.
YP	y coordinate of ground motion location.
ZP	z coordinate of ground motion location.
GMX	Acceleration load curve or ground motion ID for motion in the (local) x-direction.
GMY	Acceleration load curve or ground motion ID for motion in the (local) y-direction.
GMZ	Acceleration load curve or ground motion ID for motion in the (local) z-direction.

Card 2	1	2	3	4	5	6	7	8
Variable	SF	CID	BIRTH	DEATH	ISG	IGM	PSET	VDIR
Type	F	I	F	F	I	I	I	I
Default	1.	0	0.	10^{28}	0	0	none	3

VARIABLE	DESCRIPTION
SF	Ground motion scale factor.
CID	Coordinate system ID; see *DEFINE_COORDINATE_SYSTEM.
BIRTH	Time at which specified ground motion is activated.
DEATH	Time at which specified ground motion is removed: EQ.0.0: default set to 10^{28}
ISG	Definition of soil-structure interface: EQ.0: SSID is the ID for the soil-structure interface defined by *INTERFACE_SSI_ID for a non-matching mesh between soil and structure. For the DECONV keyword option, ISG = 0 additionally flags that the free field within motion is computed at depth.

VARIABLE	DESCRIPTION
	EQ.1: SSID is the ID of the segment set on the soil side of the soil-structure interface. This segment set should be oriented toward the structure. For the DECONV, ISG = 1 additionally flags that the free-field outcrop motion is computed at depth.
IGM	Specification of ground motions GMX, GMY, GMZ: EQ.0: Ground motions are specified as acceleration load curves. See *DEFINE_CURVE EQ.1: Both ground accelerations and velocities specified using *DEFINE_GROUND_MOTION.
PSET	Soil part set through which ground motion travels (DECONV option only)
VDIR	Vertical direction (local if CID ≠ 0) for ground motion propagation (DECONV option only): EQ.-1: -x-direction EQ.-2: -y-direction EQ.-3: -z-direction EQ.1: x-direction EQ.2: y-direction EQ.3: z-direction

Remarks:

1. **Ground Motion at Soil-Structure Interface.** The ground motion at any node on a soil-structure interface is computed as follows:

   a) If the node coincides with a location where ground motion is specified, that ground motion is used for that node.

   b) If the node does not coincide with a location where ground motion is specified, the ground motion at that node along a particular degree of freedom is taken as a weighted average of all the ground motions set on the interface along that degree of freedom. The weights are inversely proportional to the node's distance from the ground motion location.

2. **Multiple Ground Motions.** Multiple ground motions specified at the same location are added together to obtain the resultant ground motion.

3. **Spatially-Uniform Ground Motion.** To model a spatially-uniform ground motion m on a soil-structure interface, set the ground motion at only one location on that interface. Specifying the ground motion at more than one point on a soil-structure interface results in spatially-varying ground motion on that interface.

4. **Deconvolution input.** The DECONV option provides a simplified ground motion input for a soil-structure interface that goes below the flat, top surface of the soil. The free-field ground motion can be set at the top of the soil or below the surface. The computation for the corresponding motion at interface nodes at depth assumes the free-field movement only propagates vertically. ISG = 0 flags that the within motion is computed, while ISG = 1 flags that the outcrop motion is calculated. Currently, this keyword option is available only when modeling the soil domain with either *MAT_ELASTIC or *MAT_BIOT_HYSTERETIC. Other materials can be supported upon request.

*LOAD_SEISMIC_SSI_AUX_{OPTION}

OPTION allows an optional ID to be given:

ID

Purpose: Apply earthquake load due to free-field earthquake ground motion on a soil-structure interface, where the free-field motions were recorded from a previous analysis using *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED. The specified motions are used to compute a set of effective forces in the soil elements adjacent to the soil-structure interface, according to the effective seismic input–domain reduction method [Bielak and Christiano (1984)].

This card supersedes the optional Card 3 in *INTERFACE_SSI

Card Sets. Include as many pairs of Cards 1 and 2 as necessary. This input ends at the next keyword ("*") card. If the ID keyword option is used, include a separate ID card for each pair.

ID Card. Additional card for the ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Card 1	1
Variable	FILENAME
Type	A80

Card 2	1	2	3	4	5	6	7	8
Variable	SSID	GMSET	SF	BIRTH	DEATH	ISG	MEMGM	
Type	I	I	F	F	F	I	I	
Default	none	none	1.	0.	10^{28}	0	2500000	

VARIABLE	DESCRIPTION
ID	Optional ID. This ID does not need to be unique.
HEADING	An optional descriptor for the given ID
FILENAME	Name of binary file containing recorded motions
SSID	Soil-structure interface ID
GMSET	Identifier for set of recorded motions; see *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED
SF	Recorded motion scale factor
BIRTH	Time at which specified recorded motion is activated
DEATH	Time at which specified recorded motion is removed. EQ.0.0: Set to the default, 10^{28}
ISG	Definition of soil-structure interface: EQ.0: SSID is an ID for the soil-structure interface defined by *INTERFACE_SSI_ID for a non-matching mesh between soil and structure. EQ.1: SSID is the ID of the segment set on the soil side of the soil-structure interface. This segment set should be oriented toward the structure.
MEMGM	Size in words of buffer allocated to read in recorded motions

Remarks:

1. **Binary File for Recorded Motions.** The free-field motions are read in from a binary file created in a previous run using *INTERFACE_SSI_AUX or *INTERFACE_SSI_AUX_EMBEDDED.

2. **Multiple Ground Motion Inputs.** Multiple ground motion files can be specified at the same interface. All the ground motions are added together to obtain the resultant free-field motion at that interface.

3. **Interface Geometry Match.** It is assumed that the nodes of the soil-structure interface lie on the surface of the interface used in the previous run to record the ground motion.

***LOAD_SHELL**_*OPTION1_{OPTION2}*

Available options for *OPTION1* include:

ELEMENT

SET

Available options for *OPTION2* include:

ID

If the ID is defined, an additional card is required.

Purpose: Apply the distributed pressure load over a single shell element or a shell element set. The numbering of the shell nodal connectivities must follow the right hand rule with positive pressure acting in the negative *t*-direction. See Figure 33-12. This option applies to the three-dimensional shell elements only.

ID Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	HEADING						
Type	I	A70						

Shell Cards. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID/ESID	LCID	SF	AT				
Type	I	I	F	F				
Default	none	none	1.	0.				

VARIABLE	DESCRIPTION
EID/ESID	Shell ID (EID) or shell set ID (ESID), see *ELEMENT_SHELL or *SET_SHELL.

VARIABLE	DESCRIPTION
LCID	Load curve ID, see *DEFINE_CURVE. EQ.-1: Brode function; see *LOAD_BRODE. EQ.-2: ConWep function; see *LOAD_BLAST.
SF	Load curve scale factor. See Remark 1.
AT	Arrival time for pressure or birth time of pressure.

Remarks:

1. **Multipliers.** The load curve multipliers may be used to increase or decrease the pressure. The time value is not scaled.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *LOAD_SHELL_ELEMENT
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  From a sheet metal forming example.  A blank is hit by a punch, a holder is
$  used to hold the blank on its sides.  All shells on the holder are given a
$  pressure boundary condition to clamp down on the blank.  The pressure
$  follows load curve 3, but is scaled by -1 so that it applies the load in the
$  correct direction.  The load starts at zero, but quickly rises to 5 MPa
$  after 0.001 sec.  (Units of this model are in: ton, mm, s, N, MPa, N-mm)
$
*LOAD_SHELL_ELEMENT
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      eid      lcid        sf        at
     30001         3 -1.00E+00       0.0
     30002         3 -1.00E+00       0.0
     30003         3 -1.00E+00       0.0
     30004         3 -1.00E+00       0.0
     30005         3 -1.00E+00       0.0
     30006         3 -1.00E+00       0.0
     30007         3 -1.00E+00       0.0
$
$  Note: Just a subset of all the shell elements of the holder is shown above,
$        in practice this list contained 448 shell element id's.
$
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
         3
$
$           abscissa            ordinate
               0.000                 0.0
               0.001                 5.0
               0.150                 5.0
$
$
```

$$$

*LOAD_SPCFORC

Purpose: When used in a full-deck restart run, this card will apply the SPC constraint forces from the initial run on the corresponding degrees of freedom in the current run. This is useful when modeling unbounded domains using a non-reflecting boundary while incorporating static stresses computed in the initial run: the fixed constraints on the outer boundary in the initial static analysis are removed in the transient analysis and replaced by equivalent static forces.

While *BOUNDARY_NON_REFLECTING acts similarly if dynamic relaxation is used for the static analysis, this approach works for any method used to preload the model.

No parameters are necessary for this card.

*LOAD_SSA

Purpose: The Sub-Sea Analysis (SSA) capability allows a simple and efficient way of loading the structure to account for the effects of the primary shock wave and the subsequent bubble oscillations of an underwater explosion. It achieves its efficiency by approximating the pressure scattered by air and water-backed plates and the pressure transmitted through a water-back plate. The loading incorporates the plane wave approximation for direct shock response and the virtual mass approximation for bubble response. *LOAD_SSA does not implement a doubly asymptotic approximation of transient fluid-structure interaction.

Card 1	1	2	3	4	5	6	7	8
Variable	VS	DS	REFL	ZB	ZSURF	FPSID	PSID	NPTS
Type	F	F	I	F	F	I	I	I
Default	none	none	0	0.	0.	0	0	1

Card Sets. Include as many pairs of Cards 2 and 3 as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	A	ALPHA	GAMMA	KTHETA	KAPPA			
Type	F	F	F	F	F			
Default	none	none	none	none	none			

Card 3	1	2	3	4	5	6	7	8
Variable	XS	YS	ZS	W	TDELY	RAD	CZ	
Type	F	F	F	F	F	F	F	
Default	none	none	none	none	none	none	none	

VARIABLE	DESCRIPTION
VS	Sound speed in fluid
DS	Density of fluid
REFL	Consider reflections from sea floor. EQ.0: off EQ.1: on
ZB	Z coordinate of sea floor if REFL = 1, otherwise, not used.
ZSURF	Z coordinate of sea surface
FPSID	Part set ID of parts subject to flood control. See Remark 2. Use the *SET_PART_COLUMN option where the parameters A1 and A2 must be defined as follows: Parameter A1: Flooding status: EQ.1.0: Fluid on both sides. EQ.2.0: Fluid outside, air inside. EQ.3.0: Air outside, fluid inside. EQ.4.0: Material or part is ignored. Parameter A2: Tubular outer diameter of beam elements. For shell elements this input must be greater than zero for loading.
PSID	Part set IDs of parts defining the wet surface. The elements defining these parts must have their outward normals pointing into the fluid. See Figure 33-16. EQ.0: all parts are included. GT.0: the part set ID.
NPTS	Number of integration points for computing pressure (1 or 4)
A	Shock pressure parameter
ALPHA	α, shock pressure parameter
GAMMA	γ, time constant parameter
KTHETA	K_θ, time constant parameter
KAPPA	κ, ratio of specific heat capacities

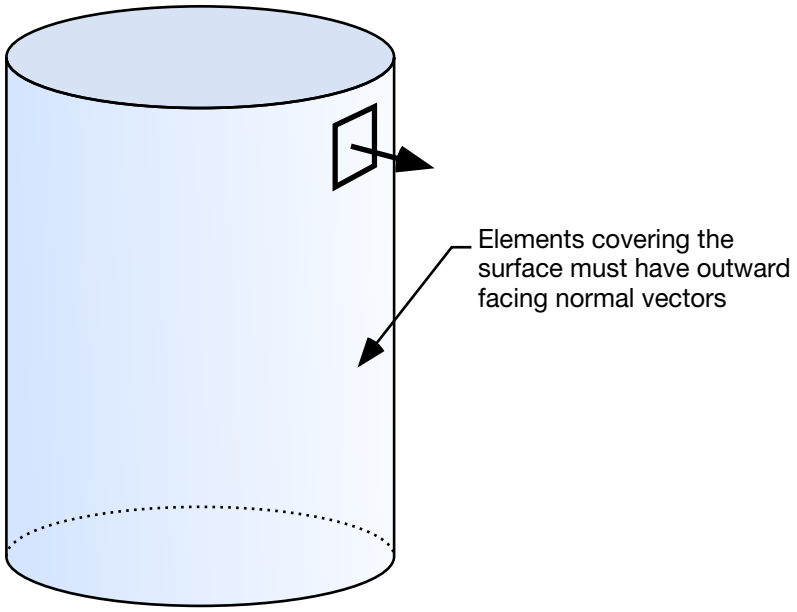


Figure 33-16. The shell elements interacting with the fluid must be numbered such that their outward normal vector points into the fluid media.

VARIABLE	DESCRIPTION
XS	*X* coordinate of charge
YS	*Y* coordinate of charge
ZS	*Z* coordinate of charge
W	Weight of charge
TDELY	Time delay before charge detonates
RAD	Charge radius
CZ	Charge depth

Remarks:

1. **Units.** SSA assumes the model is in MKS units. If it is in another system of units, *CONTROL_COUPLING should be used to account for the conversion.

2. **Flooding Status.** The "flooding status" is instrumental in determining how the model parts are loaded. If A1 = 1.0, the front of the plate as defined by the outward normal is exposed to the incident pressure. The back of the plate is not exposed to the incident pressure but feels a transmitted pressure that resists

plate motion. If A1 = 2.0, then the plate has fluid on the outside as determined by the outward normal. It is exposed to the incident pressure and feels the scattered pressure. No loading is applied to the back side. If A1 = 3.0, then air is on the front of the plate and water is on the back. Neither the front nor the back of the plate is exposed to the incident pressure, but the motion of the plate is resisted by pressure generated on the back of the plate when it moves. Transmitted pressures are assumed not to strike another plate.

3. **Pressure History of Primary Shockwave.** The pressure history of the primary shockwave at a point in space through which a detonation wave passes is given as:

$$P(t) = P_m e^{\frac{-t}{\theta}}$$

where P_m and the time constant θ below are functions of the type and weight W of the explosive charge and the distance Q from the charge.

$$P_{peak} = A\left[\frac{W^{1/3}}{Q}\right]^{\alpha}$$

$$\theta = K_\theta W^{1/3}\left[\frac{W^{1/3}}{Q}\right]^{\gamma}$$

where A, α, γ, and K_θ are constants for the explosive being used.

4. **Model Orientation.** The positive global Z-direction points vertically upward, from the sea floor to water surface. The finite element model must be oriented accordingly.

*LOAD_STEADY_STATE_ROLLING

Purpose: Steady state rolling analysis is a generalization of *LOAD_BODY, allowing the user to apply body loads to part sets due to translational and rotational accelerations in a manner that is more general than the *LOAD_BODY capability. *LOAD_STEADY_-STATE_ROLLING may be invoked multiple times as long as no part has the command applied more than once. Furthermore, the command may be applied to arbitrary meshes, that is, axisymmetric meshes are not required.

Card Sets. Include as many sets consisting of the following four cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	PSID						
Type	I	I						
Default	none	none						

Card 2	1	2	3	4	5	6	7	8
Variable	N1	N2	LCD1	LCD1R				
Type	I	I	I	I				
Default	0	0	0	LCD1				

Card 3	1	2	3	4	5	6	7	8
Variable	N3	N4	LCD2	LCD2R				
Type 4	I	I	I	I				
Default	0	0	0	LCD2				

Card 4	1	2	3	4	5	6	7	8
Variable	N5	N6	LCD3	LCD3R				
Type	I	I	I	I				
Default	0	0	0	LCD3				

VARIABLE	DESCRIPTION
ID	ID
PSID	Part set ID
N1	Node 1 defining rotational axis
N2	Node 2 defining rotational axis
LCD1	Load curve defining angular velocity around rotational axis.
LCD1R	Optional load curve defining angular velocity around rotational axis for dynamic relaxation. LCD1 is used during dynamic relaxation if LCD1R is not defined.
N3	Node 3 defining turning axis
N4	Node 4 defining turning axis
LCD2	Load curve defining angular velocity around turning axis.
LCD2R	Optional load curve defining angular velocity around turning axis for dynamic relaxation. LCD2 is used during dynamic relaxation if LCD2R is not defined.
N5	Node 5 defining translational direction
N6	Node 6 defining translational direction
LCD3	Load curve defining translational velocity in translational direction.
LCD3R	Optional load curve defining translational velocity in translational direction. LCD3 is used during dynamic relaxation if LCD3R is not defined.

Remarks:

1. **Model.** The steady state rolling capability adds inertial body loads in terms of a moving reference defined by the user input. The current coordinates are defined in terms of the displacement, u, and the moving reference frame, Y,

$$x_{SSR} = u + Y \quad \dot{x}_{SSR} = \dot{u} + \dot{Y} \quad \ddot{x}_{SSR} = \ddot{u} + \ddot{Y}$$
$$Y = R(\omega_2 t)[R(\omega_1 t)(X - X_O) - X_C] + Y_T(t)$$

where R is the rotation matrix obtained by integrating the appropriate angular velocity, the magnitude of the angular velocities ω_1 and ω_2 are defined by load curves LCD1 and LCD2 respectively, and the directions are defined by the current coordinates of the node pairs N1-N2 and N3-N4. The velocity corresponding to the translational term, $Y_T(t)$, is defined in magnitude by LCD3 and in direction by the node pair N5-N6. The initial coordinates of the nodes are X, X_O is the initial coordinate vector of node N1 and X_C is the initial coordinate vector of node N3. If data defining an angular velocity is not specified, the velocity is defaulted to zero, and R is the identity matrix. In a similar manner, if the translational velocity is not specified, it is defaulted to zero.

2. **Restrictions.** This capability is useful for initializing the stresses and velocity of tires during dynamic relaxation and rolling processes in manufacturing. It is available for solid formulations 1, 2, 10, 13, and 15, and for shell formulations 2, 4, 5, 6, 16, 25, 26, and 27. It is *not* available for beams and tshells. It is available for implicit and explicit simulations, but it is *not* available for eigenvalue problems. To invoke this keyword for dynamic relaxation, specify that the load curves are used during dynamic relaxation (SIDR = 1 or 2 on *DEFINE_CURVE). At the end of the dynamic relaxation, the velocities of the parts with this load are set to $\dot{x}_{SSR}$ while the remaining parts are initialized according to the input file.

3. **Convergence and Stability.** Users must ensure that the appropriate load curves are turned on during the relaxation process, and if implicit dynamic relaxation is used, that sufficient constraints are applied during the initialization to remove any rigid body motion and that they are removed at the end of the dynamic relaxation. The implicit iteration convergence rate is often improved by adding the geometric stiffness matrix using *CONTROL_IMPLICIT_GENERAL. A consistent tangent matrix is available by using *CONTROL_IMPLICIT_GENERAL, and while it improves the convergence rate with problems with small strains, it is often unstable for problems with large strains. The *CONTROL_STEADY_STATE_ROLLING options should be used to ramp up the frictional forces to obtain smooth solutions and good convergence rates.

4. **Angular Velocity and Hoop Strain.** To obtain the free-rolling angular velocity, the tire should be first inflated, then brought into contact with the road while the frictional force is ramped up with a load curve and a large value of SCL_K

specified in *CONTROL_STEADY_STATE_ROLLING. The angular velocity of the tire is then slowly varied over a range that covers the free rolling velocity. The free rolling velocity is obtained when either the frictional force in the direction of rolling or the moment about the tire axis is near zero. For a tire with an initial radius of R and a translational velocity of V, the approximate value for the free rolling value of the rolling velocity is

$$\omega = \frac{V}{(1+\varepsilon)R} \ ,$$

where ε is the hoop strain of the rolling tire. For a first guess, the hoop strain can be set to 0.0, and the rolling velocity will be within 10% of the actual value. After the first calculation, a smaller range bracketing the free rolling velocity should be used in a second calculation to refine the free rolling velocity. An accurate value of the free rolling velocity is necessary for subsequent analyses, such as varying the slip angle of the tire.

5. **Slip Angle.** A time varying slip angle can be specified by moving one of the nodes defining the direction vector of the translational velocity. To check that the stiffness scale factor in *CONTROL_STEADY_STATE_ROLLING is high enough, a complete cycle from a zero slip angle to a maximum value, then back to zero, should be performed. If the loading and unloading values are reasonably close, then the stiffness scale factor is adequate.

*LOAD_STIFFEN_PART_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Staged construction. Available for solid, shell, thick shell, and beam elements. See also *DEFINE_STAGED_CONSTRUCTION_PART, which provides an alternative, simpler input method for the same capability.

Construction Cards. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	LC	(blank)	STGA	STGR			
Type	I	I		I	I			
Default	none	0		0	0			

VARIABLE	DESCRIPTION
PID	Part ID (or Part Set ID for the SET keyword option)
LC	Load curve defining factor as a function of time. See Remark 5.
STGA	Construction stage at which part is added (optional)
STGR	Construction stage at which part is removed (optional)

Remarks:

1. **Removing and Adding Parts.** For parts that are initially present but are removed during the analysis (for example, soil that is excavated), the stiffness factor starts at 1.0. Before the part is deleted, the stiffness factor should ramp down to a small value, such as 10^{-6}, over a duration long enough to avoid introducing shock or dynamic effects. The elements of a part introduced during the construction, such as retaining walls, are initially present in the model; however, the factor for this part is initially set to a low value, such as 10^{-6}. The stiffness factor increases to 1.0 when the part is added. A factor that increases from 10^{-6} to 1.0

and then reduces back to 10^{-6} can be used for temporary retaining walls, props, etc.

2. **Increasing and Decreasing Factor.** When the factor is increasing, it applies *only* to the stiffness and strength of the material in response to *subsequent* strain increments, not to any existing stresses. When the factor is decreasing, it applies to existing stresses as well as to the stiffness and strength.

3. **Compatible Material Models.** This feature works with all material models when used only to *reduce* the stiffness (e.g. parts that are excavated, not parts that are added during construction). When used to *increase* the stiffness from an initial low value, this feature works for most material types but not for hyperelastic types as well as certain others. There is no error check at present to detect STIFFEN_PART being used with an inappropriate material model. Symptoms of resulting problems would include non-physically large stresses when a part stiffens due to the accumulated strains in the "dormant" material since the start of the analysis.

4. **Gravity Factor.** This feature is generally used with *LOAD_GRAVITY_PART. The same curve is often used for the stiffness factor and the gravity factor.

5. **Factor Curve.** There are 3 methods for defining the factor as a function of time:

   a) LC overrides STGA and STGR if LC is non-zero.

   b) STGA and STGR refer to stages at which the part is added and removed: the stages are defined in *DEFINE_CONSTRUCTION_STAGES. If STGA is zero, the part has full stiffness at time zero. If not, it starts with a value equal to FACT (on *CONTROL_STAGED_CONSTRUCTION) and increases immediately to 1.0 at the start of stage STGA. If STGR is zero, the stiffness factor continues at 1.0 until the end of the analysis. If not, it ramps down from 1.0 to FACT over the ramp time at the start of stage STGR.

   c) *DEFINE_STAGED_CONSTRUCTION_PART can be used instead of *LOAD_STIFFEN_PART to define this loading. During initialization, a *LOAD_STIFFEN_PART card will be created, and the effect is the same as using the STGA and STGR method described above. In many cases using *DEFINE_STAGED_CONSTRUCTION_PART is more convenient than this keyword.

*LOAD_SUPERPLASTIC_FORMING

Purpose: Perform superplastic forming (SPF) analysis. This option can be applied to 2D and 3D solid elements and to 3D shell elements, and has been implemented for both explicit and implicit analyses. The pressure loading controlled by the load curve ID given below is scaled to maintain a constant maximal strain rate or other target value.

This option must be used with material model 64, *MAT_RATE_SENSITIVE_POWER-LAW_PLASTICITY, for strain rate sensitive, powerlaw plasticity. For the output of data, see *DATABASE_SUPERPLASTIC_FORMING. Mass scaling is recommended in SPF applications.

New options to compute the target value with various averaging techniques and autojump options to control the simulation are implemented. Strain-rate speedup is also available. See Remarks 5, 6, and 7 for details.

Card Sets. Include as many sets consisting of the following four cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	LCP1	CSP1	NCP1	LCP2	CSP2	NCP2	PCTS1	PCTS2
Type	I	I	F	I	I	F	F	F
Default	none	none	none	optional	optional	optional	100.0	100.0
Remarks				1	1	1		

Card 2	1	2	3	4	5	6	7	8
Variable	ERATE	SCMIN	SCMAX	NCYL	NTGT	LEVEL	TSRCH	AT
Type	F	F	F	I	I	I	F	F
Default	none	none	none	0	1	0	none	0.0
Remarks				2		5		

Card 3	1	2	3	4	5	6	7	8
Variable	TPEAK	TNEG	TOSC	POSC	PDROP	RILIM	RDLIM	STR
Type	F	F	F	F	F	F	F	F
Default	10.0	5.0	10.0	1.0	2.0	1.0	1.0	0.0
Remarks								6

Card 4	1	2	3	4	5	6	7	8
Variable	THRES	LOWER	UPPER	TFACT	NTFCT	BOX		
Type	F	F	F	F	I	I		
Default	5.0	90.0	99.0	1.0	10	0		
Remarks				7	7	8		

VARIABLE	DESCRIPTION
LCP1	Load curve number for Phase I pressure loading; see *DEFINE_CURVE. The scaled version of this curve as calculated by LS-DYNA is output to curve1. See Remark 3.
CSP1	Contact surface number to determine completion of Phase I.
NCP1	Percent of nodes in contact to terminate Phase I; see *CONTACT_*OPTION*.
LCP2	Load curve number for Phase II pressure loading (reverse); see *DEFINE_CURVE. The scaled version of this curve as calculated by LS-DYNA is output to curve2. See Remark 3.
CSP2	Contact surface to determine completion of Phase II; see *CONTACT_OPTION.
NCP2	Percent of nodes in contact to terminate Phase II.

VARIABLE	DESCRIPTION
PCTS1	Percentage of nodes-in-contact to active autojump in Phase I forming.
PCTS2	Percentage of nodes-in-contact to active autojump in Phase II forming.
ERATE	Desired target value. If it's strain rate, this is the time derivative of the logarithmic strain.
SCMIN	Minimum allowable value for load curve scale factor. To maintain the target value, the pressure curve is scaled. In the case of a snap through buckling the pressure may be removed completely. By putting a value here the pressure will continue to act but at a value given by this scale factor multiplying the pressure curve.
SCMAX	Maximum allowable value for load curve scale factor. Generally, it is a good idea to put a value here to keep the pressure from going to unreasonable values after full contact has been attained. When full contact is achieved the strain rates will approach zero and pressure will go to infinity unless it is limited or the calculation terminates.
NCYL	Number of cycles for monotonic pressure after reversal.
NTGT	Type of the target (controlling) variable: EQ.1: strain rate. EQ.2: effective stress.
LEVEL	Criterion to compute averaged maximum of controlling variable: EQ.0: no average used. GE.1: averaging over neighbors of element with peak value of the controlling variable. This parameter determines the level of neighbor search. EQ.-1: averaging over elements within selective range of peak controlling variable.
TSRCH	Time interval to conduct neighbors search.
AT	Time when SPF Phase I simulation starts.
TPEAK	Additional run time to terminate simulation when maximum pressure is reached.

VARIABLE	DESCRIPTION
TNEG	Additional run time to terminate simulation when percentage change of nodes-in-contact is zero or negative.
TOSC	Additional run time to terminate simulation when percentage change of nodes-in-contact oscillates within a specific value.
POSC	Percentage change to define the oscillation of percentage of nodes-in-contact.
PDROP	Drop in percentage of nodes-in-contact from the maximum to terminate simulation after the specified termination percentage has been reached.
STR	Autojump option or strike-through time (period of time without autojump check): EQ.0: no autojump EQ.-1: autojump controlled by peak pressure EQ.-2: autojump controlled by percentage of nodes in contact EQ.-3: autojump controlled by both above GT.0: strike-through time, then same as STR = -3
THRES	Threshold percentage that gives the threshold value above which elements are considered for average.
LOWER	Lower percentile of elements above the threshold value to be included for average.
UPPER	Upper percentile of elements above the threshold value to be included for average.
RILIM	Maximum percentage change for pressure increment.
RDLIM	Maximum percentage change for pressure decrement.
TFACT	Strain rate speedup factor.
NTFCT	Number of computing cycles to ramp up speedup.
BOX	Box ID or Box Set ID. See Remark 8. GT.0: Box ID; see *DEFINE_BOX. LT.0: |BOX| is box set ID; see *SET_BOX.

Remarks:

1. **Second Phase.** Optionally, a second phase can be defined. In this second phase a *unique* set of pressure segments must be defined whose pressure is controlled by LCP2. During the first phase, the pressure segments of LCP2 are inactive, and likewise, during the second phase the pressure segments of the first phase are inactive. When shell elements are used, the complete set of pressure segments can be repeated in the input with a sign reversal used on the load curve. When solid elements are used, the pressure segments for each phase, in general, must be unique.

2. **NCYCL.** This is an ad hoc parameter which should probably not be used.

3. **Output Files.** There are three output files pressure, curve1, and curve2 from the SPF simulation, and they may be plotted using ASCII → superpl in LS-PrePost. The file curve2 is created only if the second phase is active. The time interval for writing out these files is controlled by *DATABASE_SUPERPLASTIC_FORMING. The files curve1 and curve2 contain the adjusted pressure histories calculated by SPF solver. The file pressure contains time histories of scaling factor, maximal target value, averaged target value, and percentage of contact.

4. **Contact.** The constraint method contact, *CONTACT_CONSTRAINT_NODES_TO_SURFACE, is recommended for superplastic forming simulations since the penalty methods are not as reliable when mass scaling is applied. Generally, in superplastic simulations mass scaling is used to enable the calculation to be carried out in real time.

5. **Averaging Algorithms.** In order to reduce the oscillation in pressure, the maximal target value used to adjust the pressure load is calculated by a special averaging algorithm. There are two options available:

   a) *Averaging over neighbors of element with maximum target value*: In this method, the element that has the maximum strain rate or other controlling variable is stored in each cycle of the computation. The elements close to the element with the maximum value are searched and stored in an array. The averaged maximal target value is computed over the neighboring elements. The user can input an integer number to control the level of neighbor search, which will affect the total number of elements for average. Because the neighbor search is time consuming, the user can input a time interval to limit the occurrence of searching. The neighbor search is conducted only when the simulation time reaches the specified time or the element with maximum target value falls out of the array of neighbors.

   b) *Averaging over elements within selective range of target value:* In this method, all elements that have target value above a threshold value (a threshold

percentage of maximum target value) are sorted according to their target value, and the elements between the user specified lower percentile and upper percentile are selected to compute the average of the maximal target value.

6. **Autojump.** The SPF simulation can be controlled by various autojump options. When autojump conditions are met, the SPF simulation will be either terminated or continued from phase I to phase II simulation. The autojump check can be held inactive by setting a strikethrough time. In this case the SPF simulation will continue for that period of time and only be interrupted when the percentage of nodes-in-contact reaches 100% for a specified time. The available autojump conditions are:

   a) *Peak pressure is reached and stays for certain time:* The peak pressure is determined by the maximum allowable scale factor and the load curve. The simulation will continue for a user specified time before termination.

   b) *User specified percentage of nodes-in-contact is reached:* The simulation will be terminated or continued to Phase II automatically if one of the following conditions is met:

      i) If the change of the percentage of nodes-in-contact is zero or negative for a specified time.

      ii) If the percentage of nodes-in-contact oscillates in a specified range for a specified time.

      iii) If the percentage of nodes-in-contact drops more than a specified value from the maximum value recorded.

      iv) If the percentage of nodes-in-contact reaches a user specified stop value.

7. **Computation Time.** In order to speed up the simulation of the superplastic forming process, we scale down the computation time. By doing this we increase the strain rate allowed in the SPF process, resulting in reduced simulation time. However, caution should be utilized with this speedup as it may affect the accuracy of the results. We recommend no or small strain rate speedup for simulations with complex geometry or tight angles.

8. **Critical Regions.** If the user knows the area(s) in the workpiece that are critical in the SPF process, he can use the box option to limit the region(s) where the elements are checked for computing the average of the maximal target value.

*LOAD_SURFACE_STRESS_{OPTION}

Available options include:

<BLANK>

SET

Purpose: This keyword modifies the behavior of shell elements causing them to pass pressure-type loads to material models 37 and 125 (see Remarks 1 and 2); shells usually omit such effects.

This keyword causes calculating segment pressures from contact and applied pressure loads on both the upper and lower surfaces of the shell and applying them as negative local z-stresses during the simulation. It is found in some cases this capability can improve the accuracy of metal forming simulations.

Card Sets. Include as many sets consisting of the following three cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID							
Type	I							

Card 2	1	2	3	4	5	6	7	8
Variable	LSCID1	LSCID2	LSCID3	LSCID4	LSCID5	LSCID6	LSCID7	LSCID8
Type	I	I	I	I	I	I	I	I

Card 3	1	2	3	4	5	6	7	8
Variable	USCID1	USCID2	USCID3	USCID4	USCID5	USCID6	USCID7	USCID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
PID/PSID	Part ID or, if option SET is active, part set ID.
LSCID*n*	Lower surface contact IDs. Up to eight IDs can be defined. These contacts definitions contribute to the pressure acting on the lower surface of the shell. If the pressure on the lower surface is due to applied pressure loads, specify a value -1 instead of a contact ID. Only one of the LSCID*n* may be set to -1. Lower surface of a part is on the opposite side of the shell element normals, which must be made consistent (in LS-PrePost).
USCID*n*	Upper surface contact IDs. Up to eight IDs can be defined. These contacts definitions contribute to the pressure acting on the upper surface of the shell. If the pressure on the upper surface is due to applied pressure loads, specify a value of -1 instead of a contact ID. Only one of the USCID*n* may be set to -1. Upper surface of a part is on the same side of the shell element normals, which must be made consistent (in LS-PrePost).

Remarks:

1. **MAT_TRANSVERSLY_ANISOTROPIC_ELASTIC_PLASTIC (37).** This keyword can be used with *MAT_037 when ETAN is set to a negative value (see *MAT_037 manual pages), which triggers normal stresses (local *z*-stresses) resulting either from sliding contact or applied pressure to be considered in the shell formulation. The normal stresses can be significant in male die radius and corners in forming of Advanced High Strength Steels (AHSS). The negative local *z*-stresses are written to the d3plot files and can be plotted in LS-PrePost by selecting *z*-stress under *FCOMP* → *Stress* and select *local* under *FCOMP*.

2. **MAT_KINEMATIC_HARDENING_TRANSVERSLY_ANISOTROPIC (125).** This keyword can also be used in a simulation with *MAT_125 to account for the normal stresses (see *MAT_125 manual pages). For this material inserting this keyword *anywhere* in the input deck will invoke the shell normal stress calculation.

LOAD_THERMAL_*OPTION

Available options include:

BINOUT

CONSTANT

CONSTANT_ELEMENT_*OPTION*

CONSTANT_NODE

D3PLOT

LOAD_CURVE

RSW

TOPAZ

VARIABLE

VARIABLE_ELEMENT_*OPTION*

VARIABLE_NODE

VARIABLE_SHELL_*OPTION*

Purpose: To define nodal temperatures that thermally load the structure. Nodal temperatures defined by the *LOAD_THERMAL_*OPTION* method are all applied in a structural only analysis. They are ignored in a thermal only or coupled thermal/structural analysis, see *CONTROL_THERMAL_*OPTION*.

All the *LOAD_THERMAL options cannot be used in conjunction with each other. Only those of the same thermal load type, as defined below in column 2, may be used together.

*LOAD_THERMAL_CONSTANT	- Thermal load type 1
*LOAD_THERMAL_ELEMENT	- Thermal load type 1
*LOAD_THERMAL_CONSTANT_NODE	- Thermal load type 1
*LOAD_THERMAL_LOAD_CURVE	- Thermal load type 2
*LOAD_THERMAL_TOPAZ	- Thermal load type 3
*LOAD_THERMAL_VARIABLE	- Thermal load type 4
*LOAD_THERMAL_VARIABLE_ELEMENT	- Thermal load type 4

*LOAD_THERMAL_VARIABLE_NODE - Thermal load type 4

*LOAD_THERMAL_VARIABLE_SHELL - Thermal load type 4

*LOAD_THERMAL_BINOUT

Purpose: Nodal temperatures computed in one or more prior simulations are used to load a mechanical-only analysis. Each of the prior simulations can either be a thermal-only or coupled thermal/structural analysis. The temperatures are read from the TPRINT section in the binary database binout or any other file containing the necessary information in LSDA-format. The data is mapped based on the nodal IDs.

Card 1	1	2	3	4	5	6	7	8
Variable	DEFTEMP							
Type	F							
Default	0.							

Card Sets. Include as many sets consisting of the following two cards as desired. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							

Card 3	1	2	3	4	5	6	7	8
Variable	STARTT	TSF	TMPOFF	NIDOFF				
Type	F	F	F	I				
Default	0.	1.	0.	0				

VARIABLE	DESCRIPTION
DEFTEMP	Default temperature applied to nodes that do not have temperature information provided in the binout file(s)
FILENAME	Name of the file that contains the temperature information

VARIABLE	DESCRIPTION
STARTT	Start time, t_{start}, for the temperature mapping. Until this time the nodal temperature for the first step provided in the file is used.
TSF	Time scale factor, α_t, which represents the speed-up factor of the mechanical analysis to the previous thermal analysis
TMPOFF	Temperature offset that is added to the temperature data in the specified LSDA input file.
NIDOFF	Offset to the user node IDs. The input value is added to the IDs as specified in the LSDA file before mapping temperature information.

Remarks:

1. **Overview of keyword.** This feature provides a method for using temperature data from a previous thermal-only or even as coupled thermal/structural analysis as an external thermal load for a mechanical-only simulation. For any node of the mechanical model, the algorithm searches for temperature data in the given files. This search is based on the user node ID. If no information is found, the default temperature is assumed. Therefore, the algorithm requires only a partial overlap of the meshes.

2. **Thermal and mechanical simulation time.** Each of the thermal results can be applied starting at different simulation times, t_{start}, of the mechanical run. Furthermore, the algorithm allows running an artificially accelerated mechanical analysis. It takes into account possibly different acceleration factors, α_t, for each of the thermal runs. Those factors are equivalent to the parameter TSF as given in the keyword *CONTROL_THERMAL_SOLVER. As a result, a time, t_{th}, in the thermal run corresponds to

$$t_{\text{mech}} = t_{\text{start}} + \frac{t_{\text{th}}}{\alpha_t} \ .$$

3. **Temperature information file.** With this keyword, binary LSDA files must be specified as input. The files should provide the same information as the TPRINT section in the binary database binout. More details for this output are defined in *DATABASE_*OPTION*.

*LOAD_THERMAL_CONSTANT

Purpose: Define the constant temperature that is applied to a given nodal set. The reference temperature state is assumed to be a null state with this option. A nodal temperature state, read in and held constant throughout the analysis, dynamically loads the structure. Thus, the temperature defined can also be seen as a relative temperature to a surrounding or initial temperature.

Card Sets. Include as many sets consisting of the following two cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	NSIDEX	BOXID					
Type	I	I	I					
Default	{all}	0	0					

Card 2	1	2	3	4	5	6	7	8
Variable	T	TE						
Type	F	F						
Default	0.	0.						

VARIABLE	DESCRIPTION
NSID	Nodal set ID containing nodes for initial temperature EQ.0: all nodes are included
NSIDEX	Nodal set ID containing nodes that are exempted from the imposed temperature (optional).
BOXID	All nodes in box which belong to NSID are initialized. Others are excluded (optional).
T	Temperature

VARIABLE	DESCRIPTION
TE	Temperature of exempted nodes (optional)

LOAD_THERMAL_CONSTANT_ELEMENT_*OPTION

Available options include:

BEAM

SHELL

SOLID

TSHELL

Purpose: Define a uniform element temperature that remains constant for the duration of the calculation. The reference temperature state is assumed to be a null state. An element temperature, read in and held constant throughout the analysis, dynamically loads the structure. The defined temperature can also be seen as a relative temperature to a surrounding or initial temperature.

Element Cards. Include as many cards in this format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	EID	T						
Type	I	F						
Default	none	0.						

VARIABLE	DESCRIPTION
EID	Element ID
T	Temperature, see Remark 1.

Remarks:

1. **Reference Temperature.** The temperature range for the constitutive constants in the thermal materials must include the reference temperature of zero. If not, termination will occur with a temperature out-of-range error immediately after the execution phase is entered.

*LOAD_THERMAL_CONSTANT_NODE

Purpose: Define nodal temperature that remains constant for the duration of the calculation. The reference temperature state is assumed to be a null state with this option. A nodal temperature state, read in and held constant throughout the analysis, dynamically loads the structure. Thus, the temperature defined can also be seen as a relative temperature to a surrounding or initial temperature.

Node Cards. Include as many cards in this format as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	NID	T						
Type	I	F						
Default	none	0.						

VARIABLE	DESCRIPTION
NID	Node ID
T	Temperature, see Remark 1.

Remarks:

1. **Reference Temperature.** The temperature range for the constitutive constants in the thermal materials must include the reference temperature of zero. If not, termination will occur with a temperature out-of-range error immediately after the execution phase is entered.

*LOAD_THERMAL_D3PLOT

Purpose: Temperatures computed in a prior thermal-only analysis are used to load a mechanical-only analysis. The rootname of the d3plot database from the thermal-only analysis is specified on the execution line of the mechanical-only analysis using T = tpf, where tpf is that rootname, such as T = d3plot.

Warnings:

1. If using a double precision LS-DYNA executable in making the two runs, do not write the d3plot data using 32ieee format in the thermal-only run, that is, the environment variable LSTC_BINARY must not be set.

2. The rootnames of the d3plot databases from the two runs must not conflict. Such conflict can be avoided, for example, by using jobid = *jobname* on the execution line of the second (mechanical-only) run.

*LOAD_THERMAL_LOAD_CURVE

Purpose: Nodal temperatures will be uniform throughout the model and will vary according to a load curve. The temperature at time = 0 becomes the reference temperature for the thermal material. The reference temperature is obtained from the optional curve for dynamic relaxation if this curve is used. The load curve option for dynamic relaxation is useful for initializing preloads.

Thermal Load Curve Cards. Include as many cards in this format as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	LCID	LCIDDR						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
LCID	Load curve ID, see *DEFINE_CURVE, to define temperature as a function of time.
LCIDDR	An optional load curve ID, see *DEFINE_CURVE, to define temperature as a function of time during the dynamic relaxation phase.

*LOAD_THERMAL_RSW

Purpose: Define thermal loading conditions within an ellipsoidal region of the solid or shell structure for structural-only simulation. Temperatures are prescribed to nodes found in a region defined by this keyword. The load condition is tailored to represent the so-called weld nuggets evolving during resistive spot welding (RSW) processes and surrounding heat affected zones. For any node that is not part of a spot weld region, a default temperature is assumed.

Card Summary:

Card 1. This card is required.

DEFTEMP							

Card 2. Include as many sets of Cards 2 through 4 as needed (Card 4 is only included in the set if OPTION = 1). This input ends at the next keyword ("*") card.

SID	OPTION	NID1	NID2	TDEATH	TBIRTH	LOC	GEOUP

Card 3. Include in set with Card 2.

DIST	H1	H2	R	TEMPC	TEMPB	LCIDT	

Card 4. Include in set with Cards 2 and 3 if OPTION = 1 on Card 2.

HZ1	HZ2	RZ	TEMPZB				

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	DEFTEMP							
Type	F							
Default	0.							

VARIABLE	DESCRIPTION
DEFTEMP	Default temperature outside the nuggets and heat affected zones

Include as many sets of Cards 2 through 4 as needed (Card 4 is only included in the set if OPTION = 1). This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SID	OPTION	NID1	NID2	TDEATH	TBIRTH	LOC	GEOUP
Type	I	I	I	I	F	F	I	I
Default	none	0	none	none	10^{20}	0.	0	0

VARIABLE	DESCRIPTION
SID	Node set ID; see *SET_NODE_*OPTION*. Nodes in the set will be checked to see if they are in the nugget or heat affected zone. If they are, the load will be applied. The load will not be applied to nodes in these regions if they are not included in the set.
OPTION	Option for heat affected zone around the weld nugget: EQ.0: No heat affected zone EQ.1: Ellipsoidal region considered
NID1	Node defining the tail of the orientation vector (axis of rotation of the ellipsoidal region) and the base for positioning of the nugget. See Remarks 1 and 2.
NID2	Node defining the head of the orientation vector (axis of rotation of the ellipsoidal region). See Remarks 1 and 2.
TDEATH	Deactivation time for temperature boundary condition. At this point in time, the temperature constraint is removed.
TBIRTH	Activation time for temperature boundary condition. Before this point in time, the temperature constraint is ignored.
LOC	Application of surface for thermal shell elements; see parameter, THSHEL, in the *CONTROL_SHELL input: EQ.-1: Lower surface of thermal shell element EQ.0: Middle surface of thermal shell element EQ.1: Upper surface of thermal shell element

VARIABLE	DESCRIPTION
GEOUP	Number of times the geometry of the spot weld is updated between TBIRTH and TDEATH EQ.0: Update geometry every time step

Geometry and Temperature Card. Include in set with Card 2.

Card 3	1	2	3	4	5	6	7	8
Variable	DIST	H1	H2	R	TEMPC	TEMPB	LCIDT	
Type	F	F	F	F	F	F	I	
Default	0.	0.	0.	0.	0.	0.	none	

VARIABLE	DESCRIPTION
DIST	Position of center of nugget on the axis of rotation. Parameter defines the distance to NID1 along the orientation vector. See Remark 1.
H1	Half width h_1 of nugget in the lower half, meaning in direction to NID1. See Remark 2.
H2	Half width h_2 of nugget in the upper half, meaning in direction to NID2. See Remark 2.
R	Radius r_{weld} of the nugget in surface normal to orientation vector. See Remark 2.
TEMPC	Base temperature at the center of the nugget. See Remark 3.
TEMPB	Base temperature at the boundary of the nugget. See Remark 3.
LCIDT	|LCIDT| refers to the load curve ID prescribing the temperature evolution in the nugget as a function of time. The abscissa of the load curve will be normalized between the birth and death times of the boundary condition. GT.0: The ordinate values of the load curve scale the respective base temperature of a particular point. EQ.0: No temperature evolution. Base temperatures are used.

VARIABLE	DESCRIPTION
	LT.0: The ordinate values of the load curve are used to define a linear combination between the temperature at the birth time and the base temperature of a particular point. Load curve ordinate values should range between 0.0 and 1.0. We recommend LCIDT < 0 to ensure a smooth temperature evolution.
	See Remark 3.

Heat Affected Zone Card. Include in set with Cards 2 and 3 if OPTION = 1 on Card 2.

Card 4	1	2	3	4	5	6	7	8
Variable	HZ1	HZ2	RZ	TEMPZB				
Type	R	F	F	R				
Default	0.	0.	0.	0.				

VARIABLE	DESCRIPTION
HZ1	Half width h_{z1} of heat affected zone in the lower half, meaning in direction to NID1. See Remark 4.
HZ2	Half width h_{z2} of heat affected zone in the upper half, meaning in direction to NID1. See Remark 4.
RZ	Radius r_{haz} of the heat affected zone in surface normal to orientation vector. See Remark 4.
TEMPBZ	Base temperature at the boundary of the heat affected zone. See Remark 4.

Remarks:

1. **Positioning.** The position of the center of the nugget is defined starting from a base node (NID1). It is then translated by a distance DIST along the orientation vector **v** of the nugget. Vector **v** points from NID1 to NID2. See Figure 33-17.

2. **Geometry of the Nugget.** Orientation vector **v** defines the axis of rotational symmetry for the ellipsoidal region, which reflects the weld nugget. The radius around this axis is defined by r_{weld}. The nugget consists of two half ellipsoids

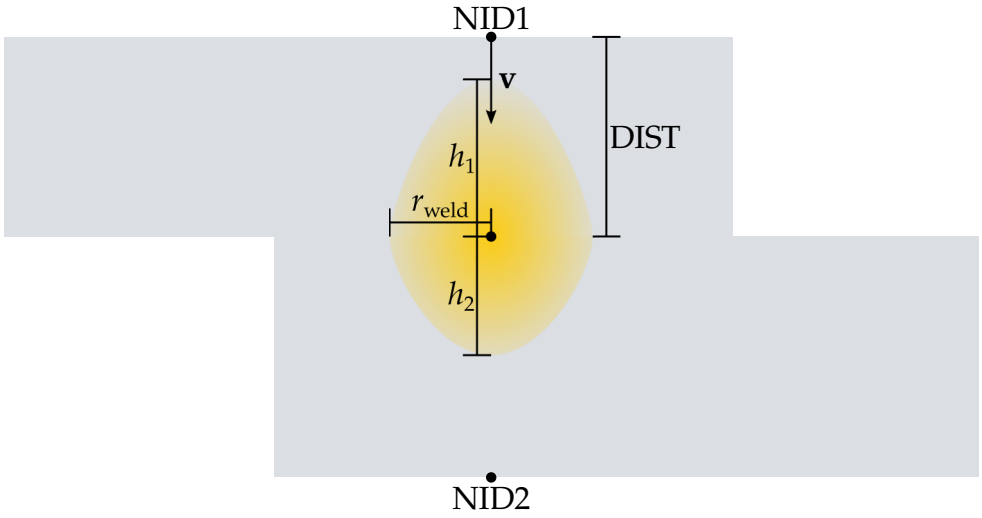


Figure 33-17. Example of geometry for OPTION = 0

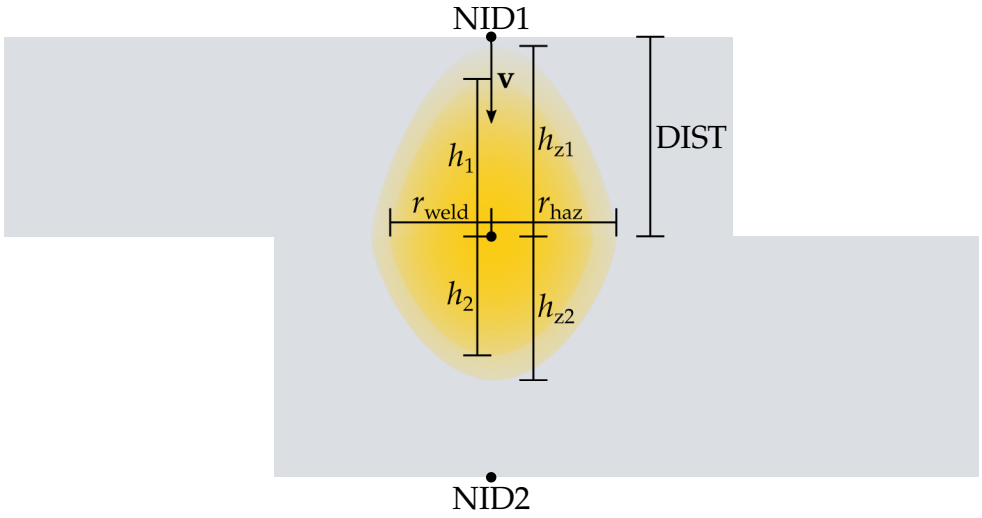


Figure 33-18. Example of geometry with OPTION = 1 (heat affected zone)

of possibly different height. The lower half, i.e. the half between NID1 and the center, has a height of h_1, wheras the upper half has a height of h_2. See Figure 33-17.

3. **Prescribing Temperature Values.** You can prescribe the base temperatures at the boundary and the center of the nugget with TEMPC and TEMPB, respectively. In between LS-DYNA calculates the base temperature in the nugget, $T_{\mathrm{nug},b}(x,y,z)$, with a quadratic interpolation.

   For this loading condition, you must provide activation and deactivation times, t_{ac} and t_{deac}, respectively. With LCIDT, you can specify the temperatures as a function, $\theta(\tilde{t})$, of the normalized time, $\tilde{t} = (t - t_{\mathrm{ac}})/(t_{\mathrm{deac}} - t_{\mathrm{ac}})$, within this time span. For positive values of field LCIDT, a simple scaling is applied:

$$T_{\mathrm{nug}}(x,y,z,t) = T_{\mathrm{nug},b}(x,y,z) \times \theta(\tilde{t}(t)) \quad .$$

For negative values of LCIDT, a smooth evolution of the temperature from the birth time is ensured, so we recommend this type of load curve. The temperature is given by:

$$T_{\text{nug}}(x,y,z,t) = T_{\text{ac}}(x,y,z) + \Big(T_{\text{nug},b}(x,y,z) - T_{\text{ac}}(x,y,z)\Big) \times \theta(\tilde{t}) \ .$$

Here T_{ac} corresponds to the temperature of a particular point right before the birth time.

4. **Heat affected Zone.** With input field OPTION set to 1, the prescribed load expands to a heat affected zone that has the same general shape definition as the weld nugget described in Remark 2. The dimensions are here given by r_{haz}, h_{z1}, and h_{z2}. See Figure 33-18.

   You can prescribe the base temperature value at the boundary of the heat affected zone. LS-DYNA finds the base temperature in the heat affected zone, $T_{\text{HAZ},b}(x,y,z)$ with a linear interpolation between the base boundary temperature of the nugget and the base boundary temperature of the heat affected zone. The load curve LCIDT governs the temperature evolution of the heat affected zone as described in Remark 3.

*LOAD_THERMAL_TOPAZ

Purpose: Nodal temperatures will be read in from the TOPAZ3D database. This file is defined on the execution line by the specification: T=**tpf**, where **tpf** is a binary database file (such as T3PLOT).

*LOAD_THERMAL_VARIABLE

Purpose: Define nodal temperature using node set(s) and temperature as a function of time curve(s).

Card Sets. Include as many sets consisting of the following two cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	NSIDEX	BOXID					
Type	I	I	I					
Default	{all}	{∅}	0					

Card 2	1	2	3	4	5	6	7	8
Variable	TS	TB	LCID	TSE	TBE	LCIDE	LCIDR	LCIDEDR
Type	F	F	I	F	F	I	I	I
Default	0.	0.	none	0.	0.	↓	none	↓
Remark	1	1	1	1	1			

VARIABLE	DESCRIPTION
NSID	Nodal set ID containing nodes (see *SET_NODE_*OPTION*): EQ.0: all nodes are included.
NSIDEX	Nodal set ID containing nodes that are exempted (optional), (see *SET_NODE_OPTION).
BOXID	All nodes in box which belong to NSID are initialized. Others are excluded.
TS	Scaled temperature.
TB	Base temperature.

VARIABLE	DESCRIPTION
LCID	Load curve ID (see *DEFINE_CURVE) or function ID (see *DEFINE_FUNCTION and Remark 2) that multiplies the scaled temperature.
TSE	Scaled temperature of the exempted nodes (optional).
TBE	Base temperature of the exempted nodes (optional).
LCIDE	Load curve ID that multiplies the scaled temperature of the exempted nodes (optional), (see *DEFINE_CURVE).
LCIDR	Load curve ID that multiplies the scaled temperature during the dynamic relaxation phase
LCIDEDR	Load curve ID that multiplies the scaled temperature of the exempted nodes (optional) during the dynamic relaxation phase.

Remarks:

1. **Temperature Model.** The total temperature is defined as

$$T = \mathrm{TB} + \mathrm{TS} \times f(t) \ ,$$

where $f(t)$ is the current value of the load curve, TS is the scaled temperature, and TB is the base temperature. The initial temperature and thus the reference temperature for a thermal loading is consistently defined by $T_0 = \mathrm{TB} + \mathrm{TS} \times f(0)$.

2. ***DEFINE_FUNCTION for LCID.** If LCID references a function ID, then the following function arguments are allowed: $f(x0, y0, z0, x, y, z, t)$, where $x0$, $y0$ and $z0$ are the original coordinates; x, y and z are the current coordinates; and t is the solution time

3. **Thermal Strain.** The rate of thermal strain is based on the rate of temperature change. In other words, the thermal load arises from change in total temperature. Furthermore, the calculation of the thermal strain from the coefficient of thermal expansion depends on the material model, and some material models, e.g., *MAT_255, may offer multiple options. Temperature-dependent material properties are based on the total temperature.

*LOAD_THERMAL_VARIABLE_BEAM_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Define a known temperature time history as a function of the section coordinates for Hughes-Liu beam elements. To set the temperature for the whole element see *LOAD_THERMAL_VARIABLE_ELEMENT_BEAM.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	EID/SID	IPOLAR					
Type	I	I	I					
Default	none	none	0					

Temperature Cards. Include as many cards in the following format as desired. See Remarks 2 and 3. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	TBASE	TSCALE	TCURVE	TCURDR	SCOOR	TCOOR		
Type	F	F	I	I	F	F		
Default	0.	1.0	0	TCURVE	none	none		

VARIABLE	DESCRIPTION
ID	Load case ID
EID/SID	Beam ID or beam set ID
IPOLAR	Flag to use polar coordinates instead of rectangular coordinates. GT.0: The coordinates SCOOR and TCOOR are given in polar coordinates. See Remark 3.
TBASE	Base temperature

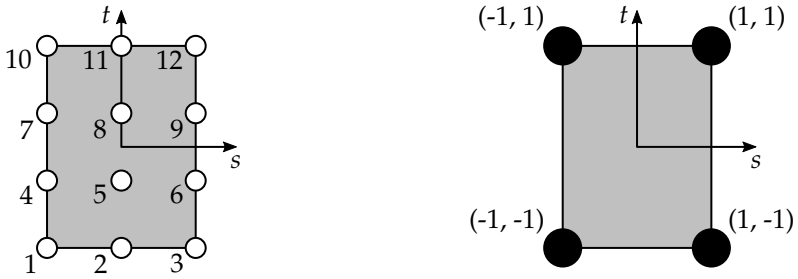


Figure 0-1. Figure illustrating point ordering.

VARIABLE	DESCRIPTION
TSCALE	Constant scale factor applied to temperature from curve
TCURVE	Curve ID for temperature as a function of time
TCURDR	Curve ID for temperature as a function of time used during dynamic relaxation
SCOOR	Normalized coordinate in local s-direction (-1.0 to +1.0)
TCOOR	Normalized coordinate in local t-direction (-1.0 to +1.0)

Remarks:

1. **Temperature Curve.** The temperature T is defined as:

$$T = TBASE + TSCALE \times f(t)$$

where $f(t)$ is the current ordinate value of the curve. If the curve is undefined, then T = TBASE at all times.

2. **Cross Section Coordinates.** At least four points (four Card 2's) must be defined in a rectangular grid since each edge can have a different temperature history. The required order of the points is as shown in Figure 0-1. First, define the bottom row of points (most negative t), left to right in order of increasing s. Then increment t to define the next row of points, left-to-right in order of increasing s, and so on. The s-t axes are in the plane of the beam cross-section with the s-axis in the plane of nodes N1, N2, N3 defined in *ELEMENT_BEAM.

3. **Polar Cross Section Coordinates.** For this option only two points (two Card 2's) must be defined for the center and edge of the section because the temperature changes are assumed to only be in the radial direction. For the polar option, SCOOR is the non-dimensional radius R/R_0 where R_0 is the outer radius of the section; and TCOOR is defined as θ/π, where θ is the angle in radians from the s-axis and ranges from $-\pi$ to π.

4. **Integration Point Temperatures.** Temperatures will be assigned to the integration points by linear interpolation from the points defined using this command.

LOAD_THERMAL_VARIABLE_ELEMENT_*OPTION

Available options include:

BEAM

SHELL

SOLID

TSHELL

Purpose: Define element temperature that is variable during the calculation. The reference temperature state is assumed to be the temperature at time = 0.0 with this option.

Element Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	EID	TS	TB	LCID				
Type	I	F	F	I				
Default	none	0.	0.	none				

VARIABLE	DESCRIPTION
EID	Element ID
TS	Scaled temperature
TB	Base temperature
LCID	Load curve ID defining a scale factor that multiplies the scaled temperature as a function of time (see *DEFINE_CURVE).

Remarks:

1. The temperature is defined as:

$$T = \mathrm{TB} + \mathrm{TS} \times f(t)$$

where $f(t)$ is the current value of the load curve, TS is the scaled temperature, and TB is the base temperature

*LOAD_THERMAL_VARIABLE_NODE

Purpose: Define nodal temperature that is variable during the calculation. A nodal temperature state read in and varied according to the load curve dynamically loads the structure.

Node Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	NID	TS	TB	LCID				
Type	I	F	F	I				
Default	none	0.	0.	none				

VARIABLE	DESCRIPTION
NID	Node ID
TS	Scaled temperature
TB	Base temperature
LCID	Load curve ID that multiplies the scaled temperature, (see *DEFINE_CURVE).

Remarks:

1. The temperature is defined as:

$$T = \mathrm{TB} + \mathrm{TS} \times f(t)$$

where $f(t)$ is the current value of the load curve, TS is the scaled temperature, and TB is the base temperature. The initial temperature and thus the reference temperature for a thermal loading is consistently defined by $T_0 = \mathrm{TB} + \mathrm{TS} \times f(0)$.

*LOAD_THERMAL_VARIABLE_SHELL_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Define a known temperature time history as a function of the through-thickness coordinate for the shell elements.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	EID/SID						
Type	I	I						
Default	none	none						

Temperature Cards. Include as many cards of this type as desired. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	TBASE	TSCALE	TCURVE	TCURDR	ZC0			
Type	F	F	I	I	F			
Default	0.0	1.0	Rem 1	TCURVE	none			

VARIABLE	DESCRIPTION
ID	Load case ID
EID/SID	Shell ID or shell set ID
TBASE	Base temperature
TSCALE	Constant scale factor applied to temperature from curve
TCURVE	Curve ID for temperature as a function of time

VARIABLE	DESCRIPTION
TCURDR	Curve ID for temperature as a function of time used during dynamic relaxation
ZCO	Normalized through-thickness coordinate (-1.0 to +1.0). See Remarks 2 and 3.

Remarks:

1. **Temperature Definition.** The temperature T is defined as:

$$T = \text{TBASE} + \text{TSCALE} \times f(t) ,$$

where $f(t)$ is the current ordinate value of the curve. If the curve is undefined, then T = TBASE at all times.

2. **Through-Thickness Coordinate.** Through-thickness points must be defined in order of increasing ZCO (-1.0 to +1.0). ZCO = +1.0 is the top surface of the element, that is, the element surface in the positive outward normal vector direction from the mid-plane.

3. **Shell Temperature Distribution.** At least two points (two Card 2's) must be defined. Temperatures will be assigned in the through-thickness direction by linear interpolation from the points defined using this command. If the element has multiple in-plane integration points, the same temperature distribution is used at each in-plane integration point.

4. **LOAD_THERMAL_NODE.** If a shell's temperature distribution is defined using this card, any values defined by *LOAD_THERMAL_NODE are ignored for that shell.

*LOAD_VOLUME_LOSS

Purpose: To represent the effect of tunneling on surrounding structures, it is common to assume that a pre-defined fraction (e.g., 2%) of the volume occupied by the tunnel is lost during the construction process. This feature is available for solid elements only.

Part Set Cards. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card	1	2	3	4	5	6	7	8
Variable	PSID	COORD	LCUR	FX	FY	FZ	PMIN	FACTOR
Type	I	I	I	F	F	F	F	F
Default	none	global	0	1	1	1	-10^{20}	.01

VARIABLE	DESCRIPTION
PSID	Part Set ID
COORD	Coordinate System ID (default - global coordinate system)
LCUR	Curve ID containing volume fraction lost as a function of time
FX	Fraction of strain occurring in *x*-direction
FY	Fraction of strain occurring in *y*-direction
FZ	Fraction of strain occurring in *z*-direction
PMIN	(Leave blank)
FACTOR	Feedback factor

Remarks:

1. **Volume Loss Algorithm.** Volume loss is modeled by a process similar to thermal contraction: if the material is unrestrained, it will shrink while remaining unstressed; if restrained, stresses will become more tensile. Typically, the material surrounding the tunnel offers partial restraint; the volume loss algorithm adjusts the applied "thermal" strains to attempt to achieve the desired volume loss. Optionally, FX, FY and FZ may be defined: these will be treated as ratios

for the *x*, *y* and *z* strains; this feature can be used to prevent contraction parallel to the tunnel axis.

2. **Output.** The total volume of all the parts in the part set is monitored and output at the time-history interval (on *DATABASE_BINARY_D3THDT) to a file named vloss_output. This file contains lines of data (*time, volume1, volume2, volume3…*) where *volume1* is the total volume of elements controlled by the first *LOAD_VOLUME_LOSS card, *volume2* is the total volume of elements controlled by the second *LOAD_VOLUME_LOSS card, etc.

3. **Material Models.** This feature works only with material types that use incremental strains to compute stresses. Thus, hyperelastic materials (e.g. MAT_027) are excluded, as are certain foam material types (e.g. MAT_083).

4. **Feedback Factor.** The feedback factor (see FACTOR field) controls how strongly the algorithm tries to impose the desired change of volumetric strain. The default value is recommended. If the volumetric response appears noisy or unstable, it may be necessary to reduce FACTOR. Alternatively, if the actual volumetric strain changes much more slowly than the input curve, it may be necessary to increase FACTOR.

*MODULE

The keyword *MODULE provides a way to load user compiled libraries at runtime, to support user defined capabilities such as material models, equations of state, element formulations, etc. The following keywords implement this capability:

*MODULE_LOAD

*MODULE_PATH

*MODULE_USE

***MODULE_LOAD**_*{OPTION}*

Available options:

<BLANK>

SP

DP

Purpose: Load a dynamic library for user subroutines. When using the SP / DP keyword option, LS-DYNA only loads the dynamic library when running a single / double precision binary.

Card Sets. Repeat as many sets of data cards as desired (Cards 1 and 2) to load multiple libraries. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	MDLID		TITLE					
Type	A20		A60					
Default	none		none					

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	A80							
Default	none							

VARIABLE	DESCRIPTION
MDLID	Module identification. A unique string label must be specified.
TITLE	Description of the module
FILENAME	File name of the library to be loaded, 80 characters maximum. If the file name has no path component, LS-DYNA will search in all directories specified in *MODULE_PATH first. If not found and

VARIABLE	DESCRIPTION
	the file name starts with "+" (a plus sign), LS-DYNA will continue to search all directories specified in the system environment variable LD_LIBRARY_PATH.

Remarks:

1. **Library disclaimer.** The MODULE capability described here and in *MODULE_USE is still under development. We consider this capability experimental.

2. **Execution line option.** If loading only one dynamic library that requires no rules (*MODULE_USE), you can use the execution line option "module=*dll*" to specify this dynamic library. If the file name starts with +, LS-DYNA searches for the dynamic library using the system environment variable LD_LIBRARY_PATH. We implemented this execution line option to support the classic user-subroutine subroutines without modifying the input deck.

***MODULE_PATH**_{OPTION}

Available options:

<BLANK>

RELATIVE

LS-DYNA's default behavior is to search for modules in the current working directory. If a module file is not found and the filename has no path component, LS-DYNA will search in all directories specified on the cards following a *MODULE_PATH keyword. Multiple paths can be specified using one *MODULE_PATH keyword card, that is,

```
*MODULE_PATH
Directory_path1
Directory_path2
Directory_path3
```

Directory paths are read until the next keyword ("*") card is encountered. A directory path can have up to 236 characters. See Remark 2.

When the RELATIVE option is used, all directories are relative to the location of the input file. For example, if "i=/home/test/problems/input.k" is given on the command line, and the input contains

```
*MODULE_PATH_RELATIVE
lib
../lib
```

then the two directories /home/test/problems/lib and /home/test/lib will be searched for module files.

Remarks:

1. **Input Format.** Filenames and pathnames are limited to 236 characters spread over up to three 80 character lines. When 2 or 3 lines are needed to specify the filename or pathname, end the preceding line with "␣+" (space followed by a plus sign) to signal that a continuation line follows. Note that the "␣+" combination is, itself, part of the 80 character line; hence the maximum number of allowed characters is 78 + 78 + 80 = 236.

*MODULE_USE

Purpose: Define the rules for mapping the user subroutines loaded in dynamic libraries to the model. The rules can be applied to:

*MAT_USER_DEFINED_MATERIAL_MODELS	(MAT 41 - 50)
*MAT_THERMAL_USER_DEFINED	(MAT T11 - T15)
*EOS_USER_DEFINED	(EOS 21 – 30)
*SECTION_SOLID	(ELFORM 101 - 105)
*SECTION_SHELL	(ELFORM 101- 105)
*USER_INTERFACE_FRICTION	
*CONTACT_..._MORTAR_TIED_WELD	(TEMP.LT.0)

as well as other subroutines in the LS-DYNA user-subroutine package.

LS-DYNA requires that subroutines in modules (see *MODULE_LOAD) be named as if they were part of the traditional user-subroutine framework. Each module, however, can contain a complete set of those subroutines. It is, therefore, possible to import from different modules the same subroutines of the same name several times. Each module is, essentially, an independent copy of the traditional user-subroutine framework. This keyword, *MODULE_USE, deals with namespace conflicts by defining how each subroutine in the module is presented to the other keywords.

The rules defined in *MODULE_USE are applied to only one dynamic library.

Card 1	1	2	3	4	5	6	7	8
Variable	MDLID							
Type	A20							
Default	none							

Rule Cards. Card 2 defines the rules for the module specified in Card 1. Include one instance of this card for each subroutine to be mapped. If two rules conflict, new rules override existing rules. Input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	TYPE		PARAM1		PARAM2			
Type	A20		A20		A20			
Default	none		blank		blank			

VARIABLE	DESCRIPTION
MDLID	Module identification defined in *MODULE_LOAD
TYPE	Rule type. See TYPE definitions below.
PARAM1	Type-dependent parameter
PARAM2	Type-dependent parameter.

User-Defined Materials:

TYPE	DESCRIPTION
UMAT	Implements the user-material type PARAM1 in the model via a possibly different type PARAM2 in the dynamic library. Types in the extended range from 1001 to 2000 can be used as material types in the model. For example, if PARAM1 = 1001 and PARAM2 = 42, then model material 1001 will use subroutine `umat42` from this library. If PARAM1 is blank and PARAM2 is not, then all user materials in the model will use the indicated subroutine from this library. If PARAM2 is blank and PARAM1 is not, the material type PARAM1 in the model will use the traditional subroutine from this library. For example, if PARAM1 = 43 and PARAM2 is blank, material type 43 in the model will use the subroutine `umat43` from this library.

TYPE	DESCRIPTION
	If PARAM1 and PARAM2 are blank, then all user-defined materials in the model will use the traditional subroutines from this library (41 → `umat41`, 42 → `umat42`, etc.).
MATID	Implements the user material having ID = PARAM1 via a possibly different material type PARAM2 in the dynamic library. PARAM2 may be blank if the material type defined in the model is the same as the one in the dynamic library. PARAM1 can be a numerical ID or label of the material as defined in the model. Materials beyond user material models can be overloaded.

Thermal User-Defined Materials:

TYPE	DESCRIPTION
TUMAT	Implements the user thermal material type PARAM1 in the model (PARAM1 = 11-15) via type PARAM2 in the dynamic library. See type UMAT for the default rules.
TMATID	Implements the user thermal material having ID = PARAM1 via thermal material type PARAM2 in the dynamic library. PARAM2 may be blank if the material type defined in the model is the same as the one in the dynamic library. PARAM1 can be a numerical ID or label of the thermal material as defined in the model. Materials beyond user material models can be overloaded.

User-Defined EOS:

TYPE	DESCRIPTION
UEOS	Implements the user EOS model PARAM1 in the model (PARAM1 = 21-30) via EOS model PARAM2 in the dynamic library. See type UMAT for the default rules.

User-Defined Elements:

TYPE	DESCRIPTION
UELEM	Implements the user element type PARAM1 in the model (PARAM1 = 101-105) via element type PARAM2 in the dynamic library. See type UMAT for the default rules.
SECTIONID	Implements the user element type with section ID = PARAM1 via element type PARAM2 in the dynamic library. PARAM2 may be blank if the element type defined in the model is the same as the one in the dynamic library.

> **Note**: Solid element types are mapped to subroutine `usrsld`, and shell element types are mapped to subroutine `usrshl`. These interfaces are documented in the Fortran user-subroutine files.

Other User Subroutines:

If TYPE = UTIL, the model uses the following subroutines implemented in the dynamic library. LS-DYNA ignores any subroutines with the same name in other dynamic libraries. PARAM1 must be unique in the model, meaning two libraries cannot use the same PARAM1.

PARAM1	DESCRIPTION
UMATFAIL	matusr_24 matusr_103
UFRICTION	usrfrc mortar_usrfrc mortar_usrtie
UWEAR	userwear
UADAP	useradap uadpnorm uadpval
UWELDFAIL	uweldfail uweldfail12 uweldfail22
USPH	hdot

PARAM1	DESCRIPTION
UTHERMAL	usrhcon usrflux
ULOAD	ujntfdrv loadsetud loadud
UELEMFAILCTL	matfailusercontrol matuserfail
UMATFPERT	usermatfpert
UREBAR	rebar_bondslip_get_nhisvar rebar_bondslip_get_force
ULAGPOROUS	lagpor_getab_userdef
UAIRBAG	airusr user_inflator
UALE	al2rfn_criteria5 al2rmv_criteria5 alerfn_criteria5 alermv_criteria5 shlrfn_criteria5 shlrmv_criteria5 sldrfn_criteria5 sldrmv_criteria5 f3dm9ale_userdef1
UCOU-PLE2OTHER	couple2other_boxminmax couple2other_comm couple2other_dt couple2other_extra couple2other_getf couple2other_givex couple2other_reader chkusercomm usercomm usercomm1

PARAM1	DESCRIPTION
UOUTPUT	ushout
UEM	*EM user subroutines
UICFD	*ICFD user subroutines
UCESE	*CESE user subroutines
UDUALCESE	*DUALCESE user subroutines
UCTRL	uctrl1 uctrl2
UTIME	utimestep

Remarks:

1. **Omitting *MODULE_USE.** To simplify the development of user-defined materials via modules, the *MODULE_USE keyword can be omitted in one case. If only a single module library will be used and no remapping of material types is desired, then a *MODULE_LOAD keyword may appear with a single library and no other *MODULE keywords. In this case, all the user-defined subroutines in this library will be used in the usual way. For example, if the library contains `umat41` and `umat43`, those routines will be used for all materials of type 41 and 43, respectively. The *MODULE_USE keyword describing this library may be omitted.

EXAMPLES:

The following examples demonstrate the input for the MODULE keywords:

*MODULE_PATH

*MODULE_LOAD

*MODULE_USE

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Using *MODULE_PATH to define additional directories where
$  dynamic libraries are saved.
$
$-------------------------------------------------------------------------------
$
*MODULE_PATH
/home/lsdynauser/lib
$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Using *MODULE_LOAD to load all dynamic libraries for this model.
$
$  three libraries are loaded here for demonstration:
$
$  M_LIB: my library, which is under development.  A debug version is built
$         in my local directory.
$         it contains: UMAT41, UMAT42, and UMAT45
$
$  LIB_A: a hypoelastic model, provided by a third party, for shell & solid.
$          it contains UMAT41 only, with an optional FLUID
$
$  LIB_B: contains two material models provided by another company, with
$         UMAT41: a elasto-plastic model
$         UMAT45: a hyper-elastic model for rubber
$
$---------------------------------------------------------------------------
*MODULE_LOAD
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
M_LIB               My library
/my_development_path/mylib_r123_dbg.so
LIB_A               library from company A
Lib_hypoelastic.so
LIB_B               library from company B
Lib_plastic.so
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Using *MODULE_USE to map to user subroutines
$
$  CASE 1:
$
$  M_LIB is used for UMAT41,UMAT42,UMAT45 in the model, as default
$  UMAT41 in LIB_A is used for MT=1001 in the model for shell, and MT=1002 for solid
$  UMAT45 in LIB_B is used for MATID=202, which also happens to hvae MT=1002
$
$---------------------------------------------------------------------------
*MODULE_USE
M_LIB
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
UMAT

*MODULE_USE
LIB_A
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
UMAT                1001                41
UMAT                1002                41
*MODULE_USE
LIB_B
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
MATID               202                 45
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$  CASE 2:
$
$  M_LIB is used for UMAT41,UMAT42 as,
$         MATID=101 with MT=41
$         MATID=102 with MT=42
$  UMAT45 in LIB_B is used with different material properties, as,
$         MATID=201 with MT=1001
$         MATID=202 with MT=1002
$         MATID=203 with MT=1003
$
*MODULE_USE
LIB_B
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
```

```
UMAT                                    45
*MODULE_USE
M_LIB
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
MATID                101
MATID                102
```

*NODE

The keywords defined in this section include:

NODE_*{OPTION}

*NODE_MERGE_SET

*NODE_MERGE_TOLERANCE

*NODE_REFERENCE

*NODE_RIGID_SURFACE

NODE_SCALAR_*{OPTION}

*NODE_THICKNESS

*NODE_TO_TARGET_VECTOR

*NODE_TRANSFORM

***NODE**_*{OPTION}*

Available options include:

<BLANK>

MERGE

Purpose: Define a node and its coordinates in the global coordinate system. Also, the boundary conditions in global directions can be specified. Generally, nodes are assigned to elements; however, exceptions are possible (see Remark 2 below). The nodal point ID must be unique relative to other nodes defined in the *NODE section.

The MERGE option is usually applied to boundary nodes on disjoint parts and only applies to nodes defined when the merge option is invoked. With this option, nodes with identical coordinates are replaced during the input phase by the first node encountered that shares the coordinate. During the merging process a tolerance is used to determine whether a node should be merged. This tolerance can be defined using the keyword *NODE_MERGE_TOLERANCE keyword, which is recommended over the default value. See the *NODE_MERGE_TOLERANCE input description in the next section.

Node Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z		TC	RC	
Type	I	F		F		F		I	I	
Default	none	0.		0.		0.		0	0	
Remarks								1	1	

VARIABLE	DESCRIPTION
NID	Node number
X	*x* coordinate
Y	*y* coordinate
Z	*z* coordinate

VARIABLE	DESCRIPTION
TC	Translational Constraint: EQ.0: no constraints, EQ.1: constrained x displacement, EQ.2: constrained y displacement, EQ.3: constrained z displacement, EQ.4: constrained x and y displacements, EQ.5: constrained y and z displacements, EQ.6: constrained z and x displacements, EQ.7: constrained x, y, and z displacements.
RC	Rotational constraint: EQ.0: no constraints, EQ.1: constrained x rotation, EQ.2: constrained y rotation, EQ.3: constrained z rotation, EQ.4: constrained x and y rotations, EQ.5: constrained y and z rotations, EQ.6: constrained z and x rotations, EQ.7: constrained x, y, and z rotations.

Remarks:

1. **Boundary Conditions.** Boundary conditions can also be defined on nodal points in a local (or global) system by using the keyword *BOUNDARY_SPC. For other possibilities also see the *CONSTRAINED keyword section of the manual. No attempt should be made to apply boundary conditions to nodes belonging to rigid bodies (see *MAT_RIGID for application of rigid body constraints).

2. **Massless Nodes.** A node without an element or a mass attached to it will be assigned a very small amount of mass and rotary inertia. Generally, massless nodes should not cause any problems but in rare cases may create stability problems if these massless nodes interact with the structure. Warning messages are printed when massless nodes are found. Also, massless nodes are used with rigid bodies to place joints, see *CONSTRAINED_EXTRA_NODES_*OPTION* and *CONSTRAINED_NODAL_RIGID_BODY.

*NODE_MERGE_SET

Purpose: The MERGE_SET option is applied to a set of boundary nodes on disjoint part. With this option, nodes with identical coordinates that are members of any node set ID defined by this keyword are replaced during the input phase by one node within the set or sets. Of the nodes sharing the same coordinates, the node chosen is the one with the smallest ID. During the merging process a tolerance is used to determine whether a node should be merged. This tolerance can be defined using the keyword *NODE_MERGE_-TOLERANCE keyword, which is recommended over the default value. See the *NODE_-MERGE_TOLERANCE input description in the next section. Only nodes contained within the specified sets will be merged. Nodes contained within the set are defined by the *NODE keyword. With this option, the keyword *NODE_MERGE is not needed.

Node Set Cards. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
NSID	Node set ID containing list of nodes to be considered for merging.

*NODE_MERGE_TOLERANCE

Purpose: Define a tolerance that determines whether a node listed in *NODE_MERGE should be merged.

Card 1	1	2	3	4	5	6	7	8
Variable	T0LR							
Type	F							
Default	Rem 1							

VARIABLE	DESCRIPTION
TOLR	Physical distance used to determine whether to merge a nodal pair of nearby nodes.

Remarks:

1. **Default Tolerance.** If the tolerance, TOLR, is undefined or if it is defaulted to zero, a value is computed as:

$$\text{TOLR} = 10^{-5} \cdot \frac{\text{XMAX} + \text{YMAX} + \text{ZMAX} - \text{XMIN} - \text{YMIN} - \text{ZMIN}}{3 \times \sqrt[3]{\text{NUMNP}}},$$

where XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX represent the minimum and maximum values of the (x, y, z) nodal point coordinates in the global coordinate system, and NUMNP is the number of nodal points.

*NODE_REFERENCE

Purpose: Define a nodal coordinate in the reference configuration of a solid element. Only *ELEMENT_SOLID_REFERENCE refers to this node.

Node Card. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	NID	X1		X2		X3				
Type	I	F		F		F				
Default	none	0.0		0.0		0.0				

VARIABLE	DESCRIPTION
NID	Node ID for the reference configuration node. Only *ELEMENT_SOLID_REFERENCE can refer to this node.
X1, X2, X3	Coordinates in the reference configuration

*NODE_RIGID_SURFACE

Purpose: Define a rigid node and its coordinates in the global coordinate system. These nodes are used to define rigid road surfaces, and they have no degrees of freedom. The nodal points are used in the definition of the segments that define the rigid surface. See *CONTACT_RIGID_SURFACE. The nodal point ID must be unique relative to other nodes defined in the *NODE section.

Node Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	NID	X		Y		Z				
Type	I	F		F		F				
Default	none	0.		0.		0.				

VARIABLE	DESCRIPTION
NID	Node number
X	*x* coordinate
Y	*y* coordinate
Z	*z* coordinate

***NODE_SCALAR**_*{OPTION}*

Available options include:

<BLANK>

VALUE

Purpose: Define a scalar nodal point which has a specified number of degrees-of-freedom. The scalar point ID must be unique relative to other nodes defined in the *NODE section.

Node Card. Card 1 for no keyword option (option set to <BLANK>). Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8	9	10
Variable	NID	NDOF								
Type	I	I								
Default	none	0								

Node Card. Card 1 for the VALUE keyword option. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8	9	10
Variable	NID	X1		X2		X3		NDOF		
Type	I	F		F		F		I		
Default	none	0.0		0.0		0.0		0		

VARIABLE	DESCRIPTION
NID	Scalar node ID
NDOF	Number of degrees-of-freedom: EQ.0: Fully constrained EQ.1: One degree-of-freedom

VARIABLE	DESCRIPTION
	EQ.2: Two degrees-of-freedom
	EQ.3: Three degrees-of-freedom
X*i*	Initial value of i^{th} degree of freedom

***NODE_THICKNESS**_*{OPTION1}_{OPTION2}*

For OPTION1 the available options include:

<BLANK>

SET

For OPTION2 the available options include:

<BLANK>

GENERATE

Purpose: Define nodal thickness that overrides nodal thickness otherwise determined via *SECTION_SHELL, *PART_COMPOSITE, or *ELEMENT_SHELL_THICKNESS. The option GENERATE generates a linear thickness distribution between a starting node (or node set) and a ending node (or node set).

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	THK	ID2	INC				
Type	I	F	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
ID1	Node ID, or node set ID if SET option is active. If GENERATE option is active, ID1 serves as the starting node (or node set).
THK	Thickness at node ID1, or node set ID1 if SET option is active (ignored if GENERATE option is active).
ID2	Ending node (or node set) if GENERATE option is active.
INC	Increment in node numbers if GENERATE option is active.

Remarks:

When the GENERATE option is active, both the starting and ending nodes (or node sets) must have a nodal thickness as defined by *NODE_THICKNESS or NODE_THICK-

NESS_SET. The sample commands shown below create a linear thickness distribution between node set 100 and node set 200.

```
*SET_NODE_LIST
100
1, 15, 39
*SET_NODE_LIST
200
7, 21, 45
*NODE_THICKNESS_SET
$ assign thickness of 2.0 to node 1, 15 and 39
100, 2.0
*NODE_THICKNESS_SET
$ assign thickness of 5.0 to node 7, 21 and 45
200, 5.0
*NODE_THICKNESS_SET_GENERATE
$ assign thickness of 3. (= 2.+1.) to node 3 (=1+2), 17 (=15+2) and 41 (=39+2)
$ assign thickness of 4. (= 2.+2.) to node 5 (=1+4), 19 (=15+4) and 43 (=39+4)
100,, 200, 2
```

*NODE_TO_TARGET_VECTOR

Purpose: When the input contains a *CONTROL_FORMING_BESTFIT_VECTOR card, LS-DYNA writes as output a keyword file named bestfit.out. Unlike most keyword cards, which are read by LS-DYNA to define a model, LS-DYNA's keyword reader ignores this card. This card and its associated data cards are a kind of output for the user's benefit (or for post-processing).

Recall that the *CONTROL_FORMING_BESTFIT_VECTOR keyword (mentioned above) rigidly moves a part to best fit a target, such as a scanned part. For each node of the best fit part LS-DYNA writes a data card (Card 1, below) consisting of the components of the vector that is the shortest distance from the node to the target.

Node Cards. As many cards as needed are output in the following format. This output ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8	9	10
Variable	NID	XDELTA		YDELTA		ZDELTA				
Type	I	F		F		F				
Default	none	0.		0.		0.				

VARIABLE	DESCRIPTION
NID	Node ID on a part best fitted to the target.
XDELTA	x-component of the shortest vector from the node to the target
YDELTA	y-component of the shortest vector from the node to the target.
ZDELTA	z-component of the shortest vector from the node to the target.

Revision Information:

The keyword is available starting from revision 112655.

*NODE_TRANSFORM

Purpose: Perform a transformation on a node set based on a transformation defined by the keyword *DEFINE_TRANSFORMATION.

Transformation Cards. Include as many cards in the following format as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	TRSID	NSID	IMMED					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
TRSID	The ID of the transformation defined under *DEFINE_TRANSFORMATION.
NSID	Node set ID of the set of nodes to be subject to the transformation.
IMMED	Optional flag for transformation processing: EQ.0: Node transformation is performed after all input cards are read through. It is more efficient, and the definition sequence of *NODE_TRANSFORM and its NSID is irrelevant, meaning the referred NSID doesn't have to be defined prior to *NODE_TRANSFORM. However, if nodes in NSID are used in, for instance, POS6N of *DEFINE_TRANSFORMATION, the original coordinates, not the transformed coordinates, will be used to define the transformation matrix. EQ.1: Node transformation is performed immediately after *NODE_TRANSFORM is read. The referred NSID and its nodes must be defined prior to *NODE_TRANSFORM.

*PARAMETER

The *PARAMETER family of commands assign numerical values or expressions to named parameters. The parameter names can be used subsequently in the input in place of numerical values.

PARAMETER_*OPTION

*PARAMETER_DUPLICATION

*PARAMETER_EXPRESSION

*PARAMETER_TYPE

***PARAMETER**_*{OPTION}*_*{OPTION}*_*{OPTION}*

The available options are

<BLANK>

LOCAL (see Remark 5)

MUTABLE (see Remark 6)

NOECHO (see Remark 7)

The keyword name may include any combination of the above keyword options in any order.

Purpose: Define the numerical values of parameter names referenced throughout the input file. The parameter definitions, if used, should be placed at the beginning of the input file following *KEYWORD or at the beginning of an include file if the LOCAL option is specified.

Parameter Cards. Include as many cards as necessary.

Card 1	1	2	3	4	5	6	7	8
Variable	PRMR1	VAL1	PRMR2	VAL2	PRMR3	VAL3	PRMR4	VAL4
Type	A	I, F or C	A	I, F or C	A	I, F or C	A	I, F or C
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
PRMR*n*	PRMR*n* sets both the n^{th} parameter and its storage type. $$\text{PRMR} = \texttt{T}\underbrace{\texttt{xxxxxxxxx}}_{\text{9 character name}}$$ The first character, "T", is decoded as follows: T.EQ."R": Parameter is a real number T.EQ."I": Parameter is an integer T.EQ."C": Parameter is a character The remaining 9 characters specify the name of the parameter. A parameter name of "time" (case insensitive) is disallowed. Appendix U lists reserved names. Parameter names can only include numbers, letters, and "_". The first character in a name cannot be

VARIABLE	DESCRIPTION
	a number. For example, to define a shell thickness named, "SHLTHK", the input "RSHLTHK", "R␣␣␣SHLTHK", or "R␣␣SHLTHK␣" are all equivalent 10 character strings ("␣" is space). For instructions regarding how to use the variable "SHLTHK," see Remark 1.
VAL*n*	Define the value of the n^{th} parameter as either a real or integer number, or a character string consistent with preceding definition for PRMR*n*.

Remarks:

1. **Syntax for using parameters.** Parameters can be referenced anywhere in the input by placing an "&" immediately preceding the parameter name. If a minus sign "-" is placed directly before "&", i.e., "-&", with no space the sign of the numerical value will be switched.

2. **Including character parameters in larger strings.** A character parameter can be included as part of a larger string. The included character parameter should start with '&' and end with a delimiter. The default delimiter is '^'. The delimiter will be substituted with a replacement string in the resultant larger string. The default replacement string is '_'. For example, a string definition of "&TORSO^lower arm" will result in "P101_lower arm" if parameter "TORSO" is defined as "C" type with a value of "P101".

   To define a customer delimiter and replacement string, use the special character parameter "CPINLSDNR" with the delimiter and replacement separated by '/'. For instance, the following definition includes the file "/home/PARAM/test.inc":

```
*PARAMETER
CCPINLSDNR      ^^//CINCDIR   PARAM
*INCLUDE
file_&TST^.k
/home/&incdir^^test.inc
```

   If no '/' is found in the definition of CPINLSDNR, the delimiter in the large string is skipped, and no string substitution is applied. Taking the above case as an example, instead of "^^//", if CPINLSDNR is defined as "^^", the file to be included becomes /home/PARAMtest.inc. Only one CPINLSDNR will be used. If more than one is specified in input deck, only the last one is used.

Note that prior to R12.1, specifying an include file name with a character parameter in a larger string did not work properly. LS-DYNA added erroneous spaces to the name. For instance,

```
*PARAMETER
C TST              01
*INCLUDE
file_&TST^.k
```

led to an error termination in earlier versions since LS-DYNA interpreted the name as "file_01 _.k" ("␣" is a space).

3. **Comma-delimited format for character parameters**. Note that prior to R12.1, the comma-delimited input format for character parameters did not work correctly. You should use the fixed format for prior LS-DYNA versions.

4. **Reserved variable names.** See Appendix U for a list of variable names reserved internally by LS-DYNA.

5. **LOCAL option.** *PARAMETER_LOCAL behaves like the *PARAMETER keyword with one difference. A parameter defined by *PARAMETER without the LOCAL option is visible and available at any later point in the input processing. Parameters defined with the LOCAL versions disappear when the input parser finishes reading the file in which they appear. LOCAL variables can temporarily mask non-LOCAL variables.

   For example, suppose you have the following input files:

   main.k:

```
*PARAMETER
R  VAL1    1.0
*PARAMETER
R  VAL2    2.0
*PARAMETER
R  VAL3    3.0
*CONTROL_TERMINATION
   &VAL1
*INCLUDE
file1
```

   file1:

```
*PARAMETER
R  VAL1    10.0
*PARAMETER_LOCAL
R  VAL2    20.0
*PARAMETER_LOCAL
R  VAL4    40.0
*INCLUDE
```

```
file2
⋮
```

Then, inside file2 we will see VAL1 = 10.0, VAL2 = 20.0, VAL3 = 3.0 and VAL4 = 40.0. In main.k, after returning from file1, we will see VAL1 = 10.0, VAL2 = 2.0, and VAL3 = 3.0. VAL4 will not exist. This allows for include files that can set all their own parameters without clobbering the parameters in the rest of the input.

6. **MUTABLE option for redefining.** The MUTABLE option indicates that an integer or real parameter may be redefined at some later point in the input processing (it is ignored for character parameters). Redefinition is allowed regardless of the setting of *PARAMETER_DUPLICATION. The MUTABLE qualifier must appear for the first definition of the parameter. It is not required for any later redefinition.

7. **NOECHO option.** The NOECHO keyword option tells LS-DYNA to not echo the defined parameters to the d3hsp file. This feature is useful for indicating which parameters in an encrypted file should not be echoed.

*PARAMETER_DUPLICATION

Purpose: The purpose is to control how the code behaves if a duplicate parameter definition is found in the input. This command must appear in the keyword deck before all *PARAMETER definitions.

Card 1	1	2	3	4	5	6	7	8
Variable	DFLAG							
Type	I							
Default	1							

VARIABLE	DESCRIPTION
DFLAG	Flag to control treatment of duplicate parameter definitions: EQ.1: issue a warning and ignore the new definition (default) EQ.2: issue a warning and accept the new definition EQ.3: issue an error and ignore (terminates at end of input) EQ.4: accept silently EQ.5: ignore silently

Remarks:

1. **LOCAL and non-LOCAL Variables.** A LOCAL variable appearing in a file, which masks a non-LOCAL parameter, won't trigger these actions; however, a LOCAL that masks another LOCAL or a non-LOCAL that masks a non-LOCAL will.

2. **Multiple Cards.** Only one *PARAMETER_DUPLICATION card is allowed. If more than one is found, a warning is issued and any after the first are ignored.

***PARAMETER_EXPRESSION**_{OPTION}_{OPTION}_{OPTION}

The available options are

<BLANK>

LOCAL

MUTABLE (see Remark 4)

NOECHO (see Remark 5)

The keyword name may include any combination of the above keyword options in any order.

Purpose: Define the numerical values of parameter names referenced throughout the input file. Like the *PARAMETER keyword, but allows for general algebraic expressions, not simply fixed values. The LOCAL option allows for include files to contain their own unique expressions without clobbering the expressions in the rest of the input. See the *PARAMETER keyword.

Parameter Cards. Include as many cards as necessary.

Card 1	1	2	3	4	5	6	7	8
Variable	PRMR1	EXPRESSION1						
Type	A	A						
Default	none	none						

VARIABLE	DESCRIPTION
PRMR*n*	Define the *n*th parameter in a field of 10. Within this field the first character must be either an "R" for a real number, "I" for an integer, or "C" for a character string. Lower or upper case for "I/C/R" is okay. Following the type designation, define the name of the parameter using up to, but not exceeding nine characters. For example, when defining a shell thickness named, "SHLTHK", both inputs "RSHLTHK" or "R SHLTHK" can be used and placed anywhere in the field of 10. When referencing SHLTHK in the input file, see Remark 1 below.
EXPRESSION*n*	General expression which is evaluated, having the result stored in PRMR*n*. The following functions are available (see Remarks 2

VARIABLE	DESCRIPTION
	and 3):
	sin, cos, tan, csc, sec, ctn, asin, acos, atan, atan2, sinh, cosh, tanh, asinh, acosh, atanh, min, max, sqrt, mod, abs, sign, int, aint, nint, anint, exp, log, log10, float, pi, and general arithmetic expressions involving +, -, *, /, and **.
	The standard rules regarding operator precedence are obeyed, and nested parentheses are allowed. The expression can reference previously defined parameters (with or without the leading &). The expression can be continued on multiple lines simply by leaving the first 10 characters of the continuation line blank.
	For type "C" parameters, the expression is not evaluated in any sense, just stored as a string.

Remarks:

1. **Referencing parameters.** Parameters can be referenced anywhere in the input by placing an "&" immediately preceding the parameter name. Expressions can be included in the input when placed between brackets "<>" as long as the total line length does not exceed 80 columns and fields are comma-delimited. For example, this…

```
*parameter
rterm, 0.2, istates,  80
*parameter_expression
rplot,term/(states-30)
*DATABASE_BINARY_D3PLOT
&plot
```

   is equivalent to

```
*parameter
rterm, 0.2, istates,  80
*DATABASE_BINARY_D3PLOT
<term/(states-30)>,
```

2. **Expression properties.** Properties used when evaluating the expressions are listed below.

   a) The integer and real properties of constants and parameters are honored when evaluating expressions. So 2/5 becomes 0, but 2.0/5 becomes 0.4.

b) The sign, atan2, min, max, and mod functions all take two arguments. The others all take only 1. The mod function supports integers only, and any real arguments are rounded.

c) Functions that use an angle as their argument, e.g., sin or cos, assume the angle is in radians.

d) The unary minus has higher precedence than exponentiation, that is, the formula -3**2 is interpreted as $(-3)^2 = 9$.

3. **Expression functions.** Descriptions for the available functions in parameter expressions are as follows:

sin	The sine function. It takes one argument, sin(x).
cos	The cosine function. It takes one argument, cos(x).
tan	The tangent function. It takes one argument, tan(x).
csc	The cosecant function. It takes one argument, csc(x).
sec	The secant function. It takes one argument, sec(x).
ctn	The cotangent function. It takes one argument, ctn(x).
asin	The inverse sine function. It takes one argument, asin(x).
acos	The inverse cosine function. It takes one argument, acos(x).
atan	The inverse tangent function. It takes one argument, atan(x).
atan2	Two-argument arctangent (or inverse tangent) function. It takes two arguments, atan(y,x) where $x + iy$ is a complex number.
sinh	The hyperbolic sine function. It takes one argument, sinh(x).
cosh	The hyperbolic cosine function. It takes one argument, cosh(x).
tanh	The hyperbolic tangent function. It takes one argument, tanh(x).
asinh	The inverse hyperbolic sine function. It takes one argument, asinh(x).
acosh	The inverse hyperbolic cosine function. It takes one argument, acosh(x).
atanh	The inverse hyperbolic tangent function. It takes one argument, atanh(x).
min	The minimum function. It takes two arguments, min(x,y), and returns the minimum value from the arguments.
max	The maximum function. It takes two arguments, max(x,y), and returns the maximum value from the arguments.
sqrt	Computes the square root of the argument. It takes one argument, sqrt(x).

mod The modulo operation. It takes two arguments, mod(x,y), and gives the remainder of the division of x by y. This function only supports integers. Real arguments are rounded.

abs Computes the absolute value of the argument. It takes one argument, abs(x).

sign Function that returns the value of the first argument with the sign of the second argument. It takes two arguments, sign(x,y). For example, sign(-4,8) is 4.

int Converts the argument into a value that has type integer. It takes one argument, int(x). The magnitude of the returned integer does not exceed the magnitude of the argument. For example, int(-48.1) is -48, and int(0.9) is 0.

aint Function that returns a whole number by truncating the argument. The type of the value returned is real. It takes one argument, abs(x). If the magnitude of the argument is less than one, then it returns 0. The magnitude of the returned whole number does not exceed the magnitude of the argument. For example, aint(-48.1) is -48.

nint Rounds the magnitude of the argument to the nearest whole number while retaining the sign. The type of the returned value is integer. It takes one argument, nint(x). For example, nint(-48.8) is -49.

anint Rounds the magnitude of the argument to the nearest whole number while retaining the sign. The type of the returned value is real. It takes one argument, anint(x). For example, anint(-48.8) is -49.

exp Exponential function, e^x, where x is the argument. It takes one argument, exp(x).

log Computes the natural logarithm of the argument. It takes one argument, log(x).

log10 Computes the base 10 logarithm of the argument. It takes one argument, log10(x).

float Converts the argument into a real value. It takes one argument, float(x).

pi Number π.

4. **Redefining parameters.** The MUTABLE option indicates that an integer or real parameter may be redefined at some later point in the input processing (it is ignored for character parameters). Redefinition is allowed regardless of the setting of *PARAMETER_DUPLICATION. The MUTABLE qualifier must appear for the first definition of the parameter. It is not required for any later redefinition.

5. **NOECHO option.** The NOECHO keyword option tells LS-DYNA to not echo the defined parameters to the d3hsp file. This feature is useful for indicating which parameters in an encrypted file should not be echoed.

*PARAMETER_TYPE

*PARAMETER_TYPE is a variation on the *PARAMETER keyword command. In addition to its basic function of associating a parameter name (PRMR) with a numerical value (VAL), the *PARAMETER_TYPE command also includes information (PRTYP) about how the parameter is used by LS-DYNA, such as a Part ID or as a segment set ID.

*PARAMETER_TYPE is useful only when (1) the parameter is used to represent an integer ID number, and (2) LS-PrePost is used to combine two or more models (keyword decks) into a larger model.

Only by knowing how the parameter is used by LS-DYNA is LS-PrePost able to increment the parameter value by the proper "offset" when LS-PrePost combines two or more input decks together into a larger deck. These offsets are necessary so that IDs of a certain type, such as Part IDs, are not duplicated in the assembled model. Figure 36-1 (b) shows the offset input dialog box of LS-PrePost where offset values for specific ID types are assigned.

Background:

This command is designed to support workflows involving models that are built up from discrete subassemblies created by independent workgroups. As the subassembly models evolve through the design process, Part IDs, material IDs, etc. in the models may change with each design iteration; therefore, it is advantageous to parameterize those IDs. In this way, though the parameter values may change, the parameter names remain the same. When the subassembly models are combined by LS-PrePost to create a larger model of an assembly or of a complete system, for instance, an aircraft engine model, parameter values assigned using *PARAMETER_TYPE are incremented by the proper ID offset value as prescribed when LS-PrePost imports each keyword file.

Card Format:

Parameter Cards. For each parameter with type information, include an additional card. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PRMR	VAL	PRTYP					
Type	A	I	A					
Default	none	none	none					

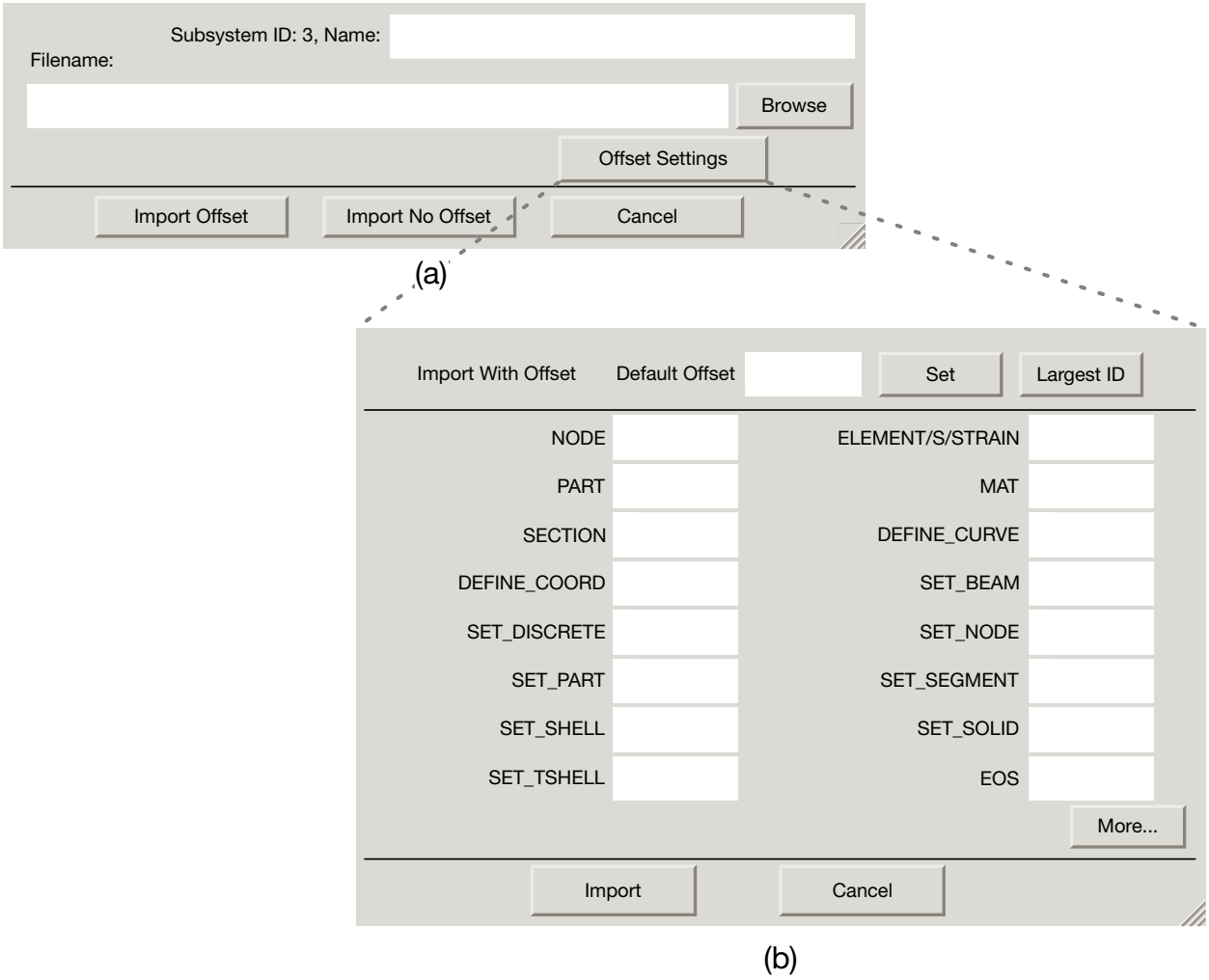


Figure 36-1. (a) the *file* → *import* → *keyword dialog box*; (b) the LS-PrePost dialog that takes the offset as a function of ID type.

VARIABLE	DESCRIPTION
PRMR	PRMR must be in the following format $\text{PRMR} = \texttt{I}\underbrace{\texttt{xxxxxxxxx}}_{\text{9 character name}}$ The first character is the type indicator and must be set to "I" for integer. The remaining 9 characters specify the name of the parameter. For example, to define a part ID "WHLPID", the input "IWHLPID", "I␣␣␣WHLPID", or "I␣␣WHLPID␣" are all equivalent 10 character strings ("␣" is space). For instructions regarding how to use the variable "WHLPID" see Remark 1.

VARIABLE	DESCRIPTION
VAL	Define the value of the parameter. The VAL field must contain an integer.
PRTYP	Describes, for the benefit of LS-PrePost only, how the parameter PRMR is used by LS-DYNA. PRTYP is ignored by LS-DYNA. For example, if VAL represents a Part ID, then PRTYP should be set to "PID". Knowing how the parameter is used by LS-DYNA, LS-PrePost can apply the appropriate offset to VAL when input decks are combined using LS-PrePost.

EQ.NID: Node ID,

EQ.NSID: Node set ID,

EQ.PID: Part ID,

EQ.PSID: Part set ID,

EQ.MID: Material ID,

EQ.EOSID: Equation of state ID,

EQ.BEAMID: Beam element ID,

EQ.BEAMSID: Beam element set ID,

EQ.SHELLID: Shell element ID,

EQ.SHELLSID: Shell element set ID,

EQ.SOLIDID: Solid element ID,

EQ.SOLIDSID: Solid element set ID,

EQ.TSHELLID: Tshell element ID,

EQ.TSHELLSID: Tshell element set ID,

EQ.SSID: Segment set ID

Remarks:

1. **Referencing Parameters.** Parameters can be referenced anywhere in the input by placing an "&" at the first column of its field followed by the name of the parameter without blanks. For example, if PRMR is set to "I␣␣WHLPID␣", then the appropriate reference is "&WHLPID".

   Example:

```
*PARAMETER_TYPE
I WHLPID  100       PID
I WHLMID  300       MID
```

```
*PART
Wheel
&WHLPID,200,&WHLMID
```

2. **LS-PrePost.** *PARAMETER_TYPE is only supported by LS-PrePost 4.1 or later.

3. **Use Limitations.** Combining *INCLUDE_TRANSFORM with *PARAMETER_-TYPE is unsupported. This will introduce conflicting parameter offset values, and offset values specified in *INCLUDE_TRANSFORM will override offset values associated with *PARAMETER_TYPE.

*PART

The following keywords are used in this section:

PART_*{OPTION1}_{OPTION2}_{OPTION3}_{OPTION4}_{OPTION5}_{OPTION6}

*PART_ADAPTIVE_FAILURE

*PART_ANNEAL

PART_COMPOSITE_*{OPTION}

*PART_DUPLICATE

*PART_MODES

*PART_MOVE

*PART_SENSOR

*PART_STACKED_ELEMENTS

***PART**_*{OPTION1}_{OPTION2}_{OPTION3}_{OPTION4}_{OPTION5}_{OPTION6}*

For *OPTION1*, the available options are

<BLANK>

INERTIA

REPOSITION

For *OPTION2*, the available options are

<BLANK>

CONTACT

For *OPTION3*, the available options are

<BLANK>

PRINT

For *OPTION4*, the available options are

<BLANK>

ATTACHMENT_NODES

For *OPTION5*, the available options are

<BLANK>

AVERAGED

For *OPTION6*, the available options are

<BLANK>

FIELD

Options 1, 2, 3, 4, 5, and 6 may be specified in any order on the *PART card.

Purpose: Define parts, that is, combine material information, section properties, hourglass type, thermal properties, and a flag for part adaptivity.

The INERTIA option allows the inertial properties and initial conditions to be defined rather than calculated from the finite element mesh. This applies to rigid bodies (see *MAT_RIGID) only.

The REPOSITION option applies to deformable materials and is used to reposition deformable materials attached to rigid dummy components whose motion is controlled by either CAL3D or MADYMO. At the beginning of the calculation, each component controlled by CAL3D/MADYMO is automatically repositioned to be consistent with the CAL3D/MADYMO input. However, deformable materials attached to these components will not be repositioned unless this option is used.

The CONTACT option allows part-based contact parameters to be used with contact types a3, 4, a5, b5, a10, 13, a13, 15 and 26, that is

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE,

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_MORTAR,

*CONTACT_SINGLE_SURFACE,

*CONTACT_AUTOMATIC_NODES_TO_SURFACE,

*CONTACT_AUTOMATIC_BEAMS_TO_SURFACE,

*CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE,

*CONTACT_AUTOMATIC_SINGLE_SURFACE,

*CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR,

*CONTACT_AIRBAG_SINGLE_SURFACE,

*CONTACT_ERODING_SINGLE_SURFACE,

*CONTACT_AUTOMATIC_GENERAL.

The default values for these contact parameters can be specified on the input for *CONTACT_...

The PRINT option allows user control over whether output data is written into the ASCII files matsum and rbdout. See *DATABASE_ASCII.

The AVERAGED option may be applied *only* to parts consisting of a single (non-branching) line of truss elements. The average strain and strain rate over the length of the truss elements in the *part* are calculated, and the resulting average axial force is applied to all the elements in the part. Truss elements in an averaged part form one long continuous "macro-element." The time step size for an AVERAGED part is based on the total length of the assembled trusses rather than on the shortest truss.

Effectively, the truss elements of an AVERAGED part behave as a string under uniform tension. In an AVERAGED part, no internal forces act to keep the nodes separated, and

other force contributions from the surrounding system *must* play that role. Therefore, the nodes connected to the truss elements should be attached to other structural members. This model is prototypically used for modeling cables in mechanical actuators. The AVERAGED option can be activated for all material types available for truss elements.

The FIELD option enables defining baseline orientation vectors for non-isotropic materials and initializing history variables for all materials with fields of data. The field data cards determine a unique correlation between the current part and the corresponding fields of data. Field data are specified in the physical space on sets of points; see *DEFINE_FIELD and *DEFINE_POINT_CLOUD. The interpolation parameters are specified on *DEFINE_FIELD. The FIELD option is supported for:

- isogeometric shell elements defined via *IGA keywords,
- isogeometric solid element defined via *IGA keywords, and
- solid element formulations 1, 2, 10, 13, 15, and 98.

Card Summary:

Card Sets. Repeat as many sets of data cards as desired (Cards 1 through 12.1, depending on the keyword options). This input ends at the next keyword ("*") card.

Card 1. This card is required.

HEADING	

Card 2. This card is required.

PID	SECID	MID	EOSID	HGID	GRAV	ADPOPT	TMID

Card 3. Include this card if using the INERTIA option.

XC	YC	ZC	TM	IRCS	NODEID		

Card 4. Include this card if using the INERTIA option.

IXX	IXY	IXZ	IYY	IYZ	IZZ		

Card 5. Include this card if using the INERTIA option.

VTX	VTY	VTZ	VRX	VRY	VRZ		

Card 6. Include this card if using the INERTIA option. It is optional unless IRCS = 1.

XL	YL	ZL	XLIP	YLIP	ZLIP	CID	

Card 7. Include this card if using the REPOSITION option.

CMSN	MDEP	MOVOPT					

Card 8. Include this card if using the CONTACT option.

FS	FD	DC	VC	OPTT	SFT	SSF	CPARM8

Card 9. Include this card if using the PRINT option.

PRBF							

Card 10. Include this card if using the ATTACHMENT_NODES option.

ANSID							

Card 11. Include this card if using the FIELD option.

FIDBO							

Card 12. Include this card if using the FIELD option.

NHV							

Card 12.1. Include ceil(NHV/4) instances of this card if NHV≠ 0.

IHV1	FIDHV1	IHV2	FIDHV2	IHV3	FIDHV3	IHV4	FIDHV4

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	A							

VARIABLE	DESCRIPTION
HEADING	Heading for the part

Card 2	1	2	3	4	5	6	7	8
Variable	PID	SECID	MID	EOSID	HGID	GRAV	ADPOPT	TMID
Type	I/A	I/A	I/A	I/A	I/A	I	I	I/A
Default	none	none	none	0	0	0	0	0

VARIABLE	DESCRIPTION
PID	Part identification. A unique number or label must be specified.
SECID	Section identification defined in a *SECTION keyword. See Remark 7.
MID	Material identification defined in the *MAT section. See Remark 7.
EOSID	Equation of state identification defined in the *EOS section. Nonzero only for solid elements using an equation of state to compute pressure. See Remark 7.
HGID	Hourglass/bulk viscosity identification defined in the *HOURGLASS section. See Remark 7. EQ.0: Default values are used.
GRAV	Flag to turn on gravity initialization according to *LOAD_DENSITY_DEPTH. EQ.0: Part will be initialized only if included in the part set PSID in *LOAD_DENSITY_DEPTH. EQ.1: Part will be initialized irrespective of PSID in *LOAD_DENSITY_DEPTH.
ADPOPT	Indicate if this part is adapted. (See also *CONTROL_ADAPTIVE): LT.0: *r*-adaptive remeshing for 2D solids, |ADPOPT| gives the load curve ID that defines the element size as a function of time. EQ.0: Adaptive remeshing is inactive for this part ID. EQ.1: *h*-adaptivity for 3D shells and shell/solid/shell sandwich composites. EQ.2: *r*-adaptive remeshing for 2D solids, 3D tetrahedrons and

VARIABLE	DESCRIPTION
	3D EFG. EQ.3: Axisymmetric *r*-adaptive remeshing for 3D solid (see Remark 6). EQ.9: Passive *h*-adaptivity for 3D shells. The elements in this part will not be split unless their neighboring elements in other parts need to be split more than one level.
TMID	Thermal material property identification defined in the *MAT_THERMAL section. Thermal properties must be specified for all solid, shell, and thick shell parts for thermal or coupled thermal-structural analyses. Discrete elements are not considered in thermal analyses. See Remark 7.

Inertia Card 1. Additional card for the INERTIA option. See Remarks 2, 3, and 4.

Card 3	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	TM	IRCS	NODEID		
Type	F	F	F	F	I	I		

VARIABLE	DESCRIPTION
XC	Global *x*-coordinate of center of mass. If nodal point NODEID is defined, XC, YC, and ZC are ignored, and the coordinates of NODEID are taken as the center of mass.
YC	Global *y*-coordinate of center of mass
ZC	Global *z*-coordinate of center of mass
TM	Translational mass
IRCS	Flag for inertia tensor reference coordinate system: EQ.0: Global inertia tensor EQ.1: Local inertia tensor given in a system defined by the orientation vectors
NODEID	Nodal point defining the center of mass of the rigid body. This node should be, but is not required to be, included as an extra node for the rigid body. If this node is free, its motion will not be

VARIABLE	DESCRIPTION
	updated to correspond with the rigid body after the calculation begins.

Inertia Card 2. Additional card for the INERTIA option.

Card 4	1	2	3	4	5	6	7	8
Variable	IXX	IXY	IXZ	IYY	IYZ	IZZ		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
IXX	I_{xx}, xx component of the inertia tensor (see Remark 4)
IXY	I_{xy},, xy component of the inertia tensor (see Remark 4)
IXZ	I_{xz}, xz component of the inertia tensor (see Remark 4)
IYY	I_{yy}, yy component of the inertia tensor (see Remark 4)
IYZ	I_{yz}, yz component of the inertia tensor (see Remark 4)
IZZ	I_{zz}, zz component of the inertia tensor (see Remark 4)

Inertia Card 3. Additional card for the INERTIA option.

Card 5	1	2	3	4	5	6	7	8
Variable	VTX	VTY	VTZ	VRX	VRY	VRZ		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
VTX	Initial translational velocity of the rigid body in the global x-direction (see Remark 5)
VTY	Initial translational velocity of the rigid body in the global y-direction (see Remark 5)

VARIABLE	DESCRIPTION
VTZ	Initial translational velocity of the rigid body in the global z-direction (see Remark 5)
VRX	Initial rotational velocity of the rigid body about the global x-axis (see Remark 5)
VRY	Initial rotational velocity of the rigid body about the global y-axis (see Remark 5)
VRZ	Initial rotational velocity of the rigid body about the global z-axis (see Remark 5)

Inertial Coordinate System Card. Optional card required for IRCS = 1 with INERTIA option. Define two local vectors or a local coordinate system ID.

Card 6	1	2	3	4	5	6	7	8
Variable	XL	YL	ZL	XLIP	YLIP	ZLIP	CID	
Type	F	F	F	F	F	F	I	
Remark	1	1	1	1	1	1		

VARIABLE	DESCRIPTION
XL	x-coordinate of the local x-axis. The origin lies at (0,0,0).
YL	y-coordinate of the local x-axis
ZL	z-coordinate of the local x-axis
XLIP	x-coordinate of vector in the local xy-plane
YLIP	y-coordinate of vector in the local xy -plane
ZLIP	z-coordinate of vector in the local xy-plane
CID	Local coordinate system ID, see *DEFINE_COORDINATE_… With this option, leave fields 1 - 6 blank.

Reposition Card. Include this card for the REPOSITION option.

Card 7	1	2	3	4	5	6	7	8
Variable	CMSN	MDEP	MOVOPT					
Type	I	I	I					

VARIABLE	DESCRIPTION
CMSN	CAL3D segment number/MADYMO system number. See the numbering in the corresponding program.
MDEP	MADYMO ellipse/plane number: GT.0: Ellipse number EQ.0: Default LT.0: Absolute value is plane number.
MOVOPT	Flag to deactivate moving for merged rigid bodies; see *CONSTRAINED_RIGID_BODIES. This option allows a merged rigid body to be fixed in space while the nodes and elements of the generated CAL3D / MADYMO parts are repositioned: EQ.0: The merged rigid body is repositioned. EQ.1: The merged rigid body is not repositioned.

Contact Card. An additional card is required for the CONTACT option.

Card 8	1	2	3	4	5	6	7	8
Variable	FS	FD	DC	VC	OPTT	SFT	SSF	CPARM8
Type	F	F	F	F	F	F	F	F

NOTE: If FS, FD, DC, and VC are specified, they will not be used unless FS is set to a negative value (-1.0) in the *CONTACT input. These frictional coefficients apply only to contact types:

SINGLE_SURFACE,
AIRBAG_SINGLE_SURFACE,
AUTOMATIC_GENERAL,
AUTOMATIC_SINGLE_SURFACE,
AUTOMATIC_SINGLE_SURFACE_MORTAR,
AUTOMATIC_NODES_TO_...,
AUTOMATIC_SURFACE_...,
AUTOMATIC_SURFACE_..._MORTAR,
AUTOMATIC_ONE_WAY_...,
ERODING_SINGLE_SURFACE

Default values are input via *CONTROL_CONTACT input.

VARIABLE	DESCRIPTION
FS	Static coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity, v_{rel}, of the surfaces in contact, $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}.$$ For Mortar contact $\mu_c = \text{FS}$, meaning dynamic effects are ignored.
FD	Dynamic coefficient of friction. The frictional coefficient is assumed to be dependent on the relative velocity, v_{rel}, of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}.$$ For Mortar contact $\mu_c = \text{FS}$, meaning dynamic effects are ignored.
DC	Exponential decay coefficient. The frictional coefficient is assumed to be dependent on the relative velocity, v_{rel}, of the surfaces in contact $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}.$$ For Mortar contact, $\mu_c = \text{FS}$ (dynamical effects are ignored).
VC	Coefficient for viscous friction. This is necessary to limit the friction force to a maximum. A limiting force is computed by $$F_{\text{lim}} = \text{VC} \times A_{\text{cont}},$$ where A_{cont} is the area of the segment contacted by the node. The

VARIABLE	DESCRIPTION
	suggested value for VC is to use the yield stress in shear; that is, VC = $\sigma_0/\sqrt{3}$, where σ_0 is the yield stress of the contacted material.
OPTT	Optional contact thickness. For SOFT = 2, it applies to solids, shells, and beams. For SOFT = 0 and 1 and Mortar contacts, it only applies to shells and beams. However, for the MPP version only, OPTT does affect the contact behavior of solid elements for SOFT = 0 and 1 but *not* by changing the contact thickness. In the case of MPP with SOFT = 0 and 1 for solids, OPTT overrides the thickness of the solid elements used for the calculation of the contact penetration release (see Table 11-2) but does *not* affect the contact thickness. This is *not* available in SMP. OPTT does *not* affect the contact thickness when applied to the parts on the tracked side (SURFA) of an AUTOMATIC_NODES_-TO_SURFACE contact. However, it affects the contact thickness of this type of contact's reference side (SURFB).
SFT	Optional thickness scale factor in automatic contact (scales the true thickness). This option applies to contact with shell and beam elements. The true thickness is the element thickness.
SSF	Scale factor on default tracked surface penalty stiffness for this part ID whenever it appears in the contact definition. If zero, SSF is taken as unity.
CPARM8	Flag to exclude beam-to-beam contact from the same PID for *CONTACT_AUTOMATIC_GENERAL. This flag applies only to MPP. The global default may be set using CPARM8 on MPP1 of *CONTACT_…_MPP. EQ.0: Flag is not set (default). EQ.1: Flag is set. EQ.2: Flag is set. CPARM8 = 2 has the additional effect of permitting contact treatment of spot weld (type 9) beams in AUTOMATIC_GENERAL contacts; spot weld beams are otherwise disregarded entirely by AUTOMATIC_GENERAL contacts.

Print Card. An additional card is required for the PRINT option. This option applies to rigid bodies and provides a way to turn off ASCII output in files rbdout and matsum.

Card 9	1	2	3	4	5	6	7	8
Variable	PRBF							
Type	I							

VARIABLE	DESCRIPTION
PRBF	Print flag for rbdout and matsum files. EQ.0: Default is taken from *CONTROL_OUTPUT. EQ.1: Write data into the rbdout file only. EQ.2: Write data into the matsum file only. EQ.3: Do not write data into rbdout and matsum.

Attachment Nodes Card. An additional card is required for the ATTACHMENT_-NODES option. See Remark 8.

Card 10	1	2	3	4	5	6	7	8
Variable	ANSID							
Type	I							

VARIABLE	DESCRIPTION
ANSID	Attachment node set ID. See Remark 8. This option should be used very cautiously and applies only to rigid bodies. The attachment point nodes are updated each cycle, whereas other nodes in the rigid body are updated only in the output databases. All loads seen by the rigid body must be applied through this nodal subset or directly to the center of gravity of the rigid body. This set must include all interacting nodes if the rigid body is in contact. EQ.0: All nodal updates are skipped for this rigid body. The null option can be used if the rigid body is fixed in space or does not interact with other parts. For example, the rigid body is only used for some visual purpose.

Field Card 1. This card is required for the FIELD option.

Card 11	1	2	3	4	5	6	7	8
Variable	FIDBO							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
FIDBO	Field ID for baseline orientation vectors for the material referenced by MID. See Remark 9. Set to 0 to not use this feature, such as in the case of unsupported material types.

Field Card 2. This card is required for the FIELD option.

Card 12	1	2	3	4	5	6	7	8
Variable		NHV						
Type		I						
Default		0						

VARIABLE	DESCRIPTION
NHV	Number of history variables defined as functions of spatial coordinates for the material referenced by MID. This is used for initialization. It is supported for all material types. See Remark 10.

Field Card 3. Include ceil(NHV/4) instances of this card if NHV≠ 0.

Card 12.1	1	2	3	4	5	6	7	8
Variable	IHV1	FIDHV1	IHV2	FIDHV2	IHV3	FIDHV3	IHV4	FIDHV4
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
IHV*i*	History variable number from the material model that is initialized with the i^{th} history variable. For example, suppose you are using a supported solid element formulation with *MAT_015, and you want to initialize history variables 5 and 6 for this material with a field. Set NHV = 2 and specify IHV1 = 5 and IHV2 = 6. Field FIDHV1 then provides the field data for history variable #5 and FIDHV2 for 6.
FIDHV*i*	Field ID that defines the i^{th} history variable

Remarks:

1. **Local inertia tensor coordinate system.** The local Cartesian coordinate system is defined as described in *DEFINE_COORDINATE_VECTOR. The local *z*-axis vector is the vector cross product of the *x*-axis and the in-plane vector. The local *y*-axis vector is finally computed as the vector cross product of the *z*-axis vector and the *x*-axis vector. The local coordinate system defined by CID has the advantage that the local system can be defined by nodes in the rigid body, making repositioning the rigid body in a pre-processor much easier since the local system moves with the nodal points.

2. **INERTIA keyword option and shared rigid/deformable nodes.** When specifying mass properties for a rigid body using the INERTIA keyword option, the mass contributions of deformable bodies to nodes shared by the rigid body should be considered part of the rigid body.

3. **INERTIA keyword option lacks default values.** If the INERTIA keyword option is used, all mass and inertia properties of the body *must* be specified. *There are no default values.*

4. **Inertia tensor characteristics.** The inertia terms are always with respect to the center of mass of the rigid body. The reference coordinate system defines the orientation of the axes, not the origin. Note that the off-diagonal terms of the inertia tensor are opposite in sign from the products of inertia.

5. **Initial velocity card for rigid bodies.** The *INITIAL_VELOCITY card may overwrite the initial velocity of the rigid body.

6. **Axisymmetric remeshing.** Axisymmetric remeshing is for 3D orbital forming. The adaptive part using this option needs to meet the following requirements for both geometry and discretization:

a) The geometry is (quasi-) symmetric with respect to the local z-axis, which in turn must be parallel to the global z-axis. See CID in *CONTROL_REMESHING.

b) A set of 2D cross-sections with a uniform angular interval around the z-axis is discretized by mixed triangular and quadrilateral elements in a similar pattern.

c) A set of circular lines around the z-axis passes through the nodes of the cross-sections and forms the orbital pentahedrons and hexahedrons.

7. **Allowed ID values.** The variables SECID, MID, EOSID, HGID, and TMID in *PART, and in *SECTION, *MAT, *EOS, *HOURGLASS, and *MAT_THERMAL, respectively, may be input as a 10-character alphanumeric variable (20-characters if the long format is used), such as "HS Steel," or as a 10-digit integer (20-digit integer if the long format is used) that may not exceed the limit of a 4-byte integer (2,147,483,647) if a single precision executable is used. If any of the aforementioned variables are non-numeric, the execution line should include "pla-bel=y" for proper (but, unfortunately, slower) input processing.

8. **Attachment nodes option.** All nodes are treated as attachment nodes if this option is not used. Attachment nodes apply to rigid bodies only. The motion of these nodes, which must belong to the rigid body, is updated each cycle. Other nodes in the rigid body are updated only for output purposes. Include all nodes in the attachment node set that interact with the structure through joints, contact, merged nodes, applied nodal point loads, and applied pressure. Include all nodes in the attachment node set if their displacements, accelerations, and velocities are to be written into an ASCII output file. Body force loads are applied to the center of gravity of the rigid body.

9. **Spatially varying baseline orientation vectors.** Specifying the baseline orientation vectors using the FIELD option is only active for material models that allow the specification of baseline orientation vectors with parameter AOPT in the keyword *MAT. If FIDBO is used to define baseline orientation vectors, then the vectors associated with the parameter AOPT are disregarded for the current part.

   This feature is supported for isogeometric shell elements in combination with material types 22, 33, 36, 41-50, 103, 104, 108, 122, 133, 135, 136, 215, 242, 249, and 274. For isogeometric solid elements it is supported for material types 22, 91-92, 103, 133, 215, 261, 262, 266, 267, 274, and 295. Solid element formulations 1, 2, 10, 13, 15, and 98 are supported for material types 2, 21, 22, 23, 26, 33, 40, 54, 55, 58, 59, 91-92, 103, 104, 115_O, 122, 126, 128, 132, 133, 142, 143, 157, 189, 199, 213, 215, 219, 221, 233, 245, 261, 262, 263, 264, 266, 267, 274, 278, 291, and 295.

For shells, three values per point are required, so NV should be set to 3 in *DEFINE_FIELD. These values correspond to the components of the baseline orientation vector, which are automatically interpolated at the integration points. The interpolated vector is projected to the reference surface of the shell. The first material axis coincides with the projected vector unless an additional rotation, with respect to the element's normal vector, is specified. The rotation angle between the projected baseline orientation vector and the first material axis can be defined with the material orientation angle cards on *SECTION_IGA_SHELL by setting ICOMP equal to 1.

For solids, six values per point are required, so NV should be set to 6 in *DEFINE_FIELD. The first three per-point values define the first baseline orientation vector **a**, while the last three per-point values define the vector **d**. The components of these vectors are automatically interpolated at the integration points. The interpolated vectors $\mathbf{a}_I$ and $\mathbf{d}_I$ are used to compute the material axes: $\mathbf{a}_{\text{mat}} = \mathbf{a}_I$, $\mathbf{c}_{\text{mat}} = \mathbf{a}_{\text{mat}} \times \mathbf{d}_I$ and $\mathbf{b}_{\text{mat}} = \mathbf{c}_{\text{mat}} \times \mathbf{a}_{\text{mat}}$, where the symbol × denotes the cross product between two vectors.

10. **Spatially varying history variables.** The FIELD option allows initializing an arbitrary number of history variables for all material types as long as the element formulation is supported. The IHV*i* history variable is automatically interpolated at the integration points based on the data point(s) of the FIDHV*i* field. Each history variable is considered a scalar quantity, so NV should be set to 1 in *DEFINE_FIELD. This option supersedes the initialization of history variables specified via *INITIAL_STRESS_*{OPTION}*. For a list of the history variable indices, please refer to the *MAT keyword for the material type referenced by MID.

*PART_ADAPTIVE_FAILURE

Purpose: This option applies to two-dimensional adaptivity and allows a part that is singly connected to be split into two parts.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	T	TERM					
Type	I	F	I					

VARIABLE	DESCRIPTION
PID	Part ID
T	Thickness. When the thickness of the part reaches this minimum value, the part is split into two parts. *The value for T should be on the order of the element thickness of a typical element.*
TERM	Control adaptivity after the part separates: EQ.0: continue to adapt part (default) EQ.1: remove only this part from the adaptivity. Other parts will continue to adapt as normal. If there are no remaining parts to be adapted, adaptivity is disabled EQ.2: adaptivity is disabled for all parts.

*PART_ANNEAL

Available options include:

<BLANK>

SET

Purpose: To initialize the stress states at integration points within a specified part to zero at a given time during the calculation. This option is valid for parts that use constitutive models where the stress is incrementally updated. This option applies to the Hughes-Liu beam elements, the integrated shell elements, thick shell elements, and solid elements. In addition to the stress tensor components, the effective plastic strain is also set to zero.

Part Cards. Include as many parts cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	TIME						
Type	I	F						
Default	none	none						

VARIABLE	DESCRIPTION
PID/PSID	Part ID or part set ID if the SET option is active.
TIME	Time when the stress states are reinitialized.

***PART_COMPOSITE**_*{OPTION1}_{OPTION2}_{OPTION3}*

For *OPTION1* the available options are:

<BLANK>

TSHELL

IGA_SHELL

For *OPTION2* the available option is:

<BLANK>

LONG

For *OPTION3* the available option is:

CONTACT

Purpose: The following input provides a simplified method of defining a composite material model for shell elements and thick shell (TSHELL) elements that eliminates the need for user defined integration rules and part IDs for each composite layer. When *PART_COMPOSITE is used, a section definition, *SECTION_SHELL or *SECTION_TSHELL, and integration rule definition, *INTEGRATION_SHELL, are unnecessary. Using *PART_COMPOSITE, the user specifies layer thicknesses, and integration points reside at the mid-points of layers (trapezoidal rule), if not otherwise defined through IRPL on irregular optional Card 2 (see Remark 6). So, while *PART_COMPOSITE is more straightforward than *INTEGRATION_SHELL, it does not offer the same generality as *INTEGRATION_SHELL in terms of defining an arbitrary integration rule.

The material ID, thickness, material angle and thermal material ID for each through-thickness layer of a composite shell or thick shell are provided in the input for this command. The total number of layers is determined by the number of entries on these cards.

Unless the *ELEMENT_SHELL_THICKNESS card is set, the thickness is assumed to be constant on each shell element. The thickness, then, is given by summing the THICK*i* values from Card 5a/b below over the layers *i*. When the *ELEMENT_SHELL_THICKNESS card is included, the THICK*i* values are scaled to fit the nodal thickness values assigned using the *ELEMENT_SHELL_THICKNESS keyword. For thick shells, the total thickness is obtained from the positions of the nodes on the top and bottom surfaces. In this case, the THICK*i* are also scaled to conform to the geometry defined by the element's nodes.

For a more general method of defining composite shells and thick shells, see *ELEMENT_SHELL_COMPOSITE and *ELEMENT_TSHELL_COMPOSITE. These

commands permit unique layer stack-ups for each element without requiring a unique part ID for each element.

With *PART_COMPOSITE, each Layer Properties Card provides two layers with 4 constants each. With the LONG keyword option, one Layer Properties Card specifies a single layer. The LONG option can also be used with options TSHELL and IGA_SHELL.

To maintain a direct association of through-thickness integration point numbers with physical plies in the case where plies span over more than one part ID, see Remark 5.

The CONTACT option allows part-based contact parameters to be used with the automatic contact types a3, 4, a5, a10, 13, a13, 15 and 26, which are listed under the *PART definition above.

The IGA_SHELL option allows *PART_COMPOSITE to be used with IGA shells. The part ID specified in *IGA_SHELL can reference the part ID in *PART_COMPOSITE.

Card Summary:

Card 1. This card is required.

HEADING	

Card 2. The keyword reader will interpret the card following Card 1 as optional Card 2 if the first column of the card is occupied by the string "OPTCARD." Otherwise, it is interpreted as Card 3a, 3b, or 3c.

OPTC	IRPL						

Card 3a. This card is included for all keyword options other than TSHELL or IGA_SHELL.

PID	ELFORM	SHRF	NLOC	MAREA	HGID	ADPOPT	THSHEL

Card 3b. This card is included if the TSHELL option is used.

PID	ELFORM	SHRF			HGID		TSHEAR

Card 3c. This card is included if the IGA_SHELL option is used.

PID	ELFORM	SHRF	NLOC		IRL		

Card 4. This card is included if the CONTACT option is used.

FS	FD	DC	VC	OPTT	SFT	SSF	

Card 5a. This card is included if the LONG option is not used. This card provides the layer data. Include as many cards as necessary. This input ends at the next keyword ("*") card.

MID1	THICK1	B1	TMID1	MID2	THICK2	B2	TMID2

Card 5b. This card is included if the LONG option is used. This card provides the layer data. Include as many cards as necessary. This input ends at the next keyword ("*") card.

MID1	THICK1	B1	TMID1	PLYID1	SHRFAC1		

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	A							

VARIABLE	DESCRIPTION
HEADING	Heading for the part

Irregular Optional Card. The keyword reader will interpret the card following Card 1 as optional Card 2 if the first column of the card is occupied by the string "OPTCARD." Otherwise, it is interpreted as Card 3 (a, b or c); see below.

Card 2	1	2	3	4	5	6	7	8
Variable	OPTC	IRPL						
Type	A	I						

VARIABLE	DESCRIPTION
OPTC	Include string "OPTCARD" to activate this card.
IRPL	Through thickness integration rule per layer (see Remark 6) EQ.103: 3-point Simpson

Thin Shell Card. The following card is required for thin shell composites that are not IGA. Omit this card if the TSHELL option or the IGA_SHELL option is used.

Card 3a	1	2	3	4	5	6	7	8
Variable	PID	ELFORM	SHRF	NLOC	MAREA	HGID	ADPOPT	THSHEL
Type	I	I	F	F	F	I	I	I
Default	none	none	1.0	0.0	0.0	0	0	0

VARIABLE	DESCRIPTION
PID	Part ID
ELFORM	Element formulation options for thin shells: EQ.1: Hughes-Liu, EQ.2: Belytschko-Tsay, EQ.3: BCIZ triangular shell, EQ.4: C^0 triangular shell, EQ.6: S/R Hughes-Liu, EQ.7: S/R co-rotational Hughes-Liu, EQ.8: Belytschko-Leviathan shell, EQ.9: Fully integrated Belytschko-Tsay membrane, EQ.10: Belytschko-Wong-Chiang, EQ.11: Fast (co-rotational) Hughes-Liu, EQ.16: Fully integrated shell element (very fast), EQ.-16: Fully integrated shell element modified for higher accuracy, EQ.17: Fully integrated DKT, triangular shell element, EQ.18: Fully integrated linear DK quadrilateral/triangular shell, EQ.20: Fully integrated linear assumed strain C0 shell, EQ.21: Fully integrated linear assumed strain C0 shell (5 DOF), EQ.23: 8-node quadratic quadrilateral shell (see IRQUAD in *CONTROL_SHELL),

VARIABLE	DESCRIPTION
	EQ.24: 6-node quadratic triangular shell, EQ.25: Belytschko-Tsay shell with thickness stretch, EQ.26: Fully integrated shell with thickness stretch, EQ.27: C0 triangular shell with thickness stretch, EQ.30: Fast fully integrated element with 2 in-plane integration points based on ELFORM 16 EQ.41: Mesh-free (EFG) shell local approach (more suitable for crashworthiness analysis), EQ.42: Mesh-free (EFG) shell global approach (more suitable for metal forming analysis), EQ.101: User defined shell, EQ.102: User defined shell, EQ.103: User defined shell, EQ.104: User defined shell, EQ.105: User defined shell.
SHRF	Shear correction factor which scales the transverse shear stress.
NLOC	Location of reference surface, available for thin shells only. If nonzero, the offset distance from the plane of the nodal points to the reference surface of the shell in the direction of the shell normal vector is a value: $\text{offset} = -0.50 \times \text{NLOC} \times (\text{average shell thickness})$. This offset is not considered in the contact subroutines unless CNTCO is set to 1 or 3 in *CONTROL_SHELL. Alternatively, the offset can be specified by using the OFFSET option in the *ELEMENT_SHELL input section. EQ.1.0: Top surface, EQ.0.0: Mid-surface (default), EQ.-1.0: Bottom surface.
MAREA	Non-structural mass per unit area. This is additional mass which comes from materials such as carpeting. This mass is not directly included in the time step calculation.
HGID	Hourglass/bulk viscosity identification defined in the *HOURGLASS Section:

VARIABLE	DESCRIPTION
	EQ.0: Default values are used.
ADPOPT	Indicate if this part is adapted or not. Also see, *CONTROL_-ADAPTIVITY: EQ.0: No adaptivity, EQ.1: *h*-adaptive for 3-D thin shells.
THSHEL	Thermal shell formulation EQ.0: Default is governed by THSHEL on *CONTROL_SHELL EQ.1: Thick thermal shell EQ.2: Thin thermal shell

Thick Shell Card. This is an additional card for the TSHELL option.

Card 3b	1	2	3	4	5	6	7	8
Variable	PID	ELFORM	SHRF			HGID		TSHEAR
Type	I	I	F			I		I
Default	none	none	1.0			0		0

VARIABLE	DESCRIPTION
PID	Part ID
ELFORM	Element formulation options for thick shells: EQ.1: One point reduced integration, EQ.2: Selective reduced 2 × 2 in plane integration, EQ.3: Assumed strain 2 × 2 in plane integration, EQ.5: Assumed strain reduced integration with brick materials EQ.6: Assumed strain reduced integration with shell materials EQ.7: Assumed strain 2 × 2 in plane integration
SHRF	Shear correction factor which scales the transverse shear stress.

VARIABLE	DESCRIPTION
HGID	Hourglass/bulk viscosity identification defined in the *HOURGLASS Section: EQ.0: Default values are used.
TSHEAR	Flag for transverse shear stress distribution (see Remarks 3 and 4): EQ.0: Parabolic, EQ.1: Constant through thickness.

IGA Shell Card. The following card is required for the IGA_SHELL option.

Card 3c	1	2	3	4	5	6	7	8
Variable	PID	ELFORM	SHRF	NLOC		IRL		
Type	I	I	F	F		I		
Default	none	0	1.0	0.0		0		

VARIABLE	DESCRIPTION
PID	Part ID
ELFORM	Element formulation options for IGA shells: EQ.0: Reissner-Mindlin with fibers at the control points EQ.1: Kirchhoff-Love with fibers at the control points EQ.2: Kirchhoff-Love with fibers at the integration points EQ.3: Reissner-Mindlin with fibers at the integration points EQ.5: Shell with thickness stretch based on the ELFORM = 0 EQ.6: Shell with thickness stretch based on ELFORM = 3
SHRF	Shear correction factor which scales the transverse shear stress.
NLOC	Location of reference surface; see the definition of NLOC in *SECTION_IGA_SHELL for more details.
IRL	Lamina integration rule: EQ.0: Reduced Gauss-Legendre

VARIABLE	DESCRIPTION
	EQ.1: Gauss-Legendre
	EQ.2: Patchwise reduced Gauss-Legendre (for biquadratic NURBS only)

Contact Card. Additional card is required for the CONTACT option.

Card 4	1	2	3	4	5	6	7	8
Variable	FS	FD	DC	VC	OPTT	SFT	SSF	
Type	F	F	F	F	F	F	F	

NOTE: If FS, FD, DC, and VC are specified they will not be used unless FS is set to a negative value (-1.0) in the *CONTACT section. These frictional coefficients apply only to contact types:

SINGLE_SURFACE,
AUTOMATIC_GENERAL,
AUTOMATIC_SINGLE_SURFACE,
AUTOMATIC_NODES_TO_...,
AUTOMATIC_SURFACE_...,
AUTOMATIC_ONE_WAY_...,
ERODING_SINGLE_SURFACE

Default values are input via *CONTROL_CONTACT input.

VARIABLE	DESCRIPTION
FS	Static coefficient of friction. The functional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact as $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}.$$
FD	Dynamic coefficient of friction. The functional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact as $$\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}.$$

VARIABLE	DESCRIPTION
DC	Exponential decay coefficient. The functional coefficient is assumed to be dependent on the relative velocity v_{rel} of the surfaces in contact as $\mu_c = \text{FD} + (\text{FS} - \text{FD})e^{-\text{DC}\times\lvert v_{\text{rel}}\rvert}$.
VC	Coefficient for viscous friction. This is necessary to limit the friction force to a maximum. A limiting force is computed $F_{\text{lim}} = \text{VC} \times A_{\text{cont}}$. A_{cont} being the area of the segment contacted by the node in contact. The suggested value for VC is to use the yield stress in shear VC $= \sigma_o/\sqrt{3}$ where σ_0 is the yield stress of the contacted material.
OPTT	Optional contact thickness. This applies to shells only.
SFT	Optional thickness scale factor for PART ID in automatic contact (scales true thickness). This option applies only to contact with shell elements. True thickness is the element thickness of the shell elements.
SSF	Scale factor on default tracked surface penalty stiffness for this part ID whenever it appears in the contact definition. If zero, SSF is taken as unity.

Layer Data Cards without Long Option. The material ID, thickness, and material angle for each through-thickness layer of a composite shell are provided below (up to two layers per card). The layer data should be given sequentially starting with the bottommost layer. The total number of layers is determined by the number of entries on these cards. Include as many cards as necessary. The next keyword ("*") card terminates this input.

Card 5a	1	2	3	4	5	6	7	8
Variable	MID1	THICK1	B1	TMID1	MID2	THICK2	B2	TMID2
Type	I	F	F	I	I	F	F	I

Layer Data Cards for Long Option. The material ID, thickness, and material angle for each through-thickness layer of a composite shell are provided below (one layer per card). The layer data should be given sequentially starting with the bottommost layer. The total number of layers is determined by the number of entries on these cards. Include as many cards as necessary. The next keyword ("*") card terminates this input.

Card 5b	1	2	3	4	5	6	7	8
Variable	MID1	THICK1	B1	TMID1	PLYID1	SHRFAC1		
Type	I	F	F	I	I	F		

VARIABLE	DESCRIPTION
MID*i*	Material ID of layer *i*, see *MAT_… Section.
THICK*i*	Thickness of layer *i*.
B*i*	Material angle of layer *i*. This material angle applies only to material types 21, 22, 23, 33, 33_96, 34, 36, 40, 41-50, 54, 55, 58, 59, 103, 103_P, 104, 108, 116, 122, 133, 135, 135_PLC, 136, 157, 158, 190, 219, 226, 233, 234, 235, 242, 243, 261, 262, 278, and 293.
TMID*i*	Thermal material ID of layer *i*
PLYID*i*	Ply ID of layer *i* (for post-processing purposes)
SHRFAC*i*	Transverse shear stress scale factor

Remarks:

1. **Orthotropic Materials.** In cases where there is more than one orthotropic material model referenced by *PART_COMPOSITE, the orthotropic material orientation parameters (AOPT, BETA, and associated vectors) from the material model of the first orthotropic integration point apply to all the orthotropic integration points. AOPT, BETA, etc. input for materials of subsequent integration points are ignored. B*i*, not to be confused with BETA, is applied into account for each integration point.

2. **SHRF Field and Zero Traction Condition.** Thick shell formulations 1, 2, and 3, and all shell formulations, except for BCIZ and DK elements, are based on first order shear deformation theory that yields constant transverse shear strains which violates the condition of zero traction on the top and bottom surfaces of the shell. For these elements, setting SHRF = 0.83333 will compensate for this

error and result in the correct transverse shear deformation, so long as all layers have the same transverse stiffness. SHRF is not used by thick shell forms 3, 5, or 7 except for materials 33, 36, 133, 135, and 243.

3. **Thick Shell 5 or 6 and Shear Stress.** When thick shell formulation 5 and 6 are used, LS-DYNA will use either a parabolic transverse shear stress distribution when TSHEAR = 0 or a constant shear stress distribution when TSHEAR = 1. The parabolic option is recommended when elements are used in a single layer to model a plate or beam. The constant option may be better when elements are stacked so there are two or more elements through the thickness.

4. **Laminated Shear Stress Theory to Minimize Discontinuities.** For composites that have a transverse shear stiffness that varies by layer, laminated shell theory, activated by LAMSHT on *CONTROL_SHELL, will correct the transverse shear stress to minimize stress discontinuities between layers and at the bottom and top surfaces by imposing a parabolic transverse shear stress. SHRF should be set to the default value of 1.0 when the shear stress distribution is parabolic. If thick shells are stacked so that there is more than one element through the thickness of a plate or beam model, setting TSHEAR = 1 will cause a constant shear stress distribution which may be more accurate than parabolic. The TSHEAR parameter is available for all thick shell forms when laminated shell theory is active. Alternatively, a scale factor can be defined for transverse shear stress in each layer of shell or thick shell composites using the SHRFAC parameter. The inputted SHRFAC values are normalized so that overall shear stiffness is unaffected by the distribution. Therefore, only the ratio of parameter values is significant.

5. **Assignment of Zero Thickness to Layers.** The ability to assign zero thickness layers in the stacking sequence allows the number of layers to remain constant even as the number of physical plies varies from part to part and eases post-processing since a particular layer corresponds to a physical ply. Such a capability is important when one or more of the physical plies are not continuous across a composite structure. To represent a missing ply in *PART_COMPOSITE, set THICK*i* to 0.0 for the corresponding layer and additionally, set either MID = -1 or, if the LONG option is used, set PLYID to any nonzero value.

   When the number of physical plies varies from element to element in a part, one can assign zero thickness to integration points in exactly the same manner as described above but on an element-by-element basis using *ELEMENT_SHELL_COMPOSITE(_LONG) or *ELEMENT_TSHELL_COMPOSITE.

   When post-processing the results using LS-PrePost version 4.5, read both the keyword deck and d3plot database into the code and then select *Option* →*N/A gray fringe*. Then, when viewing fringe plots for a particular integration point

(*FriComp* → *IPt* → *intpt#*), the element will be grayed out if the selected integration point is missing (or has zero thickness) in that element.

6. **Through Thickness Integration Rule per Layer.** If not otherwise stated, each layer will be numerically integrated with a one-point trapezoidal rule which should be sufficient for most composite layups and which will be the recommended setting. However, by adding an irregular optional card 1a, the user may define other through thickness integration rules that should be applied for the numerical integration of an individual ply. So far, a 3-point Simpson integration per layer is supported, where the integration points are located at the bottom, the middle and the top of each individual layer. If IRPL is set to an unknown integration rule, the standard default one-point integration will be used.

***PART_DUPLICATE**_*{OPTION}*

The available *OPTION* is:

<BLANK>

NULL_OVERLAY

NULL_OVERLAY is used to generate null shells for contact. See Remark 5.

Purpose: To provide a method of duplicating parts or part sets without the need to use the *INCLUDE_TRANSFORM option.

Duplication Cards. This format is used when the keyword option is left <BLANK>. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	PTYPE	TYPEID	IDPOFF	IDEOFF	IDNOFF	TRANID	BOXID	ZMIN
Type	A	I	I	I	I	I	I	F
Default	none	none	0	0	0	0	0	0.0

Null Duplication Cards. This format is used when the keyword option is set to NULL_-OVERLAY. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	PTYPE	TYPEID	IDPOFF	IDEOFF	DENSITY	E	PR	
Type	A	I	I	I	F	F	F	
Default	none	none	0	0	0	0	0	

VARIABLE	DESCRIPTION
PTYPE	Type of entity to duplicate: EQ.PART: Duplicate a single part EQ.PSET: Duplicate a part set

VARIABLE	DESCRIPTION
TYPEID	ID of part or part set to be duplicated
IDPOFF	ID offset of newly created parts
IDEOFF	ID offset of newly created elements
IDNOFF	ID offset of newly created nodes
TRANID	ID of *DEFINE_TRANSFORMATION to transform the existing nodes in a part or part set; see Remark 4.
BOXID	Optional *DEFINE_BOX ID used to define the boundary of the transformed configuration; see Remark 6.
ZMIN	Minimum value of the Z-coordinate of the transformed part or part set if set to a value other than zero. This field applies to all transformations except for SCALE.
DENSITY	Density
E	Young's modulus
PR	Poisson's ratio

Remarks:

1. **Parts with Common Nodes.** All parts sharing common nodes must be grouped in a *PART_SET and duplicated in a single *PART_DUPLICATE command so that the newly duplicated parts still share common nodes.

2. **Allowed Elements.** The following elements which need a *PART to complete their definition can be duplicated by using this command: *ELEMENT_SOLID, *ELEMENT_DISCRETE, *ELEMENT_SHELL, *ELEMENT_TSHELL, *ELEMET_BEAM, and *ELEMENT_SEATBELT.

3. **Constraints.** This command only duplicates definition of nodes, elements and parts, not the associated constraints. For example, TC and RC defined in *NODE will not be passed to the newly created nodes.

4. **Transformation.** When IDNOFF = IDPOFF = IDEOFF = 0, the existing part, or part set, will be transformed as indicated by TRANID; no new node or elements will be created.

5. **Null Elements.** The NULL_OVERLAY option may be used to generate 3 and 4-node null shell elements from the 6- and 8-node quadratic elements for use in contact. No additional nodes are generated.

6. **Boundary Box.** The XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX of *DEFINE_BOX define the boundary of the transformed configuration. If any node falls out of the box boundary after transformation, TRANID, is applied, an additional rigid body translation will be applied automatically to the whole part(s) to assure no box boundary violation.

Example:

The following partial keyword example will rotate a part set 45° about the global Y-axis, passing through the global origin. The minimum coordinate of any nodes in the part set after the transformation will be 431.0 mm.

```
*PART_DUPLICATE
$     PTYPE    TYPEID    IDPOFF    IDEOFF    IDNOFF    TRANID     BOXID       ZMIN
      PSET       300                                     989                 431.
*DEFINE_TRANSFORMATION
$ TRANID
989
$OPTION, A1, A2, A3, A4, A5, A6, A7
ROTATE,0.0,1.0,0.0,,,,45.0
```

Revision Information:

The variable ZMIN is available starting in Dev 138833.

*PART_MODES

Purpose: Treat a part defined with *MAT_RIGID as a linearized flexible body (LFB) whereby deformations are calculated from mode shapes. Unlike superelements (*ELEMENT_DIRECT_MATRIX_INPUT), linearized flexible bodies are accurate for systems undergoing large displacements and large rotations.

The modes shapes obtained experimentally or in a finite element analysis, e.g., NASTRAN .pch file or an LS-DYNA d3eigv or d3mode file, enable modeling the deformations. These files may contain a combination of normal modes, constraint modes, and attachment modes. For stress recovery in linearized flexible bodies, we recommend using linear element formulations. The implementation assumes a lump mass matrix. See also *CONTROL_RIGID.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NMFB	FORM	ANSID	FORMAT	KMFLAG	NUPDF	SIGREC
Type	I	I	I	I	I	I	I	

Card 2	1	2	3	4	5	6	7	8
Variable	FILENAME							
Type	C							
Default	none							

Kept Mode Cards. Additional card if KMFLAG = 1. Use as many cards as necessary to specify the NMFB kept modes. After defining NMFB modes, no further input is expected.

Card 3	1	2	3	4	5	6	7	8
Variable	MODE1	MODE2	MODE3	MODE4	MODE5	MODE6	MODE7	MODE8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Optional Modal Damping Cards. This input ends at the next keyword ("*") card.

Card 4	1	2	3	4	5	6	7	8
Variable	MSTART	MSTOP	DAMPF					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
PID	Part identification. This part must be a rigid body. See Remark 4.
NMFB	Number of kept modes in linearized flexible body. The number of modes in the file, FILENAME, must equal or exceed NMFB. If KMFLAG = 0, the first NMFB modes in the file are used.
FORM	Linearized flexible body formulation. See Remark 6 below. EQ.0: Exact EQ.1: Fast EQ.2: General formulation (default) EQ.3: General formulation without rigid body mode orthogonalization
ANSID	Attachment node set ID (optional).
FORMAT	Input format of modal information:

VARIABLE	DESCRIPTION
	EQ.0: NASTRAN .pch file.
	EQ.1: (not supported)
	EQ.2: NASTRAN .pch file (LS-DYNA binary version). The binary version of this file is automatically created if a NASTRAN .pch file is read. The name of the binary file is the name of the NASTRAN .pch file but with .bin appended. The binary file is smaller and can be read much faster.
	EQ.3: LS-DYNA d3eigv binary eigenvalue database (see *CONTROL_IMPLICIT_EIGENVALUE).
	EQ.4: LS-DYNA d3mode binary constraint/attachment mode database (see *CONTROL_IMPLICIT_MODE).
KMFLAG	Kept mode flag. It selects the method for identifying modes to keep. This flag is not supported for FORMAT = 4 (d3mode).
	EQ.0: The first NMFB modes in the file, FILENAME, are used.
	EQ.1: Define NMFB kept modes with additional input.
NUPDF	Nodal update flag. If active, an attachment node set, ANSID, must be defined.
	EQ.0: All nodes of the rigid part are updated each cycle.
	EQ.1: Only attachment nodes are fully updated. All nodes in the body are output based on the rigid body motion without the addition of the modal displacements. For maximum benefit an attachment node set can also be defined with the PART_ATTACHMENT_NODES option. The same attachment node set ID should be used here.
SIGREC	Stress recovery flag:
	EQ.0: Do not recover stress.
	EQ.1: Recover stress.
	EQ.2: Recover stress and then set the recovery stress as initial stress when switching to a deformable body using *DEFORMABLE_TO_RIGID_AUTOMATIC. (shell formulations 16, 18, 20, and 21 and solid formulation 2).
	EQ.3: Recover stress based on shell formulation 21, and then set the recovery stress as initial stress for shell formulation 16 when switching to a deformable body using *DEFORMABLE_TO_RIGID_AUTOMATIC (shell formulation 16

VARIABLE	DESCRIPTION
	only).
FILENAME	The path and name of a file which contains the modes for this rigid body.
MODE*n*	Keep normal mode, MODE*n*.
MSTART	First mode for damping, (1 ≤ MSTART ≤ NMFB).
MSTOP	Last mode for damping, MSTOP, (1 ≤ MSTOP ≤ NMFB). All modes between MSTART and MSTOP inclusive are subject to the same modal damping coefficient, DAMPF.
DAMPF	Modal damping coefficient, ζ. See Remark 5.

Remarks:

1. **Shared nodes and interconnections.** Currently, linearized flexible bodies cannot share nodes with other linearized flexible bodies or rigid bodies; however, interconnections to other linearized flexible bodies or to rigid bodies can use the penalty joint option. The linearized flexible bodies are not implemented with the Lagrange multiplier joint option (see LMF in *CONTROL_RIGID).

2. **Normal modes file format.** The format of the file which contains the normal modes follows the file formats of NASTRAN output for modal information.

3. **Mode set.** The mode set typically combines both normal modes and attachment modes. The eigenvalues for the attachment modes are computed from the stiffness and mass matrices.

4. **PID.** The part ID specified must be either a single rigid body or a lead rigid body (see *CONSTRAINED_RIGID_BODIES) which can be made up of many rigid parts.

5. **Modal damping.** The modal damping is defined by the modal damping coefficient ζ, where a value of 1.0 equals critical damping. For a one degree of freedom model system, the relationship between the damping and the damping coefficient is $c = 2\zeta\omega_n m$, where c is the damping, m is the mass, and ω_n is the natural frequency, $\sqrt{k/m}$.

6. **Linearized flexible body formulations.** There are four linearized flexible body formulations. The first is a formulation that contains all the terms of the linearized flexible body equations, and its cost grows approximately as the square of the number of modes. The second formulation ignores most of the second-order

terms appearing in the exact equations, and its cost grows linearly with the number of modes. If the angular velocities are small and if the deflections are small with respect to the geometry of the system, the cost savings of the second formulation may make it more attractive than the first method.

Please note that the first two formulations are only applicable when the modes are eigenmodes computed for the free-free problem, that is, the 6 rigid body modes are included. The third formulation, the default, is a more general formulation that allows more general mode shapes. The fourth formulation does not orthogonalize the modes with respect to the rigid body modes and may allow boundary conditions to be imposed more simply in some cases than the third formulation. This last formulation is selected regardless of user input if attachment modes are used.

***PART_MOVE**_*{OPTION}*

For *OPTION* the available options are:

<BLANK>

N2N

Purpose: Translate a part by an incremental displacement in either a local or global coordinate system or by a distance along a defined vector. This option currently applies to parts defined by either shell or solid elements. All nodal points of the given part ID are moved. Care must be observed since parts that share boundary nodes with the part being moved must also be moved to avoid severe mesh distortions – the variable IFSET can be used to handle the situation.

With the N2N option, the movement can be defined in terms of the displacement from node N1 to node N2.

Card Summary:

Card 1a. Include as many instances of this card as desired when not using a keyword option (<BLANK>). This input ends at the next keyword ("*") card.

PID	XMOV	YMOV	ZMOV	CID	IFSET	

Card 1b. Include as many instances of this card as desired when the keyword option is set to N2N. This input ends at the next keyword ("*") card.

PID	N1	N2	IFSET	DONOW					

Data Card Definitions:

Part/Part Set Move Cards for no Keyword Option. Include as many instances of this card as desired when not using a keyword option (<BLANK>). This input ends at the next keyword ("*") cards.

Card 1a	1	2	3	4	5	6	7	8	9	10
Variable	PID	XMOV		YMOV		ZMOV		CID	IFSET	
Type	I	F		F		F		I	I	
Default	none	0.0		0.0		0.0		0	0	

VARIABLE	DESCRIPTION
PID	Part or part set ID
XMOV	Move shell/solid part ID, PID, in the x-direction by the incremental distance, XMOV. Ignored if CID < 0.
YMOV	Move shell/solid part ID, PID, in the y-direction by the incremental distance, YMOV. Ignored if CID < 0.
ZMOV	Move shell/solid part ID, PID, in the z-direction by the incremental distance, ZMOV. CID.LT.0: ZMOV is the distance the part is translated along CID.
CID	Coordinate system ID to define incremental displacement or vector ID to define the direction for the translation. GT.0: Local coordinate system ID. All displacements, XMOV, YMOV, and ZMOV, are with respect to CID. EQ.0: Global coordinate system. All displacements, XMOV, YMOV, and ZMOV, are with respect to the global coordinate system. LT.0: |CID| is a vector ID for a vector that points in the direction of the part translation. XMOV and YMOV are ignored in this case.
IFSET	Indicate if a part set ID is used in the PID definition: EQ.0: Part ID is used. EQ.1: Part set ID is used.

Part/Part Set Move Cards for the N2N keyword option. Include as many instances of this card as desired when the keyword option is set to N2N. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8	9	10
Variable	PID	N1	N2	IFSET	DONOW					
Type	I	I	I	I	I					
Default	none	none	none	0	0					

VARIABLE	DESCRIPTION
PID	Part or part set ID
N1	Node ID giving the starting point for the vector that defines the motion
N2	Node ID giving the ending point for the vector that defines the motion
IFSET	Indicate if a part set ID is used in the PID definition: EQ.0: Part ID is used. EQ.1: Part set ID is used.
DONOW	Flag specifying when the part is moved during keyword processing: EQ.0: The part is moved after all input cards have been processes. Thus the movement is based on the transformed coordinates of N1 and N2. EQ.1: The part is moved based on the coordinates at the time this keyword is read. Therefore, nodes N1 and N2 must be defined before *PART_MOVE_N2N is processed. Note that the difference between DONOW = 0 and 1 can be significant if N1 and N2 are affected by *INCLUDE_TRANSFORM or *NODE_TRANSFORM.

Remarks:

1. **Multiple parts.** IFSET addresses the movement of multiple parts that share common boundary nodes, such as tailor-welded blanks. This field allows for all parts in a part set to move simultaneously.

2. **Draw beads.** Draw beads can be modeled as beam elements that move the same distance and direction as either the die or punch, depending on the draw types.

3. **Supported keywords.** Keywords *SET_PART_COLLECT and *SET_PART_ADD are supported.

Example:

The example partial deck given below automatically positions all tools in a toggle draw of a decklid inner, with a tailor welded blank consisting of PID 1 and PID5, as shown in

Figure 37-1. The keyword *CONTROL_FORMING_AUTOPOSITION_PARAMETER_-SET calculates where to position various part sets as follows:

a) the tailor-welded blank part set ID 1 is to be positioned in the global *Z*-direction on top of the lower die cavity (part set ID 4);

b) the binder (part set ID 3) is to be positioned on top of the blank;

c) the upper punch (part set ID 2) is to be positioned on top of the blank.

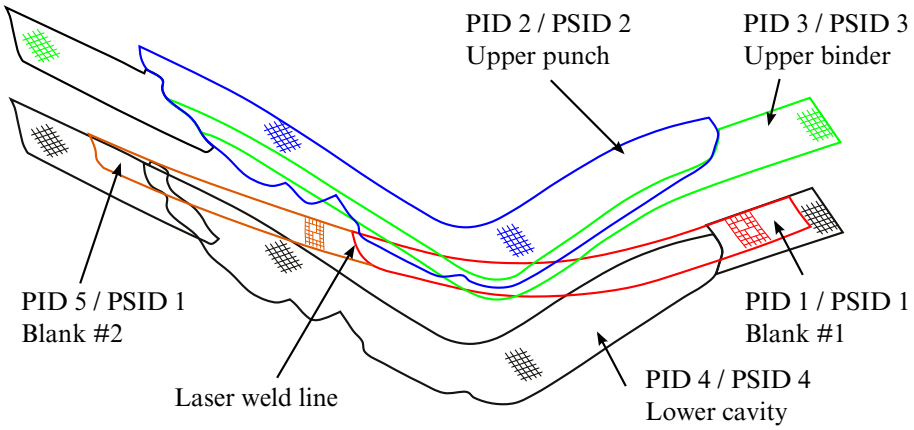


Figure 37-1. A tailor welded blank is positioned in a decklid (toggle draw).

The three positioning distances for the blank, upper binder and upper punch are calculated and stored in variables &blnkmv, &upbinmv, and &uppunmv, respectively. The keyword *PART_MOVE, with IFSET set to 1, is responsible for moving the three part sets, using the three corresponding positioning variables.

```
*PARAMETER
R   blnkmv       0.0
R  upbinmv       0.0
R  uppunmv       0.0
*SET_PART_LIST
1
1,5
*SET_PART_LIST
2
2
*SET_PART_LIST
3
3
*SET_PART_LIST
4
4
*CONTROL_FORMING_AUTOPOSITION_PARAMETER_SET
$  PID/SID       CID       DIR MPID/MSID  Position   PREMOVE     THICK
PARORDER
         1                   3         4         1                 1.5
blnkmv
         3                   3         1         1                 1.5
upbinmv
         2                   3         1         1                 1.5
uppunmv
```

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
-8
*PART_MOVE
$     PID             XMOV             YMOV             ZMOV      CID    IFSET
        1              0.0              0.0          &blnkmv                1
        3              0.0              0.0         &upbinmv                1
        2              0.0              0.0         &uppunmv                1
```

*PART_SENSOR

Purpose: Activate and deactivate parts, based on sensor defined in *ELEMENT_SEAT-BELT_SENSOR. This option applies to discrete beam element only.

Sensor Part Coupling Cards. Include as many of the following cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	SIDA	ACTIVE					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
PID	Part ID which is controlled by sensor
SIDA	Sensor ID to activate or deactivate part
ACTIVE	Part activation flag: EQ.0: part is active from time zero until a signal is received by the part to deactivate. EQ.1: part is inactive from time zero and becomes active when a signal is received by the part to activate. The history variables for inactive parts are initialized at time zero.

*PART_STACKED_ELEMENTS

Purpose: This keyword provides a method of defining a stacked element model for shell-like structures. These types of plane load-bearing components possess a thickness which is small compared to their other (in-plane) dimensions. Their physical properties vary in the thickness direction according to distinct layers. Application examples include sandwich plate systems, composite laminates, plywood, and laminated glass.

With this keyword it is possible to discretize layered structures by an arbitrary sequence of shell and/or solid elements over the thickness. Whether a physical ply should be discretized by shell or solid elements depends on the individual thickness and other mechanical properties. Every single layer can consist of one shell element over the thickness or one or several solid elements over thickness.

The stacked element mesh can either be provided directly or it be automatically generated by LS-DYNA itself. For automatic generation extrusion methods are used to determine the new node locations that characterize the out-of-plane geometry. Each layer gets its own predefined properties such as individual thickness, material characteristic, and element type.

A more detailed description of this feature including a detailed description of the appropriate mesh generation procedure is given in Erhart [2015].

Card 1	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	C							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	PIDREF	NUMLAY	ADPOPT	INPLCMP				
Type	I	I	I	I				
Default	none	none	0	0				

Layer Data Cards. The part ID, section ID, material ID, hourglass ID, thermal material ID, thickness, and number of through thickness solid elements for each layer *i* of a stacked element model are provided below. The layer data should be given sequentially starting with the bottommost layer. This card should be included NUMLAY times (one for each layer). The definitions in Card 3 replace the usual *PART cards.

Card 3	1	2	3	4	5	6	7	8
Variable	PID*i*	SID*i*	MID*i*	HGID*i*	TMID*i*	THK*i*	NSLD*i*	
Type	I	I	I	I	I	F	I	
Default	none	none	none	0	0	none	none	

VARIABLE	DESCRIPTION
HEADING	Heading for the part composition
PIDREF	Part ID of reference shell element mesh
NUMLAY	Number of layers
ADPOPT	Indicate if parts are adapted or not. (See also *CONTROL_ADAPTIVE): EQ.0: Inactive EQ.1: *h*-adaptive refinement
INPLCMP	Option for in-plane composed parts or to invoke a more robust search algorithm: EQ.0: Off EQ.1: On; see Remarks 6 and 7.
PID*i*	Part identification
SID*i*	Section identification for layer *i* defined in a *SECTION keyword
MID*i*	Material identification for layer *i* defined in a *MAT keyword
HGID*i*	Hourglass identification for layer *i* defined in a *HOURGLASS keyword

VARIABLE	DESCRIPTION
TMID*i*	Thermal material identification for layer *i* defined in a *MAT_-THERMAL keyword
THK*i*	Thickness of layer *i*
NSLD*i*	Number of through-thickness solid elements for layer *i*

Remarks:

1. **Provided versus automatically generated meshes.** In general, there are two different options for this keyword:

   a) The user provides a finished mesh comprising stacked shell and/or solid elements and then combines the corresponding part IDs using this keyword. This mode does not require a reference mesh in the PIDREF field nor does it require that either the layer thickness (THK*i* fields) and the the number of through-thickness solids elements (NSLD*i* fields) be specfied.

   b) The user may provide a shell reference mesh (PIDREF) together with the layup sequence. The stacked element mesh is automatically generated during the initialization phase of LS-DYNA. In that second case, layer thickness (THK*i*) and number of through-thickness solids (NSLD*i*) must be defined.

2. **Shell solid overlap.** In the mesh generation case, two consecutive layers (solid-solid or solid-shell) are firmly connected, meaning that they share nodes in the most obvious way possible (except they are both shell element layers, see Remark 3 for that case). This condition leads to the necessity that shell and solid elements partly overlap if they follow each other in the stacking sequence. This deficiency can be corrected afterwards by subsequent relocation of the shell mid-surfaces via NLOC on *SECTION_SHELL (only works for first or last layer in this stacked element approach) or appropriate adjustment of the material stiffness for the solid elements.

3. **Stacked shells.** Starting with the release of LS-DYNA version R10, it is possible to define shell element layers directly on top of each other (i.e. without solid elements in between). A potential connection/interaction of such layers has to be declared separately by additional contact definitions (standard, tied, or tie-break) otherwise they are free to penetrate each other.

4. **Chained calculations.** This keyword (*PART_STACKED_ELEMENTS) can also be used for modeling multi-stage processes. The *INTERFACE_SPRING-

BACK_LSDYNA card can be used (where PSID should contain all the parts PID*i*, not PIDREF) to save a final state including deformed geometry, stresses, and strains to a dynain file. In a subsequent calculation the *INCLUDE keyword can be used with that dynain to apply the layup sequence without regenerating the mesh. LS-DYNA automatically detects if a reference shell element mesh is present or not.

5. **Example.** The following input gives an example of specifying a three-layer shell-solid-shell structure:

```
*PART_STACKED_ELEMENTS
$ title
sandwich
$   pidref    numlay
        11         3
$#     pid       sid       mid      hgid      tmid       thk      nsld
       100       200         1         1         0      0.25         0
       101       201         2         0         0      0.60         3
       102       200         1         1         0      0.15         0
*SECTION_SHELL
$      sid    elform      shrf       nip     propt   qr/irid
       200         2     0.833       5.0       1.0       0.0
$       t1        t2        t3        t4
      0.25      0.25      0.25      0.25
*SECTION_SOLID
$      sid    elform
       201        -1
```

Shell elements (SID = 200) on the outer layers with part identifiers 100 and 102 discretize the sandwich structure. The interior of the "sandwich" consists of three solid elements (SID = 201, NSLD = 3) with part identifier 101. In this case, the reference shell mesh belongs to part 11 (PIDREF). The value of the field THK defines the thickness of each layer. It overwrites the thickness values from *SECTION_SHELL. Related materials (MID, TMID) and hourglass types (HGID) receive the usual treatment, so we do not show them here. .

6. **In-Plane composed parts.** In the case of a reference mesh made up of several parts with matching nodes at their boundaries (such as tailor welded blanks), define one *PART_STACKED_ELEMENTS keyword for each reference part. Each of these *PART_STACKED_ELEMENTS keywords must have INPLCMP set to 1. This feature requires merging the coincident nodes of the reference parts in a pre-processor before running the simulation. Additionally, the input deck must include *NODE_MERGE_SET for these newly generated parts so that they stay together after LS-DYNA generates the new mesh.

7. **Robust search algorithm.** The option INPLCMP = 1 also invokes a more robust search algorithm that can be used if the regular method fails, even if parts are not in-plane composed. For large models, this algorithm leads to some additional computational effort, meaning more CPU time.

8. **Boundary conditions.** All node sets (*SET_NODE) containing nodes of the reference shell element mesh are automatically converted to node sets that include the corresponding new nodes of the generated mesh. Therefore, it is possible to transfer boundary conditions of the reference mesh to the stacked model when corresponding keywords *BOUNDARY_... or *LOAD_... refer to such node sets.

*PERTURBATION

The keyword *PERTURBATION provides a means of defining deviations from the designed structure such as buckling imperfections. These perturbations can be viewed in LS-PrePost as user-defined fringe plots. Available options are:

*PERTURBATION_MAT

*PERTURBATION_NODE

*PERTURBATION_SHELL_THICKNESS

***PERTURBATION**_*OPTION*

Available options are:

MAT

NODE

SHELL_THICKNESS

Purpose: Define a perturbation (stochastic field) over the whole model or a portion of the model, typically to trigger an instability. The NODE option modifies the three-dimensional coordinates for the whole model or a node set. For the SHELL_THICKNESS option, the shell thicknesses are perturbed for the whole model or a shell set. The MAT option perturbs a material parameter value for all the elements associated with that material.

Card Summary:

Card 1a. This card is included if and only if the MAT option is used.

TYPE	PID	SCL	CMP	ICOORD	CID		

Card 1b. This card is included if and only if the NODE option is used.

TYPE	NSID	SCL	CMP	ICOORD	CID		

Card 1c. This card is used if and only if the SHELL_THICKNESS option is used.

TYPE	EID	SCL	CMP	ICOORD	CID		

Card 2a. This card is included if and only if TYPE = 1.

AMPL	XWL	XOFF	YWL	YOFF	ZWL	ZOFF	

Card 2b. This card is included if and only if TYPE = 2.

FADE							

Card 2c. This card is included if and only if TYPE = 3.

FNAME							

Card 2d. This card is included if and only if TYPE = 4.

CSTYPE	ELLIP1	ELLIP2	RND				

Card 2d.1. Depending upon the value of CSTYPE, include one, two, or three cards of this format.

CFTYPE	CFC1	CFC2	CFC3				

Card 2e. This card is included if and only if TYPE = 8.

AMPL	DTYPE						

Data Card Definitions:

Material Perturbation Card. Card 1 for MAT keyword option. Perturb a material parameter.

Card 1a	1	2	3	4	5	6	7	8
Variable	TYPE	PID	SCL	CMP	ICOORD	CID		
Type	I	I	F	I	I	I		
Default	1	0	1.0	5	0	0		

VARIABLE	DESCRIPTION
TYPE	Type of perturbation: EQ.1: Harmonic Field (see Remark 3) EQ.3: Read perturbations from a file EQ.4: Spectral field
PID	Part ID
SCL	Scale factor
CMP	Component. See Remark 10 and *MAT_238.
ICOORD	Coordinate system to use (see Remarks 7, 8 and 9): EQ.0: Global Cartesian EQ.1: Cartesian EQ.2: Cylindrical (computed and applied) EQ.3: Spherical (computed and applied)

VARIABLE	DESCRIPTION
	EQ.-2: Computed in cartesian but applied in cylindrical
	EQ.-3: Computed in cartesian but applied in spherical
CID	Coordinate system ID; see *DEFINE_COORDINATE_NODES.

Node Perturbation Card. Card 1 for NODE keyword option. Perturb the coordinates of a node set (or all nodes).

Card 1b	1	2	3	4	5	6	7	8
Variable	TYPE	NSID	SCL	CMP	ICOORD	CID		
Type	I	I	F	I	I	I		
Default	1	{all}	1.0	7	0	0		

VARIABLE	DESCRIPTION
TYPE	Type of perturbation:
	EQ.1: Harmonic Field (see Remark 3)
	EQ.2: Fade out all perturbations at this node set (see Remark 4)
	EQ.3: Read perturbations from a file
	EQ.4: Spectral field
	EQ.8: Random value from uniform distribution
NSID	Node set ID. Specify 0 to perturb all the nodes in the model.
SCL	Scale factor
CMP	Component as given below:
	EQ.1: x coordinate
	EQ.2: y coordinate
	EQ.3: z coordinate
	EQ.4: x and y coordinates
	EQ.5: y and z coordinates
	EQ.6: z and x coordinates

VARIABLE	DESCRIPTION
	EQ.7: x, y, and z coordinates
ICOORD	Coordinate system to use (see Remarks 7, 8 and 9): EQ.0: Global Cartesian EQ.1: Cartesian EQ.2: Cylindrical (computed and applied) EQ.3: Spherical (computed and applied) EQ.-2: Computed in cartesian but applied in cylindrical EQ.-3: Computed in cartesian but applied in spherical
CID	Coordinate system ID; see *DEFINE_COORDINATE_NODES

Shell Thickness Card. Card 1 for SHELL_THICKNESS keyword option. Perturb the thickness of a set of shells (or all shells).

Card 1c	1	2	3	4	5	6	7	8
Variable	TYPE	EID	SCL	CMP	ICOORD	CID		
Type	I	I	F	I	I	I		
Default	1	{all}	1.0	none	0	0		

VARIABLE	DESCRIPTION
TYPE	Type of perturbation: EQ.1: Harmonic Field (see Remark 3) EQ.2: Fade out all perturbations at this element set (see Remark 4) EQ.3: Read perturbations from a file EQ.4: Spectral field EQ.8: Random value from uniform distribution
EID	Element set ID. Specify 0 to perturb all the elements in the model.
SCL	Scale factor

VARIABLE	DESCRIPTION
CMP	Component as given below: EQ.1: x coordinate EQ.2: y coordinate EQ.3: z coordinate EQ.4: x and y coordinates EQ.5: y and z coordinates EQ.6: z and x coordinates EQ.7: x, y, and z coordinates
ICOORD	Coordinate system to use (see Remarks 7, 8 and 9): EQ.0: Global Cartesian EQ.1: Cartesian EQ.2: Cylindrical (computed and applied) EQ.3: Spherical (computed and applied) EQ.-2: Computed in cartesian but applied in cylindrical EQ.-3: Computed in cartesian but applied in spherical
CID	Coordinate system ID; see *DEFINE_COORDINATE_NODES.

Harmonic Perturbation Cards (TYPE = 1). Card format 2 for TYPE = 1. Include as many cards of the following card as necessary. The input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	AMPL	XWL	XOFF	YWL	YOFF	ZWL	ZOFF	
Type	F	F	F	F	F	F	F	
Default	1.0	0.0	0.0	0.0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
AMPL	Amplitude of the harmonic perturbation
XWL	x wavelength of the harmonic field

VARIABLE	DESCRIPTION
XOFF	*x* offset of harmonic field
YWL	*y* wavelength of the harmonic field
YOFF	*y* offset of harmonic field
ZWL	*z* wavelength of the harmonic field
ZOFF	*z* offset of harmonic field

Fade Field Perturbation Card (TYPE = 2). Card format 2 for TYPE = 2. See Remark 4.

Card 2b	1	2	3	4	5	6	7	8
Variable	FADE							
Type	F							
Default	1.0							

VARIABLE	DESCRIPTION
FADE	Parameter controlling the distance over which faded perturbations are faded (material perturbations are not faded).

Perturbation from File Card (TYPE = 3). Card format 2 for TYPE = 3.

Card 2c	1	2	3	4	5	6	7	8
Variable	FNAME							
Type	A							

VARIABLE	DESCRIPTION
FNAME	Name of file containing the perturbation definitions

Spectral Field Perturbation Card (TYPE = 4). Card format 2 for TYPE = 4 (fade field).

Card 2d	1	2	3	4	5	6	7	8
Variable	CSTYPE	ELLIP1	ELLIP2	RND				
Type	I	F	F	I				
Default	none	1.0	1.0	0				

VARIABLE	DESCRIPTION
CSTYPE	Correlation structure: EQ.1: 3D isotropic. The *x*, *y* and *z* correlations are described using one correlation function. Define one instance of Card 2d.1. EQ.2: 3D product. The *x*, *y* and *z* correlations are described using a correlation function each. Define three instances of Card 2d.1. EQ.3: 2D isotropic. A correlation function describes the *x* correlation while the *yz* isotropic relationship is described using another correlation function. Define two instances of Card 2d.1. EQ.4: 2D isotropic. The *xz* isotropic relationship is described using a correlation function, while another correlation function describes the *y* correlation function. Define two instances of Card 2d.1 . EQ.5: 2D isotropic. The *xy* isotropic relationship is described using a correlation function, while another correlation function describes the *z* correlation function. Define two instances of Card 2d.1. EQ.6: 3D elliptic. Define ELLIP1, ELLIP2, and one instance of Card 2d.1. EQ.7: 2D elliptic. A correlation function describes the *x* correlation while the *yz* elliptic relationship is described using another correlation function. Define ELLIP1 and two instances of Card 2d.1. EQ.8: 2D elliptic. A correlation function describes the *y* correlation, while the *zx* elliptic relationship is described using another correlation function. Define ELLIP1 and two

VARIABLE	DESCRIPTION
	instances of Card 2d.1. EQ.9: 2D elliptic. The *xy* elliptic relationship is described using a correlation function, while another correlation function describes the *z* correlation function. Define ELLIP1 and two instances of Card 2d.1.
ELLIP1	Elliptic constant for 2D and 3D elliptic fields
ELLIP2	Elliptic constant for 3D elliptic field
RND	Seed for random number generator. EQ.0: The solver generates a random seed. GT.0: Value to be used as seed

Spectral Perturbation Parameter Cards. Include one, two, or three cards of this format, depending on the value of CSTYPE.

Card 2d.1	1	2	3	4	5	6	7	8
Variable	CFTYPE	CFC1	CFC2	CFC3				
Type	I	F	F	F				
Default	none	1.0	1.0	1.0				

VARIABLE	DESCRIPTION
CFTYPE	Correlation function (see Remark 6) EQ.1: Gaussian EQ.2: Exponential EQ.3: Exponential Cosine EQ.4: Rational EQ.5: Linear
CFC*i*	Correlation function constant *i*. The number of constants varies with CFTYPE. See Remark 6.

Random Value Perturbation Card (TYPE = 8). Card format 2 for TYPE = 8.

Card 2e	1	2	3	4	5	6	7	8
Variable	AMPL	DTYPE						
Type	F	F						
Default	1.0	0.0						

VARIABLE	DESCRIPTION
AMPL	Amplitude of the random perturbation
DTYPE	Distribution type: EQ.0.0: Uniform distribution between SCL × [0, AMPL] EQ.1.0: Uniform distribution between SCL × [−AMPL, AMPL]

Remarks:

1. **Post-processing.** The perturbation can be viewed in LS-PrePost. For the NODE option, files named pert_node_x/y/z/res are created, which can be viewed as user-defined fringe plots. For the SHELL_THICKNESS and MAT options, the files are named pert_shell_thickness and pert_mat, respectively. If a coordinate system with a radial component is used, then the file pert_node_radial is also written.

2. **Linear combinations and maximum amplitudes.** Perturbations specified using separate *PERTURBATION cards are created separately and then added together. This is true as well for special cases, such as CMP = 7 in which case the x, y and z fields are created separately and added together afterwards, which can result in an absolute amplitude greater than specified using AMPL or SCL.

3. **Harmonic perturbations.** The harmonic perturbation is

$$p_{\mathrm{CMP}}(x,y,z) = \mathrm{SCL} \times \mathrm{AMPL} \times \left[\sin\left(2\pi\frac{x+\mathrm{XOFF}}{\mathrm{XWL}}\right) + \sin\left(2\pi\frac{y+\mathrm{YOFF}}{\mathrm{YWL}}\right) + \sin\left(2\pi\frac{z+\mathrm{ZOFF}}{\mathrm{ZWL}}\right)\right]$$

Note that the harmonic perturbations can sum to values greater than SCL × AMPL.

4. **Fade perturbation.** The fade perturbation is

$$p'(x,y,z) = \text{SCL} \times \left(1 - \frac{1}{e^{-\frac{\ln 0.05}{\text{FADE}} \times x'}}\right) p(x,y,z)$$

where x' is the shortest distance to a node in the node set or element set specified, and FADE is the parameter controlling the sharpness of the fade perturbation.

5. **Keyword format for FNAME.** The file FNAME must contain the perturbation in keyword format. This file can be created from d3plot results using the LS-PrePost output capability. The data must be arranged into two columns with the first column being the node IDs. Lines starting with the character $ will be ignored.

6. **Correlation functions.** The correlation functions are defined as follows:

   a) Gaussian: $B(t) = e^{-(at)^2}$

   b) Exponential: $B(t) = e^{-|at|^b}$

   c) Exponent and Cosine: $B(t) = e^{-|at|}\cos(bt)$

   d) Rational: $B(t) = (1 + |at|^b)^{-c}$

   e) Piecewise Linear: $B(t) = (1 - |at|)\chi(1 - |at|)$

   with χ the Heaviside step function and a, b and c corresponding to CFC1, CFC2, and CFC3, respectively.

7. **Cylindrical coordinates.** For the cylindrical coordinate system option (ICOORD = 2), the default is to use the global coordinate system for the location of the cylindrical part, with the base of the cylinder located at the origin, and the global z-axis aligned with the cylinder axis. For cylindrical parts not located at the global origin, define a coordinate system (numbered CID) using *DEFINE_COORDINATE_NODES by selecting any three nodes on the base of the cylinder in a clockwise direction (resulting in the local z-axis to be aligned with the cylinder).

8. **Spherical coordinates.** For the spherical coordinate system (ICOORD = 3), the coordinates are the radius, zenith angle $[0,\pi]$, and the azimuth angle $[0,2\pi]$. The default is to use the global coordinate system with the zenith measured from the z-axis and the azimuth measured from the x-axis in the xy-plane. For spherical parts not located at the global origin, define a coordinate system using *DEFINE_COORDINATE_NODES by selecting any three nodes as follows: the first node is the center of the sphere, the second specifies the x-axis of the coordinate

system, while the third point specifies the plane containing the new y-axis. The z-axis will be normal to this plane.

9. **Computed in cartesian applied to cylindrical or spherical.** It is possible to compute the perturbations in a Cartesian coordinate system, but to apply them in a cylindrical or spherical coordinate system (ICOORD = -2, -3). This is the natural method of doing say a radial perturbation of a sphere using a spectral perturbation field. We expect that computing the perturbation in the spherical coordinate system should be rare (ICOORD = 3). Computing a perturbation in a cylindrical coordinate system should be common though; for example, a circumferential harmonic perturbation.

10. **Material perturbation feature.** Only *MAT_238 (*MAT_PERT_PIECEWISE_LINEAR_PLASTICITY) and solid elements in an explicit analysis can be perturbed using *PERTURBATION_MAT. See the documentation of this material for allowable components. Only one part per model can be perturbed. For some perturbed quantity c, the material perturbation is applied on an element-by-element basis as

$$c_{\text{new}} = (1 + p)c_{\text{base}}$$

where p is a random number, which is written to the pert_mat file during the calculation. Values of p less than -1 are not allowed because the material behavior is not defined.

Completely independent of *PERTURBATION_MAT, see *DEFINE_STOCHASTIC_VARIATION for a way to define a stochastic variation of yield stress and/or failure strain in material models 10, 15, 24, 81, and 98 and the shell version of material 123.

*RAIL

Two keywords are defined in this section.

*RAIL_TRACK

*RAIL_TRAIN

*RAIL_TRACK

Purpose: Wheel-rail contact algorithm intended for railway applications but can also be used for other purposes. The wheel nodes (defined on *RAIL_TRAIN) represent the contact patch between wheel and rail. A penalty method is used to constrain the wheel nodes to slide along the track. A track consists of two rails, each of which is defined by a set of beam elements.

Card Sets. For each track include one pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	BSETID1	NORGN1	LCUR1	OSET1	SF1	GA1	IDIR
Type	I	I	I	I	F	F	F	I
Default	none	none	none	↓	0.0	1.0	0.0	0

Card 2	1	2	3	4	5	6	7	8
Variable		BSETID2	NORGN2	LCUR2	OSET2	SF2	GA2	
Type		I	I	I	F	F	F	
Default		none	none	↓	0.0	1.0	0.0	

VARIABLE	DESCRIPTION
ID	Track ID
BSETID1,2	Beam set ID for rails 1 and 2 containing all beam elements that make up the rail; see *SET_BEAM.
NORGN1,2	Reference node at one end of each rail, used as the origin for the roughness curve. The train will move in a direction away from this node.
LCUR1,2	Load curve ID (see *DEFINE_CURVE) defining track roughness (vertical displacement from line of beam elements) of the rail as a function of distance from the reference node NORIGIN. Distance

VARIABLE	DESCRIPTION
	from reference node is the x-axis of the curve while roughness is on the y-axis. Default: no roughness.
OSET1,2	Origin of curve LCUR is shifted by distance OSET towards the reference node.
SF1,2	Roughness values are scaled by SF. Default: 1.0.
GA1,2	Shear stiffness of rail per unit length (used to calculate local rail shear deformation within each beam element). GA = shear modulus × cross-sectional area. Default: local shear deformation is ignored.
IDIR	Contact forces are calculated in local directions relative to the plane containing the two rails at the contact point. IDIR determines which side of the plane is "up", that is, the direction in which the wheel can lift off the rail. "Up" is either **c** or $-$**c**,where **c** = **a** × **b**. **a** is the direction along rail 1 heading away from node NORGN1 and **b** is the vector from rail 1 to rail 2. Both **a** and **b** are determined locally. EQ.0: Whichever out of **c** or $-$**c** has a positive global Z component is "up" (default). EQ.1: $-$**c** is "up". EQ.-1: **c** is "up".

Remarks:

*RAIL_TRACK and *RAIL_TRAIN were written by Arup to represent wheel-rail contact. They have been used to generate loading on models of bridges for vibration predictions, for stress calculations, and for estimating accelerations experienced by passengers. Other non-railway uses are possible: the algorithm causes the "train" nodes to follow the line defined by the "rail" beam elements and transfers forces between them. In some cases (especially vibration modeling), double precision versions of LS-DYNA may give superior results because of the small relative deflections between wheel and rail.

Track Modeling:

The rails of the track should be modeled by two parallel lines of beam elements. The track can be curved or straight and the rails can be modeled as deformable or rigid. If required, rail pads, sleepers and ballast may also be modeled – typically with spring, damper and beam elements. It is also possible to use this algorithm to control the motion

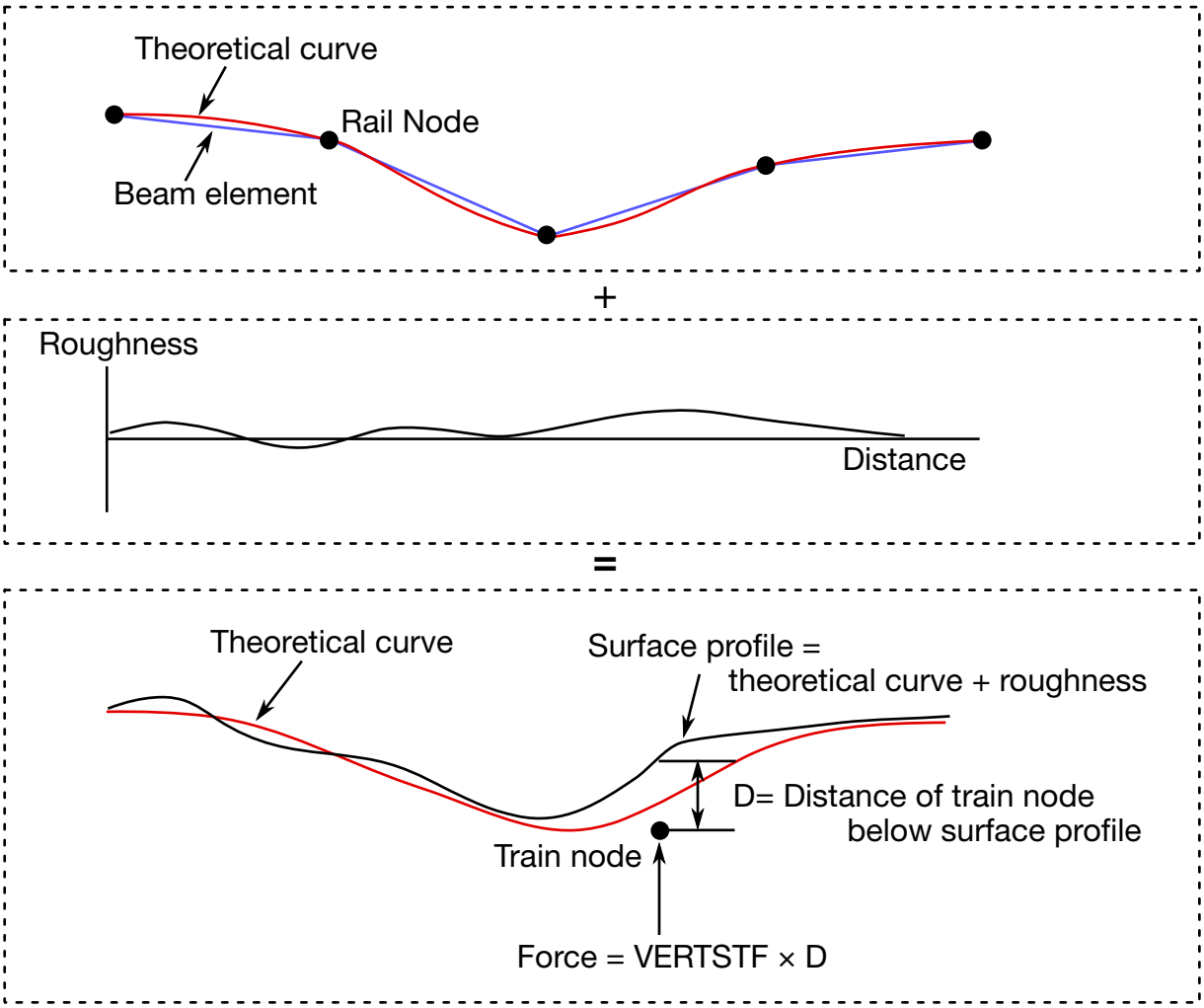


Figure 39-1. Track Model

of simple road vehicle models: beam element "rails" made of null material can be embedded in the road surface. It is recommended that the mesh size of the two rails should be similar: LS-DYNA calculates a local coordinate system for each train node based on the alignment of the currently contacted beam element and the nearest node on the other rail.

Because wheel-rail contact stiffness is generally very high, and wheel masses are large, small deviations from a straight line or smooth curve can lead to large transient forces. It is recommended that great care be taken in generating and checking the geometry for the track, especially where the track is curved. Some pre-processors write the coordinates with insufficient precision to the LS-DYNA input file which can cause unintended roughness in the geometry. For the same reason, if the line of the track were taken as straight between nodes, spurious forces would be generated when the wheel passes from one rail element to the next. This is avoided because the *RAIL algorithm calculates a theoretical curved centerline for the rail element to achieve continuity of slope from one element to the next. Where the length of the rail elements is similar to or shorter than the maximum section dimension, shear deformation may be significant, and it is possible to include this

in the theoretical centerline calculation to further reduce spurious forces at the element boundaries (inputs GA1, GA2).

Roughness (small deviations in the vertical profile from a perfect straight line) does exist in real life and is a principal source of vibration. *RAIL allows the roughness to be modeled by a load curve giving the vertical deviation (in length units) of the rail surface from the theoretical centerline of the beam elements as a function of distance along the track from the origin node of the rail. The roughness curve is optional. Ideally, roughness profiles measured from both rails of the same piece of track should be used so that the relationship between bump and roll modes is correctly captured.

Whether roughness is included or not, it is important to select as the origin nodes (NORIGIN1 and NORIGIN2) the nodes at the end of the rails away from which the train will be traveling. The train can start at any point along the rails but must travel away from the origin nodes.

Train Modeling:

The vehicles are typically modeled using spring, damper and rigid elements, or simply a point mass at each wheel position. Each node in the set referred to on *RAIL_TRAIN represents the contact patch of one wheel (note: not the center of the wheel). These nodes should be initially on or near the line defined by either of the two rails. LS-DYNA will move the train nodes initially onto the rails to achieve the correct initial wheel-rail forces. If the results are viewed with magnified displacements, the initial movements can appear surprising.

Wheel roughness input is available. This will be applied in addition to track roughness. The input curve must continue for the total rolled distance – it is not assumed to repeat with each wheel rotation. This is to avoid problems associated with ensuring continuity between the start and end of the profile around the wheel circumference, especially since the profiles might be generated from roughness spectra rather than taken directly from measured data.

Wheel-Rail Interface:

The wheel-rail interface model is a simple penalty function designed to ensure that the train nodes follow the line of the track. It does not attempt to account for the shape of the rail profile. Vertical and lateral loads are treated independently. For this reason, the algorithm is not suitable for rail vehicle dynamics calculations.

Wheel-rail contact stiffness is input on *RAIL_TRAIN. “Vertical” means normal to the plane containing the two rails at the contact point. “Horizontal” means the direction in the plane of the two rails perpendicular to the contacted rail. Vertical contact works like a normal penalty-based contact. A linear force-deflection relationship is assumed in

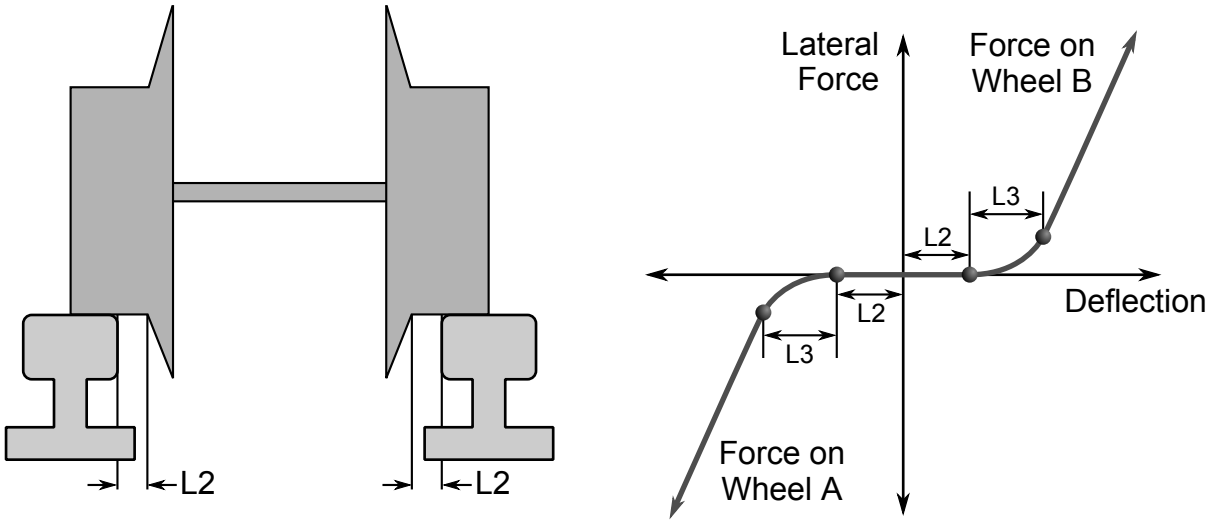


Figure 39-2. Illustration of lateral drift parameters L2 and L3 from *RAIL_-TRAIN. Diagram shows configuration for LATDIR = 0.

compression (see VERTSTF on *RAIL_TRAIN); no tensile force is generated if the train wheel lifts up off the rail. The direction treated as "up" is controlled by IDIR. Typical contact stiffness is 2 MN/mm. Lateral deflections away from the theoretical centerline of the rail beams are penalized by a linear stiffness LATSTF (see *RAIL_TRAIN).

Optionally, a "gap" can be defined in input parameter L2 such that the wheel-set can drift laterally by L2 length units before any lateral force is generated. A further option is to allow smooth transition between "gap" and "contact" by means of a transition distance input as parameter L3. Figure 39-2 illustrates the geometry of parameters L2 and L3. A further option is FRIC on *RAIL_TRAIN. The friction force resists lateral motion of the train wheels relative to the track and is applied along with the forces calculated from LATSTF, L2 and L3 described above.

Generally, with straight tracks a simple linear stiffness is sufficient. With curved tracks, a reasonable gap and transition distance should be defined to avoid unrealistic forces being generated in response to small inaccuracies in the distance between the rails.

Gravity loading is expected, in order to maintain contact between rail and wheel. This is normally applied by an initial phase of dynamic relaxation. To help achieve convergence quickly, or in some cases avoid the need for dynamic relaxation altogether, the initial force expected on each train node can be input (field FINIT on *RAIL_TRAIN). LS-DYNA positions the nodes initially such that the vertical contact force will be FINIT at each node. If the suspension of the rail vehicles is modeled, it is recommended that the input includes pre-compression of the spring elements; if this is not done, the train model may take a long time to reach equilibrium under gravity loading.

The *RAIL algorithm ensures that the train follows the rails but does not provide forward motion. This is generally applied using *INITIAL_VELOCITY, or for straight tracks, *BOUNDARY_PRESCRIBED_MOTION.

Output:

LS-DYNA generates an additional output file train_force.csv, containing force time-histories for each train node, output at the same time intervals as the binary time history file (DT on *DATABASE_BINARY_D3THDT). R12 and previous versions of LS-DYNA generated a file in a different ASCII format, train_force_*n*, where *n* is an integer. For backward compatibility of post-processing tools that format is still available by setting FMT = 1 on *RAIL_TRAIN.

Checking:

We recommend testing the track and train models separately before adding the *RAIL cards. You should check that the models respond stably to impulse forces and that they achieve equilibrium under gravity loading. Most problems we have encountered have been due to unstable behavior of train or track. Often, these are first detected by the *RAIL algorithm, and an error message will result.

*RAIL_TRAIN

Purpose: Define train properties. A train is defined by a set of nodes in contact with a rail defined by *RAIL_TRACK. See description under *RAIL_TRACK.

Card Sets. For each train include one pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	NSETID	FMT	FINIT	VGAP	TRID	LCUR	OFFS
Type	I	I	F	F	F	I	I	F
Default	none	none	0.0	0.0	0.0	0	↓	0.0

Card 2	1	2	3	4	5	6	7	8
Variable	VERTSTF	LATSTF	V2	V3	L2	L3	LATDIR	FRIC
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
ID	Train ID
NSETID	Node set ID containing all nodes that are in contact with rails
FMT	Format of output file containing force time-histories for all train nodes: EQ.0.0: CSV format EQ.1.0: ASCII file, same as R12 and previous versions
FINIT	Estimate of initial vertical force on each wheel (optional) – speeds up the process of initial settling down under gravity loading.
VGAP	Initial gap between wheel and track. If VGAP is defined, FINIT must be zero.

VARIABLE	DESCRIPTION
TRID	ID of track for this train (see *RAIL_TRACK)
LCUR	Load curve ID (see *DEFINE_CURVE) containing wheel roughness (distance of wheel surface away from perfect circle) as a function of distance traveled. The curve does not repeat with each rotation of the wheel – the last point should be at a greater distance than the train is expected to travel. Default: no wheel roughness.
OFFS	Offset distance used to generate different roughness curves for each wheel from the roughness curve LCUR. The curve is offset on the x-axis by a different whole number multiple of OFFS for each wheel.
VERTSTF	Vertical stiffness of rail contact
LATSTF	Lateral stiffness of rail contact
V2, V3	Unused variables – leave blank.
L2	Lateral clearance from rail to wheel flange (see Figure 39-2 of *RAIL_TRACK). Lateral force is applied to a wheel only when it has moved more than L2 laterally relative to the rail in the direction determined by LATDIR.
L3	Further lateral distance before full lateral stiffness applies (force-deflection curve follows a parabola up to this point). See Figure 39-2 of *RAIL_TRACK.
LATDIR	Determines the lateral direction (relative to the track) in which wheel movement is resisted by flange contact. If two wheels are fixed to an axle, lateral force is generally applied to one or other of the two wheels, depending on the direction of lateral movement. EQ.0.0: Wheel flanges run on inside faces of rails EQ.1.0: Wheel flanges run on outside faces of rails EQ.2.0: Wheel flanges on both faces of rails (both wheels resist lateral motion in both directions).
FRIC	Coefficient for additional friction force resisting lateral motion of wheel relative to rail.

*RIGIDWALL

Two keywords are used in this section to define rigid surfaces:

*RIGIDWALL_GEOMETRIC*_OPTION_{OPTION}_{OPTION}}_{OPTION}*

*RIGIDWALL_PLANAR*_{OPTION}_{OPTION}_{OPTION}*

The RIGIDWALL option provides a simple way of treating contact between a rigid surface and nodal points of a deformable body, called tracked nodes. Tracked nodes which belong to rigid parts are not, in general, checked for contact with only one exception. The RIGIDWALL_PLANAR option may be used with nodal points of rigid bodies if the planar wall defined by this option is fixed in space and the RWPNAL parameter is set to a positive nonzero value on the control card, *CONTROL_CONTACT.

When the rigid wall defined in this section moves with a prescribed motion, the equations of rigid body mechanics are not involved. For a general rigid body treatment with arbitrary surfaces and motion, refer to the *CONTACT_ENTITY definition. The *CONTACT_ENTITY option is for treating contact between rigid and deformable surfaces only.

Energy dissipated due to rigidwalls (sometimes called stonewall energy or rigid wall energy) is computed only if the parameter RWEN is set to 2 in *CONTROL_ENERGY.

The following keyword causes the forces acting on a rigid wall to be computed:

*RIGIDWALL_FORCE_TRANSDUCER

*RIGIDWALL_FORCE_TRANSDUCER

Purpose: Define a force transducer for a rigid wall. The output of the transducer is written to the rwforc file.

Card 1	1	2	3	4	5	6	7	8
Variable	TID	RWID						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
TID	Transducer ID.
RWID	Rigid wall ID.

Card 2	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	C							
Default	none							
Remarks	1							

VARIABLE	DESCRIPTION
HEADING	Description for force transducer

Node Set Cards. For each node set add one card. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	NSID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
NSID	Node set ID.

Remarks:

1. **Reporting for Node Set.** The forces acting on rigid wall RWID are reported separately for each NSID.

2. **Segment Option for Rigid Walls.** For rigid walls using the segment option, the forces acting on each segment are reported separately for each NSID.

***RIGIDWALL_GEOMETRIC**_*SHAPE_{OPTION}_{OPTION}_{OPTION}*

RIGIDWALL_GEOMETRIC_*SHAPE is a family of keywords all sharing a common set of data cards and option flags. The available shape variants are:

*RIGIDWALL_GEOMETRIC_FLAT

*RIGIDWALL_GEOMETRIC_PRISM

*RIGIDWALL_GEOMETRIC_CYLINDER

*RIGIDWALL_GEOMETRIC_SPHERE

If prescribed motion is desired an additional option is available:

MOTION

One of the shape types **[FLAT, PRISM, CYLINDER, SPHERE]** must be specified, followed by the optional definition of **MOTION**, both on the same line with ***RIGIDWALL_GEOMETRIC.** If an ID number is specified, the additional option is available:

ID

If active, the ID card is the first card following the keyword. To view the rigid wall, the option:

DISPLAY

is available. With this option a rigid body is automatically defined which represents the shape, the physical position of the wall, and follows the walls motion if the MOTION option is active. Additional input is optional if DISPLAY is active.

For the CYLINDER and SPHERE, the option:

INTERIOR

is available. Nodes are confined to the interior of these geometric forms.

For the CYLINDER, the option:

DEFORM

is available. With this option you can rotate and change the shape of the cylinder.

The order of the *OPTIONS* is arbitrary, that is, LS-DYNA will read *RIGIDWALL_GEOMETRIC_*SHAPE*_MOTION_DISPLAY the same as *RIGIDWALL_GEOMETRIC_*SHAPE*_DISPLAY_MOTION. However, the data cards have a strict order as indicated in the Card Summary.

Purpose: Define a rigid wall with an analytically described form. Four forms are possible. A prescribed motion is optional. For general rigid bodies with arbitrary surfaces and motion, refer to the *CONTACT_ENTITY definition. This option is for treating contact between rigid and deformable surfaces only.

Card Summary:

Card Sets. For each rigid wall include one set of the following data cards. This input ends at the next keyword ("*") card.

Card ID. First data card of the card set if the ID keyword option is used in the keyword name. Otherwise Card 1 is the first data card.

RWID	HEADING

Card 1. This card is required.

NSID	NSIDEX	BOXID	BIRTH	DEATH			

Card 2. This card is required.

XT	YT	ZT	XH	YH	ZH	FRIC	

Card 3a. This card is included only for the FLAT shape.

XHEV	YHEV	ZHEV	LENL	LENM			

Card 3b. This card is included only for the PRISM shape.

XHEV	YHEV	ZHEV	LENL	LENM	LENP		

Card 3c. This card is included only for the CYLINDER shape.

RADCYL	LENCYL	NSEGS					

Card 3c.1. NSEGS instances of this card are included only for the CYLINDER shape.

VL	HEIGHT						

Card 3c.2. This card is included only for the CYLINDER shape with DEFORM option.

XP	YP	ZP	NL	NARC	NR		

Card 3c.3. This card is included only for the CYLINDER shape with DEFORM option.

LCIDR	LCIDA	LCIDB	LCIDG				

Card 3d. This card is included only for the SPHERE shape.

RADSPH							

Card 4. This card is required if the MOTION keyword option is used.

LCID	OPT	VX	VY	VZ			

Card 5. This card is read only if the DISPLAY keyword option is used. It is optional and may be omitted unless more than one card set is being read in; if more than one card set is being used, then at least a blank line must be included for all, but the last card set. If not input, the defaults will be used.

PID	RO	E	PR				

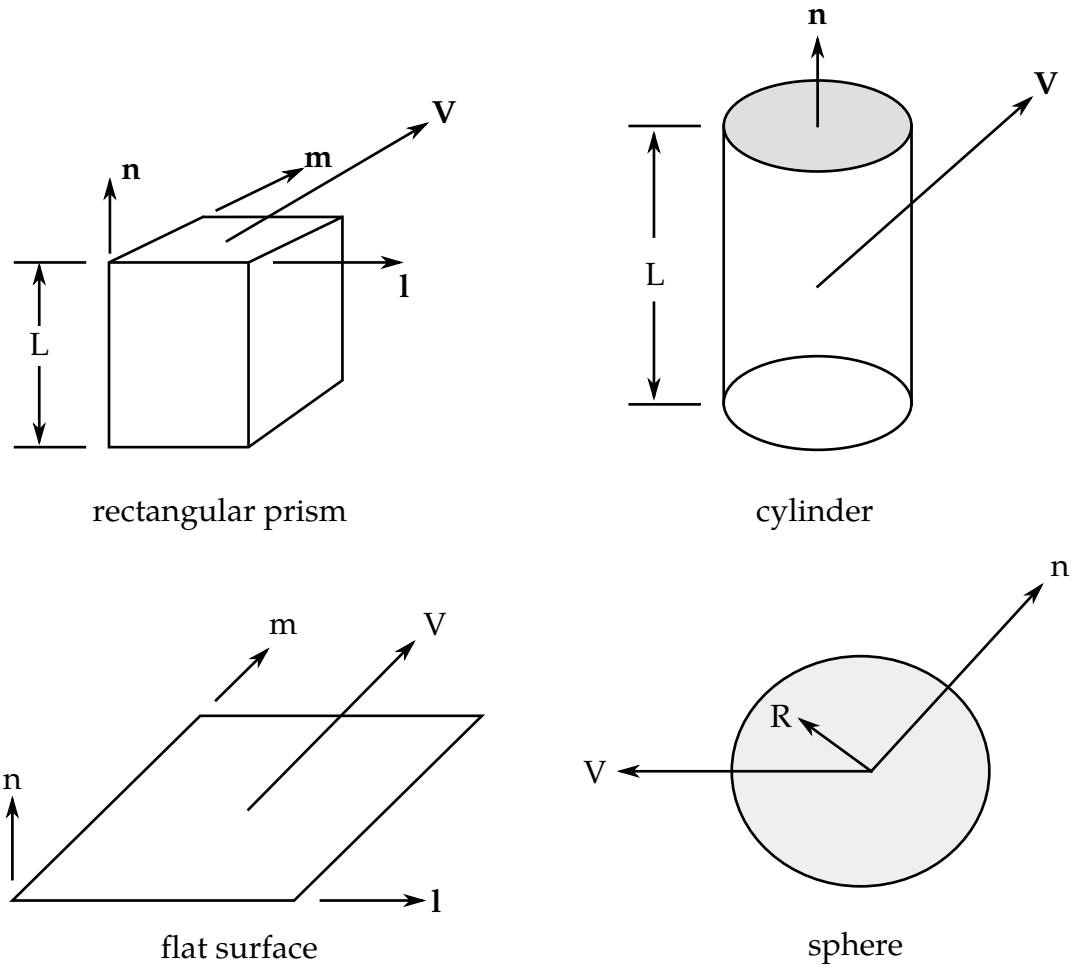


Figure 40-1. Vector **n** determines the orientation of the rigidwall. By including the MOTION option, motion of the rigidwall can be prescribed in any direction **V** as defined by variables VX, VY, VZ.

Data Card Definitions:

ID Card. Additional card for ID keyword option. This heading is picked up by some of the peripheral LS-DYNA codes to aid in post-processing.

Card ID	1	2	3	4	5	6	7	8
Variable	RWID	HEADING						
Type	I	A70						

VARIABLE	DESCRIPTION
RWID	Rigid wall ID. This must be a unique number.
HEADING	Rigid wall descriptor. It is suggested that unique descriptions be used. This field can be left undefined.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	NSIDEX	BOXID	BIRTH	DEATH			
Type	I	I	I	F	F			
Default	{all}	{∅}	{∅}	0.	10^{20}			

VARIABLE	DESCRIPTION
NSID	Nodal set ID containing tracked nodes; see *SET_NODE_*OPTION*: EQ.0: All nodes are tracked with respect to the rigid wall.
NSIDEX	Nodal set ID containing nodes exempted as tracked nodes; see *SET_NODE_*OPTION*.
BOXID	If defined, only nodes in box are included as tracked nodes for the rigid wall.
BIRTH	Birth time of rigid wall. The time values of the load curves that control the motion of the wall are offset by the birth time.

VARIABLE	DESCRIPTION
DEATH	Death time of rigid wall. At this time the wall is deleted from the calculation. If dynamic relaxation is active at the beginning of the calculation and if BIRTH = 0.0, the death time is ignored during the dynamic relaxation.

Card 2	1	2	3	4	5	6	7	8
Variable	XT	YT	ZT	XH	YH	ZH	FRIC	
Type	F	F	F	F	F	F	F	
Default	0.	0.	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
XT	x-coordinate of tail of any outward drawn normal vector, **n**, originating on wall (tail) and terminating in space (head); see Figure 40-1.
YT	y-coordinate of tail of normal vector **n**
ZT	z-coordinate of tail of normal vector **n**
XH	x-coordinate of head of normal vector **n**
YH	y-coordinate of head of normal vector **n**
ZH	z-coordinate of head of normal vector **n**
FRIC	Coulomb friction coefficient, except as noted below: EQ.0.0: Frictionless sliding when in contact EQ.1.0: No sliding when in contact

Flat Rigidwall Card. Card 3 for FLAT keyword option. A plane with a finite size or with an infinite size can be defined; see Figure 40-1. The vector **m** is computed as the vector cross product **n** × **l**. The origin, which is the tail (the start) of the normal vector, is the corner point of the finite size plane.

Card 3a	1	2	3	4	5	6	7	8
Variable	XHEV	YHEV	ZHEV	LENL	LENM			
Type	F	F	F	F	F			
Default	0.	0.	0.	infinity	infinity			

VARIABLE	DESCRIPTION
XHEV	x-coordinate of head of edge vector **l**; see Figure 40-1.
YHEV	y-coordinate of head of edge vector **l**
ZHEV	z-coordinate of head of edge vector **l**
LENL	Length of **l** edge. A zero value defines an infinite size plane.
LENM	Length of **m** edge. A zero value defines an infinite size plane.

Prismatic Rigidwall Card. Required card for PRISM keyword option that is input after Card 2. The description of the definition of a plane with finite size is enhanced by an additional length in the direction negative to **n**; see Figure 40-1.

Card 3b	1	2	3	4	5	6	7	8
Variable	XHEV	YHEV	ZHEV	LENL	LENM	LENP		
Type	F	F	F	F	F	F		
Default	none	0.	0.	infinity	infinity	infinity		

VARIABLE	DESCRIPTION
XHEV	x-coordinate of head of edge vector **l**, see Figure 40-1.
YHEV	y-coordinate of head of edge vector **l**

VARIABLE	DESCRIPTION
ZHEV	z-coordinate of head of edge vector **l**
LENL	Length of **l** edge. A zero value defines an infinite size plane.
LENM	Length of **m** edge. A zero value defines an infinite size plane.
LENP	Length of prism in the direction negative to **n**; see Figure 40-1.

Cylindrical Rigidwall Card. Required card for CYLINDER keyword option that is input after Card 2. The tail of **n** specifies the top plane of the cylinder. The length is defined in the direction negative to **n**. See Figure 40-1.

Card 3c	1	2	3	4	5	6	7	8
Variable	RADCYL	LENCYL	NSEGS					
Type	F	F	I					
Default	none	infinity	0					

VARIABLE	DESCRIPTION
RADCYL	Radius of cylinder
LENCYL	Length of cylinder; see Figure 40-1. Only if a value larger than zero is specified is a finite length assumed. For the DEFORM keyword option, the length of the cylinder must be finite.
NSEGS	Number of subsections along cylinder to output forces for post-processing. The force vector for each subsection is output to rwforc. This gives a better idea of the force distribution along the length of the cylinder.

NSEGS Card. For the CYLINDER option, NSEGS of this card must be input after Card 3c.

Card 3c.1	1	2	3	4	5	6	7	8
Variable	VL	HEIGHT						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
VL	Distance from the top plane of the cylinder where the subsection begins.
HEIGHT	Section height. The subsection starts at VL and extends the length, HEIGHT, in the *negative* **n**-direction.

DEFORM Card 1. If the DEFORM option is used for the CYLINDER shape, this card must be input after Card 3c.1.

Card 3c.2	1	2	3	4	5	6	7	8
Variable	XP	YP	ZP	NL	NARC	NR		
Type	F	F	F	I	I	I		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
XP, YP, ZP	Coordinates of a point in the local *xz*-plane of the local coordinate system for cylinder. See Remark 1.
NL	Number of auto-generated elements in the longitudinal direction. If DISPLAY option is not used, NL will be ignored. See Remark 2.
NARC	Number of auto-generated elements in the circumferential direction. If DISPLAY option is not used, NARC will be ignored.

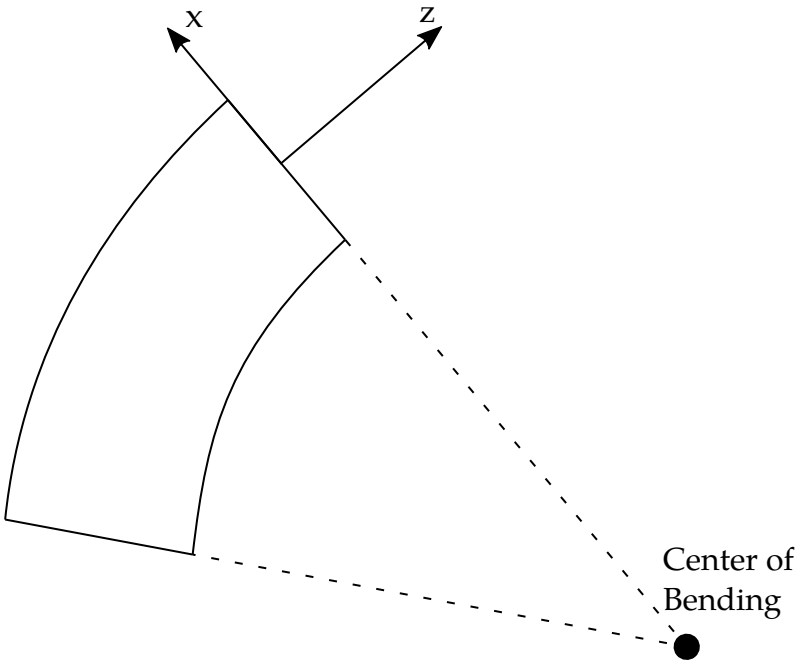


Figure 40-2. Example of cylinder bending for DEFORM keyword option

VARIABLE	DESCRIPTION
NR	Number of auto-generated elements in the radius direction. If DISPLAY option is not used, NR will be ignored.

DEFORM Card 2. If the DEFORM option is used for the CYLINDER shape, this card must be input after Card 3c.2.

Card 3c.3	1	2	3	4	5	6	7	8
Variable	LCIDR	LCIDA	LCIDB	LCIDG				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
LCIDR	Load curve ID to describe the change of the radius over time
LCIDA	Load curve ID to specify the change of the rotation in radians about the local *x*-axis over time.

VARIABLE	DESCRIPTION
LCIDB	Load curve ID to describe the change of the bending curvature over time. Bending occurs in the local *xz*-plane with the center of the bending lying on the negative side of the local *x*-axis. See Figure 40-2.
LCIDG	Load curve ID to describe the change of the rotation in radians about the local *z*-axis over time.

Card 3d	1	2	3	4	5	6	7	8
Variable	RADSPH							
Type	F							
Default	0.							

VARIABLE	DESCRIPTION
RADSPH	Radius of sphere

Motion Card. Additional card for MOTION keyword option.

Card 4	1	2	3	4	5	6	7	8
Variable	LCID	OPT	VX	VY	VZ			
Type	I	I	F	F	F			
Default	none	0	none	none	none			

VARIABLE	DESCRIPTION
LCID	Rigidwall motion curve ID; see *DEFINE_CURVE.
OPT	Type of motion: EQ.0: velocity specified, EQ.1: displacement specified.

VARIABLE	DESCRIPTION
VX	x-direction cosine of velocity/displacement vector
VY	y-direction cosine of velocity/displacement vector
VZ	z-direction cosine of velocity/displacement vector

Display Card. Optional card for DISPLAY keyword option. If this card is omitted, default values are set. The values set here have no effect on the solution other than the PID appearing in the post-processing.

Card 5	1	2	3	4	5	6	7	8
Variable	PID	RO	E	PR				
Type	I	I	I	F				
Default	↓	10^{-9}	10^{-4}	0.3				

VARIABLE	DESCRIPTION
PID	Unique part ID for moving geometric rigid wall. If zero or left empty for default behavior, a part ID will be set that is larger than the maximum of all user defined part IDs.
RO	Density of rigid wall. The default is set to 10^{-9}.
E	Young's modulus. The default is set to 10^{-4}.
PR	Poisson's ratio. The default is set to 0.30.

Remarks:

1. **Local coordinate system of the cylinder.** When keyword option DEFORM is used, the deformations of the cylinder are based upon a local coordinate system. The origin of the coordinate system is (XT,YT,ZT). The local z-direction is aligned with vector **n**. The cross product of vector **n** with the vector that has its the tail at the origin and head at (XP,YP,ZP) gives the local y-axis. The direction of the local x-axis is the cross product of a vector on the local y-axis and vector **n**.

2. **Visualization of the cylinder with DEFORM option**. Elements generated by defining NL, NARC and NR are only for visualization. These elements are not used in analysis.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *RIGIDWALL_GEOMETRIC_SPHERE_MOTION
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a rigid sphere:
$     - with a radius of 8
$     - centered at (x,y,z) = (20,20,9)
$     - that moves in the negative z-direction with a specified displacement
$          given by a load curve (load curve: lcid = 5)
$     - which prevents all nodes within a specified box from penetrating the
$          sphere (box number: boxid = 3), these nodes can slide on the sphere
$          without friction
$
*RIGIDWALL_GEOMETRIC_SPHERE_MOTION
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid    nsidex     boxid
                             3
$
$       xt        yt        zt        xh        yh        zh      fric
      20.0      20.0       9.0      20.0      20.0       0.0       0.0
$
$   radsph
       8.0
$
$     lcid       opt        vx        vy        vz
         5         1       0.0       0.0      -1.0
$
$
*DEFINE_BOX
$    boxid       xmn       xmx       ymn       ymx       zmn       zmx
         3       0.0      40.0       0.0      40.0      -1.0       1.0
$
$
*DEFINE_CURVE
$     lcid      sidr      scla      sclo      offa      offo
         5
$           abscissa            ordinate
                 0.0                 0.0
              0.0005                15.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

***RIGIDWALL_PLANAR**_{OPTION}_{OPTION}_{OPTION}

Available options include:

<BLANK>

ORTHO

FINITE

MOVING

FORCES

The ordering of the input below as specified in the Card Summary must be observed, but the ordering of the options in the keyword name is unimportant. For example, both *RIGIDWALL_PLANAR_ORTHO_FINITE and *RIGIDWALL_PLANAR_FINITE_ORTHO are valid and have the same effect. The ORTHO option does not apply if the MOVING option is used.

An ID number may be assigned to the rigid wall using the following option:

ID

If this option is active, the ID card is the first card following the keyword.

Display of a non-moving, planar rigid wall is on by default (see SKIPRWG in *CONTROL_CONTACT). The option

DISPLAY

is available for display of moving rigid walls. With this option active, a rigid body is automatically created which represents the shape of the rigid wall and tracks its position without need for additional input. The part ID of the rigid body defaults to RWID if the ID option is active and RWID is a unique ID within the set of all part IDs.

Purpose: Define planar rigid walls with either finite (FINITE) or infinite size. Orthotropic friction can be defined (ORTHO). Also, the plane can possess a mass and an initial velocity (MOVING); otherwise, the wall is assumed to be stationary. The FORCES option allows the specification of segments on the rigid walls on which the contact forces are computed. For a more physical reaction related to the force as function of time curve, the SOFT value on the FORCES card can be specified.

Card Summary:

Card Sets. For each rigid wall matching the specified keyword options include one set of the following data cards. This input ends at the next keyword ("*") card.

Card ID. This card is included if the ID keyword option is used.

ID							

Card 1. This card is required.

NSID	NSIDEX	BOXID	OFFSET	BIRTH	DEATH	RWKSF	

Card 2. This card is required.

XT	YT	ZT	XH	YH	ZH	FRIC	WVEL

Card 3. This card is included if the ORTHO option is used.

SFRICA	SFRICB	DFRICA	DFRICB	DECAYA	DECAYB		

Card 4. This card is included if the ORTHO option is used.

NODE1	NODE2	D1	D2	D3			

Card 5. This card is included if the FINITE option is used.

XHEV	YHEV	ZHEV	LENL	LENM			

Card 6. This card is included if the MOVING option is used.

MASS	V0						

Card 7. This card is included if the FORCES option is used.

SOFT	SSID	N1	N2	N3	N4		

Data Card Definitions:

ID. Card. Additional card for ID keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	RWID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
RWID	Rigid wall ID. Up to 8 characters can be used.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID	NSIDEX	BOXID	OFFSET	BIRTH	DEATH	RWKSF	
Type	I	I	I	F	F	F	F	
Default	all nodes	{∅}	{∅}	0.	0.	10^{20}	1.0	

VARIABLE	DESCRIPTION
NSID	Nodal set ID containing tracked nodes; see *SET_NODE: EQ.0: All nodes are tracked for interacting with the rigid wall.
NSIDEX	Nodal set ID containing nodes that are exempted as tracked nodes; see *SET_NODE.
BOXID	All nodes in box are included as tracked nodes for interacting with the rigid wall; see *DEFINE_BOX. If options NSID or NSIDEX are active, then only the subset of nodes activated by these options are checked to see if they are within the box.
OFFSET	All nodes within a normal offset distance, OFFSET, to the rigid wall are included as tracked nodes for the rigid wall. If options NSID, NSIDEX, or BOXID are active, then only the subset of nodes activated by these options are checked to see if they are within the offset distance.
BIRTH	Birth time of rigid wall. The time values of the load curves that control the motion of the wall are offset by the birth time.
DEATH	Death time of rigid wall. At this time the wall is deleted from the calculation. If dynamic relaxation is active at the beginning of the calculation and BIRTH = 0.0, the death time is ignored during the dynamic relaxation.
RWKSF	Stiffness scaling factor. If RWKSF is also specified in *CONTROL_CONTACT, the stiffness is scaled by the product of the two values.

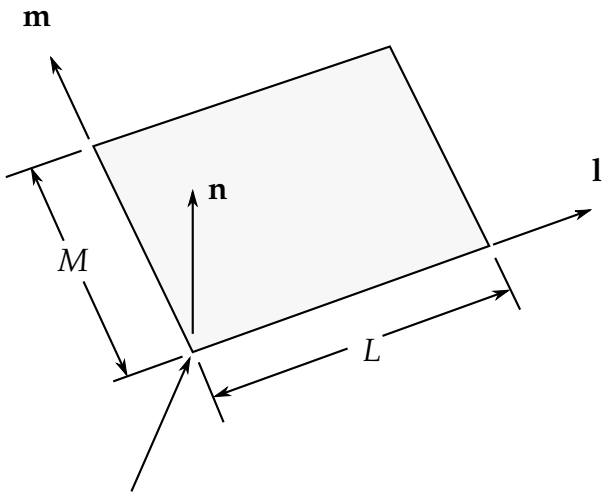


Figure 40-3. Vector **n** is normal to the rigidwall. An optional vector **l** can be defined such that $\mathbf{m} = \mathbf{n} \times \mathbf{l}$. The extent of the rigidwall is limited by defining L (LENL) and M (LENM). A zero value for either of these lengths indicates that the rigidwall is infinite in that direction.

Card 2	1	2	3	4	5	6	7	8
Variable	XT	YT	ZT	XH	YH	ZH	FRIC	WVEL
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
XT	x-coordinate of tail of any outward drawn normal vector, **n**, originating on wall (tail) and terminating in space (head), see Figure 40-3.
YT	y-coordinate of tail of normal vector **n**
ZT	z-coordinate of tail of normal vector **n**
XH	x-coordinate of head of normal vector **n**
YH	y-coordinate of head of normal vector **n**

VARIABLE	DESCRIPTION
ZH	z-coordinate of head of normal vector **n**
FRIC	Coulomb friction coefficient except as noted below: EQ.0.0: frictionless sliding after contact, EQ.1.0: no sliding after contact, EQ.2.0: node is welded after contact with frictionless sliding. Welding occurs if and only if the normal value of the impact velocity exceeds the critical value specified by WVEL. EQ.3.0: node is welded after contact with no sliding. Welding occurs if and only if the normal value of the impact velocity exceeds the critical value specified by WVEL. In summary, FRIC could be any positive value. Three special values of FRIC trigger special treatments as follows:
WVEL	Critical normal velocity at which nodes weld to wall (FRIC = 2 or 3).

FRIC	1.0	2.0	3.0
Bouncing back from wall	allowed	not allowed	not allowed
Sliding on wall	not allowed	allowed	not allowed

Orthotropic Friction Card 1. Additional card for ORTHO keyword option. See Figure 40-4 for the definition of orthotropic friction.

Card 3	1	2	3	4	5	6	7	8
Variable	SFRICA	SFRICB	DFRICA	DFRICB	DECAYA	DECAYB		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

Orthotropic Friction Card 2. Additional card for ORTHO keyword option. See Figure 40-4 for the definition of orthotropic friction.

Card 4	1	2	3	4	5	6	7	8
Variable	NODE1	NODE2	D1	D2	D3			
Type	I	I	F	F	F			
Default	0	0	0.	0.	0.			

VARIABLE	DESCRIPTION
SFRICA	Static friction coefficient in local a-direction, μ_{sa}; see Figure 40-4 and Remark 1.
SFRICB	Static friction coefficient in local b-direction, μ_{sb}
DFRICA	Dynamic friction coefficient in local a-direction, μ_{ka}
DFRICB	Dynamic friction coefficient in local b-direction, μ_{kb}
DECAYA	Decay constant in local a-direction, d_{ya}
DECAYB	Decay constant in local b-direction, d_{yb}
NODE1	Node 1, alternative to definition of vector **d** using components below. See Figure 40-4. With the node definition, the direction changes if the nodal pair rotates. See Remark 2.
NODE2	Node 2
D1	d_1, x-component of vector, alternative to definition with nodes above. See Figure 40-4. This vector is fixed as a function of time. See Remark 2.
D2	d_2, y-component of vector
D3	d_3, z-component of vector

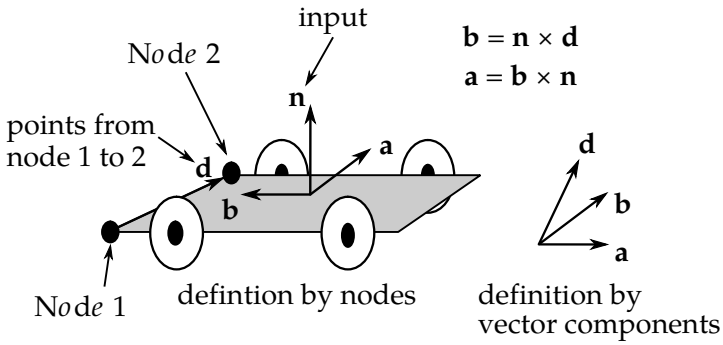


Figure 40-4. Definition of orthotropic friction vectors. The two methods of defining the vector, **d**, are shown. If vector **d** is defined by nodes 1 and 2, the local coordinate system may rotate with the body which contains the nodes; otherwise, **d** is fixed in space, thus on the rigid wall, and the local system is stationary.

Finite Wall Size Card. Additional card for FINITE keyword option. See Figure 40-4 for the definition of orthotropic friction. See Figure 40-3. The m vector is computed as the vector cross product $m = n \times l$. The origin, the tail of the normal vector, is taken as the corner point of the finite size plane.

Card 5	1	2	3	4	5	6	7	8
Variable	XHEV	YHEV	ZHEV	LENL	LENM			
Type	F	F	F	F	F			
Default	0.	0.	0.	infinity	infinity			

VARIABLE	DESCRIPTION
XHEV	x-coordinate of head of edge vector l, see Figure 40-3.
YHEV	y-coordinate of head of edge vector l
ZHEV	z-coordinate of head of edge vector l
LENL	Length of l edge
LENM	Length of m edge

Moving Wall Card. Additional card for MOVING keyword option. Note: The MOVING option is *not* compatible with the ORTHO option.

Card 6	1	2	3	4	5	6	7	8
Variable	MASS	V0						
Type	F	F						
Default	none	0.						

VARIABLE	DESCRIPTION
MASS	Total mass of rigidwall
V0	Initial velocity of rigidwall in direction of defining vector, **n**

Forces Card. Additional card for FORCES keyword option. This option allows the force distribution to be monitored on the plane. Also four points can be defined for visualization of the rigid wall. A shell or membrane element must be defined with these four points as the connectivity for viewing in LS-PREPOST.

Card 7	1	2	3	4	5	6	7	8
Variable	SOFT	SSID	N1	N2	N3	N4		
Type	I	I	I	I	I	I		
Default	0	{∅}	no node	no node	no node	no node		

VARIABLE	DESCRIPTION
SOFT	Number of cycles to zero relative velocity to reduce force spike
SSID	Segment set ID for defining areas for force output; see *SET_SEGMENT and Remark 3 below.
N1-N4	Optional node for visualization. See Remark 4.

Remarks:

1. **Orthotropic Coefficients of Friction.** The coefficients of friction are defined in terms of the static, dynamic and decay coefficients and the relative velocities in the local a and b-directions as

$$\mu_a = \mu_{ka} + (\mu_{sa} - \mu_{ka})e^{-d_{va}|V_{\text{relative},a}|}$$
$$\mu_b = \mu_{kb} + (\mu_{sb} - \mu_{kb})e^{-d_{vb}|V_{\text{relative},b}|}$$

2. **Modeling Rolling Objects.** Orthotropic rigid walls can be used to model rolling objects on rigid walls where the frictional forces are substantially higher in a direction transverse to the rolling direction. To use this option, define a vector **d** to determine the local frictional directions by:

$$\mathbf{b} = \mathbf{n} \times \mathbf{d}, \qquad \mathbf{a} = \mathbf{b} \times \mathbf{n}$$

where **n** is the normal vector to the rigid wall. If **d** is in the plane of the rigid wall, then **a** is identical to **d**.

3. **Resultant Forces.** The segment set defines areas for computing resultant forces. These segments translate with the moving rigidwall and allow the force distribution to be determined. The resultant forces are written in file rwforc.

4. **Visualization with LS-PrePost.** These four nodes are for visualizing the movement of the wall, that is, they move with the wall. To view the wall in LS-PREPOST a single shell element with these four nodes as its connectivity need to be defined. The single element must be deformable (non-rigid) or else the segment will be treated as a rigid body and the nodes will have their motion modified independently of the rigidwall.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *RIGIDWALL_PLANAR_MOVING_FORCES
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a moving planar rigid wall:
$     - that is parallel to the y-z plane starting at x = 250 mm
$     - with an initial velocity of 8.94 mm/ms in the negative z-direction
$     - that has a mass of 800 kg
$     - which prevents all nodes in the model from penetrating the wall
$     - with a friction coefficient for nodes sliding along the wall of 0.1
$     - track the motion of the wall by creating a node (numbered 99999)
$         at the tail of the wall and assigning the node to move with the wall
$
*RIGIDWALL_PLANAR_MOVING_FORCES
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$     nsid    nsidex     boxid
         0         0         0
$
$       xt        yt        zt        xh        yh        zh      fric
```

```
     250.0       0.0       0.0       0.0       0.0       0.0       0.1
$
$  SW mass    SW vel
    800.00      8.94
$
$     soft      ssid     node1     node2     node3     node4
         0         0     99999
$
$
*NODE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$    nid               x               y               z      tc      rc
   99999           250.0             0.0             0.0       0       0
$
$
*DATABASE_HISTORY_NODE
$   Define nodes that output into nodout
$      id1       id2       id3
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
     99999
$
*DATABASE_NODOUT
$       dt
       0.1
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*SECTION

In this section, the element formulation, integration rule, nodal thicknesses, and cross sectional properties are defined. All section identifiers (SECIDs) defined in this section must be unique, i.e., if a number is used as a section ID for a beam element then this number cannot be used again as a section ID for a solid element. The keyword cards in this section are defined in alphabetical order:

*SECTION_ALE1D

*SECTION_ALE2D

*SECTION_BEAM

*SECTION_BEAM_AISC

*SECTION_DISCRETE

*SECTION_FPD

*SECTION_IGA_SHELL

*SECTION_IGA_SOLID

*SECTION_ISPG

*SECTION_POINT_SOURCE

*SECTION_POINT_SOURCE_MIXTURE

*SECTION_SEATBELT

SECTION_SHELL_*{OPTION}

SECTION_SOLID_*{OPTION}

**SECTION_SOLID_PERI*

SECTION_SPH_*{OPTION}

*SECTION_TSHELL

The location and order of these cards in the input file are arbitrary.

An additional option **TITLE** may be appended to all the ***SECTION** keywords. If this option is used then an addition line is read for each section in 80a format which can be

used to describe the section. At present, the title serves no purpose other than to perhaps lend clarity to input decks.

*SECTION_ALE1D

Purpose: Define section properties for 1D ALE elements

Card Sets. For each ALE1D section add one pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ALEFORM	AET	ELFORM				
Type	I/A	I	I	I				
Default	none	none	0	none				

Card 2	1	2	3	4	5	6	7	8
Variable	THICK	THICK						
Type	F	F						
Default	none	none						

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ALEFORM	ALE formulation: EQ.11: Multi-Material ALE formulation
AET	Ambient Element Type: EQ.4: Pressure inflow
ELFORM	Element formulation: EQ.7: plane strain EQ.8: axisymmetric (per radian) EQ.-8: spherical (per unit of solid angle)

VARIABLE	DESCRIPTION
THICK	Nodal thickness. See Remark 1.

Remarks:

1. **Nodal Thickness.** *SECTION_ALE1D is using the common *SECTION_BEAM reader which expects two thickness values. The ALE 1D will simply take the average of these two values as the beam thickness. The thickness, however, is not used for ELFORM = -8, but the reader routine expects values on the 2nd line.

2. **Output to d3plot.** Unlike normal beams, ALE beams have history variables for more than one material. Since ALE beams only have one integration point, we use BEAMIP in *DATABASE_EXTENT_BINARY in a similar fashion to NEIPH for solids to tell LS-DYNA how much data needs to be exported to the d3plot file. Normally BEAMIP just tells LS-DYNA how many integration points for which data is output. By doing this, we can access for each ALE group in an element, 6 stresses, 1 plastic strain and a number of history variables that depend on the material and equation of state for the ALE group. Note that each increase in the value of BEAMIP leads to an increase of 5 data slots in the output file.

   Because this is an overloaded use of BEAMIP, LS-PrePost labels will not match the data being output. The data being output will be under beam integration points (bintpt) which can be accessed with Post → FriComp → Beam. The interpretation of the data will be described with examples below.

   For all cases of BEAMIP including BEAMIP = 0, d3plot will include 5 volume averaged items that will be under the first integration point. These five values will be under the first integration point in LS-PrePost and are described in the table below. Their real meaning is in the column labeled Variable Description.

LS-PrePost Label	Variable Description	ALE Group
axial stress	pressure	averaged
rs-shear stress	density	averaged
tr-shear stress	compression ratio	averaged
plastic strain	internal energy/vol	averaged
axial strain	ALE group ID	

Now suppose that we have 2 ALE groups, one with 8 history variables and one with 3 history variables. If we want all the output data, the number of data variables that we need output to d3plot would be

$$5 + \text{number of ALE groups} \times 7 + \sum_{i=1}^{n} \text{history variables of ALE group } i$$

since the 5 averaged variables listed above are always output, each ALE group has 7 values for the stress and effective plastic strain, and each ALE group has history variables. In our current example, this would $5 + 2 \times 7 + 8 + 3 = 30$. Since each integration points has 5 data slots as understood by LS-PrePost, then 6 integration points of data are needed, so BEAMIP should be set to 5 (note that this is because BEAMIP = 0 causes one integration point worth of data to be output with this implementation). If it is not set to 5, then d3plot will only include the data up to the number of integration points being output. The description for the values starting with the second integration point are listed in the table below.

Integration Point in LS-PrePost	LS-PrePost Label	Variable Description	ALE Group
2	axial stress	*xx*-stress	1
2	*rs*-shear stress	*yy*-stress	1
2	*tr*-shear stress	*zz*-stress	1
2	plastic strain	*xy*-stress	1
2	axial strain	*yz*-stress	1
3	axial stress	*zx*-stress	1
3	*rs*-shear stress	effective plastic strain	1
3	*tr*-shear stress	history variable # 1	1
3	plastic strain	history variable # 2	1
3	axial strain	history variable # 3	1
4	axial stress	history variable # 4	1
4	*rs*-shear stress	history variable # 5	1
4	*tr*-shear stress	history variable # 6	1

Integration Point in LS-PrePost	LS-PrePost Label	Variable Description	ALE Group
4	plastic strain	history variable # 7	1
4	axial strain	history variable # 8	1
5	axial stress	*xx*-stress	2
5	*rs*-shear stress	*yy*-stress	2
5	*tr*-shear stress	*zz*-stress	2
5	plastic strain	*xy*-stress	2
5	axial strain	*yz*-stress	2
6	axial stress	*zx*-stress	2
6	*rs*-shear stress	effective plastic strain	2
6	*tr*-shear stress	history variable # 1	2
6	plastic strain	history variable # 2	2
6	axial strain	history variable # 3	2

*SECTION_ALE2D

Purpose: Define section properties for 2D ALE elements. This supersedes the old way of defining section properties for 2D ALE elements using *SECTION_SHELL.

Section Cards. For each ALE2D section include a card. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ALEFORM	AET	ELFORM				
Type	I/A	I	I	I				
Default	none	none	0	none				

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ALEFORM	ALE formulation: EQ.11: Multi-Material ALE formulation.
AET	Part type flag: EQ.0: This is a regular or non-ambient part (default) EQ.4: Reservoir or ambient type part EQ.5: Reservoir or ambient type part, but only used together with *LOAD_BLAST_ENHANCED (LBE). It defines this part as an "ambient receptor part" for the transient blast load supplied by a corresponding LBE KW (see *LOAD_BLAST_ENHANCED, available only for ALEFORM = 11).
ELFORM	Element formulation: EQ.13: Plane strain (*xy*-plane) EQ.14: Axisymmetric solid (*xy*-plane, *y*-axis of symmetry) – area weighted (see Remark 2) EQ.15: Axisymmetric solid (*xy*-plane, *y*-axis of symmetry) – volume weighted

Remarks:

1. **Lagrangian-ALE coupling.** For coupling between 2D Lagrangian elements and 2D ALE elements, use *CONSTRAINED_LAGRANGE_IN_SOLID rather than *CONTACT_2D_AUTOMATIC_SURFACE_IN_CONTINUUM.

2. **Axisymmetric analysis.** For an axisymmetric analysis, ELFORM for *SECTION_ALE2D can be set to 14 (area-weighted) or 15 (volume-weighted). In the same analysis, axisymmetric Lagrangian elements are not restricted to the selected 2D element formulations for ALE. In other words, shell formulation 14 or 15 is permitted for Lagrangian shells, and beam formulation 8 is permitted for Lagrangian beams. Coupling forces between the axisymmetric ALE elements and axisymmetric Lagrangian elements are automatically adjusted as needed.

*SECTION_BEAM

Purpose: Define cross-sectional properties for beam, truss, discrete beam, and cable elements.

Card Summary:

Card Sets. For each BEAM section in the model, add one set of Cards 1 and 2 (maybe additionally Card 3 for ELFORM = 12) cards. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SECID	ELFORM	SHRF	QR/IRID	CST	SCOOR	NSM	NAUPD

Card 2a. Include this card when ELFORM is 1 or 11.

TS1	TS2	TT1	TT2	NSLOC	NTLOC	ITORM	

Card 2b. Include this card when ELFORM is 2, 3, or 12 and the first 7 characters of the card spell out "SECTION."

STYPE	D1	D2	D3	D4	D5	D6	ITORM

Card 2b.1. This card is optional. It is only read when ELFORM = 2, the preceding card is Card 2b, and the first column includes the character "OPTCARD."

OPTC	ASA						

Card 2c. Include this card when ELFORM is 2, 12, or 13 and the first 7 characters of the card *do not* spell out "SECTION."

A	ISS	ITT	J	SAS	IST	ITORM	SAT

Card 2c.1. Include this card when ELFORM equals 12 and the preceding card is Card 2c.

YS	ZS	IYR	IZR	IRR	IW	IWR	

Card 2d. Include this card when ELFORM is 3 and the first 7 characters of the card *do not* spell "SECTION."

A	RAMPT	STRESS					

Card 2e. Include this card for ELFORM equal to 4 or 5.

TS1	TS2	TT1	TT2				

Card 2f. Include this card for ELFORM equal to 6 for any material other than material type 146.

VOL	INER	CID	CA	OFFSET	RRCON	SRCON	TRCON

Card 2g. Include this card for ELFORM equal to 6 for material type 146.

VOL	INER	CID	DOFN1	DOFN2			

Card 2h. Include this card for ELFORM equal to 7 or 8.

TS1	TS2						

Card 2i. Include this card for ELFORM equal to 9.

TS1	TS2	TT1	TT2	PRINT		ITOFF	

Card 2j. Include this card for ELFORM equal to 14.

PR	IOVPR	IPRSTR					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	SHRF	QR/IRID	CST	SCOOR	NSM	NAUPD
Type	I/A	I	F	F	F	F	F	I
Default	none	1	1.0	2.0	0.0	0.0	0.0	0

VARIABLE	DESCRIPTION
SECID	Section ID. The *PART card references SECID. A unique number or label must be specified.
ELFORM	Element formulation options: EQ.1: Hughes-Liu with cross-section integration (default) EQ.2: Belytschko-Schwer resultant beam (resultant) EQ.3: Truss (resultant) EQ.4: Belytschko-Schwer full cross-section integration

VARIABLE	DESCRIPTION
	EQ.5: Belytschko-Schwer tubular beam with cross-section integration EQ.6: Discrete beam/cable EQ.7: 2D plane strain shell element (*xy*-plane) EQ.8: 2D axisymmetric volume weighted shell element (*xy*-plane, *y*-axis of symmetry) EQ.9: Spot weld beam; see *MAT_SPOTWELD. EQ.11: Integrated warped beam. See Remark 3. EQ.12: Resultant warped beam EQ.13: Small displacement, linear Timoshenko beam with exact stiffness. See Remark 5. EQ.14: Elbow integrated tubular beam element. A user-defined integration rule with a tubular cross section (9) must be used. Note that the 2D and 3D element types must not be mixed, and different types of 2D elements must not be used together. For example, the plane strain element type must not be used with the axisymmetric element type. In 3D, the different beam element types, that is, 1-6, 9, and 11-14, can be freely mixed together.
SHRF	Shear factor. This factor is not needed for truss, resultant beam, discrete beam, and cable elements. The recommended value for rectangular sections is 5/6; the default is 1.0.
QR/IRID	Quadrature rule or rule number for a user-defined rule for integrated beams. For rectangular cross sections, Gauss or Lobatto quadrature will be employed. For circular cross sections, the integration points are evenly spaced on the perimeter of the surface. See Remark 9 regarding beam formulations 7 and 8. EQ.1.0: One integration point EQ.2.0: 2×2 Gauss or 4-point circular quadrature (default beam) EQ.3.0: 3×3 Gauss or 9-point circular quadrature EQ.4.0: 3×3 Lobatto or 9-point circular quadrature EQ.5.0: 4×4 Gauss or 16-point circular quadrature EQ.-n: |n| is the number of the user-defined rule. IRID integration rule n is defined using *INTEGRATION_BEAM

VARIABLE	DESCRIPTION
	card.
CST	Cross-section type, not needed for truss, resultant beam, discrete beam, and cable elements: EQ.0.0: Rectangular, EQ.1.0: Tubular (circular only), EQ.2.0: Arbitrary (user-defined integration rule).
SCOOR	Affects the discrete beam formulation (see Remarks 6 and 14) and the update of the local coordinate system of the discrete beam element. This field does not apply to cable elements. The force and moment resultants in the output databases are output in the local coordinate system. See Remark 8 for more on the local coordinate system update. EQ.-14.0: Like -13.0, but with improved rotational stability EQ.-13.0: Like -3.0, but with correction for beam rotation EQ.-12.0: Like -2.0, but with correction for beam rotation EQ.-3.0: Beam node 1; the angular velocity of node 1 rotates the triad. EQ.-2.0: Beam node 1: The angular velocity of node 1 rotates the triad, but the *r*-axis is adjusted to lie along the line between the two beam nodal points. This option is not recommended for zero-length discrete beams. EQ.-1.0: Beam node 1; the angular velocity of node 1 rotates the triad. EQ.0.0: Centered between beam nodes 1 and 2. The triad rotates with the average angular velocity of nodes 1 and 2. EQ.+1.0: Beam node 2: The angular velocity of node 2 rotates the triad. EQ.+2.0: Beam node 2: The angular velocity of node 2 rotates the triad, but the *r*-axis is adjusted to lie along the line between the two beam nodal points. This option is not recommended for zero-length discrete beams. EQ.+3.0: Beam node 2; the angular velocity of node 2 rotates the triad. EQ.+12.0: Like +2.0, but with correction for beam rotation

VARIABLE	DESCRIPTION
	EQ.+13.0: Like +3.0, but with correction for beam rotation
	EQ.+14.0: Like +13.0 but with improved rotational stability
NSM	Nonstructural mass per unit length. This option applies to beam types 1-5 and does not apply to discrete, 2D, and spot weld beams.
NAUPD	Neutral axis update option. See Remark 11.
	EQ.0: Not used
	EQ.1: Update the neutral axis when damage or failure occurs at one or more integration points.

Integrated Beam Card (types 1 and 11). This card is Card 2 for ELFORM set to either type 1 or 11.

Card 2a	1	2	3	4	5	6	7	8
Variable	TS1	TS2	TT1	TT2	NSLOC	NTLOC	ITORM	
Type	F	F	F	F	F	F	F	

VARIABLE	DESCRIPTION
TS1	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_1. The thickness defined on *ELEMENT_BEAM_THICKNESS overrides the definition given here.
TS2	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_2.
TT1	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_1.
TT2	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_2.
NSLOC	Location of the reference surface normal to the s-axis for Hughes-Liu beam elements only. See Remark 4.
	EQ.1.0: Side at $s = 1.0$
	EQ.0.0: Center
	EQ.-1.0: Side at $s = -1.0$

VARIABLE	DESCRIPTION
NTLOC	Location of the reference surface normal to the t-axis for Hughes-Liu beam elements only. See Remark 4. EQ.1.0: Side at $t = 1.0$ EQ.0.0: Center EQ.-1.0: Side at $t = -1.0$
ITORM	Flag for improved representation of torsional and rotational modes for beam type 1. LT.0.0: The nodal mass related to torsional and rotational modes is computed from the product of the absolute value of ITORM and the nodal translational mass. This option is frame invariant and applies to explicit and implicit analyses. EQ.0.0: Not active. The torsional and rotational inertia from the structural analysis are used. This may result in too large eigenvalues related to torsional and rotational modes. EQ.1.0: More accurately recompute the mass related to torsional modes. This option applies to eigenvalue analysis only. EQ.2.0: More accurately recompute the mass related to torsional and rotational modes. This option applies to eigenvalue analysis only. EQ.3.0: Average the mass related to torsional and rotational modes from ITORM = 2. This option is frame invariant and applies to explicit and implicit analyses. Note that ITORM < 0 and ITORM = 3 may require reducing the time step size for stability in explicit analysis.

Resultant Beam with Shape Card (types 2, 3, and 12). Include this card as Card 2 when ELFORM equals 2, 3, or 12 and the first 7 characters of the card spell out "SECTION."

Card 2b	1	2	3	4	5	6	7	8
Variable	STYPE	D1	D2	D3	D4	D5	D6	ITORM
Type	A10	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
STYPE	Section type (A format) of resultant beam (see Figure 41-1):

EQ.SECTION_01: I-Shape
EQ.SECTION_02: Channel
EQ.SECTION_03: L-shape
EQ.SECTION_04: T-shape
EQ.SECTION_05: Tubular box
EQ.SECTION_06: Z-Shape
EQ.SECTION_07: Trapezoidal
EQ.SECTION_08: Circular
EQ.SECTION_09: Tubular
EQ.SECTION_10: I-Shape 2
EQ.SECTION_11: Solid Box
EQ.SECTION_12: Cross
EQ.SECTION_13: H-Shape
EQ.SECTION_14: T-Shape 2
EQ.SECTION_15: I-Shape 3
EQ.SECTION_16: Channel 2
EQ.SECTION_17: Channel 3
EQ.SECTION_18: T-Shape 3
EQ.SECTION_19: Box-Shape 2
EQ.SECTION_20: Hexagon
EQ.SECTION_21: Hat-Shape
EQ.SECTION_22: Hat-Shape 2

D1-D6: Input parameters for section option using STYPE above. See Figures 41-1 through 41-22.

ITORM: Flag for improved representation of torsional and rotational modes for beam type 2.

LT.0.0: The nodal mass related to torsional and rotational modes is computed from the product of the absolute value of ITORM and the nodal translational mass. This option is frame invariant and applies to explicit and implicit analyses.

EQ.0.0: Not active. The torsional and rotational inertia from the structural analysis are used. This may result in too large eigenvalues related to torsional and rotational modes.

EQ.1.0: More accurately recompute the mass related to torsional modes. This option applies to eigenvalue analysis only.

EQ.2.0: More accurately recompute the mass related to torsional and rotational modes. This option applies to eigenvalue analysis only.

EQ.3.0: Average the mass related to torsional and rotational modes from ITORM = 2. This option is frame invariant and applies to explicit and implicit analyses.

Note that ITORM < 0 and ITORM = 3 may require reducing the time step size for stability in explicit analysis.

Resultant Beam with Optional Shape Card (type 2 only). This card is optional. It is read when ELFORM = 2, the preceding card is Card 2b, and the first column includes the string "OPTCARD."

Card 2b.1	1	2	3	4	5	6	7	8
Variable	OPTC	ASA						
Type	A	F						

VARIABLE	DESCRIPTION
OPTC	Include the string "OPTCARD" to activate this card.
ASA	Automized shear area computation for section types defined with STYPE on Card 2b. EQ.0.0: Not active. The shear area is set equal to the cross-sectional area. EQ.1.0: Approximate the shear area based on an energy consideration. It currently applies to Tubular (SECTION_09) and Solid Box (SECTION_11) section types.

Resultant Beam Card 1 (types 2, 12, and 13). Include this card as Card 2 when ELFORM equals 2, 12, or 13 and the first 7 characters of the card *do not* spell "SECTION."

Card 2c	1	2	3	4	5	6	7	8
Variable	A	ISS	ITT	J	SAS	IST	ITORM	SAT
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
A	Cross-sectional area. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here.
ISS	I_{ss}, area moment of inertia about local s-axis. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here.
ITT	I_{tt}, area moment of inertia about local t-axis. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here.
J	J, torsional constant. The definition on *ELEMENT_BEAM_-

VARIABLE	DESCRIPTION
	THICKNESS overrides the value defined here. If J is zero, then J is reset to the sum of ISS + ITT as an approximation for the warped beam.
SAS	Shear area if ELFORM = 12 or 13. It is the shear area perpendicular to the local s-axis if ELFORM = 2. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here.
IST	I_{st}, product area moment of inertia with respect to the local s- and t-axes. This is only nonzero for asymmetric cross sections, and it can take positive and negative values; for example, it is negative for SECTION_03.
ITORM	Flag for improved representation of torsional and rotational modes for beam types 2 and 13. LT.0.0: The nodal mass related to torsional and rotational modes is computed from the product of the absolute value of ITORM and the nodal translational mass. This option is frame invariant and applies to explicit and implicit analyses. EQ.0.0: Not active. The torsional and rotational inertia from the structural analysis are used. This may result in too large eigenvalues related to torsional and rotational modes. EQ.1.0: More accurately recompute the mass related to torsional modes. This option applies to eigenvalue analysis only. EQ.2.0: More accurately recompute the mass related to torsional and rotational modes. This option applies to eigenvalue analysis only. EQ.3.0: Average the mass related to torsional and rotational modes from ITORM = 2. This option is frame invariant and applies to explicit and implicit analyses. Note that ITORM < 0 and ITORM = 3 may require reducing the time step size for stability in explicit analysis.
SAT	Shear area perpendicular to the local t-axis if ELFORM = 2. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here. This field is ignore ELFORM ≠ 2.

Resultant Beam Card 2 (type 12 only). Include this card if ELFORM equals 12 and the preceding card is Card 2c.

Card 2c.1	1	2	3	4	5	6	7	8
Variable	YS	ZS	IYR	IZR	IRR	IW	IWR	
Type	F	F	F	F	F	F	F	

VARIABLE	DESCRIPTION
YS	s coordinate of the shear center of the cross section. The coordinate system is located at the centroid.
ZS	t coordinate of the shear center of the cross section. The coordinate system is located at the centroid.
IYR	$\int_A sr^2 dA$, where $r^2 = s^2 + t^2$
IZR	$\int_A tr^2 dA$, where $r^2 = s^2 + t^2$
IRR	$\int_A r^4 dA$, where $r^2 = s^2 + t^2$
IW	Warping constant. $\int_A \omega^2 dA$, where ω is the sectorial area.
IWR	$\int_A \omega r^2 dA$

Resultant Beam Card (type 3). Include this card for ELFORM equal to 3.

Card 2d	1	2	3	4	5	6	7	8
Variable	A	RAMPT	STRESS					
Type	F	F	F					

VARIABLE	DESCRIPTION
A	Cross-sectional area. The definition on *ELEMENT_BEAM_THICKNESS overrides the value defined here.
RAMPT	Optional ramp-up time for dynamic relaxation. At the end of the ramp-up time, a uniform stress, STRESS, exists in the truss element. This option does not work for hyperelastic materials.

VARIABLE	DESCRIPTION
STRESS	Optional initial stress for dynamic relaxation. At the end of dynamic relaxation, a uniform stress equal to this value should exist in the truss element.

Integrated Beam Card (types 4 and 5). Include this card when ELFORM equals 4 or 5.

Card 2e	1	2	3	4	5	6	7	8
Variable	TS1	TS2	TT1	TT2				
Type	F	F	F	F				

VARIABLE	DESCRIPTION
TS1	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_1. The thickness defined on *ELEMENT_BEAM_THICKNESS overrides the definition given here.
TS2	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_2.
TT1	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_1.
TT2	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_2.

Discrete Beam Card (type 6). Include this card when ELFORM equals 6 for any material *other* than material type 146.

Card 2f	1	2	3	4	5	6	7	8
Variable	VOL	INER	CID	CA	OFFSET	RRCON	SRCON	TRCON
Type	F	F/I	F	F	F	F	F	F

VARIABLE	DESCRIPTION
VOL	Volume of discrete beams that is used in calculating mass. If VOL = 0 for cable elements, the volume is calculated as the product of cable length and cable area. If the mass density of the material model for the discrete beam is set to unity, the magnitude of the

VARIABLE	DESCRIPTION
	lumped mass can be defined here instead. This lumped mass is partitioned equally between the two nodes of the element. See Remark 12.
INER	Mass moment of inertia for discrete beam form 6 when used with materials that have rotational stiffness defined. Materials that support rotational stiffnesses are 66, 67, 68, 93, 95, 97, 119, and 196. This lumped inertia is partitioned equally between the two nodes of the element. See Remarks 12 and 13. GT.0.0: Inertia partitioned to the nodes. EQ.-1.0: The element inertia is computed as a solid sphere of volume VOL. This inertia, which is reported to d3hsp, is then partitioned to the nodes. EQ.-2.0: The element inertia is computed such that the rotational time step matches the translational. It is currently only available for *MAT_196. The computed element inertias are written to the message file(s).
CID	Coordinate system ID for orientation (material types 66-69, 93, 95, 97), see *DEFINE_COORDINATE_*OPTION*. If CID = 0, a default coordinate system is defined in the global system or on the third node of the beam, which is used for orientation. This option is not defined for material types that act between two nodal points, such as cable elements. The coordinate system rotates with the discrete beam, see SCOOR above.
CA	Cable area. See material type 71, *MAT_CABLE_DISCRETE_BEAM. For a crushable beam using *MAT_119 with IFLAG = 2, CA is the cross-sectional area ratio between end 2 and end 1. A crushable beam of uniform cross section has CA = 1, the default value.
OFFSET	Optional offset for cable. See material type 71, *MAT_CABLE_DISCRETE_BEAM.
RRCON	*r*-rotational constraint for the local coordinate system (see Remark 7) EQ.0.0: Coordinate ID rotates about the *r*-axis with nodes. EQ.1.0: Rotation is constrained about the *r*-axis
SRCON	*s*-rotational constraint for the local coordinate system (see Remark 7)

VARIABLE	DESCRIPTION
	EQ.0.0: Coordinate ID rotates about the *s*-axis with nodes. EQ.1.0: Rotation is constrained about the *s*-axis
TRCON	*t*-rotational constraint for the local coordinate system (see Remark 7) EQ.0.0: Coordinate ID rotates about the *t*-axis with nodes. EQ.1.0: Rotation is constrained about the *t*-axis.

Discrete Beam Card (type 6, mat 146). Include this card when ELFORM equals 6 for material type 146.

Card 2g	1	2	3	4	5	6	7	8
Variable	VOL	INER	CID	DOFN1	DOFN2			
Type	F	F/I	F	F	F			

VARIABLE	DESCRIPTION
VOL	Volume of discrete and scalar (*MAT_146) beams, used in calculating mass. If VOL = 0 for cable elements, the volume is calculated as the product of cable length and cable area. If the mass density of the material model for the discrete beam is set to unity, the magnitude of the lumped mass can be defined here instead. This lumped mass is partitioned equally between the two nodes of the element. See Remark 12.
INER	Mass moment of inertia for the six degrees of freedom discrete beam and scalar (*MAT_146) beam. This parameter does not apply to cable elements. This lumped inertia is partitioned equally between the two nodes of the element. See Remarks 12 and 13. GT.0.0: Inertia partitioned to the nodes. EQ.-1.0: The element inertia is computed as a solid sphere of volume VOL. This inertia, which is reported to d3hsp, is then partitioned to the nodes. EQ.-2.0: The element inertia is computed such that the rotational time step matches the translational. It is currently only available for *MAT_196. The computed element inertias are written to the message file(s).

VARIABLE	DESCRIPTION
CID	Coordinate system ID for orientation, material type 146, see *DEFINE_COORDINATE_SYSTEM. If CID = 0, a default coordinate system is defined in the global system.
DOFN1	Active degree of freedom at node 1 specified as a number between 1 and 6, where 1 is x-translation and 4 is x-rotation.
iDOFN2	Active degree of freedom at node 2, a number between 1 and 6.

2D Shell Card (types 7 and 8). Include this card when ELFORM equals 7 or 8.

Card 2h	1	2	3	4	5	6	7	8
Variable	TS1	TS2						
Type	F	F						

VARIABLE	DESCRIPTION
TS1	Thickness at node n_1.
TS2	Thickness at node n_2.

Spot Weld Card (type 9). Include this card when ELFORM equals 9.

Card 2i	1	2	3	4	5	6	7	8
Variable	TS1	TS2	TT1	TT2	PRINT		ITOFF	
Type	F	F	F	F	F		F	

VARIABLE	DESCRIPTION
TS1	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_1. Note that the thickness defined on the *ELEMENT_BEAM_THICKNESS card overrides the definition given here.
TS2	Beam thickness (CST = 0.0, 2.0) or outer diameter (CST = 1.0) in s-direction at node n_2.

VARIABLE	DESCRIPTION
TT1	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_1.
TT2	Beam thickness (CST = 0.0, 2.0) or inner diameter (CST = 1.0) in t-direction at node n_2.
PRINT	Output spot force resultant from spot welds. EQ.0.0: Data is output to swforc file. EQ.1.0: Output is suppressed.
ITOFF	Option to specify torsional behavior for spot weld beams. EQ.0.0: Torsional stiffness is active. EQ.1.0: Torsional stiffness is zero (free to twist).

Integrated Beam Card (type 14). Include this card when ELFORM equals 14.

Card 2j	1	2	3	4	5	6	7	8
Variable	PR	IOVPR	IPRSTR					
Type	F	F	F					

VARIABLE	DESCRIPTION
PR	Pressure inside ELBOW elements that belong to the section. The pressure acts as a stiffener and will reduce the ovalization of the pipe. Pressure acting on the inside wall is taken as positive.
IOVPR	Print flag for the ELBOW ovalization degrees of freedom. EQ.1.0: An ASCII file named elbwov is created and filled with the ovalization. By default, no file is created.
IPRSTR	Flag for adding stress due to pressure PR into the material routine. EQ.0: No stress is added to the material. In this case, the pressure only acts as a stiffener for the tube. EQ.1: The pressure PR is used to calculate additional axial and circumferential stresses due to the applied pressure PR. The stress added is given by:

VARIABLE	DESCRIPTION

$$\sigma_{\text{axial}} = \text{PR} \times \frac{D}{4T}, \qquad \sigma_{\text{circ}} = \text{pr} \times \frac{D}{2T}$$

for a straight pipe, and

$$\sigma_{\text{axial}} = \text{PR} \times \frac{D}{4T}, \qquad \sigma_{\text{circ}} = \text{PR} \times \frac{D}{4T} \frac{[2R + r\cos(\theta)]}{[R + r\cos(\theta)]}$$

for a curved pipe. D is the pipe diameter, T is the thickness, R is the curvature of the bend, r is the pipe radius (mean) and θ is an angle pointing out a point on the pipe. This option also includes the stiffening effect.

Remarks:

1. **Implicit time integrator.** For implicit calculations, all the beam element choices are implemented.

2. **Local coordinate system rotation.** The local coordinate system rotates as the nodal points that define the beam rotate. In some cases, this may lead to unexpected results if the nodes undergo significant rotational motions. In the definition of the local coordinate system using *DEFINE_COORDINATE_NODES, if the option to update the system each cycle is active, then this updated system is used. This latter technique seems to be more stable in some applications.

3. **Integrated warped beam.** The integrated warped beam (type 11) is a seven-degree-of-freedom beam that must be used with an integration rule for open standard cross sections; see *INTEGRATION_BEAM. To incorporate the additional degrees of freedom corresponding to the twist rates, one scalar node (*NODE_SCALAR) should be declared for each node attached to a warped beam. This degree of freedom is associated with the beam element using the warpage option on the *ELEMENT_BEAM card.

4. **Beam offsets.** Beam offsets are sometimes necessary for correctly modeling beams that act compositely with other elements, such as shells or other beams. A beam offset extends from the beam's n_1-to-n_2 axis to the reference axis of the beam. The beam reference axis lies at the origin of the local s- and t-axes. This origin is located at the center of the cross-section footprint for beam formulations 1 and 11, but it is located at the cross-section centroid for beam formulation 2. Note that for cross-sections that are not doubly symmetric, such as a T-section, the center of the cross-section footprint and the centroid of the cross-section do not coincide. The offset in the positive s-direction is

$$\text{s-offset} = -0.5 \times \text{NSLOC} \times (\text{beam cross-section dimension in } s\text{-direction}) .$$

Similarly, the offset in the positive t-direction is

$$\text{t-offset} = -0.5 \times \text{NTLOC} \times (\text{beam cross-section dimension in } t\text{-direction}) .$$

If IRID is used to point to an integration rule with ICST > 0, then offsets must be defined using SREF and TREF on the *INTEGRATION_BEAM card, as they will override NSLOC and NTLOC even if SREF = 0 or TREF = 0. See also *ELEMENT_BEAM_OFFSET for an alternate approach to defining beam offsets.

5. **Three-dimensional Timoshenko beam.** Element type 13 is a three-dimensional Timoshenko resultant-based beam element with two nodes, suitable for small displacement, linear isotropic elasticity. The stiffness matrix is identical to the residual stiffness formulation used in the Belytschko-Schwer element (type 2). This element only works with *MAT_ELASTIC. It uses the reference geometry to calculate the element stiffness and then determines the element forces by multiplying the element stiffness by the displacements. Offsets work, but they are fixed for all time, like the reference geometry.

6. **SCOOR.** If the magnitude of SCOOR is less than or equal to unity, then zero-length discrete beams are assumed with infinitesimal separation between the nodes in the deformed state. For large separations or nonzero length beams, set |SCOOR| to 2, 3, 12, 13, or 14, in which case true beam-like behavior is invoked to provide equilibrating torques to offset any force couples that arise due to translational stiffness or translational damping. For $|\text{SCOOR}| \leq 13$, the equilibrating torque is divided equally between Node 1 and Node 2. In some cases, this can lead to unstable rotational behavior and non-physical energy growth; $|\text{SCOOR}| = 14$ fixes that by applying all of the equilibrating torque onto the node that carries the axis system, i.e., Node 1 for $\text{SCOOR} = -14$ and Node 2 for $\text{SCOOR} = +14$. Also, rigid body rotation is measured, and the spring strain is modified so that rotation does not induce strain. A flaw in this strain modification was found in the implementation of $|\text{SCOOR}| = 2$ and 3; the improved formulation is activated by setting $|\text{SCOOR}| = 12$, 13, or 14. The original options were left in place to allow legacy data to run without change. See also Remark 14.

7. **Disabling nodal rotations.** RRCON, SRCON, and TRCON are optional and apply only to non-cable discrete beams. If set to 1, RRCON, SRCON, and TRCON will prevent nodal *rotations* about the local *r*-, *s*-, and *t*-axes, respectively, from affecting the update of the local coordinate system. These three parameters do not influence how nodal translations may affect the local coordinate system update.

8. **Note about local coordinate updates and FLAG.** If CID is nonzero for a discrete beam and the coordinate system identified by CID uses *DEFINE_COORDINATE_NODES with FLAG = 1, the beam local system is updated based on the current orientation of the three nodes identified in *DEFINE_COORDINATE_NODES. In this case, local coordinate system updates per SCOOR types 0, ±1, ±3, ±13 and ±14 are inactive, while for SCOOR types ±2 and ±12, a final adjustment is made to the local coordinate system so that the local r-axis lies along the n_1 to n_2 axis of the beam. An optional output database (*DATABASE_DISBOUT) will report relative displacements, rotations, and force resultants of discrete beams, all in the local coordinate system. See also Remark 13.

9. **Beam forms 7 and 8.** Beam formulations 7 and 8 are two-dimensional shell elements. For these two formulations, variable QR/IRID is the number of through-thickness integration points for the shell. The variable BEAMIP in *DATABASE_EXTENT_BINARY controls the output for these integration points.

10. **Beam form 1.** The integration points for beam formulation 1 are located at mid-length and are arranged in the cross-section according to the specified integration rule. Stresses are reported at integration points according to the variable BEAMIP in *DATABASE_EXTENT_BINARY.

11. **NAUPD.** Integrated beams have integration points arranged about the neutral axis. Therefore, axial stress at each point will generate both an axial force and a moment at each node. The moment is proportional to the distance from the neutral axis. For a pure axial load, the moments calculated from the integration points cancel, and only the force remains. However, if damage or failure occurs at one or more integration points, then the moments will no longer cancel, and an axial load will produce moments at the nodes. To prevent this default behavior, set NAUPD to 1 to activate the neutral axis update option. With this option, the neutral axis will be updated such that partially failed elements will continue to generate balanced moments during axial loading. This option applies to beam forms 1, 9, 11, and 14 when used with material types 3, 98, 100, 124, and 158.

12. **Discrete beam volume and inertia.** The translational time step size for the type 6 beam depends on the volume, mass density, and translational stiffness values. Similarly, the rotational time step size for the type 6 beam depends on the lumped inertia and the rotational stiffness values. Therefore, it is important to define the volume and inertia with the VOL and INER parameters. Defining the volume and inertia is also essential for mass scaling if the type 6 beam controls the time step size. To avoid instabilities, it is recommended to always set the volume and inertia parameters to reasonable, nonzero values and/or have model-dependent properties; setting the inertia to a value smaller than that of an equivalent solid sphere causes a warning message. If VOL (or INER) is not set, the element mass (or inertia) is computed as the harmonic mean of half the nodal masses (or inertias) for the time step estimation.

13. **Discrete beam rotational time step.** For discrete beams of *nonzero* length with |SCOOR| values of 2, 3, 13, and 14, the translational shear and rotational stiffnesses are coupled. More specifically, the off-axis rotational stiffness (for some deformation modes) increases by an amount proportional to the beam length times the shear stiffness. Thus, nonzero shear stiffness decreases the rotational time step. To counteract this effect, additional rotational inertia is added to the beam. This additional inertia is primarily a function of the beam length, such that the total element rotational inertia is as listed in the following table:

|SCOOR|	INER	Element inertia
0, 1, or 12	> 0	INER
0, 1, or 12	-1	I_{sphere}
0, 1, or 12	-2	$\frac{k_r}{k_t} m_e$
2, 3, 13, or 14	> 0	$\text{INER} + \max\left(m_e L^2, 16 m_e \sqrt{\text{VOL} \times L}\right)$
2, 3, 13, or 14	-1	$I_{\text{sphere}} + \max\left(m_e L^2, 16 m_e \sqrt{\text{VOL} \times L}\right)$
2, 3, 13, or 14	-2	$\left(\frac{k_\text{r}}{k_\text{t}} + \frac{0.95^2}{1-0.95^2}\frac{L^2}{4}\right) m_\text{e}$

Here, $m_\text{e} = \text{RHO} \times \text{VOL}$ is the element mass, L is the element length, k_r and k_t are the maximum rotational and translational stiffnesses, and

$$I_{\text{sphere}} = \frac{2}{5} m_\text{e} \left(\frac{3}{4\pi}\text{VOL}\right)^{\frac{2}{3}}$$

is the rotational inertia of a sphere of volume VOL.

INER < 0 is meant to help set a reasonable rotational inertia when this inertia is unknown. Otherwise, if the rotational inertias of the nodes are known, INER should be set to the sum of these inertias.

14. **Further notes on SCOOR for nonzero-length discrete beams.** This remark does not apply to cable elements (see *MAT_071). Unintended rotation of the local axes during the analysis can lead to unexpected or confusing results. To avoid such issues and ensure that the local axes retain their expected directions throughout the analysis, we recommend the following setups:

    a) For applications in which only an axial force is required, meaning a force that always acts along the line defined by the two nodes of the beam no matter how that line rotates during the analysis, the following set-up is recommended:

- The material card should contain only *r*-axis translational properties.
- SCOOR can be set to any of 2, -2, 12, or -12 with identical results.
- CID can be left blank or zero.

b) For applications intended to generate force in directions other than purely axial, and where at least one end of the discrete beam is connected to a relatively stiff component (for example, a suspension bush connecting two metal parts), fix the local axes in the stiff component as follows:

- Create a *DEFINE_COORDINATE_NODES based on three nodes in the stiff component and set FLAG = 1 (see Remark 8).
- Reference the *DEFINE_COORDINATE_NODES using CID on *SECTION_BEAM.
- Set SCOOR = 13.

c) For applications intended to generate force in directions other than purely axial, and where no suitable stiff component exists to carry the local axes (or if avoiding the need to create coordinate systems), the following process is recommended. Considering the other elements connected to Node 1 and Node 2 of the beam, estimate whether Node 1 or Node 2 will provide greater resistance to rotation in response to torques generated by the beam. For example, if Node 1 is connected to solid elements (which do not generate any rotational stiffness at their nodes), while Node 2 is connected to shell elements, choose Node 2. On the other hand, if Node 1 is connected to rigid elements, choose Node 1. Then set SCOOR = -14 if Node 1 was chosen; set SCOOR = 14 if Node 2 was chosen. These settings fix the beam's local axes in the most rotationally stable of its two nodes.

Note also that shell elements do not generate rotational stiffness for the "drilling" degree of freedom of their nodes. Creating a small rigid body or nodal rigid body to provide a rotationally stiff base for the node that carries the local axes may be helpful.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$   *SECTION_BEAM
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a Belytschko-Schwer resultant beam (elform = 2) with the following
$  properties.  This beam models the connection/stiffening beams of a
$  medium-sized roadside sign.
$
$    cross-sectional area:                        a =     515.6 mm2
$    2nd moment of area about s-axis:           iss =  99,660.0 mm4
$    2nd moment of area about t-axis:           iss =  70,500.0 mm4
$    2nd polar moment of area about beam axis:    j = 170,000.0 mm4
$
*SECTION_BEAM
```

```
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid    elform      shrf   qr/irid       cst
       111         2
$
$        a       iss       itt         j       sas
     515.6   99660.0   70500.0  170000.0
$
*SECTION_BEAM_TITLE
   Main beam member
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid    elform      shrf   qr/irid       cst
       111         2
$
$        a       iss       itt         j       sas
     515.6   99660.0   70500.0  170000.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

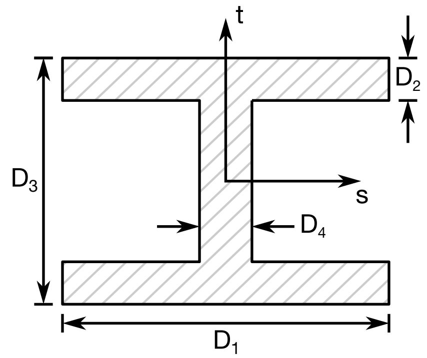


Figure 41-1. SECTION_01 ⇒ I-Shape

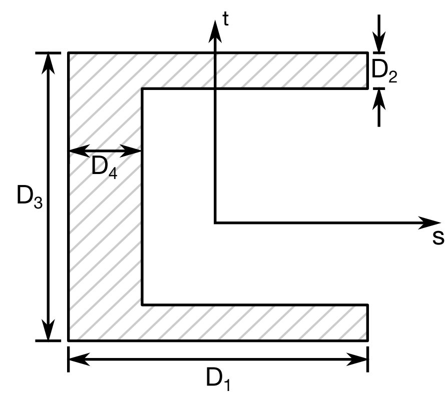


Figure 41-2. SECTION_02 ⇒ Channel

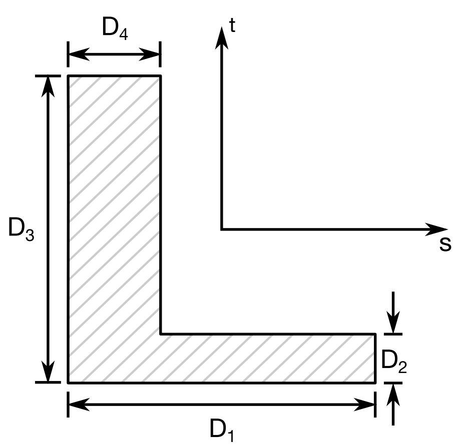


Figure 41-3. SECTION_03 ⇒ L-Shape

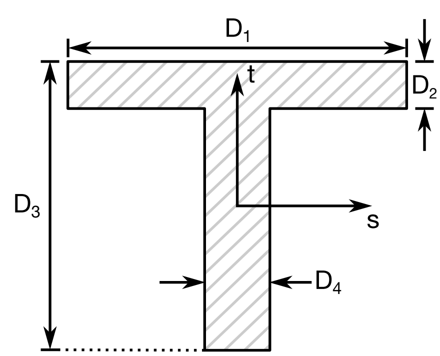


Figure 41-4. SECTION_04 ⇒ T-Shape

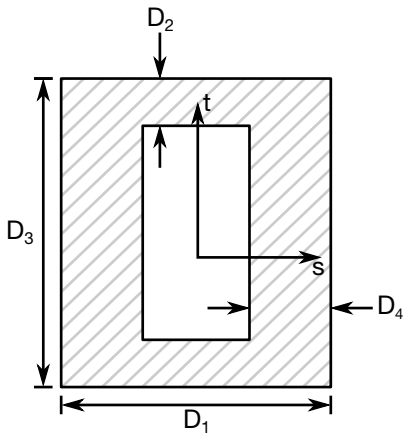


Figure 41-5. SECTION_05 ⇒ Box-Shape

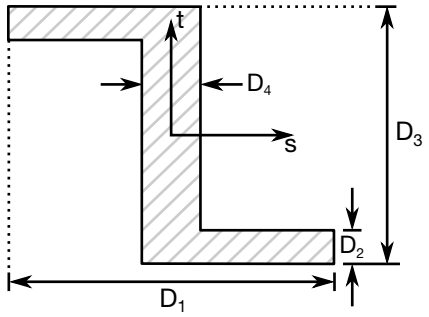


Figure 41-6. SECTION_06 ⇒ Z-Shape

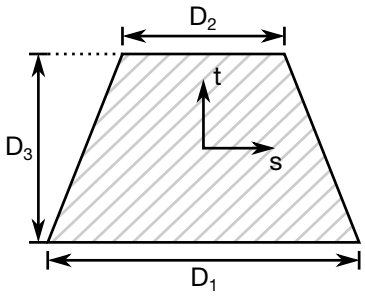


Figure 41-7. SECTION_07 ⇒ Trapezoidal

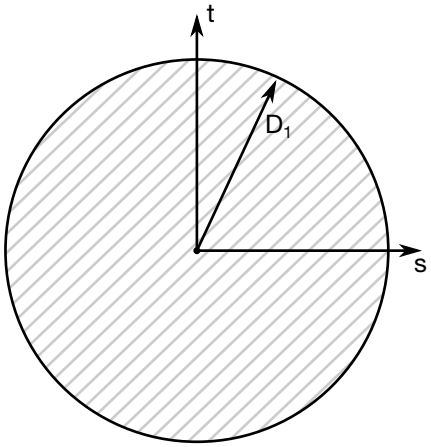


Figure 41-8. SECTION_08 ⇒ Circular

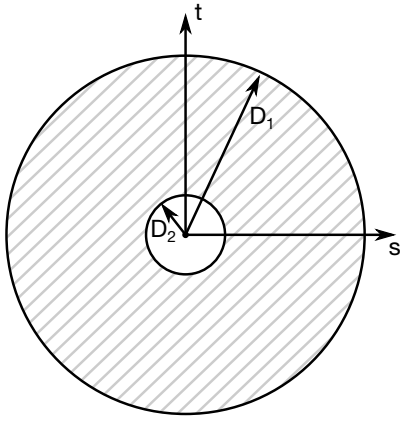


Figure 41-9. SECTION_09 ⇒ Tubular

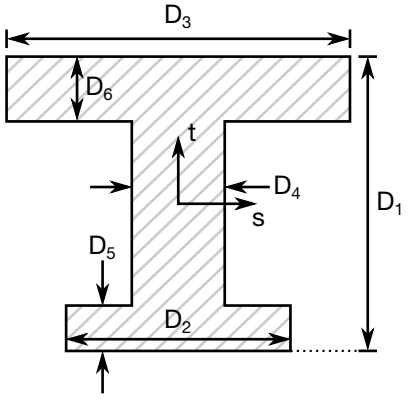


Figure 41-10. SECTION_10 ⇒ I-Shape 2

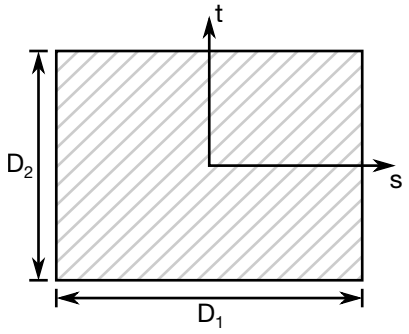


Figure 41-11. SECTION_11 ⇒ Solid Box

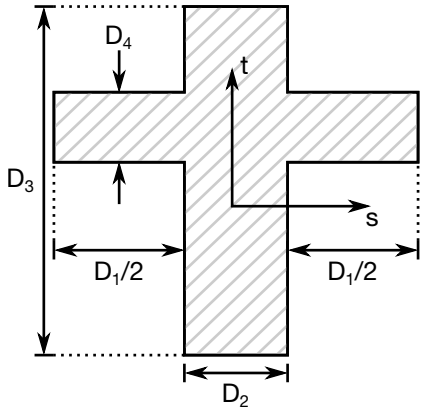


Figure 41-12. SECTION_12 ⇒ Cross

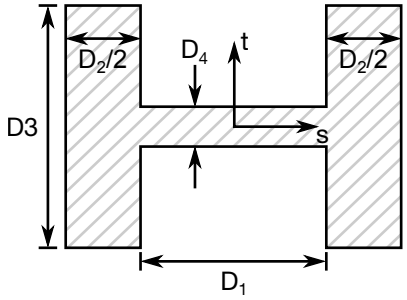


Figure 41-13. SECTION_13 ⇒ H-Shape

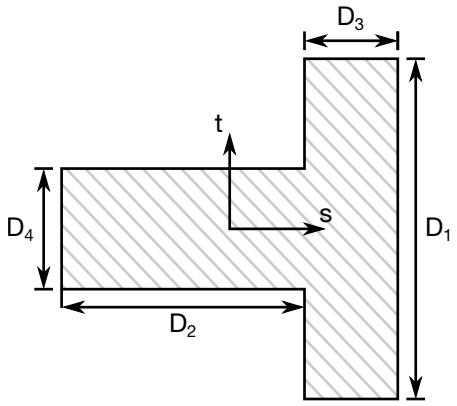


Figure 41-14. SECTION_14 ⇒ T-Shape 2

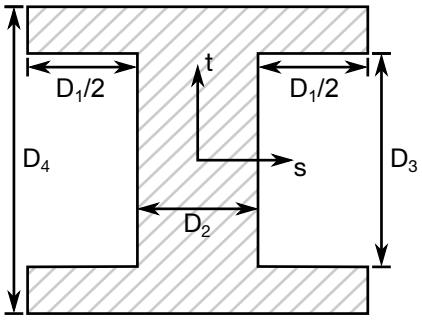


Figure 41-15. SECTION_15 ⇒ I-Shape 3

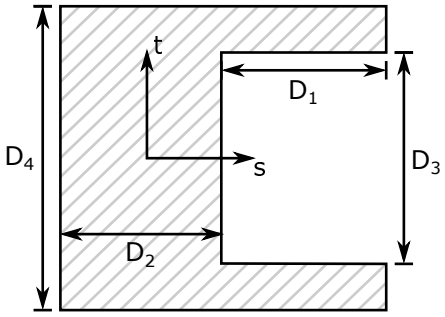


Figure 41-16. SECTION_16 ⇒ Channel 2

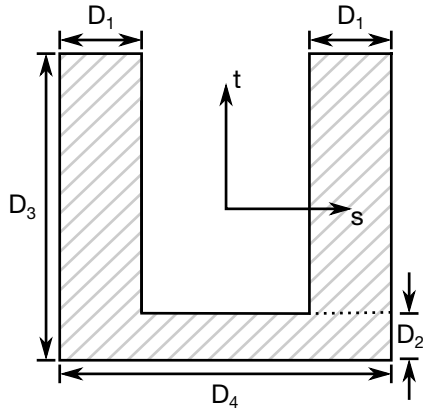


Figure 41-17. SECTION_17 ⇒ Channel 3

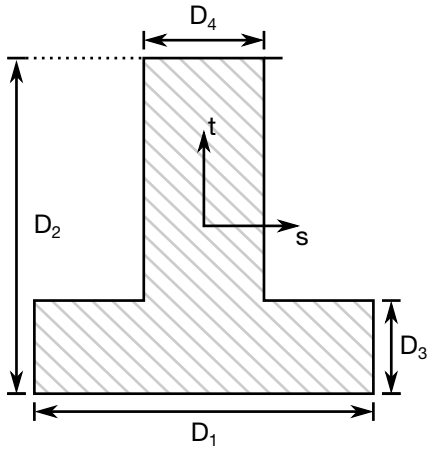


Figure 41-18. SECTION_18 ⇒ T-Shape 3

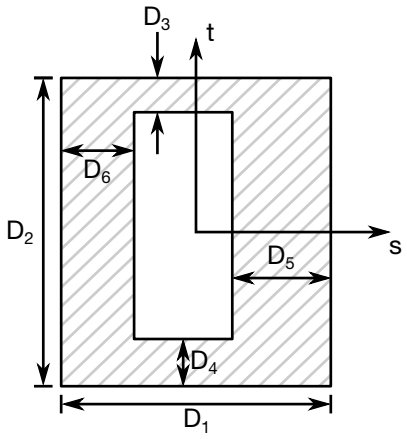


Figure 41-19. SECTION_19 ⇒ Box-Shape 2

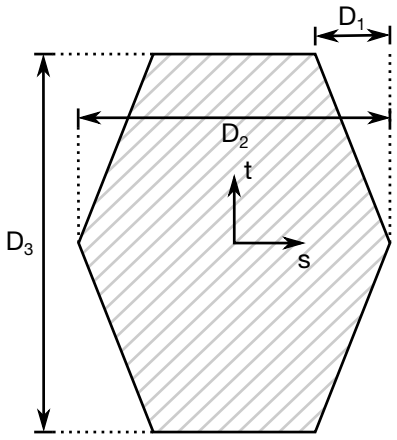


Figure 41-20. SECTION_20 ⇒ Hexagon

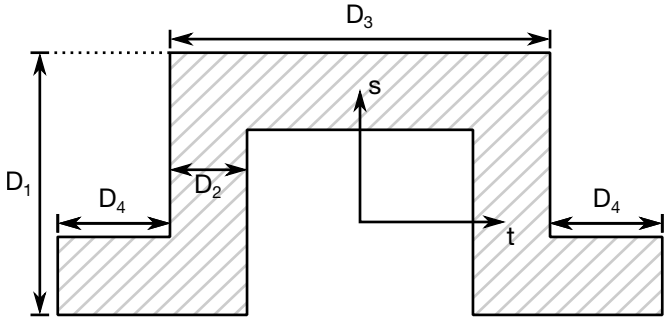


Figure 41-21. SECTION_21 ⇒ Hat Shape

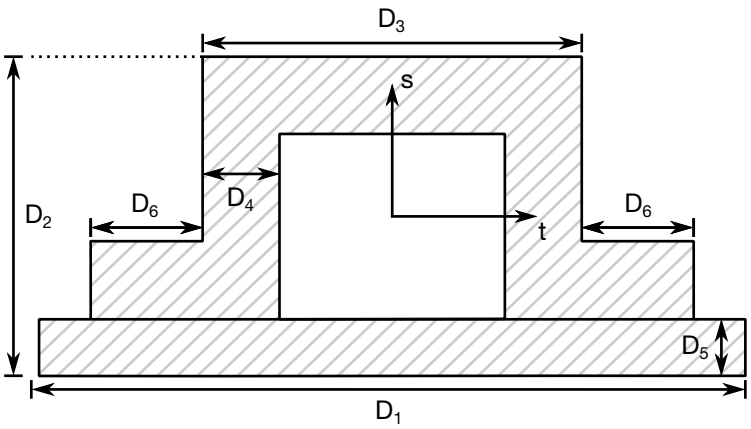


Figure 41-22. SECTION_22 ⇒ Hat Shape 2

*SECTION_BEAM_AISC

Purpose: Define cross-sectional properties for beams and trusses using section labels from the AISC Steel Construction Manual, 2005, 13th Edition, as published in the AISC Shapes Database V13.1.1

Card Summary:

Card Sets. For each BEAM_AISC section include one pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SECID	LABEL

Card 2a. Card 2 for ELFORM equal to 1 or 11.

ELFORM	SHRF	NSM	LFAC	NSLOC	NTLOC	K	

Card 2b. Card 2 for ELFORM equal to 2 or 12.

ELFORM	SHRF	NSM	LFAC				

Card 2c. Card 2 for ELFORM equal to 3.

ELFORM	LFAC	RAMPT	STRESS				

Card 2d. Card 2 for ELFORM equal to 4 or 5.

ELFORM	SHRF	NSM	LFAC	K			

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	LABEL						
Type	I	A70						

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
LABEL	AISC section label

Integrated Beam Card (types 1 and 11). Card 2 for ELFORM equal to 1 or 11.

Card 2a	1	2	3	4	5	6	7	8
Variable	ELFORM	SHRF	NSM	LFAC	NSLOC	NTLOC	K	
Type	I	F	F	F	F	F	I	

Resultant Beam Card (types 2 and 12). Card 2 for ELFORM equal to 2 or 12.

Card 2b	1	2	3	4	5	6	7	8
Variable	ELFORM	SHRF	NSM	LFAC				
Type	I	F	F	F				

Truss Beam Card (type 3). Card 2 for ELFORM equal to 3.

Card 2c	1	2	3	4	5	6	7	8
Variable	ELFORM	LFAC	RAMPT	STRESS				
Type	I	F	F	F				

Integrated Beam Card (types 4 and 5). Card 2 for ELFORM equal to 4 or 5.

Card 2d	1	2	3	4	5	6	7	8
Variable	ELFORM	SHRF	NSM	LFAC	K			
Type	I	F	F	F	F			

VARIABLE	DESCRIPTION
ELFORM	Element formulation (see *SECTION_BEAM). Only types 1–5,11,12 are allowed
SHRF	Shear factor (see *SECTION_BEAM)
NSM	Non-structural mass per unit length

VARIABLE	DESCRIPTION
LFAC	GT.0.0: Length scale factor to convert dimensions from standard units If LFAC < 0, then a predefined length factor for specific model units is used: EQ.-1.0: ft EQ.-2.0: m EQ.-3.0: in EQ.-4.0: mm EQ.-5.0: cm
NSLOC	Location of reference surface (see *SECTION_BEAM)
NTLOC	Location of reference surface (see *SECTION_BEAM)
K	Integration refinement parameter (see *INTEGRATION_BEAM)
RAMPT	Optional ramp-up time (see *SECTION_BEAM)
STRESS	Optional initial stress (see *SECTION_BEAM)

Remarks:

This keyword uses the dimensions of the standard AISC beams sections — as defined by the section label — to define *SECTION_BEAM and *INTEGRATION_BEAM cards with the appropriate parameters.

The AISC section label may be specified either as the shape designation as seen in the AISC Steel Construction Manual, 2005, or the designation according to the AISC Naming Convention for Structural Steel Products for Use in Electronic Data Interchange (EDI), 2001. As per the EDI convention, the section labels are to be case-sensitive and space sensitive, i.e. “W36X150” is acceptable but “W36 x 150” is not. Labels can be specified in terms of either the U.S. Customary units (in) or metric units (mm), which will determine the length units for the section dimensions. The parameter LFAC may be used as a multiplier to convert the dimensions to other lengths units. For user convenience, predefined conversion factors are provided for specific choices of the model length unit.

AISC requires the following legal notice specifying that LS-DYNA users may not extract AISC shapes data in any manner from LS-DYNA to create commercial software. Users are certainly allowed to create LS-DYNA input models for commercial purposes using this card.

AISC FLOW-DOWN LICENSE TERMS (TERMS OF USE)

This application contains software from the American Institute of Steel Construction, Inc. of Chicago, Illinois d/b/a AISC ("AISC"). The software from AISC (the "AISC Shapes Database") enables this application to provide dimensions and properties of structural steel shapes and to perform other functions. You may use AISC Data only by means of the intended End User functions of this application software. You agree that you will use AISC Data for non-commercial use only, meaning that it will not be used to develop or create revenue-producing software. You agree not to assign, copy, transfer or transmit the AISC Shapes Database or any AISC Data to any third party.
YOU AGREE NOT TO USE OR EXPLOIT AISC DATA OR THE AISC SHAPES DATA-BASE EXCEPT AS EXPRESSLY PERMITTED HEREIN.

*SECTION_DISCRETE

Purpose: Defined spring and damper elements for translation and rotation. These definitions must correspond with the material type selection for the elements, that is, *MAT_SPRING_... and *MAT_DAMPER_...

Card Sets. For each DISCRETE section include a pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	DRO	KD	V0	CL	FD		
Type	I/A	I	F	F	F	F		

Card 2	1	2	3	4	5	6	7	8
Variable	CDL	TDL						
Type	F	F						

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
DRO	Displacement/Rotation Option: EQ.0: the material describes a translational spring/damper, EQ.1: the material describes a torsional spring/damper.
KD	Dynamic magnification factor, k_d. See Remarks 1 and 2 below.
V0	Test velocity, V_0. See Remarks 1 and 2 below.
CL	Clearance. See Remark 3 below.
FD	Failure deflection (twist for DRO = 1). Negative for compression, positive for tension.
CDL	Deflection (twist for DRO = 1) limit in compression. See Remark 4 below.

VARIABLE	DESCRIPTION
TDL	Deflection (twist for DRO = 1) limit in tension. See Remark 4 below.

Remarks:

1. **Optional Constants.** The constants from KD to TDL are optional and do not need to be defined.

2. **Forces and Amplification Factors.** If k_d is nonzero, the forces computed from the spring elements are assumed to be the static values and are scaled by an amplification factor to obtain the dynamic value:

$$F_{\text{dynamic}} = \left(1. + k_d \frac{V}{V_0}\right) F_{\text{static}} \ ,$$

where

V = absolute value of the relative velocity between the nodes.

V_0 = dynamic test velocity.

For example, if it is known that a component shows a dynamic crush force at 15 m/s equal to 2.5 times the static crush force, use $k_d = 1.5$ and $V_0 = 15$.

3. **Clearance.** Here, "clearance" defines a compressive displacement which the spring sustains before beginning the force-displacement relation given by the load curve defined in the material selection. If a non-zero clearance is defined, the spring is compressive only.

4. **Deflection Limits.** The deflection limit in compression and tension is restricted in its application to no more than one spring per node subject to this limit and to deformable bodies only. For example, in the former case, if three springs are in series, either the center spring or the two end springs may be subject to a limit, but not all three. When the limiting deflection is reached, momentum conservation calculations are performed, and a common acceleration is computed in the appropriate direction. An error termination will occur if a rigid body node is used in a spring definition where deflection is limited.

   Constrained boundary conditions on the *NODE cards and the BOUNDARY_SPC cards must not be used for nodes of springs with deflection limits.

5. **Implicit.** Discrete elements can be included in implicit applications.

Examples:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *SECTION_DISCRETE
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Note: These examples are in kg, mm, ms, kN units.
$
$  A translational spring (dro = 0) is defined to have a failure deflection
$  of 25.4 mm (fd = 25.4).  The spring has no dynamic effects or
$  deflection limits, thus, those parameters are not set.
$
*SECTION_DISCRETE
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid       dro        kd        v0        cl        fd
       104         0                                    25.4
$
$      cdl       tdl

$
$
$  Define a translational spring that is known to have a dynamic crush force
$  equal to 2.5 times the static force at a 15 mm/ms deflection rate.
$  Additionally, the spring is known to be physically constrained to deflect
$  a maximum of 12.5 mm in both tension and compression.
$
*SECTION_DISCRETE
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid       dro        kd        v0        cl        fd
       107         0       1.5      15.0
$
$      cdl       tdl
      12.5      12.5
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*SECTION_FPD

Purpose: Define section properties for incompressible smoothed particle Galerkin (ISPG) element.

> **NOTE:** This keyword does not work with *ISPG or *MAT_ISPG keywords. It cannot be used in an ISPG input deck included with *INCLUDE_ISPG. The element formulations on this keyword are compatible with *MAT_319.

ISPG has been implemented as a 3D element formulation (ELFORM = 49) in *SECTION_FPD for fully implicit dynamic analysis. The ISPG solver automatically converts the nodes from the user's FEM model into the ISPG nodes.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM						
Type	I/A	I						

Card 2	1	2	3	4	5	6	7	8
Variable	DX	DY	DZ					
Type	F	F	F					
Default	1.5	1.5	1.5					

Card 3	1	2	3	4	5	6	7	8
Variable			TSTART	DT_IMP	RCFORCH	DTSCL		
Type			F	F	F	F		
Default			0.0	optional	optional	0.1		

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation options. EQ.49: Incompressible smoothed particle Galerkin formulation
DX, DY, DZ	Normalized dilation parameters of the kernel function in the x, y and z directions. The normalized dilation parameters of the kernel function are introduced to provide the smoothness and compact support properties on the construction of the mesh-free shape functions. Values between 1.4 and 1.8 are recommended. The nodal support size of particles are automatically adjusted with the material's deformation but cannot be decreased. An Updated Lagrangian (UL) kernel is used.
TSTART	Starting time for the fully implicit ISPG iterations. Before TSTART, only 10 ISPG iterations are done in each thermal-structural implicit step with one-way coupling (no forces from the fluid are returned to the structure) to guarantee that the fluid moves with the solid boundaries. After TSTART, the ISPG algorithm performs full iterations in the two-way coupled thermal-structural implicit analysis. This option is very useful for cases where the structural simulation time is very long (e.g., in seconds or minutes) while the reflow process to a steady state is very short. With this option, the full ISPG iterations start from TSTART, saving some computational resources.
DT_IMP	Reset the implicit structural time step size to DT_IMP after TSTART. Because the solder reflow process is very fast, a small implicit structural time step size is needed. Generally, the value of DT_IMP should be around 10 to 50 times the ISPG time step size to guarantee convergence of the solution if the gravity-driven simulation is deployed. This field is optional.
RCFORCH	Define the z coordinate of the cutting plane for evaluating the reaction forces between *all* the solders and the structure. If RCFORCH is nonzero, two files are generated in the working folders: fpdcpl_u.txt and fpdcpl_l.txt. The first file contains the reaction force history of the solders that are above the cutting plane, while the second file contains the reaction force history of those below the cutting plane.

VARIABLE	DESCRIPTION
DTSCL	The time step size scaling factor for ISPG iteration. We recommend a value between 0.1 and 0.5. Large DTSCL may cause contact detection issues.

*SECTION_IGA_SHELL

Purpose: Define section properties for isogeometric shell elements.

Card Summary:

Card Sets. For each isogeometric shell section include one set of data cards. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SECID	ELFORM	SHRF	NIP	IRL	QR / IRID	ICOMP	

Card 2. This card is required.

T				NLOC			

Card 3. Include ceil(NIP/8) cards if ICOMP = 1.

B1	B2	B3	B4	B5	B6	B7	B8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	SHRF	NIP	IRL	QR / IRID	ICOMP	
Type	I/A	I	F	F	I	F	I	
Default	none	0	1.0	2.0	0	0.0	0	

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation: EQ.0: Reissner-Mindlin with fibers at the control points EQ.1: Kirchhoff-Love with fibers at the control points EQ.2: Kirchhoff-Love with fibers at the integration points EQ.3: Reissner-Mindlin with fibers at the integration points

VARIABLE	DESCRIPTION
	EQ.5: Thick shell with thickness stretch based on the ELFORM = 0. See Remark 1.
	EQ.6: Thick shell with thickness stretch based on ELFORM = 3. See Remark 1.
SHRF	Shear correction factor which scales the transverse shear stress; see Remark 2.
NIP	Number of through thickness integration points (up to 10 points). See Remark 3.
IRL	Lamina integration rule: EQ.0: Reduced Gauss-Legendre EQ.1: Gauss-Legendre EQ.2: Patchwise reduced Gauss-Legendre (for biquadratic NURBS only) EQ.3: Generalized Gaussian Quadrature rule. It only works with a uniform knot vector with UNIR/S = 1 or 2 in *IGA_2D_NURBS_XYZ.
QR/IRID	Fiber quadrature rule or fiber integration rule ID (see *INTEGRATION_SHELL): LT.0.0: Absolute value is specified rule number. EQ.0.0: Gauss-Legendre/Gauss-Lobatto (up to 10 points) EQ.1.0: Trapezoidal, not recommended for accuracy reasons
ICOMP	Flag for anisotropic layered composite material model (see Remark 4): EQ.1: A material angle in degrees is defined for each through thickness integration point. Thus, each layer has one integration point.

Card 2	1	2	3	4	5	6	7	8
Variable	T				NLOC			
Type	F				F			
Default	0.0				0.0			

VARIABLE	DESCRIPTION
T	Shell thickness
NLOC	Location of reference surface; see Remark 5.

Material Orientation Angle Cards. Include ceil(NIP/8) cards if ICOMP = 1 in order to define material orientation angle with respect to the baseline orientation at the fiber integration points.

Card 3	1	2	3	4	5	6	7	8
Variable	B1	B2	B3	B4	B5	B6	B7	B8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
B*i*	Material angle at the i^{th} fiber integration point

Remarks:

1. **Thick Shell Formulations.** Element formulations 5 and 6 permit linear strains to develop in the thickness direction in a manner similar to the type 25 and 26 finite element shells (see *SECTION_SHELL). Thus solid materials with damage models that depend on the full stress tensor are compatible with these elements. For instance, these elements can be used with ductile fracture models that require accurate pressures for the nucleation and growth of voids.

2. **Shear Correction Factor.** Reissner-Mindlin shell formulations (FORM = 0 or 3) are based on a first order shear deformation theory that yields constant transverse shear strains. This approximation violates the condition of zero traction on the top and bottom surfaces of the shell. The shear correction factor is an

attempt to compensate for this error. A suggested value is 5/6 for isotropic materials. This value is incorrect for sandwich or laminated shells; consequently, laminated/sandwich shell theory is now an option in some of the constitutive models, that is, material types 22, 54, and 55.

3. **Fiber Integration Rule.** Either Gauss-Legendre (default) or Gauss-Lobatto integration can be used to integrate along the fiber. The flag for Gauss-Lobatto integration is set with INTGRD in *CONTROL_SHELL.

   If NIP is 0 or 1 and the *MAT_SIMPLIFIED_JOHNSON_COOK model is used, then a resultant plasticity formulation is activated. NIP is always set to 1 if a constitutive model based on resultants is used.

4. **Material Orientation.** This option applies to material types 21, 22, 23, 33, 33_96, 34, 36, 40, 41-50, 54, 55, 58, 59, 103, 103_P, 104, 108, 116, 122, 133, 135, 135_PLC, 136, 157, 158, 190, 219, 226, 233, 234, 235, 242, 243, 261, and 262. See also Remark 5 under *MAT_034 for additional information specific to fiber directions for fabrics.

5. **Shell Offset.** If NLOC $\neq 0$, the offset distance from the mid-surface to the reference surface of the shell in the direction of the shell normal vector is computed as

$$\text{offset} = -0.5 \times \text{NLOC} \times (\text{average shell thickness}).$$

   Except for Mortar contacts, this offset is not considered in the contact subroutines unless CNTCO is set to 1 or 3 in *CONTROL_SHELL.

*SECTION_IGA_SOLID

Purpose: Define section properties for isogeometric solid elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	IR					
Type	I/A	I	I					
Default	none	0	0					

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation: EQ.0: Standard
IR	Integration rule: EQ.0: Reduced Gauss-Legendre EQ.1: Gauss-Legendre EQ.2: Patchwise reduced Gauss-Legendre (for biquadratic NURBS only) EQ.3: Generalized Gaussian Quadrature rule. It only works with a uniform knot vector with UNIR/S/T = 1 or 2 in *IGA_3D_NURBS_XYZ.

*SECTION_ISPG

Purpose: Set section properties for incompressible smoothed particle Galerkin (ISPG) elements.

> **NOTE:** This keyword may only be used in an ISPG input deck included with *INCLUDE_ISPG.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM						
Type	I/A	I						

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation options. EQ.49: 3D and pseudo 2D ISPG formulation. For pseudo 2D simulations,*ISPG_BOUNDARY_SYMMETRY must specify the symmetric boundary condition. EQ.50: Planar 2D ISPG formulation. One layer of elements should be defined in the thickness direction. *ISPG_-BOUNDARY_SYMMETRY must set the symmetric boundary condition. EQ.51: Axisymmetric 2D ISPG formulation. Notice that (r,z) for the axisymmetric definition must be (x,z). Because remeshing is based on 3D remeshing, the model needs one layer of elements in the y-direction with the same thickness. *ISPG_BOUNDARY_SYMMETRY defines the symmetric boundary condition. The surface with CO-ORD1 defined in *ISPG_BOUNDARY_SYMMETRY gives the axisymmetric surface (see Figure 41-23).

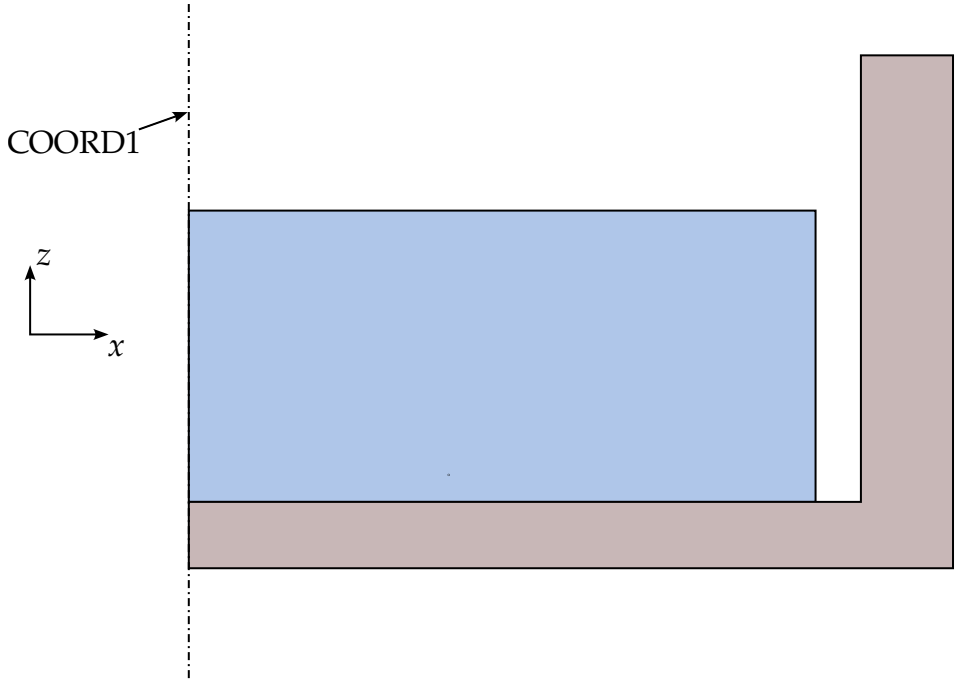


Figure 41-23. Example of an axisymmetric geometry

*SECTION_POINT_SOURCE

Purpose: This command provides the inlet boundary condition for single gas in flow (inflation potential) using a set of point source(s). It also provides the inflator orifice geometry information. It requires 3 curves defining the inlet condition for the inflator gas coming into the tank or an airbag as input ($\overline{T}_{\text{gas corrected}}(t)$, $v_r(t)$, and vel(t)). Please see also the *ALE_TANK_TEST card for additional information.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	LCIDT	LCIDVR	LCIDVEL	NIDLC1	NIDLC2	NIDLC3	
Type	I/A	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Source Node Cards. Include one card for each source node. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	NODEID	VECID	ORIFA					
Type	I	I	F					
Default	0	0	0.0					

VARIABLE	DESCRIPTION
SECID	Section ID. A unique number or label must be specified.
LCIDT	Temperature load curve ID
LCIDVR	Relative volume load curve ID
LCIDVEL	Inlet flow velocity load curve ID
NIDLC1	The 1st node ID defining a local coordinate (See Remark 2).
NIDLC2	The 2nd node ID defining a local coordinate (See Remark 2).

VARIABLE	DESCRIPTION
NIDLC3	The 3rd node ID defining a local coordinate (See Remark 2).
NODEID	The node ID(s) defining the point source(s).
VECID	The vector ID defining the direction of flow at each point source.
ORIFA	The orifice area at each point source.

Remarks:

1. **Airbag Inflator Tank Test.** In an airbag inflator tank test, the tank pressure data is measured. This pressure is used to derive $\dot{m}(t)$ and the estimated $\overline{T}_{\text{gas}}(t)$, usually using a lumped-parameter method, a system of conservation equations and EOS. Subsequently $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ (stagnation temperature) are used as input to obtain $\overline{T}_{\text{gas_corrected}}(t)$ (static temperature), $v_r(t)$, and vel(t). These 3 curves are then used to describe inflator gas inlet condition (see *ALE_TANK_TEST for more information).

2. **Local Coordinate System.** In a car crash model, the inflator housing may get displaced during the impact. The 3 node IDs defines the local reference coordinate system to which the point sources are attached. These 3 reference nodes may be located on a rigid body which can translate and rotate as the inflator moves during the impact. This allows for the point sources to move in time. These reference nodes may be used as the point sources themselves.

3. **ALE Tank Test Keyword.** If the *ALE_TANK_TEST card is present, please see the Remarks under that card.

Example:

Consider a tank test model which consists of the inflator gas (PID 1) and the air inside the tank (PID 2). The 3 load curves define the thermodynamic and kinetic condition of the incoming gas. The nodes define the center of the orifice, and the vector the direction of flow at each orifice.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*PART
inflator gas
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1         0         0         0         0         0
*SECTION_POINT_SOURCE
$    SECID     LCIDT  LCIDVOLR   LCIDVEL NIDLCOOR1 NIDLCOOR2 NIDLCOOR3
         1         3         4         5         0         0         0
$   NODEID    VECTID      AREA
     24485         3    15.066
```

```
       ...
     24557         3    15.066
*PART
air inside the tank
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         2         2         2         0         0         0         0         0 *SEC-
TION_SOLID
$    SECID    ELFORM       AET
         2        11         0
*ALE_MULTI-MATERIAL_GROUP
$      SID   SIDTYPE
         1         1
         2         1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*SECTION_POINT_SOURCE_MIXTURE

Purpose: This command provides (a) an element formulation for a solid ALE part of the type similar to ELFORM = 11 of *SECTION_SOLID, and (b) the inlet gas injection boundary condition for multiple-gas mixture in-flow using a set of point source(s). It also provides the inflator orifice geometry information. This must be used in combination with the *MAT_GAS_MIXTURE and/or *INITIAL_GAS_MIXTURE card (see Remark 1).

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	LCIDT	Not Used	LCIDVEL	NIDLC1	NIDLC2	NIDLC3	IDIR
Type	I/A	I		I	I	I	I	I
Default	none	none		0	none	none	none	0

Card 2	1	2	3	4	5	6	7	8
Variable	LCMD1	LCMD2	LCMD3	LCMD4	LCMD5	LCMD6	LCMD7	LCMD8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Source Node Cards. Include one card for each source node. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	NODEID	VECID	ORIFA					
Type	I	I	F					
Default	none	none	0.0					

VARIABLE	DESCRIPTION
SECID	Section ID. A unique number or label must be specified.

VARIABLE	DESCRIPTION
LCIDT	Inflator gas mixture average stagnation temperature load curve ID (all gases of the mixture are assumed to have the same average temperature).
LCIDVEL	User-defined inflator gas mixture average velocity load curve ID. If LCIDVEL = 0 or blank, LSDYNA will estimate the inlet gas velocity.
NIDLC001	The 1st node ID defining a local coordinate (see Remark 2).
NIDLC002	The 2nd node ID defining a local coordinate (see Remark 2).
NIDLC003	The 3rd node ID defining a local coordinate (see Remark 2).
IDIR	A flag for constraining the nodal velocity of the nodes of the ALE element containing a point source. EQ.0: ALE nodes behind the point source (relative position of nodes based on the vector direction of flow of point source) will have zero velocity (default). EQ.1: all ALE nodes will have velocity distributed based on energy conservation. This option seems to be more robust in airbag modeling (see Remark 5).
LCMD*n*	The mass flow rate load curve ID of the n^{th} gas in the mixture. See Remark 3.
NODEID	The node ID(s) defining the point sources (see Remark 7).
VECID	The vector ID defining the direction of flow at each point source.
ORIFA	The orifice area at each point source.

Remarks:

1. **Keyword Applications.** This command is used to define a part that acts as the ideal gas mixture injection source. The associated ALE material (gas mixture) may not be present at time zero but can be introduced (injected) into an existing ALE domain. For airbag applications, the input from control volume analysis, inlet mass flow rate, $\dot{m}(t)$, and, inlet stagnation gas temperature, $\overline{T}_{\text{gas}}(t)$ may be used as direct input for ALE analysis. If available, the user may input a load curve for the gas mixture average inlet velocity. If not, LS-DYNA will estimate the inlet gas velocity.

2. **Local Coordinate System.** In a car crash model, the inflator housing may get displaced during the impact. The 3 node IDs defines the local reference coordinate system to which the point sources are attached. These 3 reference nodes may be located on a rigid body which can translate and rotate as the inflator moves during the impact. This allows for the point sources to move in time. These reference nodes may be used as the point sources themselves.

3. **Mass Flow Rate.** The gas mixture is assumed to have a uniform temperature ($\overline{T} \approx T_i$) and inlet velocity. However, the species in the mixture may each have a different inlet mass flow rate.

4. **Model.** A brief review of the concept used is presented. The total energy, e_t, is the sum of internal (e_i) and kinetic ($V^2/2$) energies (per unit mass).

$$e_T = e_i + \frac{V^2}{2}$$
$$C_V T_{stag} = C_V T + \frac{V^2}{2}$$
$$T_{stag} = T + \frac{V^2}{2C_V}$$

The distinction between stagnation and static temperatures is shown above. C_V is the constant-volume heat capacity. The gas mixture average internal energy per unit mass in terms of mixture species contribution is

$$e_i = \bar{C}_V \overline{T} = \sum_i \left(\frac{\rho_i}{\rho_{mixture}}\right) C_{V_i} T_i = \left[\sum_i \left(\frac{\rho_i}{\rho_{mixture}}\right) C_{V_i}\right] \overline{T}$$
$$\bar{C}_V = \left[\sum_i \left(\frac{\rho_i}{\rho_{mixture}}\right) C_{V_i}\right]$$

Since we approximate $\overline{T} \approx T_i$, then gas mixture average static temperature is related to the mixture average internal energy per unit mass as following

$$\overline{T} = \frac{e_i}{\left[\sum_i \left(\frac{\rho_i}{\rho_{mixture}}\right) C_{V_i}\right]}$$

Note that the "i" subscript under "e" denotes "internal" energy, while the other "i" subscripts denote the "i^{th}" species in the gas mixture. The total mixture pressure is the sum of the partial pressures of the individual species.

$$\bar{p} = \sum_i p_i$$

The ideal gas EOS applies to each individual species (by default)

$$P_i = \rho_i (C_{P_i} - C_{V_i}) T_i$$

5. **Energy Conservation.** Generally, conservation of both momentum and kinetic (KE) at the same time is impossible. Typically, internal energy (IE) is conserved

and KE may not be. This may result in some KE loss (hence, total energy loss). For many analyses this is tolerable, but for an airbag, a reduction of the inflating potential of the inflator gas may result.

In *MAT_GAS_MIXTURE computation, any kinetic energy not accounted for during advection is stored in the internal energy. Therefore, no kinetic energy is lost, and the total energy of the element is conserved over the advection step. This simple, ad hoc approach is not rigorously derived for the whole system based on first principles. Therefore, it is not guaranteed to apply universally to all scenarios. The user must validate the model with data.

6. **EOS.** Since ideal gas is assumed, there is no need to define the EOS for the gases in the mixture.

7. **Point Source Location.** In general, it is best to locate a point source near the center of an ALE element. Associated with each point source is an area and a vector indicating flow direction. Each point source should occupy one ALE element by itself, and there should be at least two empty ALE elements between any two point sources. A point source should be located at least three elements away from the free surface of an ALE mesh for stability.

Example 1:

Consider a tank test model without coupling which consists of:

-a background mesh with air (PID 1 = gas 1) initially inside that mesh (tank space), and
-the inflator gas mixture (PID 2 consisting of inflator gases 2, 3, and 4).

The mixture is represented by one AMMGID and the air by another AMMGID.

The tank internal space is simply modeled with an Eulerian mesh of the same volume. The tank itself is not modeled thus no coupling is required. The inflator gases fill up this space mixing with the air initially inside the tank.

The background air (gas 1) is included in the gas mixture definition in this case because that air will participate in the mixing process. Only include in the mixture those gases that actually undergo mixing (gases 1, 2, 3 and 4). Note that for an airbag model, the "outside" air should not be included in the mixture (it should be defined independently) since it does not participate in the mixing inside the airbag. This is shown in the next example.

The nodes define the center of the orifices, and the vectors define the directions of flow at these orifices.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*PART
Tank background mesh, initially filled with air, allows gas mixture to flow in.
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1         0         0         0         0         0
*SECTION_SOLID
$    SECID    ELFORM       AET
         1        11         0
$ The next card defines the properties of the gas species in the mixture.
*MAT_GAS_MIXTURE
$      MID
         1
$      Cv1       Cv2       Cv3       Cv4       Cv5       Cv6       Cv7       Cv8
    654.47    482.00   2038.30    774.64       0.0       0.0       0.0       0.0
$      Cp1       Cp2       Cp3       Cp4       Cp5       Cp6       Cp7       Cp8
    941.32    666.67   2500.00   1071.40       0.0       0.0       0.0       0.0

$ The next card specifies that gas 1 (background air) occupies PID 1 at time 0.
*INTIAL_GAS_MIXTURE
$      SID     STYPE    AMMGID     TEMP0
         1         1         1    293.00
$     RHO1      RHO2      RHO3      RHO4      RHO5      RHO6      RHO7      RHO8
   1.20E-9       0.0       0.0       0.0       0.0       0.0       0.0       0.0
*PART
The gas mixture (inlet) definition (no initial mesh required for this PID)
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         2         2         1         0         0         0         0         0
*SECTION_POINT_SOURCE_MIXTURE
$    SECID     LCIDT   NOTUSED   LCIDVEL NIDLCOOR1 NIDLCOOR2 NIDLCOOR3      IDIR
         2         1         0         5         0         0         0         0
$  LCMDOT1   LCMDOT2   LCMDOT3   LCMDOT4   LCMDOT5   LCMDOT6   LCMDOT7   LCMDOT8
         0         2         3         4         0         0         0         0
$   NODEID    VECTID      AREA
     24485         1      25.0
       ...
     24557         1      25.0
*ALE_MULTI-MATERIAL_GROUP
$      SID   SIDTYPE
         1         1
         2         1
*DEFINE_VECTOR
$   VECTID     XTAIL     YTAIL     ZTAIL     XHEAD     YHEAD     ZHEAD
         1       0.0       0.0       0.0       0.0       1.0       0.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

Example 2:

Consider an airbag inflation model which consists of:

-a background Eulerian mesh for air initially outside the airbag (PID 1)
-the inflator gas mixture (PID 2 consisting of inflator gases 1, 2, and 3).

The mixture is represented by one AMMGID and the air by another AMMGID.

The background air (PID 1) is NOT included in the gas mixture definition in this case because that air will NOT participate in the mixing process. Only include in the mixture those gases that actually undergo mixing (gases 1, 2, and 3). Gases 1, 2, and 3 in this example correspond to gases 2, 3, and 4 in example 1. Compare the air properties in PID

1 here to that of example 1. Note that the *INITIAL_GAS_MIXTURE card is not required to initialize the background mesh in this case.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*PART
Tank background mesh, initially filled with air, allows gas mixture to flow in.
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1         0         0         0         0         0
*SECTION_SOLID
$    SECID    ELFORM       AET
         1        11         0
*MAT_NULL
$      MID       RHO      PCUT        MU     TEROD     CEROD        YM        PR
         1   1.20E-9   -1.0E-6       0.0       0.0       0.0       0.0       0.0
*EOS_IDEAL_GAS
$    EOSID       CV0       CP0     COEF1     COEF2        T0    RELVOL0
         1    654.47    941.32       0.0       0.0    293.00       1.0
$ The next card defines the properties of the gas species in the mixture.
*PART
The gas mixture (inlet) definition (no initial mesh required for this PID)
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         2         2         2         0         0         0         0         0 *SEC-
TION_POINT_SOURCE_MIXTURE
$    SECID     LCIDT   NOTUSED   LCIDVEL NIDLCOOR1 NIDLCOOR2 NIDLCOOR3      IDIR
         2         1         0         5         0         0         0         0
$  LCMDOT1   LCMDOT2   LCMDOT3   LCMDOT4   LCMDOT5   LCMDOT6   LCMDOT7   LCMDOT8
         2         3         4         0         0         0         0         0
$   NODEID    VECTID      AREA
     24485         1      25.0
       ...
     24557         1      25.0
*MAT_GAS_MIXTURE
$      MID
         2
$      Cv1       Cv2       Cv3       Cv4       Cv5       Cv6       Cv7       Cv8
    482.00   2038.30    774.64       0.0       0.0       0.0       0.0
$      Cp1       Cp2       Cp3       Cp4       Cp5       Cp6       Cp7       Cp8
    666.67   2500.00   1071.40       0.0       0.0       0.0       0.0
$ The next card specifies that gas 1 (background air) occupies PID 1 at time 0.
*ALE_MULTI-MATERIAL_GROUP
$      SID   SIDTYPE
         1         1
         2         1
*DEFINE_VECTOR
$   VECTID     XTAIL     YTAIL     ZTAIL     XHEAD     YHEAD     ZHEAD
         1       0.0       0.0       0.0       0.0       1.0       0.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*SECTION_SEATBELT

Purpose: Define section properties for the seat belt elements. This card is required for the *PART Section. Currently, only the ID is required.

Seatbelt Section Cards. Include one card for each SEATBELT section. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	AREA	THICK					
Type	I/A	F	F					
Default	none	0.01	↓					

VARIABLE	DESCRIPTION
SECID	Section ID. A unique number or label must be specified.
AREA	Optional area of cross-section used in the calculation of contact stiffness, which is proportional to the cross-section area.
THICK	Optional contact thickness which can be overwritten by a nonzero SAST defined in *CONTACT. If not defined, a value proportional to element length is used as the contact thickness.

Remarks:

Seatbelt elements are implemented for both explicit and implicit calculations.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *SECTION_SEATBELT
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a seat belt section that is referenced by part 10.  Nothing
$  more than the sid is required.
$
*SECTION_SEATBELT
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid
       111
$
$
*PART
```

```
Seatbelt material
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      pid       sid       mid     eosid      hgid    adpopt
        10       111       220
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

***SECTION_SHELL**_*{OPTION}*

Available options include:

<BLANK>

EFG

THERMAL

XFEM

MISC

Purpose: Define section properties for shell elements.

Card Summary:

Card Sets. For each shell section, of a type matching the keyword's options, include one set of data cards. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SECID	ELFORM	SHRF	NIP	PROPT	QR / IRID	ICOMP	SETYP

Card 2. This card is required.

T1	T2	T3	T4	NLOC	MAREA	IDOF	EDGSET

Card 3. Additional cards for ICOMP = 1. Include the minimum number of cards necessary to input NIP values: 8 values per card ⇒ number of cards = ceil(NIP/8) where ceil(x) = the smallest integer greater than x.

B1	B2	B3	B4	B5	B6	B7	B8

Card 4a. Include this card if using the EFG keyword option.

DX	DY	ISPLINE	IDILA	IEBT	IDIM		

Card 4b. Include this card if using the THERMAL keyword option.

ITHELFM							

Card 4c. Include this card if using the XFEM keyword option.

CMID	BASELM	DOMINT	FAILCR	PROPCR	FS	LS/FS1	NC/CL

Card 4d. Include this card if using the MISC keyword option.

THKSCL							

Card 5. Additional card for ELFORM = 101, 102, 103, 104 or 105.

NIPP	NXDOF	IUNF	IHGF	ITAJ	LMC	NHSV	ILOC

Card 5.1. Additional card for ELFORM = 101, 102, 103, 104 or 105. Define NIPP of this card.

XI	ETA	WGT					

Card 5.2. Additional card for ELFORM = 101, 102, 103, 104 or 105. Include the minimum number of cards necessary to input LMC values: 8 values per card ⇒ number of cards = ceil(LMC/8) where ceil(x) = the smallest integer greater than x.

P1	P2	P3	P4	P5	P6	P7	P8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	SHRF	NIP	PROPT	QR / IRID	ICOMP	SETYP
Type	I/A	I	F	F	F	F	I	I
Default	none	none	1.0	2.0	0.0	0.0	0	1

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation options (see Remarks 1 and 3 below): EQ.1: Hughes-Liu EQ.2: Belytschko-Tsay EQ.3: BCIZ triangular shell EQ.4: C0 triangular shell EQ.5: Belytschko-Tsay membrane

VARIABLE	DESCRIPTION
	EQ.6: Selectively reduced Hughes-Liu
	EQ.7: Selectively reduced, co-rotational Hughes-Liu
	EQ.8: Belytschko-Leviathan shell
	EQ.9: Fully integrated Belytschko-Tsay membrane
	EQ.10: Belytschko-Wong-Chiang
	EQ.11: Fast (co-rotational) Hughes-Liu
	EQ.12: Plane stress (*xy*-plane)
	EQ.13: Plane strain (*xy*-plane)
	EQ.14: Axisymmetric solid (*xy*-plane, *y*-axis of symmetry) - area weighted (see Remark 11)
	EQ.15: Axisymmetric solid (*xy*-plane, *y*-axis of symmetry) - volume weighted
	EQ.16: Fully integrated shell element (very fast). See Remark 14.
	EQ.-16: Fully integrated shell element modified for higher accuracy. See Remark 15.
	EQ.17: Fully integrated DKT, triangular shell element. See Remark 10.
	EQ.18: Fully integrated linear DK quadrilateral/triangular shell. See Remarks 2 and 3.
	EQ.20: Fully integrated linear assumed strain C0 shell. See Remark 3.
	EQ.21: Fully integrated linear assumed strain C0 shell (5 DOF)
	EQ.22: Linear shear panel element. 3 DOF per node. See Remark 4.
	EQ.23: 8-node quadratic quadrilateral shell (see IRQUAD in *CONTROL_SHELL)
	EQ.24: 6-node quadratic triangular shell
	EQ.25: Belytschko-Tsay shell with thickness stretch
	EQ.26: Fully integrated shell with thickness stretch
	EQ.27: C0 triangular shell with thickness stretch
	EQ.29: Cohesive shell element for edge-to-edge connection of shells. See Remark 13.

VARIABLE	DESCRIPTION
EQ.-29:	Cohesive shell element for edge-to-edge connection of shells (more suitable for pure shear). See Remark 13.
EQ.30:	Fast fully integrated element with 2 in-plane integration points based on ELFORM 16 (see Remark 14)
EQ.41:	Mesh-free (EFG) shell local approach (more suitable for crashworthiness analysis).
EQ.42:	Mesh-free (EFG) shell global approach (more suitable for metal forming analysis).
EQ.43:	Mesh-free (EFG) plane strain formulation (xy-plane)
EQ.44:	Mesh-free (EFG) axisymmetric solid formulation (xy-plane, y-axis of symmetry)
EQ.46:	Cohesive element for two-dimensional plane strain, plane stress, and area-weighted axisymmetric problems (use with type 14 shells). See Remark 17.
EQ.47:	Cohesive element for two-dimensional volume-weighted axisymmetric problems (use with type 15 shells). See Remark 17.
EQ.52:	Plane strain (xy-plane) XFEM, base element type 13 with full integration. See Remark 9.
EQ.54:	Shell XFEM, base element type defined by BASELM (default 2). See Remark 9.
EQ.55:	8-node singular plane strain (xy-plane) finite element. See Remark 12.
EQ.98:	Interpolation shell
EQ.99:	Simplified linear element for time-domain vibration studies. See Remark 5.
EQ.101:	User defined shell
EQ.102:	User defined shell
EQ.103:	User defined shell
EQ.104:	User defined shell
EQ.105:	User defined shell
EQ.201:	Isogeometric shells with NURBS. See *ELEMENT_SHELL_NURBS_PATCH.
GE.1000:	Generalized shell element formulation (user defined). See *DEFINE_ELEMENT_GENERALIZE_SHELL.

VARIABLE	DESCRIPTION
	Note that the 2D and 3D element types must not be mixed, and different types of 2D elements must not be used together. For example, two-dimensional axisymmetric calculations can use either element types 14 or 15, but these element types must not be mixed together. Likewise, the plane strain element type must not be used with either the plane stress element or the axisymmetric element types. In three dimensions, the different shell element types, i.e., 1-11 and 16, can be freely mixed together.
SHRF	Shear correction factor which scales the transverse shear stress. The shell formulations in LS-DYNA, with the exception of the BCIZ and DK elements, are based on a first order shear deformation theory that yields constant transverse shear strains which violates the condition of zero traction on the top and bottom surfaces of the shell. The shear correction factor is an attempt to compensate for this error. A suggested value is 5/6 for isotropic materials. This value is incorrect for sandwich or laminated shells. Consequently, laminated/sandwich shell theory is now an option in some of the constitutive models, that is, material types 22, 54, and 55.
NIP	Number of through-thickness integration points. Either Gauss (default) or Lobatto integration can be used. INTGRD in *CONTROL_SHELL specifies the integration rule. The tables Gaussian Quadrature Points and Lobatto Quadrature Points below tabulate the location of the Gauss and Lobatto integration points. EQ.0.0: Set to 2 integration points for shell elements. EQ.1.0: 1 point (no bending) EQ.2.0: 2 points EQ.3.0: 3 points EQ.4.0: 4 points EQ.5.0: 5 points EQ.6.0: 6 points EQ.7.0: 7 points EQ.8.0: 8 points EQ.9.0: 9 points EQ.10.0: 10 points GT.10.0: Trapezoidal or user-defined rule

VARIABLE	DESCRIPTION
	Through-thickness integration for two-dimensional elements (options 12-15, 43, 44, 52, and 55 above) and cohesive shells (option ±29) is not meaningful; consequently, the default is 1 integration point. Fully integrated two-dimensional elements are available for options 13 and 15 (but not 12 and 14) by setting NIP equal to a value of 4, corresponding to a 2 by 2 Gaussian quadrature. For element type 55, NIP can be 4, 9, or 16. If NIP is 0 or 1 and the *MAT_SIMPLIFIED_JOHNSON_COOK model is used, then a resultant plasticity formulation is activated. NIP is always set to 1 if a constitutive model based on resultants is used.
PROPT	Printout option (***NOT ACTIVE***): EQ.1.0: Average resultants and fiber lengths EQ.2.0: Resultants at plan points and fiber lengths EQ.3.0: Resultants, stresses at all points, fiber lengths
QR/IRID	Quadrature rule or integration rule ID, see *INTEGRATION_SHELL: LT.0.0: Absolute value is specified rule number. EQ.0.0: Gauss/Lobatto (up to 10 points are permitted) EQ.1.0: Trapezoidal, not recommended for accuracy reasons
ICOMP	Flag for orthotropic/anisotropic layered composite material model. This option applies to material types 21, 22, 23, 33, 33_96, 34, 36, 40, 41-50, 54, 55, 58, 59, 103, 103_P, 104, 108, 116, 122, 133, 135, 135_PLC, 136, 157, 158, 190, 219, 226, 233, 234, 235, 242, and 243. For these material types, see *PART_COMPOSITE as an alternative to *SECTION_SHELL. Note: Please refer to *Remark 5* under *MAT_034 for additional information specific to fiber directions for fabrics. EQ.1: A material angle in degrees is defined for each through-thickness integration point. Thus, each layer has one integration point.
SETYP	Not used (obsolete)

Card 2	1	2	3	4	5	6	7	8
Variable	T1	T2	T3	T4	NLOC	MAREA	IDOF	EDGSET
Type	F	F	F	F	F	F	F	I
Default	0.0	T1	T1	T1	0.0	0.0	0.0	↓

VARIABLE	DESCRIPTION
T1	Shell thickness at node n_1, unless the thickness is defined on the *ELEMENT_SHELL_*OPTION* card.
T2	Shell thickness at node n_2, see comment for T1 above.
T3	Shell thickness at node n_3, see comment for T1 above.
T4	Shell thickness at node n_4, see comment for T1 above.
NLOC	Location of reference surface (shell mid-thickness) for three-dimensional shell elements. If nonzero, the offset distance from the plane of the nodal points to the reference surface of the shell in the direction of the shell normal vector is a value, $\text{offset} = -0.50 \times \text{NLOC} \times (\text{average shell thickness})$. Alternatively, the offset can be specified with the OFFSET keyword option of *ELEMENT_SHELL. Except for Mortar contacts, the contact subroutines do not consider this offset unless CNTCO = 1 or 3 in *CONTROL_SHELL. For Mortar contacts, NLOC or OFFSET determines the location of the contact surface regardless of the value of CNTCO. EQ.1.0: Nodes are located at top surface of shell. EQ.0.0: Nodes are located at mid-thickness of shell (default). EQ.-1.0: Nodes are located at bottom surface of shell. For nonzero offset distances, the time step size is reduced to prevent instabilities. See NLOCDT in *CONTROL_SHELL.
MAREA	Non-structural mass per unit area. This is additional mass that comes from materials, such as carpeting. This mass is not directly included in the time step calculation. An often more convenient alternative for defining distributed mass is with *ELEMENT_MASS_PART, which allows additional non-structural mass to be

VARIABLE	DESCRIPTION
	distributed by an area-weighted distribution to all nodes of a given part ID.
IDOF	Treatment of through-thickness strain (see Remark 7). LT.0: Same as IDOF = 3, but the contact pressure is averaged over a time –IDOF to reduce noise and thus improve stability. EQ.1: The thickness field is continuous across the element edges for metal forming applications. This option applies to element types 25, 26, and 27. EQ.2: The thickness field is discontinuous across the element edges. This is necessary for crashworthiness simulations due to shell intersections, sharp included angles, and non-smooth deformations. This option applies to and is the default for element types 25, 26, and 27. EQ.3: The thickness strain is governed by stress from contact and pressure loads, meaning that the strain is adjusted for the through-thickness stress to equilibrate the contact and load pressure. Please note that the pressure is assumed positive, meaning acting towards the shell surface. Thus, vacuum loads cannot be used with this option. This option applies to element types 2, 4, and ±16. EQ.11: Same as IDOF = 1, but the through-thickness strain is simplified in the kinematics in order to reduce locking effects, which were observed for considerable thickness changes. This option applies to element type 25 only. EQ.12: Same as IDOF = 2, but the through-thickness strain is simplified in the kinematics in order to reduce locking effects, which were observed for considerable thickness changes. This option applies to element type 25 only.
EDGSET	Edge node set required for shell-type seatbelts. See Remark 8. See Figure 19-17 in *ELEMENT_SEATBELT for additional clarification.

Angle Cards. Additional cards for ICOMP = 1. Include the minimum number of cards necessary to input NIP values: 8 values per card ⇒ number of cards = ceil(NIP/8) where ceil(x) = the smallest integer greater than x.

Card 3	1	2	3	4	5	6	7	8
Variable	B1	B2	B3	B4	B5	B6	B7	B8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
Bi	β_i, material angle at the i^{th} integration point

EFG Card. Additional card for EFG keyword option. See *CONTROL_EFG.

Card 4a	1	2	3	4	5	6	7	8
Variable	DX	DY	ISPLINE	IDILA	IEBT	IDIM		
Type	F	F	I	I	I	I		
Default	1.1	1.1	0	0	-1 or 1	2 or 1		

VARIABLE	DESCRIPTION
DX, DY	Normalized dilation parameters of the kernel function in X and Y directions. The normalized dilation parameters of the kernel function are introduced to provide the smoothness and compact support properties on the construction of the mesh-free shape functions. Values between 1.0 and 2.0 are recommended. Values smaller than 1.0 are not allowed. Larger values will increase the computation time and will sometimes result in a divergence problem.
ISPLINE	Replace the choice for the EFG kernel functions definition in *CONTROL_EFG which permits defining different ISPLINE in different sections.
IDILA	Replace the choice for the normalized dilation parameter definition in *CONTROL_EFG which permits defining different IDILA in different sections.

VARIABLE	DESCRIPTION
IEBT	Essential boundary condition treatment EQ.1: Full transformation (default for ELFORM = 42) EQ.-1: Without full transformation (default for ELFORM = 41) EQ.3: Coupled FEM/EFG EQ.7: Maximum entropy approximation
IDIM	For mesh-free shell local approach (ELFORM = 41) EQ.1: First-kind local boundary condition method EQ.2: Gauss integration (default) For mesh-free shell global approach (ELFORM = 42) EQ.1: First-kind local boundary condition method (default) EQ.2: Second-kind local boundary condition method

Thermal Card. Additional Card for THERMAL keyword option.

Card 4b	1	2	3	4	5	6	7	8
Variable	ITHELFM							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
ITHELFM	Thermal shell formulation EQ.0: Default is governed by THSHEL on *CONTROL_SHELL EQ.1: Thick thermal shell EQ.2: Thin thermal shell

XFEM Card. Additional card for XFEM keyword option. See Remark 9.

Card 4c	1	2	3	4	5	6	7	8
Variable	CMID	BASELM	DOMINT	FAILCR	PROPCR	FS	LS/FS1	NC/CL
Type	I	I	I	I	I	F	F	I/F
Default	none	none	0	1	0	0.0	0.0	none

VARIABLE	DESCRIPTION
CMID	Cohesive material ID. *MAT_185 is available for both brittle and ductile fracture, and *MAT_240 is available for ductile material. See Remark 18.
BASELM	Base element type for XFEM (type 13 for 2D, types 2, 16 or 17 for shell)
DOMINT	Option for domain integration in XFEM: EQ.0: Phantom element integration (default) EQ.1: Subdomain integration with triangular local boundary integration (available in 2D only)
FAILCR	Option for different failure criteria: EQ.0: Failure criterion from *MAT_ADD_EROSION / GISSMO model EQ.1: Maximum tensile stress (failure value given in cohesive law) EQ.2: Maximum shear stress (failure value given in cohesive law) EQ.-1: Effective plastic strain (EPS) EQ.-2: Crack length dependent EPS. See Remark 16. EQ.-4: Stress triaxiality based failure plastic strain
PROPCR	PROPCR is interpreted digit-wise: $\text{PROPCR} = [IP] = P + 10 \times I$ P determines the crack propagation direction: P.EQ.0: First principal strain direction if FAILCR < 0 (default for ductile fracture), first principal stress if FAILCR = 1, or

VARIABLE	DESCRIPTION
	maximum shear stress if FAILCR = 2. P = 0 is the only option for brittle fracture (FAILCR > 0). P.EQ.2: Center of effective plastic strain P.EQ.3: Directional center of effective plastic strain *I* determines crack initiation: I.EQ.0: Crack initiates at boundary (default) I.EQ.3: Crack initiates anywhere
FS	Failure value for FAILCR = -1 or -2: FAILCR.EQ.-1: Failure strain/failure critical value FAILCR.EQ.-2: Initial failure plastic strain Curve ID or Table ID for stress triaxiality based failure plastic strain for FAILCR = -4.
LS	Length scale for strain regularization (FAILCR = -1 only)
FS1	Final failure plastic strain (FAILCR = -2 only)
NC	Number of cracks allowed in the part (FAILCR ≠ -2)
CL	Crack length at which the failure strain is FS1 (FAILCR = -2 only)

Miscellaneous Settings Card. Additional card for MISC keyword option.

Card 4d	1	2	3	4	5	6	7	8
Variable	THKSCL							
Type	F							
Default	1.0							

VARIABLE	DESCRIPTION
THKSCL	Thickness scale factor. Shell thicknesses for all elements of this section including those with thickness specified using *ELEMENT_SHELL_THICKNESS are scaled by THKSCL.

User-Defined Element Card. Additional card for ELFORM = 101,102,103,104 or 105. See Appendix C

Card 5	1	2	3	4	5	6	7	8
Variable	NIPP	NXDOF	IUNF	IHGF	ITAJ	LMC	NHSV	ILOC
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

User Defined Element Integration Point Cards. Additional cards for ELFORM = 101, 102, 103, 104 or 105. Define NIPP cards according to the following format. See Appendix C.

Card 5.1	1	2	3	4	5	6	7	8
Variable	XI	ETA	WGT					
Type	F	F	F					
Default	none	none	none					

User Defined Element Property Cards. Include the minimum number of cards necessary to input LMC values: 8 values per card $\Rightarrow$ number of cards = ceil(LMC/8) where ceil(x) = the smallest integer greater than x. See Appendix C.

Card 5.2	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
NIPP	Number of in-plane integration points for user-defined shell (0 if resultant/discrete element)

VARIABLE	DESCRIPTION
NXDOF	Number of extra degrees of freedom per node for user-defined shell
IUNF	Flag for using nodal fiber vectors in user-defined shell: EQ.0: Nodal fiber vectors are not used. EQ.1: Nodal fiber vectors are used.
IHGF	Flag for using hourglass stabilization (NIPP is greater than 0) EQ.0: Hourglass stabilization is not used. EQ.1: LS-DYNA hourglass stabilization is used. EQ.2: User-defined hourglass stabilization is used. EQ.3: Same as 2, but the resultant material tangent moduli are passed
ITAJ	Flag for setting up finite element matrices (NIPP is greater than 0) EQ.0: Set up matrices with respect to the isoparametric domain EQ.1: Set up matrices with respect to the physical domain
LMC	Number of property parameters
NHSV	Number of history variables
ILOC	Coordinate system option: EQ.0: Pass all variables in the LS-DYNA local coordinate system EQ.1: Pass all variables in the global coordinate system
XI	First isoparametric coordinate
ETA	Second isoparametric coordinate
WGT	Isoparametric weight
P*i*	i^{th} user-defined element property

Gaussian Quadrature Points

Point	1 Point	2 Points	3 Points	4 Points	5 Points
#1	.0	-.5773503	.0	-.8611363	.0
#2		+.5773503	-.7745967	-.3399810	-.9061798
#3			+.7745967	+.3399810	-.5384693
#4				+.8611363	+.5384693
#5					+.9061798
Point	6 Points	7 Points	8 Points	9 Points	10 Points
#1	-.9324695	-.9491080	-.9602899	-.9681602	-.9739066
#2	-.6612094	-.7415312	-.7966665	-.8360311	-.8650634
#3	-.2386192	-.4058452	-.5255324	-.6133714	-.6794096
#4	+.2386192	.0	-.1834346	-.3242534	-.4333954
#5	+.6612094	+.4058452	+.1834346	.0	-.1488743
#6	+.9324695	+.7415312	+.5255324	+.3242534	+.1488743
#7		+.9491080	+.7966665	+.6133714	+.4333954
#8			+.9602899	+.8360311	+.6794096
#9				+.9681602	+.8650634
#10					+.9739066

Location of through-thickness Gauss integration points. The coordinate is referenced to the shell midsurface at location 0. The inner surface of the shell is at -1, and the outer surface is at +1.

Lobatto Quadrature Points

Point	1 Point	2 Points	3 Points	4 Points	5 Points
#1			0.0	-1.0	0.0
#2			-1.0	-0.4472136	-1.0
#3			+1.0	+0.4472136	-0.6546537
#4				+1.0	+0.6546537
#5					+1.0
Point	6 Points	7 Points	8 Points	9 Points	10 Points
#1	-1.0	-1.0	-1.0	-1.0	-1.0
#2	-0.7650553	-0.8302239	-0.8717401	-0.8997580	-0.9195339
#3	-0.2852315	-0.4688488	-0.5917002	-0.6771863	-0.7387739
#4	+0.2852315	0.0	-0.2092992	-0.3631175	-0.4779249
#5	+0.7650553	+0.4688488	+0.2092992	0.0	-0.1652790
#6	+1.0	+0.8302239	+0.5917002	+0.3631175	+0.1652790
#7		+1.0	+0.8717401	+0.6771863	+0.4779249
#8			+1.0	+0.8997580	+0.7387739
#9				+1.0	+0.9195339
#10					+1.0

Location of through-thickness Lobatto integration points. The coordinate is referenced to the shell midsurface at location 0. The inner surface of the shell is at -1 and the outer surface is at +1.

Remarks:

1. **Formulations.** The default shell formulation is 2 unless overridden by THEORY in *CONTROL_SHELL. ELFORM in *SECTION_SHELL overrides THEORY.

   For implicit calculations the following element formulations are implemented: 2, 5, 6, 10, 12,12, 13, 14, 15, 16, -16, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 29, 41, 42, 55.

   If another element formulation is requested for an implicit analysis, LS-DYNA will substitute one of the above in place of the one chosen.

2. **Problem types for the type 18 element.** The type 18 element is only for linear static and normal modes. It can also be used for linear springback in sheet metal stamping.

3. **Description of linear element types 18 and 20.** The linear elements consist of an assembly of membrane and plate elements. They have six degrees of freedom per node and can, therefore, be connected to beams or used in complex shell surface intersections. These elements possess the required zero energy rigid body modes and have exact constant strain and curvature representation, that is, they pass all the first order patch tests. In addition, the elements have behavior approaching linear bending (cubic displacement) in the plate-bending configuration.

   a) The membrane component is based on an 8-node/6-node isoparametric mother element which incorporates nodal in-plane rotations through cubic displacement constraints of the sides [Taylor 1987; Wilson 2000].

   b) The plate component of element 18 is based on the Discrete Kirchhoff Quadrilateral (DKQ) [Batoz 1982]. Because the Kirchhoff assumption is enforced, the DKQ is transverse-shear rigid and can only be used for thin shells. No transverse shear stress information is available. The triangle is based on a degeneration of the DKQ. This element sometimes gives slightly lower eigenvalues when compared with element type 20.

   c) The plate component of element 20 is based on the 8-node serendipity element. At the mid-side, the parallel rotations and transverse displacements are constrained, and the normal rotations are condensed to yield a 4-node element. The element is based on thick plate theory and is recommended for thick and thin plates.

   d) The quadrilateral elements contain a warpage correction using rigid links.

   e) The membrane component of element 18 has a zero-energy mode associated with in-plane rotations. This is automatically suppressed in a non-flat shell by the plate stiffness of the adjacent elements. In contrast, element 20 has no spurious zero energy modes.

4. **Linear shear element (22).** The linear shear panel element resists tangential in plane shearing along the four edges and can only be used with the elastic material constants of *MAT_ELASTIC. Membrane forces and out-of-plane loads are not resisted.

5. **Simplified element for time domain vibrations (99).** Element type 99 is intended for vibration studies carried out in the time domain. These models may have very large numbers of elements and may be run for relatively long durations. The purpose of this element is to achieve substantial CPU savings. This

is achieved by imposing strict limitations on the range of applicability, thereby simplifying the calculations:

a) Elements must be rectangular; all edges must parallel to the global *x*-, *y*- or *z*-axis;

b) Small displacement, small strain, negligible rigid body rotation;

c) Elastic material only.

If these conditions are satisfied, the performance of the element is similar to the fully integrated shell (ELFORM = 16) but at less CPU cost than the default Belytschko-Tsay shell element (ELFORM = 2). Single element torsion and in-plane bending modes are included; meshing guidelines are the same as for fully integrated shell elements.

No damping is included in the element formulation (such as volumetric damping). It is strongly recommended that damping be applied with keywords such as *DAMPING_PART_MASS or *DAMPING_FREQUENCY_RANGE.

6. **2D formulations.** For 2D formulations (12-15, 43, 44, 46, 47, 52, and 55), nodes must lie in the global *xy*-plane, so the *z*-coordinate must be zero. Furthermore, the element normal should be in positive *z* direction. For axisymmetric element formulations, the global *y*-axis is taken as the axis of symmetry, and all nodes must have *x*-coordinate values greater than or equal to 0.

   Shell thickness values on Card 2 are ignored by formulations 13, 14, 15, 43, 44, 52, and 55. For formulation 14 input values of loads, lumped masses, discrete element stiffnesses, etc. in axisymmetric models are interpreted as values per unit circumference (that is per unit length in the circumferential direction) whereas for formulation 15 they are interpreted per radian. Output of forces for shell formulation 15 are in units of force per radian in the output files, such as bndout, nodfor, secforc, spcforc, and rcforc. The units of forces output for shell formulation 14 are, at present, inconsistent. For defining contact in 2D simulations, see the entry for the *CONTACT_2D keyword.

7. **Shells with thickness stretch.** Shell element formulation 25 and 26 are the fully integrated shell element based on the Belytschko-Tsay element but with two additional degrees of freedom allowing for a linear variation of strain through the thickness. By specifying IDOF = 1, the thickness field is continuous across the element edges implying that there can be no complex intersections since this would lock up the structure. It assumes a relatively flat surface and is intended primarily for sheets in metal forming. By default, the thickness field is decoupled between elements which makes the element suited for crash.

8. **Seat belts.** You must input a set of nodes along one of the transverse edges of a seatbelt. If there is no retractor associated with a belt, the node set can be on either edge. If the retractor exists, the edge should be on the retractor side and input in the same sequence as the retractor's node set. Therefore, each seat belt must have its own section definition and its own part.

9. **Fracture.** XFEM 2D and shell formulations are recommended for brittle or semi-brittle fracture with pre-crack (see *BOUNDARY_PRECRACK) or with geometry imperfection such as a notch or a hole, and for ductile fracture analysis with regularized effective plastic strain criterion or nonlocal continuum damage material models (provided with GISSMO model).

10. **Discrete Kirchoff Theory shell (17).** Shell element formulation 17 (DKT) is based on Discrete Kirchhoff Theory. It neglects out-of-plane shear strain energy and is thus valid only for thin plates where shear strain energy is negligible compared to bending energy.

11. **Limitations of area-weighted shells (14).** The exact stiffness matrix for the area-weighted shell formulation type 14 is nonsymmetric. The nonsymmetric terms are dropped for computational efficiency, making this formulation unsuitable for implicit linear analysis and eigenvalue analysis. For implicit modal computations element type 14 must be switched to type 15. It may, however, be used effectively for implicit nonlinear analysis.

    For explicit dynamics, viscous hourglass limits high frequency noise that may otherwise lead to nonphysical element distortion when nodes are on or near the axis of symmetry. Viscosity can be added to type 6 or 7 hourglass control by using VDC > 0 under the *HOURGLASS keyword. Alternatively, type 1 hourglass control is viscous.

12. **Eight node singular shell (55).** The eight node singular element for fracture analysis is based on the eight node quadratic quadrilateral element. There are two ways to include the singularity around the crack tip:

    a) Move the two mid-nodes on the edges connected to the crack tip to the quarter location and obtain a strain singularity of $1/\sqrt{r}$ along the edges with quarter-point mid-nodes and of weaker than one-half order elsewhere.

    b) Collapse the three nodes on one side of a quadrilateral element to the crack tip (but the three nodes remain independent) and move the two neighboring mid-nodes to the quarter location to obtain a strain singularity of exact one-half order $1/\sqrt{r}$ everywhere around the crack tip..

    This element uses 2 × 2, 3 × 3, or 4 × 4 quadrature (default 2 × 2) and is available in implicit analysis only. The reference coordinates of the quadrature points are

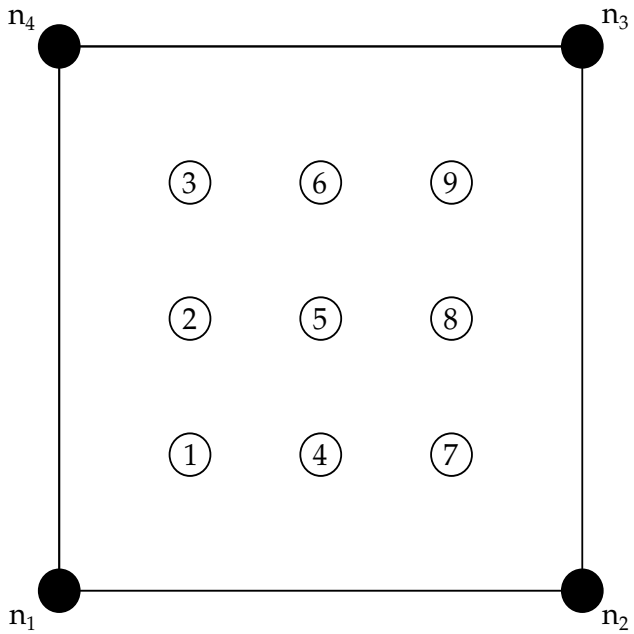


Figure 41-24. Integration point locations for a type 55 element with a 3 × 3 quadrature scheme

as in the Gaussian Quadrature Points table but have a different order (#2, #1 and #3) in one direction for the 3 × 3 quadrature scheme. The global coordinates of the quadrature points depend on the connectivity of the element, as shown in Figure 41-24.

To easily access the stress data, the stress components at all the integration points are output to the ELOUT file.

13. **Cohesive shell (+/-29).** Element type +/-29 is a cohesive element that models cohesive interfaces between shell element edges. The element takes bending forces into account and uses drilling force stabilization.

Consider two shell elements in the same plane with nodes m_1, m_2, m_3, m_4, n_1, n_2, n_3, and n_4 such that the (m_3, m_4) edge and the (n_1, n_2) edge are connected through a cohesive shell having nodes m_4, m_3, n_2, and n_1; see Figure 41-25. The initial area of the cohesive element may be zero, in which case density is defined in terms of the length of the single connecting edge.

Element type +/-29 works similarly to solid element type 20. For example, extruding the two non-cohesive shells in their respective normal directions defines two 8 node solids. The cohesive midsurface is located between the opposing faces of the solids, and tractions are calculated in four mid

surface points using differences of displacements between the opposing faces, giving rise to nodal forces and moments in the cohesive shell nodes m_4, m_3, n_2, and n_1. Note that, regardless of the value of INTFAIL, a 2 x 2 Newton-Cotes

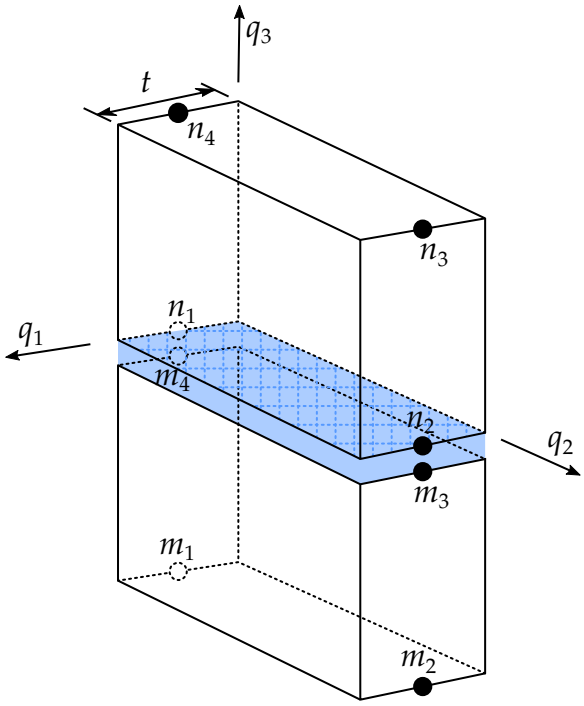


Figure 41-25. Cohesive interface coordinate system for element type +/-29

quadrature is used in the force integrals. Additional details can be found in the Theory Manual.

Difference between type 29 and -29: In both formulations, the cohesive coordinate system direction q_2 is defined by the midpoints of the (n_1, m_4) and (n_2, m_3) edges. Type 29 defines the cohesive midsurface normal q_3 using the midpoints on the far side edges (m_1, m_2) and (n_3, n_4), while type -29 defines q_1 to be the mean of the neighboring element normals. Thus, in pure out-of-plane shear, type 29 will initially have pure tangential traction that turns into a normal traction as the separation increases, while type -29 will only have tangential traction.

Cohesive shell formulations +/-29 output three tractions having units of force per unit area into the d3plot database rather than the usual six stress components. The out-of-plane shear traction replaces the x-stress (mode III), the in-plane shear traction replaces the y-stress (mode II), and the traction in the normal direction replaces the z-stress (mode I). The tractions on the four midsurface points on the cohesive interface can be saved in d3plot by setting MAXINT to -1 on *DATABASE_EXTENT_BINARY or *DATABASE_D3PART.

14. **Switching between type 16 and type 30 shells.** We implemented two methods to globally switch between types 16 and 30 without changing *SECTION_SHELL. For the first method, use the command line option "shell=16to30" or "shell=30to16" to change from type 16 to 30 or from type 30 to 16. See Execution Syntax in the Getting Started section for details about command line options.

For the second method, set flag ISWSHL on *CONTROL_SHELL to 1 to convert from 16 to 30 or to 2 to convert from 30 to 16.

15. **Accurate fully-integrated shell (-16).** Accuracy issues have been observed in shell formulation 16 under large deformations/rotations over a single time step. Formulation -16 is a version of formulation 16 which has been enhanced to address those accuracy issues. Formulation 16 is unchanged to maintain back compatibility and, although element -16 is supported in explicit mode, it is primarily intended for implicit time integration. A strongly objective version can be activated by combining ELFORM = -16 and IACC = 1 on *CONTROL_ACCURACY, thereby ensuring that arbitrarily large rigid body rotation in a single step will transform stresses correctly and not generate any spurious strains.

16. **XFEM ductile fracture.** This feature is supported by joint research among Honda, JSOL and LST. For FAILCR = -2, the failure strain is given by:

$$\mathrm{EPS} = \mathrm{FS} + (\mathrm{FS1} - \mathrm{FS}) \times \min(L/\mathrm{CL}, 1.0)$$

17. **Cohesive formulations 46 and 47.** We intend cohesive formulations 46 and 47 for cohesive elements with nonzero initial thickness. Instabilities may occur if the element has zero initial thickness.

    Like for cohesive solids, LS-DYNA outputs three tractions having units of force per unit area for plane strain and axisymmetric cohesive formulations 46 and 47 into the d3plot database rather than the usual six stress components. The in-plane shear traction along the 1-2 edge replaces the x-stress, the orthogonal in plane shear traction replaces the y-stress, and the traction in the normal direction replaces the z-stress.

18. **Cohesive material law for XFEM.** For the XFEM formulation, modeling the fracture process zone around the crack tip is simplified with a cohesive zone model. The initially rigid cohesive material law provides the kinetic relationship (traction force as a function of crack opening displacement) for the cohesive crack surface inserted in the element. The work done by the traction represents the energy release rate in brittle fracture or plastic work in ductile fracture to create a real crack surface where the traction force becomes zero. When a nonlocal continuum damage constitutive model is used for ductile fracture, the cohesive law is not needed since the stresses in the failed element become zero due to the damage induced stress softening.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *SECTION_SHELL
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  Define a shell section that specifies the following:
$     elform = 10  Belytschko-Wong-Chiang shell element formulation.
$        nip = 3   Three through the shell thickness integration points.
$    t1 - t4 = 2.0 A shell thickness of 2 mm at all nodes.
$
*SECTION_SHELL
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$      sid    elform      shrf       nip     propt   qr/irid     icomp
         1        10               3.0000
$
$       t1        t2        t3        t4      nloc
       2.0       2.0       2.0       2.0
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*SECTION_SOLID_{OPTION}

Available options include:

<BLANK>

EFG

SPG

MISC

Purpose: Define section properties for solid continuum and fluid elements.

Card Summary:

Card Sets. For each unique solid section, include one set of data cards. The EFG and SPG options cannot appear in the same model. This input ends at the following keyword ("*") card.

Card 1. This card is required.

SECID	ELFORM	AET				COHOFF	GASKETT

Card 2a.1. This card is included if the EFG keyword option is used.

DX	DY	DZ	ISPLINE	IDILA	IEBT	IDIM	TOLDEF

Card 2a.2. This card is read only if the EFG keyword option is used. It is optional.

IPS	STIME	IKEN	SF	CMID	IBR	DS	ECUT

Card 2b.1. This card is included if the SPG keyword option is used.

DX	DY	DZ	ISPLINE	KERNEL		SMSTEP	MSC

Card 2b.2. This card is read only if the SPG keyword option is used. It is optional.

IDAM	FS	STRETCH	ITB	MSFAC	ISC	BOXID	PDAMP

Card 2c. This card is read only if the MISC keyword option is used. It is optional.

COHTHK							

Card 3. This card is included if ELFORM = 101, 102, 103, 104, or 105.

NIP	NXDOF	IHGF	ITAJ	LMC	NHSV	XNOD	

Card 4. This card is included if ELFORM = 101, 102, 103, 104, or 105. Include NIP of this card.

XI	ETA	ZETA	WGT				

Card 5. This card is included if ELFORM = 101, 102, 103, 104, or 105. Include ceil(LMC/8) of this card.

P1	P2	P3	P4	P5	P6	P7	P8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	AET				COHOFF	GASKETT
Type	I/A	I	I				F	F

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation options. Remark 2 enumerates the element formulations available for implicit calculations:
	EQ.0: 1 point corotational for *MAT_MODIFIED_HONEY-COMB (see Remark 3)
	EQ.1: Constant stress solid element: default element type. By specifying hourglass type 10 with this element, a Cosserat Point Element is invoked; see *CONTROL_-HOURGLASS.
	EQ.-1: 8 point hexahedron intended for elements with poor aspect ratios, efficient formulation (see Remark 13)
	EQ.-2: 8 point hexahedron intended for elements with poor aspect ratios, accurate formulation (see Remark 13)
	EQ.2: 8 point hexahedron (see Remark 5)
	EQ.3: Fully integrated quadratic 8 node element with nodal rotations
	EQ.4: S/R quadratic tetrahedron element with nodal

VARIABLE	DESCRIPTION
	rotations
EQ.5:	1 point ALE
EQ.6:	1 point Eulerian
EQ.7:	1 point Eulerian ambient
EQ.8:	Acoustic
EQ.9:	1 point corotational for *MAT_MODIFIED_HONEY-COMB (see Remark 3)
EQ.10:	1 point tetrahedron (see Remark 1)
EQ.11:	1 point ALE multi-material element (see Remark 4)
EQ.12:	1 point integration with single material and void
EQ.13:	1 point nodal pressure tetrahedron (see Remark 12)
EQ.14:	8 point acoustic
EQ.15:	2 point pentahedron element (see Remark 1)
EQ.16:	4 or 5 point 10-node tetrahedron (see Remark 11). By specifying hourglass type 10 with this element, a Cosserat Point Element is invoked; see *CONTROL_-HOURGLASS.
EQ.17:	10-node composite tetrahedron (see Remark 11)
EQ.-18:	8 point enhanced strain solid element with 13 incompatible modes (see Remarks 5 and 17)
EQ.18:	9 point enhanced strain solid element with 12 incompatible modes (implicit only; see Remarks 5 and 17)
EQ.-19:	8-node, 4 point cohesive element with or without offset for use with both solids and shells (see Remarks 1 and 7)
EQ.19:	8-node, 4 point cohesive element without offset for use with solids (see Remarks 1 and 7)
EQ.20:	8-node, 4 point cohesive element with offsets for use with shells (see Remarks 1, 7, and 9)
EQ.-21:	6-node, 3 point pentahedron cohesive element with or without offset for use with solids and shells (see Remarks 1 and 8)
EQ.21:	6-node, 3 point pentahedron cohesive element without offset for use with solids (see Remarks 1 and 8)

VARIABLE	DESCRIPTION
	EQ.22: 6-node, 3 point pentahedron cohesive element with offsets for use with shells (see Remarks 1, 8, and 9)
	EQ.23: 20-node solid formulation
	EQ.24: 27-node, fully integrated S/R quadratic solid element (see Remark 16)
	EQ.25: 21-node, quadratic pentahedron (see Remark 16)
	EQ.26: 15-node, quadratic tetrahedron (see Remark 16)
	EQ.27: 20-node, cubic tetrahedron (see Remark 16)
	EQ.28: 40-node, cubic pentahedron (see Remark 16)
	EQ.29: 64-node, cubic hexahedron (see Remark 16)
	EQ.41: Mesh-free (EFG) solid formulation (see Remark 19)
	EQ.42: Adaptive 4-node mesh-free (EFG) solid formulation (see Remark 19)
	EQ.43: Mesh-free enriched finite element
	EQ.45: Tied mesh-free enriched finite element
	EQ.47: Smoothed Particle Galerkin (SPG) method (see Remark 14)
	EQ.60: 1 point tetrahedron (see Remark 15)
	EQ.62: 8 point brick with incompatible modes by assumed strain (see Remarks 5 and 18)
	EQ.98: Interpolation solid
	EQ.99: Simplified linear element for time-domain vibration studies (See Remark 6)
	EQ.101: User-defined solid
	EQ.102: User-defined solid
	EQ.103: User-defined solid
	EQ.104: User-defined solid
	EQ.105: User-defined solid
	EQ.115: 1 point pentahedron element with hourglass control
	GE.201: Isogeometric solids with NURBS (see *ELEMENT_SOLID_NURBS_PATCH)
	GE.1000: Generalized user-defined solid element formulation (see *DEFINE_ELEMENT_GENERALIZED_SOLID)

VARIABLE	DESCRIPTION
AET	Ambient element type (can be defined for ELFORM 7, 11, and 12): EQ.0: Non-ambient EQ.1: Temperature (not currently available) EQ.2: Pressure and temperature (not currently available) EQ.3: Pressure outflow (obsolete) EQ.4: Pressure inflow/outflow (default for ELFORM 7) EQ.5: Receptor for blast load (See *LOAD_BLAST_ENHANCED. Available only for ELFORM = 11.)
COHOFF	Applies to cohesive solid elements -19, 20, -21 and 22. COHOFF specifies the relative location of the cohesive layer. It must be a number between -1 and 1. A value of -1 will place it on the bottom face of the cohesive element, while a value of +1 will place it on the top face. This parameter is preferably used when the cohesive element connects shells with different thicknesses. In this case, the cohesive layer should not be located exactly between the bottom and top layer, which is the default location (COHOFF = 0).
GASKETT	Gasket thickness for converting ELFORM 19, 20, 21, and 22 to gasket elements and use with *MAT_COHESIVE_GASKET.

EFG Card. Additional card for the EFG keyword option. See *CONTROL_EFG.

Card 2a.1	1	2	3	4	5	6	7	8
Variable	DX	DY	DZ	ISPLINE	IDILA	IEBT	IDIM	TOLDEF
Type	F	F	F	I	I	I	I	F
Default	1.01	1.01	1.01	0	0	3	2	0.01

VARIABLE	DESCRIPTION
DX, DY, DZ	Normalized dilation parameters of the kernel function in x, y, and z-directions. The normalized dilation parameters of the kernel function are introduced to provide the smoothness and compact support properties on the construction of the mesh-free shape functions. Values between 1.0 and 1.5 are recommended. Values

VARIABLE	DESCRIPTION
	smaller than 1.0 are not allowed. Larger values will increase the computation time and sometimes result in a divergence problem.
ISPLINE	Replace the choice for the EFG kernel functions definition in *CONTROL_EFG, which allows for different spline functions in different sections. EQ.0: Cubic spline function (default) EQ.1: Quadratic spline function EQ.2: Cubic spline function with a circular shape
IDILA	Replace the choice for the normalized dilation parameter definition in *CONTROL_EFG. This allows users to define different IDILA in different sections. EQ.0: Maximum distance based on the background elements. EQ.1: Maximum distance based on surrounding nodes.
IEBT	Essential boundary condition treatment (see Remarks 20 and 21): EQ.1: Full transformation method EQ.-1: (W/o transformation) EQ.2: Mixed transformation method EQ.3: Coupled FEM/EFG method (default) EQ.4: Fast transformation method EQ.-4: (W/o transformation) EQ.5: Fluid particle method for EOS and *MAT_ELASTIC_FLUID materials, currently supports only 4-node background elements. EQ.7: Maximum entropy approximation
IDIM	Domain integration method (see Remark 22): EQ.1: Local boundary integration EQ.2: Two-point Gauss integration (default) EQ.3: Improved Gauss integration for IEBT = 4 or -4 EQ.-1: Stabilized EFG integration method (apply to 6-node cell, 8-node cell, or combination of these two) EQ.-2: EFG fracture method (apply to 4-node cell and SMP only). The EFG fracture method is obsolete – see Remark

VARIABLE	DESCRIPTION
	24.
TOLDEF	Deformation tolerance for the activation of adaptive EFG Semi-Lagrangian and Eulerian kernel. See Remark 23. EQ.0.0: Lagrangian kernel GT.0.0: Semi-Lagrangian kernel LT.0.0: Eulerian kernel

Optional EFG Card. Additional optional card for the EFG keyword option. See *CONTROL_EFG.

Card 2a.2	1	2	3	4	5	6	7	8
Variable	IPS	STIME	IKEN	SF	CMID	IBR	DS	ECUT
Type	I	F	I	I	I	I	F	F
Default	0	10^{20}	0	0.0	0	1	1.01	0.1

VARIABLE	DESCRIPTION
IPS	Pressure smoothing flag: EQ.0: No pressure smoothing (default) EQ.1: Moving-least squared pressure recovery (material response is slightly more compressible when turning on this function). Only the adaptive 4-node mesh-free (EFG) solid formulation (ELFORM = 42) is supported.
STIME	Time to switch from stabilized EFG to standard EFG formulation. Obsolete - see Remark 24.
IKEN	EQ.0: Moving-least-square approximation (default, recommended) EQ.1: Maximum entropy approximation Obsolete - see Remark 24.
SF	Failure strain. It is recommended as an additional condition for crack initiation under slow loading besides the stress-based

VARIABLE	DESCRIPTION
	cohesive law. Obsolete - see Remark 24.
CMID	Cohesive material ID for EFG fracture analysis (only Mode I crack is considered, and only *MAT_COHESIVE_TH is available). Obsolete - see Remark 24.
IBR	EQ.1: No branching allowed. EQ.2: Branching is allowed. Obsolete - see Remark 24.
DS	Normalized support defined for computing the displacement jump in fracture analysis. Obsolete - see Remark 24.
ECUT	Define the minimum distance to the node that a crack surface can cut to the edge. Obsolete - see Remark 24.

SPG Card. Additional card for the SPG keyword option.

Card 2b.1	1	2	3	4	5	6	7	8
Variable	DX	DY	DZ	ISPLINE	KERNEL		SMSTEP	MSC
Type	F	F	F	I	I		I	F
Default	↓	↓	↓	0	0		↓	0

VARIABLE	DESCRIPTION
DX, DY, DZ	Normalized dilation parameters of the kernel function in x, y, and z directions, respectively. The normalized dilation parameters of the kernel function provide the smoothness and locality on the construction of the mesh-free shape functions. Values between 1.4 and 1.8 are recommended. Values smaller than 1.0 are not allowed. Larger values will increase the computation time and sometimes result in divergence of the solution. The default value depends on the choice of KERNEL:

KERNEL	0	1	2
Default	1.6	1.8	1.5

VARIABLE	DESCRIPTION
ISPLINE	Type of kernel function: EQ.0: Cubic spline function with cubical support (default) EQ.1: Quadratic spline function with cubical support EQ.2: Cubic spline function with spherical support
KERNEL	Type of kernel support update scheme: EQ.0: Updated Lagrangian, failure or no failure analysis, tension dominant problem EQ.1: Eulerian, failure analysis, global extreme deformation EQ.2: Pseudo Lagrangian, failure analysis, local extreme deformation
SMSTEP	Interval of time steps to conduct displacement smoothing. The default value depends on the choice of KERNEL:
MSC	Smoothing scheme for momentum consistent SPG only (ITB = 3 on Card 2b.2): EQ.0: Regular smoothing scheme EQ.1: New smoothing scheme for very low-speed deformation, with better controls of the low energy modes than the regular smoothing scheme

KERNEL	0	1	2
Default	15	5	30

Optional SPG Card. Additional optional card for the SPG keyword option.

Card 2b.2	1	2	3	4	5	6	7	8
Variable	IDAM	FS	STRETCH	ITB	MSFAC	ISC	BOXID	PDAMP
Type	I	F	F	I	F	I	I	F
Default	1	↓	10^{10}	↓	↓	0	0	-0.001

VARIABLE	DESCRIPTION
IDAM	Option of bond failure mechanism EQ.0: GISSMO damage value from *MAT_ADD_GISSMO or *MAT_ADD_EROSION. STRETCH is not used in this case. EQ.1: Effective plastic strain (phenomenological strain damage, default) EQ.2: Maximum principal stress EQ.3: Maximum shear strain EQ.4: Minimum principal strain (input must be positive) EQ.5: Effective plastic strain and maximum shear strain EQ.7: Anisotropic damage for honeycomb modeled with *MAT_126 only. We recommend only using this with ITB = 3. EQ.8: Damage value from the material model. STRETCH is not used in this case. EQ.11: Pre-damage model for brittle material failure (with crack propagation). It includes both bond failure and stress degradation. We recommend using this with ITB = 3. EQ.13: Pre-damage model for ductile material failure. It includes stress degradation but not bond failure. We recommend using this with ITB = 3.
FS	Critical value of the quantity indicated by IDAM for triggering bond failure. The default value is 10^{10}, meaning no failure analysis in general. However, starting with R14.0, if an SPG part has a material law with *MAT_ADD_EROSION, bond failure will be checked for using the failure criterion defined in *MAT_ADD_EROSION, even when FS = 0.0. FS on the material cards overwrites this value for *MAT_003 and *MAT_024. When FS is defined on the data cards for these materials, bond failure will occur when it is reached, and stress will be set to zero according to material law. If FS is defined here only, bond failure will occur without the stress being set to zero.
STRETCH	Critical relative deformation (stretching or compression ratio) between the two nodes forming the bond for bond failure

VARIABLE	DESCRIPTION
ITB	Option of stabilization: EQ.1: Fluid particle approximation (accurate but slow), used with KERNEL = 0 or 1 EQ.2: Simplified fluid particle approximation (efficient and robust), used with KERNEL = 2 EQ.3: Momentum consistent SPG (MC-SPG) formulation. MC-SPG can be applied to large deformation, tension-dominant problems. For coupled thermal-mechanical problems, MC-SPG is the only option. KERNEL = 1 is recommended for MC-SPG. The default for ITB depends on the value of KERNEL:

KERNEL	0	1	2
Default	1	1	2

MSFAC	For momentum consistent SPG invoked with ITB = 3 only, quadrature factor for surface nodes to suppress shear locking in thin structures. We recommend using the latest beta version or R14. The default for a regular solid structure is 1.00 while for a thin structure is 0.75.
ISC	Self-contact indicator: EQ.0: No self-contact between the bond-failed particles in the same part. EQ.1: Self-contact is defined between the bond-failed particles in the same part. The penalty factor in the self-contact is between 0.01 to 0.1 × Young's modulus. This option is available for SMP only. Note that this method is deprecated. Use *CONTACT_SPG (available for SMP and MPP) instead.
BOXID	ID of a box defining the active SPG region. Outside this region, the particles are not included in the SPG calculation. See *DEFINE_BOX.
PDAMP	Particle-to-particle damping coefficient. It is used for momentum-consistent SPG (ITB = 3) *only*. The recommended range of values is -0.01 to -0.001. A positive value is not recommended.

Optional MISC Card. Additional optional card for the MISC keyword option.

Card 2c	1	2	3	4	5	6	7	8
Variable	COHTHK							
Type	F							
Default	optional							

VARIABLE	DESCRIPTION
COHTHK	Cohesive thickness. This value supersedes the THICK value from *MAT_240 or *MAT_ADD_COHESIVE, allowing a section- or part-wise definition of cohesive thickness.

User-Defined Element Card. Additional card for ELFORM = 101, 102, 103, 104, or 105. See Appendix C.

Card 3	1	2	3	4	5	6	7	8
Variable	NIP	NXDOF	IHGF	ITAJ	LMC	NHSV	XNOD	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

VARIABLE	DESCRIPTION
NIP	Number of integration points for user-defined solid (0 if resultant element)
NXDOF	Number of extra degrees of freedom per node for a user-defined solid
IHGF	Flag for using hourglass stabilization (NIP > 0) EQ.0: Do not use hourglass stabilization. EQ.1: Use LS-DYNA hourglass stabilization. EQ.2: Use user-defined hourglass stabilization. EQ.3: Same as 2, but pass the resultant material tangent moduli.

VARIABLE	DESCRIPTION
ITAJ	Flag for setting up finite element matrices (NIP > 0) EQ.0: Set up matrices with respect to the isoparametric domain EQ.1: Set up matrices with respect to the physical domain
LMC	Number of property parameters
NHSV	Number of history variables
XNOD	Controls how the mass of extra degrees of freedom is interpreted when defined using a user subroutine. EQ.0: The mass is to be defined per degree of freedom. EQ.1: The mass is to be defined as a nodal mass, where each scalar node has three extra degrees of freedom. See Extra Degrees of Freedom in Appendix C.

Integration Point Card. Additional card for ELFORM = 101, 102, 103, 104, or 105. Add NIP cards adhering to the format below. Because the default value for NIP is 0, these cards are read only for user-defined elements. See Appendix C.

Card 4	1	2	3	4	5	6	7	8
Variable	XI	ETA	ZETA	WGT				
Type	F	F	F	F				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
XI	First isoparametric coordinate
ETA	Second isoparametric coordinate
ZETA	Third isoparametric coordinate
WGT	Isoparametric weight

Property Parameter Cards. Additional card for ELFORM = 101, 102, 103, 104, or 105. Add LMC property parameters by packing eight parameters per card. See Appendix C.

Card 5	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F
Default	0.	0.	0.	0.	0.	0.	0.	0.

VARIABLE	DESCRIPTION
P*i*	i^{th} property parameter

Remarks:

1. **ESORT to stabilize degenerate solids.** The ESORT variable of the *CONTROL_SOLID keyword can be set to automatically convert degenerate tetrahedrons and degenerate pentahedrons into more suitable solid element formulations. The sorting is performed internally and is transparent to the user. See *CONTROL_SOLID for details.

2. **Implicit analysis.** For implicit calculations, the following element choices are implemented:

   EQ.1: Constant stress solid element

   EQ.-1: 8 point hexahedron for poor aspect ratios, efficient formulation

   EQ.-2: 8 point hexahedron for poor aspect ratios, accurate formulation

   EQ.2: 8 point hexahedron

   EQ.3: Fully integrated 8 node solid with rotational DOFs

   EQ.4: Fully integrated S/R 4 node tetrahedron with rotational DOFs

   EQ.10: 1 point tetrahedron

   EQ.13: 1 point nodal pressure tetrahedron

   EQ.15: 2 point pentahedron element

   EQ.16: 5 point 10-node tetrahedron

   EQ.17: 10-node composite tetrahedron

   EQ.-18: 8 point enhanced strain solid element (13 incompatible modes)

EQ.18: 9 point enhanced strain solid element (12 incompatible modes)

EQ.±19: 8-node, 4 point cohesive element

EQ.20: 8-node, 4 point cohesive element with offsets for use with shells

EQ.±21: 6-node, 3 point pentahedron cohesive element

EQ.22: 6-node, 3 point pentahedron cohesive element with offsets for use with shells

EQ.23: 20-node solid formulation

EQ.24: 27-node solid formulation

EQ.25: 21-node, quadratic pentahedron

EQ.26: 15-node, quadratic tetrahedron

EQ.27: 20-node, cubic tetrahedron

EQ.28: 40-node, cubic pentahedron

EQ.29: 64-node, cubic hexahedron

EQ.41: Mesh-free (EFG) solid formulation

EQ.42: 4-node mesh-free (EFG) solid formulation

EQ.43: Mesh-free enriched finite element

EQ.62: 8 point brick with incompatible modes by assumed strain

If another element formulation is requested, the solver will substitute, when possible, one of the above in place of the one chosen. The type 1 element, constant stress, is generally much more accurate than the type 2 element, the selectively reduced integrated element for implicit problems.

3. **Elements for modified honeycomb material.** Element formulations 0 and 9 behave essentially as nonlinear springs to permit severe distortions sometimes seen in honeycomb materials. They are applicable only to *MAT_MODIFIED_HONEYCOMB, but as of R14, *MAT_ADD_EROSION can also be enabled. For formulation 0, the local coordinate system follows the element rotation, whereas, for formulation 9, the local coordinate system is based on axes passing through the centroids of the element faces. Formulation 0 is preferred for severe shear deformation where the barrier is fixed in space. If the barrier is attached to a moving body, which can rotate, then formulation 9 is usually preferred.

4. **ALE element (ELFORM = 11) shapes.** We designed the ALE solver assuming hexahedral meshes of ALE elements. The ALE solver treats tetrahedral elements as degenerate hexahedrons. Thus, the advection calculation may be inaccurate and cause LS-DYNA to crash. We recommend only using ALE meshes with exclusively hexahedral elements.

5. **8/9 point hexahedrons: types 2, 18, and 62.** Solid formulation 2 often employs a B-bar method and thereby assumes that pressure is constant throughout the element to avoid pressure locking during nearly incompressible flow. But when using material model 2, 21-23, 30, 38, 54, 57, 59, 63, 83, 91-92, 126, 143, 176-179, 181 (compressible option), 183, 189, 213, 215, 218, 221, 227, 249, 261-262, 266, 269, 274-275 or 295, solid formulation 2 employs full integration to treat compressible behavior better. IHYPER=±10 on *MAT_USER_DEFINED_MATERIAL_MODELS will also invoke full integration of solid form 2 when using the corresponding hyperelastic user material. If the element aspect ratios are poor, shear locking will lead to an excessively stiff response of solid form 2. Given poor aspect ratios, a better choice is the one point solid element (formulation 1), the 8 point solid formulation -1, or the 8 point solid formulation -2 (see Remark 13). Elements with formulations -1 and -2 are insensitive to a lousy aspect ratio but become very stiff if skewed or distorted. The enhanced assumed strain solid formulations -18 (8 point) and 18 (9 point) work well for elements with poor aspect ratios and can handle some distortion. Highly distorted elements lead to excessive stiffness for element formulations -18 and 18. Element form 62 is also an enhanced assumed strain element with 8 nodes that is not susceptible to volumetric or shear locking.

6. **Element type 99 for vibration.** Element type 99 is intended for vibration studies in the time domain using explicit time integration only. This model type may have a large number of elements and run for a relatively long duration. The purpose of this element is to achieve substantial CPU savings. This is achieved by imposing strict limitations on the range of applicability, thereby simplifying the calculations:

   a) Elements must be cubed; all edges must parallel to the global *x*-, *y*- or *z*-axis;

   b) Small displacement, small strain, negligible rigid body rotation;

   c) Elastic material only

   If these conditions are satisfied, the performance of the element is similar to the fully integrated S/R solid (ELFORM = 2) but at less CPU cost than the default solid element (ELFORM = 1). Single-element bending and torsion modes are included, so meshing guidelines are the same as for fully integrated solids, e.g., relatively thin structures can be modeled with a single solid element through the thickness if required. Typically, the CPU requirement per element-cycle is roughly two-thirds that of the default solid element.

   No damping is included in the element formulation (for example, volumetric damping). It is strongly recommended that damping be applied, using keywords including *DAMPING_PART_MASS or *DAMPING_FREQUENCY_RANGE_DEFORM.

7. **8-node cohesive element: Type ±19.** Element types ±19 are cohesive elements. The tractions on the midsurface, defined as the midpoints between the nodal pairs 1-5, 2-6, 3-7, and 4-8, are functions of the differences of the displacements between nodal pairs interpolated to the four integration points. Element type 19 only includes translational DOF and performs best if there is no offset between the bottom and top nodal planes (i.e. having zero thickness). As a complement there is element type -19, which also only includes translational DOF, but accounts for an offset between the top and bottom planes. This feature means that element type -19 can be used with both solid and shell elements. For implicit calculations, element types ±19 are internally switched to a version similar to -19, but with a higher degree of accuracy for large step sizes.

   The initial volume of the cohesive element may be zero, in which case, the density may be defined in terms of the area of the midsurface. Since the cohesive element represents an infinitely thin interface between other elements, its density can theoretically be zero. However, (especially when the cohesive elements are connected using tied contacts) we recommend setting the density to some nonzero value to avoid problems during model initiation. See parameter COHTIEM on *CONTROL_CONTACT, parameter ICOH on *CONTROL_SOLID, and *MAT_ADD_COHESIVE.

   We calculate he tractions in the local coordinate system defined at the centroid of the element; see Figure 41-26. Representing the rotation matrix from the local to the global coordinate system at time t as $\mathbf{R}(t)$, the initial coordinates as $\mathbf{X}$, and the current coordinates as $\mathbf{x}$, the displacements at an integration point are

$$\Delta\mathbf{u} = \mathbf{R}^{\mathrm{T}}(t)\Delta\mathbf{x} - \mathbf{R}^{\mathrm{T}}(0)\Delta\mathbf{X}$$

$$\Delta\mathbf{x} = \sum_{i=1}^{4} N_i(s,t)\Delta\mathbf{x}_{i+4,i}$$

$$\Delta\mathbf{X} = \sum_{i=1}^{4} N_i(s,t)\Delta\mathbf{X}_{i+4,i}$$

   The forces are obtained by integrating the tractions over the midsurface and rotating them into the global coordinate system. It is the sum over integration points g = 1,2,3,4.

$$\mathbf{F}_i = \mathbf{R}(t)\sum_{g=1}^{4}\mathbf{T}_g N_i(s_g,t_g)\det(\mathrm{J}_g), \quad \text{for } 1 \le i \le 4, \quad \text{and } \mathbf{F}_{i+4} = -\mathbf{F}_i \ ,$$

   where

   $\mathbf{T}_g$ = the traction stress in the local coordinate system

   N_i = the shape function of the cohesive element at node i

   s_g and t_g = the parametric coordinates of the 4 integration points (determined by INTFAIL on the cohesive material card).

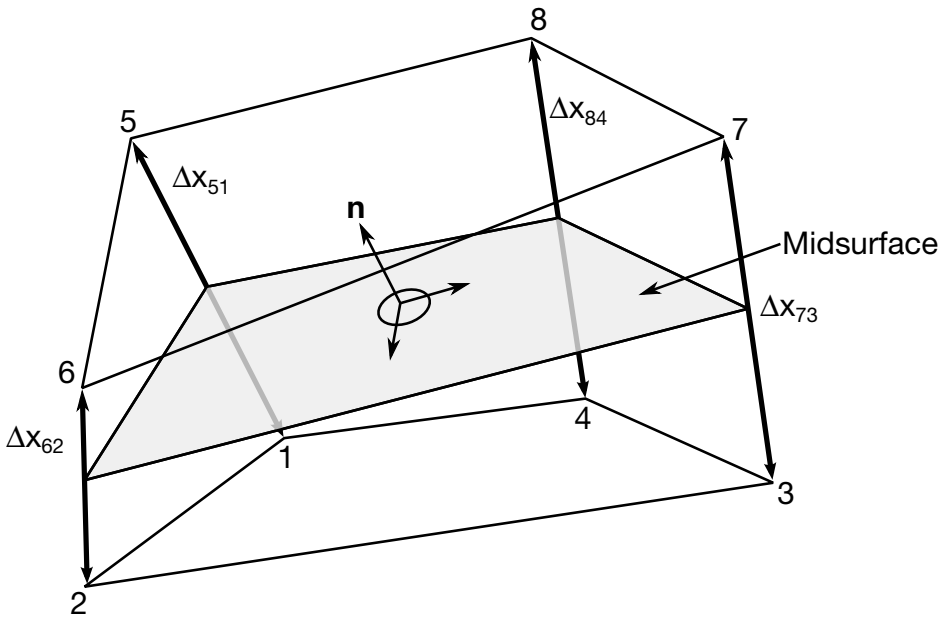


Figure 41-26. Illustration of solid local coordinates.

J_g = the integration point's portion of the determinate of the cohesive element, which is equivalent to the element volume

See Remark 10 regarding output for cohesive solids.

8. **6-node cohesive element: type ±21.** Element types ±21 are the pentahedral counterparts to element types ±19, with three nodes on the bottom and top surfaces. The tractions on the mid surface, defined as the midpoints between the nodal pairs 1-5, 2-6, and 3-7 are functions of the differences of the displacements between nodal pairs interpolated to three integration points. The ordering of the nodal points in *ELEMENT_SOLID is given by:

6-node (cohesive) pentahedron	N1, N2, N3, N3, N5, N6, N7, N7, 0, 0

Setting ESORT > 0 in *CONTROL_SOLID will automatically sort degenerated cohesive elements type ±19 to cohesive pentahedron elements type ±21. See Remark 10 regarding output for cohesive solids.

Element type 21 is internally switched to this type -21 for implicit calculations.

9. **Cohesive element with offsets: types 20 and 22.** Element type 20 is identical to element 19 but with offsets for use with shells. If COHOFF = 0, the element is assumed to be centered between two layers of shells on the cohesive element's lower (1-2-3-4) and upper (5-6-7-8) surfaces. The offset distances for both shells are one-half the initial thicknesses of the nodal pairs (1-5, 2-6, 3-7, and 4-8), separating the two shells. These offsets are used with the nodal forces to calculate moments that are applied to the shells. Element type 20 in tied contacts will

work correctly with the option TIED_SHELL_EDGE_TO_SURFACE, which transmits moments. Other tied options will leave the rotational degrees of freedom unconstrained, with the possibility that the rotational kinetic energy will cause a significant growth in the energy ratio.

Element type 22 is the pentahedron counterpart to element type 20, with three nodes on the bottom and top surfaces. The ordering of the nodal points in *ELEMENT_SOLID is identical to element type 21 (see Remark 8). Setting ESORT > 0 in *CONTROL_SOLID will automatically sort degenerated cohesive elements type 20 to cohesive pentahedron elements type 22.

For implicit calculations, element types 20 and 22 are internally switched to versions with a higher degree of accuracy for large step sizes.

See Remark 10 regarding output for cohesive solids.

10. **Cohesive element output.** Cohesive solids (formulations ±19, 20, ±21, 22) output three tractions, having units of force per unit area, into the d3plot database rather than the usual six stress components. The in-plane shear traction along the 1-2 edge replaces the x-stress, the orthogonal in-plane shear traction replaces the y-stress, and the traction in the normal direction replaces the z-stress.

    Similarly, the cohesive separation, having units of length, can be output to the strain field when STRFLG(L) = 1 on *DATABASE_EXTENT_BINARY. By default, the element average separation will be output. If NINTSLD = 8, the separation in each integration point is output as well. The orientation is the same as for the traction.

11. **10-Node tetrahedra: types 16 and 17.** Formulations 16 and 17 are 10-node tetrahedral formulations. The parameter NIPTETS in *CONTROL_SOLID controls the number of integration points for these formulations. Formulation 17 is generally preferred over formulation 16 in explicit analysis because, unlike formulation 16, the nodal weighting factors are equal, and thus nodal forces from contact and applied pressures are distributed correctly.

    When applying loads to 10-node tetrahedrons via segments, no load will be applied to the mid-side nodes if the segments contain only corner nodes. When defining contact, it is recommended that *CONTACT_AUTOMATIC_… be used and the contact surface of the 10-node tetrahedral part be specified by its part ID. In this manner, mid-side nodes receive contact forces.

    If the 10-node element connectivity is not defined in accordance with Figure 19-31 shown in *ELEMENT_SOLID, the order of the nodes can be quickly changed using a permutation vector specified with *CONTROL_SOLID. If *ELEMENT_SOLID defines 4-node tetrahedrons, the command *ELEMENT_SOLID_TET4TOTET10 can easily convert them to 10-node tetrahedrons. Because the

characteristic length of a 10-node tetrahedron is half that of a 4-node tetrahedron, the time step for the tetrahedrons will be smaller by a factor of 2. Setting the variable TET10S8 to 1 in *CONTROL_OUTPUT causes the full 10-node connectivity to be written to the d3plot and d3part databases.

12. **1-Point nodal pressure tetrahedron: type 13.** Element type 13 is identical to type 10 but includes additional averaging of nodal pressures, which significantly decreases the chance of volumetric locking. Therefore, it is well suited for applications with incompressible and nearly incompressible material behavior, such as rubber materials or ductile metals with isochoric plastic deformations (e.g., bulk forming). Compared to the standard tetrahedron (type 10), a speed penalty with a maximum of 25% can be observed. For implicit simulations, all material models supported for type 10 are also supported for this element, while for explicit currently material models *MAT_001 (excluding FLUID option), 003, 004, 006, 007, 015, 018, 019, 024, 027, 041-050, 064, 077, 081, 082, 089, 091, 092, 098, 103, 104, 105, 106, 107, 120, 122, 123, 124, 128, 129, 133, 157, 181, 183, 187, 199, 224, 225, 228, 233, 244, 251, 260, 263, 266, 269, 271, 272, 273, and 305 are fully supported. For other materials, this element behaves like the type 10 tetrahedron. Type 13 tetrahedral elements that use two different material models should not share nodes because the nodal pressure averaging will cause spurious energy. An exception to this rule is if the different materials have the same bulk modulus.

13. **8 point hexahedrons with poor aspect ratio: types -1 and -2.** Solid formulations -1 and -2 employ an assumed strain approach to avoid the shear locking behavior seen in formulation 2 elements with poor aspect ratios. Formulation -1 is a more computationally efficient implementation of formulation -2. Formulation -1 has a side effect: resistance to a particular deformation mode, similar to an hourglass mode, is weakened. This side effect is not truly hourglassing behavior, so there is no hourglass energy, and the behavior is not affected by hourglass parameters. For material models 57, 83, 181 (compressible option), and 274, formulations -1 and -2 employ full integration, rather than a B-bar method, to better deal with compressible behavior. If IHYPER = ±10 on *MAT_USER_DEFINED_MATERIAL_MODELS, this will also invoke full integration of the hyperelastic user material.

14. **Smoothed Particle Galerkin (SPG) method: Type 47.** The SPG method converts finite element nodes into particles. This method supports 4-node, 6-node, and 8-node *solid* elements. The method is suitable for severe deformation and failure analysis. SPG is only available for explicit analysis.

15. **1-Point tetrahedron: type 60.** Aside from including additional averaging of element volumetric locking, element type 60 is identical to type 10. In contrast to type 13, type 60 averages the normal stress between two adjacent elements, so nodes can be shared between the different materials with different bulk moduli.

This element supports explicit analysis but is still under development for implicit simulations.

16. **Higher order elements.** Quadratic solids (ELFORM = 24, 25, and 26) and cubic solids (ELFORM = 27, 28, and 29) remain under development.

17. **Incompatible modes elements: types -18 and 18.**

    a) Element type -18 is available for both implicit and explicit analysis. This is an 8-point integrated element, using 9 (Wilson) modes to alleviate Poisson locking in bending and another 4 to alleviate volumetric locking. The element is costly for explicit analysis; an additional factor of between 2 and 5 in simulation time is expected compared to element type 2. The exact number depends on the number of iterations needed to solve the compatibility equation for each element; typically, more iterations are needed for severely deformed elements and/or nonlinear effects in materials. In implicit analysis, the expense of the element is the same as in explicit, but this cost becomes insignificant relative to the global linear algebra. Increased accuracy seems to compensate for the cost. Therefore, the element should perform well in many types of analyses.

    b) Element type 18 is a similar element formulation to -18 but uses 9 integration points instead of 8. Similar to -18, it uses 9 modes to alleviate Poisson locking in bending. Unlike -18, it uses 3 instead of 4 incompatible modes for the volumetric treatment. This element is not available for explicit analysis, and when used, it will be switched to element formulation 1. Both elements 18 and -18 are based on the works of Simo and Rafai (1990) and Simo et al. (1993).

18. **Fully integrated QBI element.** Element form 62 is a fully integrated 8 node hexahedral element that uses an assumed strain field to prevent shear and volumetric locking and enhance bending stiffness. In Belytschko and Bindeman (1993), this assumed strain field is called the quintessential bending incompressible or QBI field. Element form 62 is available for implicit and explicit analysis. It is about 10% slower in explicit analyses than the fully integrated element formulation 2.

19. **EFG solid elements: types 41 and 42.** EFG element type 41 is supported for 4-node, 6-node, and 8-node solid elements. Element type 42 is only supported for 4-node tetrahedral meshes, but it is optimized to achieve better computational efficiency than 41. For three-dimensional tetrahedron *r*-adaptive analysis (ADPTYP = 7 in *CONTROL_ADAPTIVE and ADPOPT = 2 in *PART), the initial mesh must be purely comprised of tetrahedrons, and element type 42 should be used.

The EFG method still uses elements (so-called background elements or cells) to integrate nodal forces and stiffness in implicit analysis. The number of integration points for EFG depends on the EFG formulation and element type. For EFG type 41, the standard EFG formulation (IDIM = 1, 2, and 3) always allocates memory for 8 integration points, but the actual number of points used for the domain integration changes with the type of background element: 8 points for an 8-node element, 2 for a 6-node element, and 1 for a 4-node element. The stabilized formulation (IDIM = -1) only allocates and uses one integration point. It applies to 6-node elements, 8-node elements, or a mixed mesh of these two types of cells. EFG type 42 has only the standard formulation. It uses and allocates memory for one integration point since it applies to a pure tetrahedral mesh.

20. **Automatic sorting for EFG background mesh.** The current EFG formulation performs automatic sorting for finite element tetrahedral, pentahedron, and hexahedral elements of the background mesh to identify the mesh-free geometry and provide the contact surface definition in the computation.

21. **Essential boundary conditions.** The mixed transformation method, the coupled FEM/EFG method, and the fast transformation method were implemented for the EFG 3D solid formulation. These three features were added to improve the efficiency of the imposition of essential boundary conditions and the transfer of actual nodal and generalized nodal values. The mixed transformation method is equivalent to the full transformation method with improved efficiency. The behavior of the coupled FEM/EFG method is between FEM and EFG. The fast transformation method provides the most efficient and robust results.

22. **IDIM.** For compressible materials like foam and soil, IDIM = 1 is recommended. IDIM = 2 (default) is recommended for nearly incompressible materials like metal and rubber.

23. **TOLDEF.** This field is introduced to improve the negative volume problem usually seen during large deformation analysis. For the same analysis, the larger the magnitude of TOLDEF, the earlier the Semi-Lagrangian or Eulerian kernel is introduced into the EFG computation, so more CPU time is expected. A value between 0.0 and 0.1 is suggested for crashworthiness analysis. We suggest the Semi-Lagrangian kernel for solid materials and the Eulerian kernel for fluid and EOS materials.

24. **EFG fracture.** The EFG fracture method (IDIM = -2) is obsolete. For ductile failure, consider using *SECTION_SOLID_SPG.

Example:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$   *SECTION_SOLID
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  A bolt modeled with solids was found to have excessive hourglassing.
$  Thus, the section (sid = 116) associated with the bolt part was used
$  to specify that a fully integrated Selectively-Reduced solid element
$  formulation be used to totally eliminate the hourglassing (elform = 2).
$
*SECTION_SOLID
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
$       sid    elform
        116         2
$
*PART
bolts
$       pid       sid       mid     eosid      hgid    adpopt
         17       116         5
$
$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*SECTION_SOLID_PERI

Purpose: Define section properties for Peridynamics solid and laminate elements.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM						
Type	I/A	I						

Card 2	1	2	3	4	5	6	7	8
Variable	DR	PTYPE						
Type	F	I						
Default	1.01	1						

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation options (see Remarks): EQ.48: Peridynamics element formulation. This formulation works for both peridynamics solids (4, 6 and 8 node elements) and peridynamics laminates (4 node surface elements)
DR	Normalized horizon size, 0.6 – 1.2 is recommended
PTYPE	Peridynamics formulation: EQ.1: Bond based formulation (default)

Remarks:

1. **Peridynamics Solids.** The elements in a Peridynamics solid part should be defined with *ELEMENT_SOLID. These elements can be meshed with a 3D solid mesher. To represent material split, the elements should not share nodes. Thus, shared corners will have coincident nodes, one for each element. The total

number of nodes is then the summation over the number of elements of the number of nodes in each element.

2. **Peridynamics Laminates.** Elements in a Peridynamics laminate part should be specified with *ELEMENT_SOLID_PERI. See *ELEMENT_SOLID_PERI.

***SECTION_SPH**_*{OPTION}*

Available options include:

<BLANK>

ELLIPSE

INTERACTION (See Remark 3)

USER (See Remark 2)

Purpose: Define section properties for SPH particles.

> **NOTE:** This feature is not supported for use in implicit calculations.

Card Sets. For each SPH section add one set of the following cards (depending on the keyword option). This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	CSLH	HMIN	HMAX	SPHINI	DEATH	START	SPHKERN
Type	I/A	F	F	F	F	F	F	I
Default	none	1.2	0.2	2.0	0.0	10^{20}	0.0	0

Ellipse Card. Additional card for ELLIPSE keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	HXCSLH	HYCSLH	HZCSLH	HXINI	HYINI	HZINI		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.

VARIABLE	DESCRIPTION
CSLH	Constant used to calculate the initial smoothing length of the particles. The default value works for most problems. Values between 1.05 and 1.3 are acceptable. Taking a value less than 1 is inadmissible. Values larger than 1.3 will increase the computational time. The default value is recommended. See Remark 1.
HMIN	Scale factor for the minimum smoothing length. Ignored if FORM = 12 in *CONTROL_SPH. See Remark 1.
HMAX	Scale factor for the maximum smoothing length. Ignored if FORM = 12 in *CONTROL_SPH. See Remark 1.
SPHINI	Optional initial smoothing length (overrides true smoothing length). With this option LS-DYNA will not calculate the smoothing length during initialization, and the field CSLH is ignored.
DEATH	Time imposed SPH approximation is stopped.
START	Time imposed SPH approximation is activated.
SPHKERN	Option for SPH kernel functions (smoothing functions): EQ.0: Cubic spline kernel function (default). EQ.1: Quintic spline kernel function: a higher order smoothing function with a larger support size. It is only available for the 3D case with FORM = 0, 1, 5, 6, 9 and 10 (see *CONTROL_SPH). EQ.2: Quadratic spline kernel function: it helps to relieve the problem of compressive instability and aims for HVI problems. It is only available for the 3D case with FORM = 0, 1, 5, and 6 (see *CONTROL_SPH). EQ.3: Quartic kernel function: this kernel function is very close to cubic spline kernel function but is more stable. It is only available for the 3D case with FORM = 0, 1, 5, and 6 (see *CONTROL_SPH).
HXCSLH	Constant applied for the smoothing length in the x-direction for the ellipse case.
HYCSLH	Constant applied for the smoothing length in the y-direction for the ellipse case.
HZCSLH	Constant applied for the smoothing length in the z-direction for the ellipse case.

VARIABLE	DESCRIPTION
HXINI	Optional initial smoothing length in the x-direction for the ellipse case (overrides true smoothing length)
HYINI	Optional initial smoothing length in the y-direction for the ellipse case (overrides true smoothing length)
HZINI	Optional initial smoothing length in the z-direction for the ellipse case (overrides true smoothing length)

Remarks:

1. **Smoothing Length.** The SPH processor in LS-DYNA employs a variable smoothing length. LS-DYNA computes the initial smoothing length, h_0, for each SPH part by taking the maximum of the minimum distance between every particle and then scaling this value by CSLH. The recommended values of CSLH should be used so that the radius of the support domain covers more than two layers of SPH particles along each direction. Every particle has its own smoothing length which varies in time according to the following equation:

$$\frac{d}{dt}h(t) = h(t)\nabla \cdot \mathbf{v} \ ,$$

where $h(t)$ is the smoothing length, and $\nabla \cdot \mathbf{v}$ is the divergence of the flow. The smoothing length increases as particles separate and reduces as the concentration increases. This scheme is designed to hold constant the number of particles in each neighborhood. In addition to being governed by the above evolution equation, the smoothing length is constrained to be between a user-defined upper and lower value,

$$\text{HMIN} \times h_0 < h(t) < \text{HMAX} \times h_0.$$

Defining a value of 1 for HMIN and 1 for HMAX will result in a constant smoothing length in time and space.

When formulation 12 is employed (FORM = 12 in *CONTROL_SPH), the smoothing length remains constant with HMIN and HMAX set to 1.0 internally.

2. **USER Option.** The USER option allows the definition of customized subroutine for the variation of the smoothing length. A subroutine called *hdot* is defined in the file dyn21.F (Unix/linux) or lsdyna.f (Windows).

3. **Inter-Part Particle Interaction.** There are two fundamental ways that particles from different SPH parts can interact with each other. One way is through "particle approximation" and the other way is through *DEFINE_SPH_TO_SPH_-

COUPLING. When CONT = 0 in *CONTROL_SPH, "particle approximation" is used for all SPH parts in treating inter-part particle interaction. When CONT = 1 in *CONTROL_SPH, inter-part particle interaction by "particle approximation" occurs only for those SPH parts that use *SECTION_SPH_INTERACTION, while any SPH part that does not make use of *SECTION_SPH_INTERACTION will not participate in inter-part particle interaction, *except* as defined using *DEFINE_SPH_TO_SPH_COUPLING.

*SECTION_TSHELL

Purpose: Define section properties for thick shell elements.

Card Sets. For each TSHELL section include a set of the following cards. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SECID	ELFORM	SHRF	NIP	PROPT	QR	ICOMP	TSHEAR
Type	I/A	I	F	I	I	F	I	I
Default	none	1	1.0	2	1	0.0	0	0

Angle Cards. If ICOMP = 1 specify NIP angles putting 8 on each card. Include as many cards as necessary.

Card 2	1	2	3	4	5	6	7	8
Variable	B1	B2	B3	B4	B5	B6	B7	B8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
SECID	Section ID. SECID is referenced on the *PART card. A unique number or label must be specified.
ELFORM	Element formulation: EQ.1: one point reduced integration (default), EQ.2: selective reduced 2 × 2 in plane integration. EQ.3: assumed strain 2 × 2 in plane integration, see remark below. EQ.5: assumed strain reduced integration with brick materials EQ.6: assumed strain reduced integration with shell materials EQ.7: assumed strain 2 × 2 in plane integration.
SHRF	Shear factor. A value of 5/6 is recommended (see Remark 4).

VARIABLE	DESCRIPTION
NIP	Number of through thickness integration points for the thick shell. See the variable INTGRD in *CONTROL_SHELL for details of the through thickness integration rule. EQ.0: set to 2 integration points.
PROPT	Printout option: EQ.1: average resultants and fiber lengths, EQ.2: resultants at plan points and fiber lengths, EQ.3: resultants, stresses at all points, fiber lengths.
QR	Quadrature rule: LT.0.0: absolute value is specified rule number, EQ.0.0: Gauss (up to ten points are permitted), EQ.1.0: trapezoidal, not recommended for accuracy reasons.
ICOMP	Flag for layered composite material mode: EQ.1: a material angle is defined for each through thickness integration point. For each layer one integration point is used.
TSHEAR	Flag for transverse shear strain or stress distribution (see Remarks 4 and 5): EQ.0: Parabolic, EQ.1: Constant through thickness.
B1	β_1, material angle at first integration point. The same procedure for determining material directions is use for thick shells that is used for the 4 node quadrilateral shell.
B2	β_2, material angle at second integration point
B3	β_3, material angle at third integration point
⋮	⋮
BNIP	β_{NIP}, material angle at NIPth integration point

Remarks:

1. **Thick Shell Element Formulations.** Thick shell elements are bending elements that have 4 nodes on the bottom face and 4 on the top face. Thick shell element formulations 1, 2 and 6 are extruded thin shell elements and use thin shell material models and have an uncoupled stiffness in the z-direction. Thick shell element formulations 3, 5, and 7 are layered brick elements that use 3D brick material models. Thick shell formulations 3 and 5, and 6 are distortion sensitive and should not be used in situations where the elements are badly shaped. A single thick shell element through the thickness will capture bending response, but with thick shell formulation 3, at least two elements through the thickness are recommended to avoid excessive softness.

2. **Formulation 1 Quadrature Quirk.** When using Gaussian quadrature with element formulation 1, the number of integration points is automatically switched to 3 when NIP = 2 and 5 when NIP = 4.

3. **Implicit Time Integration.** Thick shell elements are available for implicit analysis with the exception of thick shell formulation 1. If an element of type 1 is specified in an implicit analysis, it is internally switched to type 2

4. **SHRF Field.** For ELFORM = 1, 2 and 6, the transverse shear stiffness is scaled by the SHRF parameter. Since the strain is assumed to be constant through the thickness, setting SHRF = 5/6 is recommended to obtain the correct shear energy. For ELFORM = 3 and 5, the SHRF parameter is not used, except for material types 33, 36, 133, 135, and 243. For ELFORM = 3, the shear stiffness is assumed constant through the thickness. For ELFORM = 5, 6, and 7, the shear distribution is assumed either parabolic if TSHEAR = 0, or constant if TSHEAR = 1. The parabolic assumption is good when the elements are used in a single layer to model a shell type structure, but the constant option may be better when elements are stacked one on top of the other.

5. **Modeling Composites.** Thick shell elements of all formulations can be used to model layered composites, but element formulations 5 and 6 use assumed strain to capture the complex Poisson's effects and through thickness stress distribution in layered composites. To define the layers of a composite, use QR < 0 to point to *INTEGRATION_SHELL data. Alternatively, the *PART_COMPOSITE_TSHELL keyword offers a simplified way to define the layers.

   When modeling composites, laminated shell theory may be used to correct the transverse shear strain if the shear stiffness varies by layer. Laminated shell theory is activated by setting LAMSHT = 4 or 5 on *CONTROL_SHELL. When laminated shell theory is active, the TSHEAR parameter works with all ELFORM values to select either a parabolic or constant shear stress distribution.

*SENSOR

The keyword *SENSOR provides a convenient way of activating and deactivating boundary conditions, airbags, discrete elements, joints, contact, rigid walls, single point constraints, and constrained nodes. The sensor capability is available as of the second release of version 971 and will evolve in later releases to encompass many more LS-DYNA capabilities and replace some of the existing capabilities, such as the airbag sensor logic. The keyword commands in this section are defined below in alphabetical order:

*SENSOR_CONTROL

*SENSOR_CPM_AIRBAG

*SENSOR_DEFINE_CALC-MATH

*SENSOR_DEFINE_ELEMENT

*SENSOR_DEFINE_FORCE

*SENSOR_DEFINE_FUNCTION

*SENSOR_DEFINE_MISC

*SENSOR_DEFINE_NODE

*SENSOR_SWITCH

*SENSOR_SWITCH_CALC-LOGIC

*SENSOR_SWITCH_SHELL_TO_VENT

An additional option **TITLE** may be appended to all the ***SENSOR** keywords. If this option is used, then an addition line is read for each sensor in 80a format which can be used to describe the sensor. At present, the title serves no purpose other than to perhaps lend clarity to input decks.

To define and use a sensor, three categories of sensor keyword commands are needed as shown in Figure 42-1.

1. Sensors are defined using the *SENSOR_DEFINE commands. Sensors provide a time history of model response that may be referred to by *SENSOR_SWITCH as a switching criterion. (Note: The time history of any sensor can be output using the SENSORD function in *DEFINE_CURVE_FUNCTION and *DATABASE_CURVOUT.)

   a) *SENSOR_DEFINE(_ELEMENT, _FORCE, _MISC, _NODE)

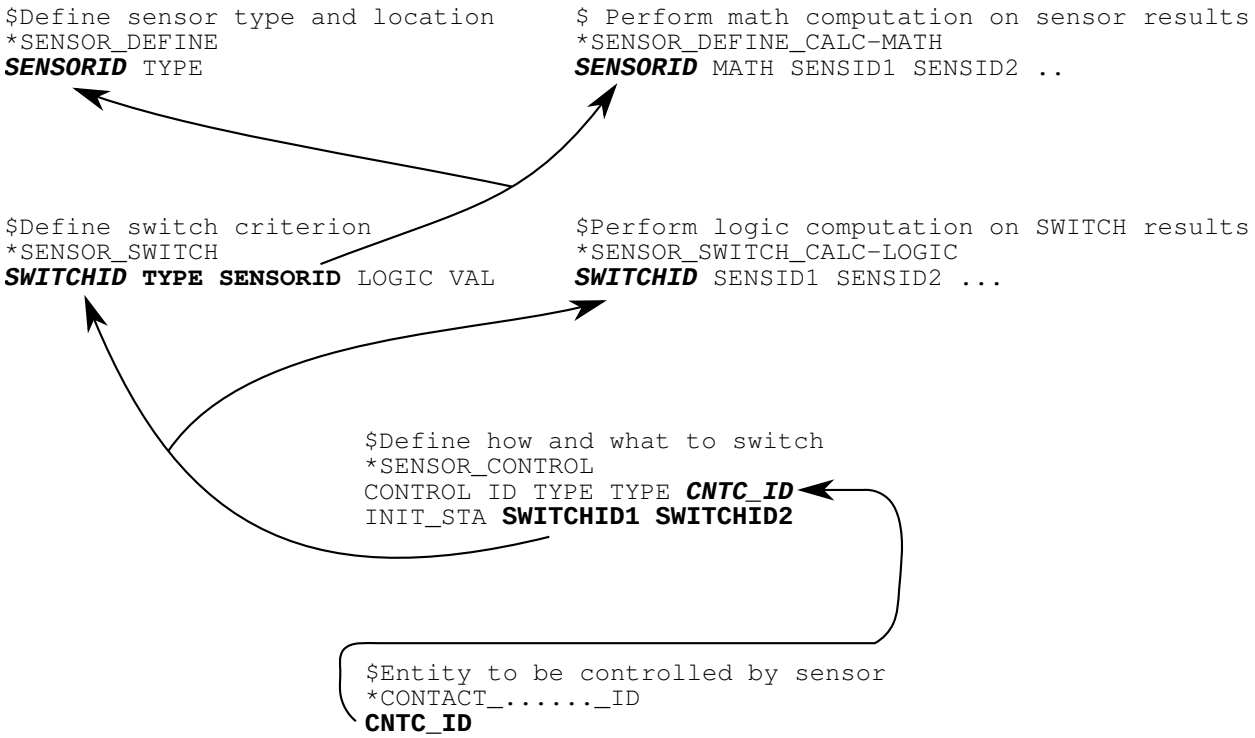


Figure 42-1. Relationship between sensor keyword definitions.

These commands define a sensor's ID, type, and location.

b) *SENSOR_DEFINE_CALC-MATH, *SENSOR_DEFINE_FUNCTION

These commands define a sensor whose value is a mathematical expression involving the values from other sensors.

2. The *SENSOR_SWITCH family of keywords define sensor switching criterion. *SENSOR_SWITCH can be combined with the logical calculation command *SENSOR_SWITCH_CALC-LOGIC for more complicated definitions. The logic value yielded by this category of commands can be referred by *SENSOR_CONTROL to determine if a status switch condition is met.

a) *SENSOR_SWITCH

This command compares the numerical value from *SENSOR_DEFINE or *SENSOR_DEFINE_CALC-MATH with the given criterion to see if a switching condition is met.

b) *SENSOR_SWITCH_CALC-LOGIC

This command performs logical calculation on the information from SENSOR_SWITCH.

3. *SENSOR_CONTROL determines how and what to switch based on the logical values from *SENSOR_SWITCH and/or *SENSOR_SWITCH_CALC-LOGIC.

*SENSOR_CONTROL

Purpose: Use switches (*SENSOR_SWITCH) to toggle on or off the effects of other keywords, such as *CONTACT or *AIRBAG.

Card Summary:

Card Sets. For each sensor control, include a set of the following cards. This input ends at the next keyword ("*") card.

Card 1a. Include this card for all values of TYPE, except CURVE, PRESC-MOT, PRESC-ORI, PRESSURE, and ELESET.

CNTLID	TYPE	TYPEID		NREP		SIDR	

Card 1b. Include this card for TYPE = CURVE, PRESC-MOT, PRESC-ORI, or PRESSURE (controlling *DEFINE_CURVE *BOUNDARY_PRESCRIBED_MOTION, *BOUNDARY_PRESCRIBED_RIGID_ORIENTATION, or *LOAD_SEGMENT_SET).

CNTLID	TYPE	TYPEID	TIMEOFF	NREP	DEFCV		

Card 1c. Include this card for TYPE = ELESET (controlling erosion of a set of elements of a specified type).

CNTLID	TYPE	TYPEID	IDISCL	NREP	ESTYP		

Card 2. This card is required.

INITSTT	SWIT1	SWIT2	SWIT3	SWIT4	SWIT5	SWIT6	SWIT7

All other values of Type Card. This card is included for all values of TYPE, except PRESC-MOT, PRESC-ORI, PRESSURE, and ELESET.

Card 1a	1	2	3	4	5	6	7	8
Variable	CNTLID	TYPE	TYPEID		NREP		SIDR	
Type	I	A	I		I		I	

VARIABLE	DESCRIPTION
CNTLID	Sensor control ID
TYPE	Entity to be controlled: EQ.AIRBAG: *AIRBAG

VARIABLE	DESCRIPTION
EQ.BAGVENTPOP:	Opening and closing the airbag venting holes (see Remark 1)
EQ.BELTPRET:	Belt pretensioner firing (see Remark 2)
EQ.BELTRETRA:	Locking the belt retractor (see Remark 2)
EQ.BELTSLIP:	Controlling the slippage of slip ring element (see Remark 3)
EQ.CONTACT:	*CONTACT
EQ.CONTACT2D:	*CONTACT_2D
EQ.CONSTRL:	*CONSTRAINED_LOCAL
EQ.CNRB:	*CONSTRAINED_NODAL_RIGID_BODY
EQ.CPM:	*AIRBAG_PARTICLE
EQ.DEF2RIG:	*DEFORMABLE_TO_RIGID_AUTOMATIC (see Remark 4)
EQ.EM:	EM solver (see Remark 6)
EQ.FUNCTION:	*DEFINE_CURVE_FUNCTION (see Remark 5)
EQ.JOINT:	*CONSTRAINED_JOINT
EQ.JOINTSTIF:	*CONSTRAINED_JOINT_STIFFNESS
EQ.LOADTHM:	*LOAD_THERMAL_VARIABLE
EQ.M PRESSURE:	*LOAD_MOVING_PRESSURE
EQ.POREAIR:	*MAT_ADD_PORE_AIR
EQ.PZBC:	*BOUNDARY_PZEPOT
EQ.RWALL:	*RIGIDWALL
EQ.SPC:	*BOUNDARY_SPC
EQ.SPOTWELD:	*CONSTRAINED_SPOTWELD
EQ.BPWPN:	*BOUNDARY_PWP_NODE/SET_ID
TYPEID	ID of entity to be controlled if TYPE ≠ FUNCTION, LOADTHM, or POREAIR. If TYPE = FUNCTION, see Remark 5. For TYPE = LOADTHM, TYPEID is the node set for which the temperature boundary condition specified by either *LOAD_THERMAL_VARIABLE or *LOAD_THERMAL_VARIABLE_NODE will be controlled. For TYPE = POREAIR, TYPEID is the ID of the part containing material with pore air.

VARIABLE	DESCRIPTION
NREP	Number of times to repeat a cycle of switches, SWIT*n*, defined on Card 2. For example, a definition of SWIT*n* like "601, 602, 601, 602, 601, 602" can be replaced by setting NREP to 3 and SWIT*n* to "601, 602". Setting NREP = -1 repeats the cycle an infinite number of times. The default is 0.
SIDR	Flag controlling the use of sensor control during dynamic relaxation: EQ.0: Used in the normal analysis phase only EQ.1: Used in the dynamic relaxation phase but not the normal analysis phase EQ.2: Used in both the dynamic relaxation phase and normal analysis phase

TYPE CURVE, PRESC-MOT, PRESC-ORI, and PRESSURE Card. Include this card for TYPE = CURVE, PRESC-MOT, PRESC-ORI, or PRESSURE.

Card 1b	1	2	3	4	5	6	7	8
Variable	CNTLID	TYPE	TYPEID	TIMEOFF	NREP	DEFCV		
Type	I	A	I	I	I	A/F		

VARIABLE	DESCRIPTION
CNTLID	Sensor control ID
TYPE	Entity to be controlled: EQ.CURVE: *DEFINE_CURVE, time-dependent curve EQ.PRESC-MOT: *BOUNDARY_PRESCRIBED_MOTION EQ.PRESC-ORI: *BOUNDARY_PRESCRIBED_ORIENTATION_RIGID EQ.PRESSURE: *LOAD_SEGMENT_SET
TYPEID	ID of entity to be controlled
TIMEOFF	Flag to offset time in curve: EQ.0: No offset is applied. EQ.1: Offset the abscissa of the time-dependent curve by the time

VARIABLE	DESCRIPTION

value at which the sensor is triggered.

Type Of Control	Curve Affected
CURVE	*DEFINE_CURVE
PRESSURE	*LOAD_SEGMENT_SET
PRESC-MOT	*BOUNDARY_PRESCRIBED_MOTION
PRESC-ORI	*BOUNDARY_PRESCRIBED_ORIENTATION_RIGID

NREP — Number of times to repeat a cycle of switches, SWIT*n*, defined on Card 2. For example, a definition of SWIT*n* like "601, 602, 601, 602, 601, 602" can be replaced by setting NREP to 3 and SWIT*n* to "601, 602". Setting NREP = -1 repeats the cycle an infinite number of times. The default is 0.

DEFCV — Default curve value when a curve is not active for TYPE = CURVE only. If DEFCRV = "LASTSTEP", the curve value right before the curve is turned off becomes the default curve value.

TYPE ELESET Card. This card is included when TYPE is ELESET.

Card 1c	1	2	3	4	5	6	7	8
Variable	CNTLID	TYPE	TYPEID	IDISCL	NREP	ESTYP		
Type	I	A	I	I	I	A		

VARIABLE	DESCRIPTION

CNTLID — Sensor control ID

TYPE — Entity to be controlled:

EQ.ELESET: Element set; see "ESTYP" below.

TYPEID — ID of entity to be controlled

IDISCL — For ESTYP = DISC, flag for updating the reference length of the discrete element with its length when it is turned on:

EQ.0: Discrete element's reference length remains the same when it is turned on.

EQ.1: Discrete element's length when it is turned on is used as its new reference length.

NREP — Number of times to repeat a cycle of switches, SWIT*n*, defined on

VARIABLE	DESCRIPTION
	Card 2. For example, a definition of SWIT*n* like "601, 602, 601, 602, 601, 602" can be replaced by setting NREP to 3 and SWIT*n* to "601, 602". Setting NREP = -1 repeats the cycle an infinite number of times. The default is 0.
ESTYP	Type of element set to be controlled. With initial status set to "ON," all the elements included in set TYPEID can be eroded when the controller status is changed to "OFF." When TYPEID is not defined, all elements of type ESTYP in the whole system will be eroded. EQ.BEAM: Beam element set EQ.DISC: Discrete element set EQ.SHELL: Thin shell element set EQ.SOLID: Solid element set EQ.TSHELL: Thick shell element set

Card 2	1	2	3	4	5	6	7	8
Variable	INITSTT	SWIT1	SWIT2	SWIT3	SWIT4	SWIT5	SWIT6	SWIT7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
INITSTT	Initial status: EQ.On: Initial status is on. EQ.Off: Initial status is off.
SWIT*n*	ID of n^{th} switch. At the start of the calculation SWIT1 is *active*, meaning that it controls the state of the feature specified in TYPE-ID. After SWIT1 triggers, then SWIT2 becomes active; after SWIT2 triggers, then SWIT3 becomes active; this process will continue until the entire stack of switches has been exhausted.

Remarks:

1. **Activation of bag venting.** BAGVENTPOP activates (opens) or deactivates (closes) the venting holes of *AIRBAG_HYBRID and *AIRBAG_-

WANG_NEFSKE. It overwrites the definitions of PVENT of *AIRBAG_HYBRID and PPOP of *AIRBAG_WANG_NEFSKE. More than one switch can be input to open/close initially closed/opened holes, and then reclose/reopen the holes.

2. **Seat belt retractors.** The locking (or firing) of the seat belt retractor (or pretensioner) can be controlled through either a general sensor, option BELTRETRA (or BELTPRET), or seat belt sensors, *ELEMENT_SEATBELT_SENSOR. When BELTRETRA (or BELTPRET) is used, the SBSID*i* in *ELEMENT_SEATBELT_-RETRACTOR (or PRETENSIONER) should be left blank.

3. **Seat belt slip ring.** For a one-way slip ring, a nonzero DIRECT in *ELEMENT_-SEATBELT_SLIPRING, BELTSLIP activates the constraint of one-way slippage when the status of *SENSOR_CONTROL is on. When the *SENSOR_CONTROL is turned off, the one-way slippage constraint is deactivated, therefore allowing slippage in both directions.

   For a two-way slip ring (DIRECT = 0), BELTSLIP activates slippage in both directions when the status of *SENSOR_CONTROL is on. When the status of *SENSOR_CONTROL is off, the slip ring locks up, allowing no slippage.

4. **Switching between rigid and deformable.** DEF2RIG provides users more flexibility in controlling the material switch between rigid and deformable. The status of ON triggers the switch, causing the deformable material to become rigid. Rigidized material can then return to a deformable status when the status becomes OFF. As many as seven switches can be input; any of them will change the status triggered by its preceding switch or the initial condition, INTSTT.

5. **Function for sensor control.** When the input parameter TYPE of *SENSOR_-CONTROL is set to "FUNCTION", the function "SENSOR(cntlid)" as described in *DEFINE_CURVE_FUNCTION takes on a value that depends on the current status of the *SENSOR_CONTROL. That status is either on or off at any given point in time. If the status is on, the value of function SENSOR(cntlid) is set to the integer value 1. If the status is off, the value of function SENSOR(cntlid) is set to the input parameter TYPEID (an integer) as specified in *SENSOR_CONTROL.

   To help clarify the relationship between *SENSOR_CONTROL and *DEFINE_-CURVE_FUNCTION, consider the following example. Suppose a *SENSOR_-CONTROL defined with CNTLID = 101, TYPE="FUNCTION", and TYPEID = -2 has a status of off. Then a *DEFINE_CURVE_FUNCTION defined as “2+3*sensor(101)” will have a value of 2 + 3(-2) = -4. On the other hand, if the status of the *SENSOR_CONTROL changes to on, the *DEFINE_CURVE_FUNCTION takes on a value of 2 + 3(1) = 5.

6. **EM solver.** TYPE = EM can be used to turn on/off EM solver. Since it serves the same function as *EM_CONTROL_SWITCH, it cannot be in an input deck with *EM_CONTROL_SWITCH.

***SENSOR_CPM_AIRBAG**_*{OPTION}*

Available options include:

ID

TITLE

OPTION provides a means to specify an airbag ID number and a heading for the airbag.

Purpose: This command will associate a CPM airbag with a sensor switch (see *SENSOR_SWITCH). When the condition flag is raised, the specified CPM airbag will deploy. All time dependent curves used for the CPM airbag are shifted by the activation time including the *AIRBAG_PARTICLE curves for the inflator and vent as well as the *MAT_FABRIC curves for TSRFAC.

ID Card. Additional card for the ID or TITLE keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ABID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	CPMID	SWITID	TBIRTH	TDEATH	TDR	DEFPS	RBPID	TFIRE
Type	I	I	F	F	F	I	I	F

VARIABLE	DESCRIPTION
ABID	Airbag ID
HEADING	Airbag descriptor
CPMID	Bag ID of *AIRBAG_PARTICLE_ID
SWITID	Switch ID of *SENSOR_SWITCH
TBIRTH	If SWITID is set, TBIRTH is not active. If SWITID is 0, TBIRTH is the activation time for the bag with ID = CPMID. All of the time-dependent curves that are used in this bag will be offset by the

VARIABLE	DESCRIPTION
	value of TBIRTH.
TDEATH	Disable the CPMID bag when the simulation time exceeds this value.
TDR	If TDR is greater than 0 the bag with ID = CPMID will be rigid starting at first cycle and switch to deformable at time TDR.
DEFPS	Part set ID specifying which parts of the bag with ID = CPMID are deformable.
RBPID	Part ID of the rigid body to which the part is merged.
TFIRE	Control activation time for mass flow rate and temperature curves

*SENSOR_DEFINE_CALC-MATH_{OPTION}

Available options include:

<BLANK>

UPDATE

With the UPDATE keyword option, you can specify the birth time, death time, and value update frequency for the sensor. Note that without the UPDATE keyword option LS-DYNA updates the sensor value every time step.

Purpose: Defines a new sensor with a unique ID. The values associated with this sensor are computed by performing mathematical calculations with the information obtained from sensors defined by the *SENSOR_DEFINE_*OPTION*.

Card Summary:

Card 1. Include this card or, if using the UPDATE keyword option, a set of this card and Card 2 for each math sensor. This input ends at the next keyword ("*") card.

SENSID	CALC	SENS1	SENS2	SENS3	SENS4	SENS5	SENS6

Card 2. Include a set of Card 1 and this card if using the UPDATE keyword option.

BIRTH	DEATH	DTUPD					

Data Card Definitions:

Math Sensor Cards. For each math sensor, include this card or, if using the UPDATE keyword option, a set of this card and Card 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	CALC	SENS1	SENS2	SENS3	SENS4	SENS5	SENS6
Type	I	A	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SENSID	Sensor ID
CALC	Mathematical calculation. See Table 42-2.

VARIABLE	DESCRIPTION
SENS*i*	i^{th} Sensor ID

UPDATE Card. Include a set of Card 1 and this card if using the UPDATE keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve defining update interval as a function of time.

Remarks:

All sensors, SENS*i*, defined with either SENSOR_DEFINE_NODE_SET or SENSOR_DEFINE_ELEMENT_SET, must refer to either the same node set or the same element set.

Mathematical Functions:

FUNCTION	DESCRIPTION	MATHEMATICAL FORM
ABSSUM	Absolute value of the sum of sensor values	$\lvert SENS1 + SENS2 + \cdots \rvert$
MIN	The minimum of sensor values	$\min(SENS1, SENS2, \ldots)$
MAX	The maximum of sensor values	$\max(SENS1, SENS2, \ldots)$
MAXMAG	The maximum of magnitude of sensor values	$\max(\lvert SENS1 \rvert, \lvert SENS2 \rvert, \ldots)$
MINMAG	The minimum of the magnitude of sensor values	$\min(\lvert SENS1 \rvert, \lvert SENS2 \rvert, \ldots)$
MULTIPLY	Multiplication of sensor values; negative for division (performed left to right)	$SENS1 \times SENS2 \times \cdots$

FUNCTION	DESCRIPTION	MATHEMATICAL FORM
SQRE	Summation of squared values of sensor values	$SENS1^2 + SENS2^2 + \cdots$
SQRTSQRE	Square root of the sum of squared values	$\sqrt{SENS1^2 + SENS2^2 + \cdots}$
SQRT	Summation of square root of sensor values; negative for subtracting values	$\sqrt{SENS1} + \sqrt{SENS2} + \cdots$
SUMABS	Summation of absolute sensor values	$\lvert SENS1\rvert + \lvert SENS2\rvert + \cdots$
SUM	Summation of sensor values; negative for subtracting values	$SENS1 + SENS2 + \cdots$

Table 42-2. Available mathematical functions.

Example:

```
$
$ assume set_2 to have 100 solid elements
*SENSOR_DEFINE_ELEMENT_SET
$ this sensor traces xx-strain of all 100 solid elements in set-2
        91     SOLID        -2        XX    STRAIN
*SENSOR_DEFINE_ELEMENT_SET
$ this sensor traces yy-strain of all 100 solid elements in set-2
        92     SOLID        -2        YY    STRAIN
*SENSOR_DEFINE_ELEMENT_SET
$ this sensor traces zz-strain of all 100 solid elements in set-2
        93     SOLID        -2        ZZ    STRAIN
*SENSOR_DEFINE_CALC-MATH
$ this sensor traces strain magnitudes of all 100 solid elements in set-2
       104  SQRTSQRE        91        92        93         0         0         0
*SENSOR_SWITCH
$ Because ELEMID of *sensor_define_element_set was input as "-2", SWITCH-1 will be
$ turned on if at least one of 100 elements has a strain magnitude>2.0E-4
$ On the other hand, If ELEMID was input as "2", SWITCH-1 will be turned on if
$ all 100 elements have strain magnitudes>2.0E-4
         1    SENSOR       104        GT    2.0E-4         0     0.001
$
```

***SENSOR_DEFINE_ELEMENT**_{OPTION1}_{OPTION2}

For *OPTION1* the available options are:

<BLANK>

SET

With the SET option active, the sensor value or values depends on a set of elements. See Remark 1.

For *OPTION2* the available options are:

<BLANK>

UPDATE

With the UPDATE keyword option, you can specify the birth time, death time, and value update frequency for the sensor. Note that without the UPDATE keyword option LS-DYNA updates the sensor value every time step.

Purpose: Define a strain gage type element sensor that checks the stress, strain, or resultant force of an element or element set.

Card Summary:

Card Sets. For each element sensor matching the specified keyword options include one set of the following data cards. Include as many sets as needed. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SENSID	ETYPE	ELEMID	COMP	CTYPE	LAYER	SF	PWR

Card 2. Include this card if using the SET option. If only one element set is specified and the UPDATE keyword option is not used, this card may be omitted if not needed. Otherwise, it must be included, but it may be left blank. Note that if this card is included and ELEMID < 0, then this card must be blank.

SETOPT							

Card 3. Include this card if using the UPDATE keyword option.

BIRTH	DEATH	DTUPD					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	ETYPE	ELEMID	COMP	CTYPE	LAYER	SF	PWR
Type	I	A	I	A	A	A/I	R	R

VARIABLE	DESCRIPTION
SENSID	Sensor ID
ETYPE	Element type. Available options include: EQ.BEAM: Beam element EQ.DISC-ELE: Discrete element EQ.SEATBELT: Seat belt element EQ.SHELL: Shell element EQ.SOLID: Solid element EQ.TSHELL: Thick shell element
ELEMID	Element ID or element set ID when the SET keyword option is active. In the case of the SET keyword option with SETOPT not defined, determining the status of a related *SENSOR_SWITCH depends on the sign of ELEMID. If SETOPT is defined, then ELEMID must be greater than 0. See Remark 1.
COMP	Component type. The definition of the component, and its related coordinate system, is consistent with that of elout. Leave blank for discrete elements. Available options for elements other than discrete elements include: EQ.AXIAL: Axial EQ.F[ID]: Value of a function, *DEFINE_FUNCTION, with all 6 stress/strain components as input. It is available for shells, thick shells, and solids. The ID of the function must follow right after F. For instance, F1000 means the function with ID of 1000 defines the sensed value. EQ.HYDR: Hydrostatic stress or strain, meaning the average

VARIABLE	DESCRIPTION
	of the three normal components, for shells, thick shells, and solids
	EQ.MAXS: Maximum shear stress or strain for shells, thick shells, and solids
	EQ.PRIN1: 1st principal stress or strain for shells, thick shells, and solids
	EQ.PRIN2: 2nd principal stress or strain for shells, thick shells, and solids
	EQ.PRIN3: 3rd principal stress or strain for shells, thick shells, and solids
	EQ.SHEARS: Local *s*-direction
	EQ.SHEART: Local *t*-direction
	EQ.VM: von Mises stress for shells, thick shells, and solids
	EQ.XX: *x*-normal component for shells, thick shells, and solids
	EQ.XY: *xy*-shear component for shells, thick shells, and solids
	EQ.YY: *y*-normal component for shells, thick shells, and solids
	EQ.YZ: *yz*-shear component for shells, thick shells, and solids
	EQ.ZX: *zx*-shear component for shells, thick shells, and solids
	EQ.ZZ: *z*-normal component for shells, thick shells, and solids
CTYPE	Sensor type:
	EQ.AREA: Area of shell element. COMP is ignored.
	EQ.DLEN: Change in length for discrete or seat belt element. COMP is ignored.
	EQ.FAIL: Failure of the element, sensor value = 1 when the element fails, = 0 otherwise. COMP is ignored.
	EQ.FORCE: Force resultants for beams, seat belt, or translational discrete element; moment resultant for rotational discrete element

VARIABLE	DESCRIPTION
	EQ.MOMENT: Moment resultants for beams EQ.STRAIN: Strain component for shells, thick shells, and solids EQ.STRESS: Stress component for shells, thick shells, and solids
LAYER	Layer of the integration point in a shell or thick shell element. Options include: EQ.BOT: Component at lower surface meaning the integration point with the smallest through-the-thickness local coordinate EQ.TOP: Component at upper surface meaning the integration point with the largest through-the-thickness local coordinate When CTYPE = STRESS, LAYER can be an integer i to monitor the stress of the i^{th} integration point.
SF, PWR	Optional parameters, scale factor and power, for users to adjust the resultant sensor value. The resultant sensor value is $[\text{SF} \times (\text{Original Value})]^{\text{PWR}}$

SET Card. Additional card for the SET keyword option. If you specify more than one element set sensor or additionally use the UPDATE keyword option, this card must be included for each set sensor even if left blank. If ELEMID < 0 and this card is included, it must be blank.

Card 2	1	2	3	4	5	6	7	8
Variable	SETOPT							
Type	A							
Default	Optional							

VARIABLE	DESCRIPTION
SETOPT	Option to process set of data when the SET keyword option is specified (see Remark 1). If you set SETOPT, then ELEMID must be greater than 0. When SETOPT is defined, a single value will be

VARIABLE	DESCRIPTION
	reported. The single reported value could be:
	EQ.AVG: The average value of the dataset
	EQ.MAX: The maximum value of the dataset
	EQ.MIN: The minimum value of the dataset
	EQ.SUM: The sum of the dataset

UPDATE Card. This card is included if the UPDATE keyword option is used.

Card 3	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve ID for a curve defining the update interval as a function of time.

Remarks:

1. **SET option.** When SETOPT is not defined for the SET option, a list of elemental data will be reported, one for each element in the element set. *SENSOR_DEFINE_FUNCTION and *SENSOR_DEFINE_CALC-MATH can process these reported elemental values, resulting in a list of processed values. Note that all sensor definitions referred to by these two processing commands must have the same number of data points. These elemental values also can determine if the status of a *SENSOR_SWITCH will be changed. Depending on the sign of ELEMID, it can take only one single data point (ELEMID < 0) or the whole data set (ELEMID > 0) to meet the switch condition to change the status of the related *SENSOR_SWITCH. The reported elemental values cannot be accessed by commands like *DEFINE_CURVE_FUNCTION; see SENSORD option.

   When SETOPT is defined, the elemental values of all elements in the element set will be processed, depending on the definition of SETOPT; the resulting value is reported as the single sensor value. The reported value can be processed by both

*SENSOR_DEFINE_FUNCTION and *SENSOR_DEFINE_CALC-MATH as well as other regular sensors. It can also be used to determine the status of a *SENSOR_SWITCH. A *DEFINE_CURVE_FUNCTION using the SENSORD option may also access this reported value. Note that ELEMID *must be greater* than 0 for this case.

***SENSOR_DEFINE_FORCE**_*{OPTION}*

Available options include:

<BLANK>

UPDATE

The UPDATE keyword option enables providing the birth time, death time, and value update frequency for the sensor. Note that without the UPDATE keyword option, LS-DYNA updates the sensor value every time step.

Purpose: Define a force transducer-type sensor.

Card Summary:

Card 1. For each force sensor, include one of this card or, if using the UPDATE keyword option, a set of this card and Card 2. This input ends at the next keyword ("*") card.

SENSID	FTYPE	TYPEID	VID	CRD	AIRBPID		

Card 2. Include this card in a set with Card 1 for the UPDATE keyword option.

BIRTH	DEATH	DTUPD					

Data Card Definitions:

Force Sensor Cards. Include one additional card for each force sensor. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	FTYPE	TYPEID	VID	CRD	AIRBPID		
Type	I	A	I	A/I	I	I		

VARIABLE	DESCRIPTION
SENSID	Sensor ID.
FTYPE	Force type. See Table 42-3 and Remark 1.
TYPEID	ID defined in the associated keyword command. See Table 42-3.

VARIABLE	DESCRIPTION
VID	Vector along which the forces is measured.
	EQ.X: *x*-direction in coordinate system CRD
	EQ.Y: *y*-direction in coordinate system CRD
	EQ.Z: *z*-direction in coordinate system CRD
	EQ.XL: *x*-direction in the local coordinate system for JOINTSTIF only
	EQ.YL: *y*-direction in the local coordinate system for JOINTSTIF only
	EQ.ZL: *z*-direction in the local coordinate system for JOINTSTIF only
	EQ.M: Force magnitude
	EQ.XMOMENT: *x*-direction moment for JOINT, JOINTSTIF, PRESC-MOT, SPC, or X-SECTION
	EQ.YMOMENT: *y*-direction moment for JOINT, JOINTSTIF, PRESC-MOT, SPC, or X-SECTION
	EQ.ZMOMENT: *z*-direction moment for JOINT, JOINTSTIF, PRESC-MOT, SPC, or X-SECTION
	EQ.XLMOMENT: *x*-direction moment in the local coordinate system for JOINTSTIF only
	EQ.YLMOMENT: *y*-direction moment in the local coordinate system for JOINTSTIF only
	EQ.ZLMOMENT: *z*-direction moment in the local coordinate system for JOINTSTIF only
	EQ.MMOMENT: Moment magnitude for JOINT, JOINTSTIF, PRESC-MOT or SPC
	VID∈{INT}: Vector ID n in coordinate system CRD
CRD	Optional coordinate system, defined by *DEFINE_COORDINATE_NODES, to which vector VID is attached. If blank, the global coordinate system is assumed.
AIRBPID	Part ID of a CPM or CPG airbag (see *AIRBAG_PARTICLE or *AIRBAG_CPG). This field is optional and is ignored unless FTYPE = CPM or CPG. If it is unset, the sensor returns airbag

VARIABLE	DESCRIPTION
	pressure. If a part ID is provided, the sensor returns the part pressure.

UPDATE card. Include this card in a set with Card 1 for the UPDATE keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve defining update interval as a function of time.

Table 42-3. Force transducer-type sensor

FTYPE	TYPEID (Enter the ID defined in the following keyword commands)	Output	ASCII file
AIRBAG	*AIRBAG	Airbag pressure	abstat
CONTACT	*CONTACT	Contact force on the surfa side	rcforc
CONTACT2D	*CONTACT_2D	Contact force on the surfa or sph side	rcforc
CPG	*AIRBAG_CPG	Airbag or part pressure	abstat_cpg
CPM	*AIRBAG_PARTICLE	Airbag or part pressure	abstat_cpm
JOINT	*CONSTRAINED_JOINT	Joint force	jntforc
JOINTSTIF	*CONSTRAINED_JOINT_-STIFFNESS	Joint stiffness force	jntforc
PRESC-MOT	*BOUNDARY_PRESCRIBED_-MOTION	Prescribed motion force	bndout

FTYPE	TYPEID (Enter the ID defined in the following keyword commands)	Output	ASCII file
RWALL	*RIGIDWALL	Rigid wall force	rwforc
SPC	*BOUNDARY_SPC	SPC reaction force	spcforc
SPOTWELD	*CONSTRAINED_POINTS	Spot weld force	swforc
X-SECTION	*DATABASE_CROSS_SECTION	Section force	secforc

Remarks:

1. **FTYPE = PRESC-MOT.** A sensor registers the same force as in the BNDOUT file. In case of multiple *BOUNDARY_PRESCRIBED_MOTION instances defining the kinematics of an entity (node/node set/rigid body, etc.), this sensor registers a resultant force of all related *BOUNDARY_PRESCRIBED_MOTION to the entity. For example, let the *BOUNDARY_PRESCRIBED_MOTION with ID 100 prescribe the x-displacement of node 50, and the *BOUNDARY_PRESCRIBED_MOTION instance with ID 200 the y-displacement of node 50. BNDOUT registers the resultant reaction force due to both of these prescribed motions. The x-direction force in BNDOUT is attributed to the prescribed motion with ID 100 and the y-direction to that with ID 200.

***SENSOR_DEFINE_FUNCTION**_*{OPTION}*

Available options include:

<BLANK>

UPDATE

With the UPDATE keyword option, you can specify the birth time, death time, and value update frequency for the sensor. Note that without the UPDATE keyword option LS-DYNA updates the sensor value every time step.

Purpose: Defines a new sensor with a unique ID. The value associated with this sensor is computed by performing mathematical calculations defined in *DEFINE_FUNCTION, with the information obtained from other sensors defined by the *SENSOR_DEFINE_*OPTION*.

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	FUNC	SENS1	SENS2	SENS3	SENS4	SENS5	SENS6
Type	I	I	I	I	I	I	I	I

Sensor Cards. Additional Cards needed when SENS1 < -5. Include as many cards as needed to specify all |SENS1| cards.

Card 2	1	2	3	4	5	6	7	8
Variable	SENS7	SENS8	SENS9	SENS10	SENS11	SENS12	SENS13	SENS14
Type	I	I	I	I	I	I	I	I

UPDATE Card. Additional card for the UPDATE keyword option to control how often the sensor value will be updated.

Card 3	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
SENSID	Sensor ID
FUNC	Function ID
SENS1	1st sensor ID, the value of which will be used as the 1st argument of function FUNC. If defined as negative, the absolute value of SENS1, |SENS1|, is the number of sensors to be input. If |SENS1| > 5, additional cards will be needed to input the ID of all sensors. The number of sensors is limited to 15.
SENS*i*	*i*th Sensor ID, the value of which will be used as the *i*th argument of function FUNC
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve defining update interval as a function of time.

***SENSOR_DEFINE_MISC**_*{OPTION}*

Available options include:

<BLANK>

UPDATE

The UPDATE keyword option enables specifying the birth time, death time, and value update frequency for the sensor. Note that *without* the UPDATE keyword option LS-DYNA updates the sensor value every time step.

Purpose: Trace the value of a miscellaneous item. This keyword replaces *SENSOR_DEFINE_ANGLE.

Force Sensor Cards. For each miscellaneous sensor, include this card or, if the UPDATE keyword option is used, sets of this card and Card 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	MTYPE	I0	I1	I2	I3	I4	I5
Type	I	A	I/A	I/A	I/A	I/A	I/A	I/A

UPDATE Card. Include this card in a set with Card 1 for each miscellaneous sensor if the UPDATE keyword option is used. This card controls how often the sensor value is updated.

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
SENSID	Sensor ID
MTYPE	Entity to be traced:
	EQ.ANGLE: Angle between two lines, $0° \le \theta \le 180°$. The fields I1 and I2 are node numbers defining the 1st line, while I3 and I4 are node numbers

VARIABLE	DESCRIPTION
	defining the 2nd line.
EQ.BNDOUT:	Boundary condition energy as reported in file bndout. I1 is the ID as defined in *BOUNDARY_PRESCRIBED_MOTION.
EQ.CURVE:	The value of a time-dependent curve defined by *DEFINE_CURVE_FUNCTION or *DEFINE_CURVE. I1 is the curve ID.
EQ.CVBAG:	Information reported in abstat for control volume airbag I1, including I0.EQ.TEMP: Airbag temperature I0.EQ.VOL: Airbag volume
EQ.EMROGO:	Information reported about Rogowsky coil I1 (see *EM_CIRCUIT_ROGO), including I0.EQ.CRNT: Current I0.EQ.CRNTRTE: Current rate I0.EQ.FX: Electromotive force along x I0.EQ.FY: Electromotive force along y I0.EQ.FZ: Electromotive force along z
EQ.ICVOL:	Information reported in icvout for incompressible control volume I1, see *DEFINE_CONTROL_VOLUME, including I0.EQ.PRES: Temperature of control volume I0.EQ.VOL: Volume of control volume
EQ.JNTSTG:	Information reported in jntforc for general stiffness joint I1 (see *CONSTRAINED_JOINT_STIFFNESS_GENERALIZED) including I0.EQ.PHI: ϕ rotation in degrees I0.EQ.PHIR: Rate of rotation of ϕ I0.EQ.THETA: θ rotation in degrees I0.EQ.THETAR: Rate of rotation of θ I0.EQ.IPSI: ψ rotation in degrees I0.EQ.PSIR Rate of rotation of ψ
EQ.MATSUM:	Information reported in matsum for part set I1, including I0.EQ.ADDMASS: Added mass I0.EQ.ERODEIE: Eroded internal energy I0.EQ.ERODEKE: Eroded kinetic energy

VARIABLE	DESCRIPTION
	I0.EQ.HOURGLASS: Hourglass energy I0.EQ.INTERNAL: Internal energy I0.EQ.KINETIC: Kinetic energy
EQ.NFAILE:	Number of failed elements of type I0 in set I1. I0, element type, can be "SOLID" for solid elements, "SHELL" for thin shell elements, "TSHELL" for thick shell elements, "BEAM" for beam elements, or "DISC" for discrete elements. I1 is the related element set number. If undefined, the failure of all elements of type I0 will be traced.
EQ.PULLEY:	Information about beam pulley I1, including: I0.EQ.SLIP: Slippage I0.EQ.WRAP: Wrap angle, θ I0.EQ.FN: Normal contact force I0.EQ.FBM1: Force in the element on side 1 I0.EQ.FBM2: Force in the element on side 2
EQ.RETRACTOR:	Information about seat belt retractor I1, including (see Remark 1): I0.EQ.PULLRATE: Pull-out rate (default) I0.EQ.PULLOUT: Total pull-out I0.EQ.FORCE: Force of the attached belt
EQ.RIGIDBODY:	Kinematics of a rigid body with ID I1. The I2 field specifies the kinematical component to be traced. It may be set to "TX", "TY", or "TZ" for X, Y, and Z translations and to "RX", "RY", or "RZ" for the X, Y, and Z components of the rotation. The I3 field specifies the kinematics type: "D" for displacement, "V" for velocity, and "A" for acceleration. Output is calculated with respect to the global coordinate system when the I4 field is set to "0", its default value; the local rigid-body coordinate system is used when I4 is set to "1".
EQ.SLIPRING:	Information about seat belt slip ring I1, including:

VARIABLE	DESCRIPTION
	I0.EQ.SLIP: Slippage I0.EQ.WRAP: Wrap angle, θ I0.EQ.SKEW: Skew angle, α I0.EQ.FC: Friction coefficient I0.EQ.FN: Normal contact force I0.EQ.FSB1: Force of element on side 1 I0.EQ.FSB2: Force of element on side 2
EQ.TIME:	Current analysis time
I0, …, I5	See MTYPE.
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve defining update interval as a function of time.

Remarks:

1. **Retractor pull-out and pull-in.** A negative value of pull-out represents pull-in in a retractor for retractor sensors (MTYPE = RETRACTOR) and sbtout (see *DATABASE_SBTOUT).

***SENSOR_DEFINE_NODE**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

SET

For *OPTION2* the available options are:

<BLANK>

UPDATE

With the SET keyword option, you can specify a sensor for a set of nodes (see Remark 2 for details). This option can be useful when you want to specify a sensor switching condition based on a group of nodes. With the UPDATE keyword option, you can specify the birth time, death time, and value update frequency for the sensor. Note that without the UPDATE keyword option the sensor value is updated every time step.

Purpose: Define an accelerometer type sensor, electric potential sensor, or temperature sensor.

1. *Accelerometer Sensor*. For an accelerometer sensor, this feature outputs the relative linear acceleration, velocity, or relative coordinate of Node 1 with respect to Node 2 along vector VID.

2. *Electric Potential Sensor*. For an electric potential sensor, this feature outputs the electric potential of a piezoelectric material (see *MAT_ADD_PZELECTRIC).

3. *Temperature Sensor.* For a temperature sensor, this feature outputs the temperature.

Card Summary:

Card Sets. For each nodal sensor matching the specified keyword options include one set of the following data cards. Include as many sets as needed. This input ends at the next keyword ("*") card.

Card 1. This card is required.

SENSID	NODE1	NODE2	VID		CTYPE	SETOPT	

Card 2. Include this card if the UPDATE keyword option is used.

BIRTH	DEATH	DTUPD					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SENSID	NODE1	NODE2	VID		CTYPE	SETOPT	
Type	I	I	I	I		A	A	

VARIABLE	DESCRIPTION
SENSID	Sensor ID
NODE1,2	For an accelerometer sensor, these fields are the nodes defining the accelerometer. If CTYPE = TEMP, then the temperature at NODE1 will be output. If both NODE1 and NODE2 are defined, then the difference in temperature between these two nodes will be output. When the keyword option SET is active, NODE1 is a node set ID. If NODE2 is needed, it must be a node set of the same length as NODE1 with SETOPT defined, but it can be either a node or node set without SETOPT defined. When the SET option is active but SETOPT is not defined, determining the status of a related *SENSOR_SWITCH depends on the sign of NODE1. See Remark 2 for details.
VID	ID of vector along which the nodal values are measured, see *DEFINE_VECTOR. The magnitude of nodal values (coordinate, velocity, or acceleration) will be output if VID is 0 or undefined.
CTYPE	Output component type: EQ.ACC: Acceleration EQ.VEL: Velocity EQ.COORD: Coordinate EQ.EPOT: Electric potential of a piezoelectric material (see *MAT_ADD_PZELECTRIC) EQ.TEMP: Temperature
SETOPT	Option to process set of data when SET option is specified. When SETOPT is specified, a single value will be reported. The single reported value could be:

VARIABLE	DESCRIPTION
EQ.AVG:	The average value of the dataset
EQ.MAX:	The maximum value of the dataset
EQ.MIN:	The minimum value of the dataset
EQ.NS2NSMIND:	The minimum distance between two sets of nodes. CTYPE is ignored.
EQ.SUM:	The sum of the dataset

UPDATE Card. Additional card for the UPDATE keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH	DTUPD					
Type	F	F	F/I					

VARIABLE	DESCRIPTION
BIRTH	Birth time of this sensor
DEATH	Death time of this sensor
DTUPD	Time interval between updates. If negative, -DTUPD is the curve defining update interval as a function of time.

Remarks:

1. **Time evolution of vector VID.** The vector direction is determined by *DEFINE_VECTOR. This vector direction is updated with time only if the coordinate system CID (see *DEFINE_VECTOR) is defined using *DEFINE_COORDINATE_NODES and the parameter FLAG is set to 1. Otherwise, the vector direction is fixed.

2. **SET option.** When SETOPT is not defined for the SET option, a list of nodal data will be reported, one for each node in the node set, NODE1. *SENSOR_DEFINE_FUNCTION and *SENSOR_DEFINE_CALC-MATH can process these reported nodal values, resulting in a list of processed values. Note that all sensor definitions referred to by these two processing commands must have the same number of data points. These nodal values also can determine if the status of a *SENSOR_SWITCH will be changed. Depending on the sign of NODE1, it can take only one single data point (NODE1 < 0) or the whole data set (NODE1 > 0)

to meet the switch condition to change the status of the related *SENSOR_SWITCH. The reported nodal values cannot be accessed by commands like *DEFINE_CURVE_FUNCTION; see SENSORD option.

When SETOPT is defined, the nodal values of all nodes in the node set will be processed, depending on the definition of SETOPT; the resulting value is reported as the single sensor value. The reported value can be processed by both *SENSOR_DEFINE_FUNCTION and *SENSOR_DEFINE_CALC-MATH as well as other regular sensors. It can also be used to determine the status of a *SENSOR_SWITCH. A *DEFINE_CURVE_FUNCTION using the SENSORD option may also access this reported value. If the reference node, NODE2, is needed, NODE2 must be a node set containing the same number of nodes as node set NODE1. The nodes in sets NODE1 and NODE2 must be arranged in the same sequence so that the nodal value of nodes in set NODE1 can be measured with respect to the correct node in set NODE2. For example, the following input traces the average of a relative coordinate.

```
*SENSOR_DEFINE_NODE_SET
$ trace the average of rel coord, i.e.,
$ (c(5)-c(12)+c(10)-c(19)+c(27)-c(38))/3
$   SENSID     NODE1     NODE2       VID       CRD     CTYPE     SETOPT
        40         5        12         1        99     COORD        AVG
*SET_NODE_LIST
5
5,10,27
*SET_NODE_LIST
12
12,19,38
```

3. **Rigid body acceleration measurement frequency.** When NODE1 is a node that belongs to a rigid body and CTTYPE = "ACC," the acceleration recorded by the sensor is approximated, not exact. *SENSOR_DEFINE_MISC with MTYPE = "RIGIDBODY" returns exact acceleration and is recommended for such application.

*SENSOR_SWITCH

Purpose: This command compares the value of a sensor, *SENSOR_DEFINE_... or *SENSOR_DEFINE_CALC-MATH, to a given criterion to check if the switch condition is met. It outputs a boolean value, TRUE or FALSE.

Sensor Switch Cards. Include one additional card for each sensor switch. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SWITID		SENSID	LOGIC	VALUE	FILTRID	TIMWIN	TDELAY
Type	I		I	A	F	I	F	F

VARIABLE	DESCRIPTION
SWITID	Switch ID, which can be referred to directly by *SENSOR_CONTROL to control the status of entities like CONTACT and AIRBAG, or can be referred to by *SENSOR_SWITCH_CALC-LOGIC for a logical computation
SENSID	ID of the sensor whose value will be compared to the criterion to determine if a switch condition is met
LOGIC	Logic operator that can be either LT (<) or GT (>)
VALUE	Critical value
FILTER	Filter ID (optional). See *DEFINE_FILTER for how to define a filter. See Remark 1.
TIMWIN	Trigger a status change when the value given by the sensor is less than or greater than (depending on LOGIC) the VALUE for a duration defined by TIMWIN
TDELAY	Optional time delay. The status change will not happen immediately when both the switch condition and TIMWIN are met. Instead, the status change is delayed by TDELAY.

Remarks:

1. **FILTER.** The filter (*DEFINE_FILTER) receives data from the sensor defined with *SENSOR_DEFINE_... either every time step or at the frequency specified with DTUPD when using the UPDATE keyword option.

*SENSOR_SWITCH_CALC-LOGIC

Purpose: Perform a logical operation on the boolean output of other *SENSOR_SWITCH or *SENSOR_SWITCH_CALC-LOGIC definitions. The output is a boolean value, TRUE or FALSE.

Log Cards. Include one additional card for each logical rule. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SWITID	SWIT1	SWIT2	SWIT3	SWIT4	SWIT5	SWIT6	SWIT7
Type	I	I	I	I	I	I	I	I

Switch Cards. Additional switch cards needed when the logical calculation involves more than seven switches. The first column must be occupied by the plus sign, "+". Include as many cards as needed.

Card 2	1	2	3	4	5	6	7	8
Variable	+	SWIT8	SWIT9	SWIT10	SWIT11	SWIT12	SWIT13	SWIT14
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SWITID	Switch ID that can be referred to directly by *SENSOR_CONTROL to control the status of entities like CONTACT and AIRBAG, or can be referred to by *SENSOR_SWITCH_CALC-LOGIC for a logical computation.
SWIT*n*	Input a positive sensor switch ID for "AND" and a negative sensor switch ID for "OR". SWIT1 must always be positive.

This keyword implements standard Boolean logic:

true = 1,
false = 0,
and = multiplication,
or = addition

An expression evaluating to 0 is false, while any expression that evaluates to greater than 0 is true, and, therefore, set to 1.

Example:

Consider 5 switches defined as follows:

switch(11) = true
switch(12) = false
switch(13) = true
switch(14) = true
switch(15) = false.

To evaluate the expression

[switch(11) or switch(12) or switch(13)] and [switch(14) or switch(15)]

and assign the value to switch(103), the following would apply:

```
*SENSOR_SWITCH_CALC-LOGIC
101,11,-12,-13
102,14,-15
103,101,102
```

This translates into

$$\begin{aligned}
\text{switch}(101) &= \text{switch}(11) \text{ or switch}(12) \text{ or switch}(13) \\
&= \min((1 + 0 + 1), 1) \\
&= 1 \text{ (true)} \\
\text{switch}(102) &= \text{switch}(14) \text{ or switch}(15) \\
&= \min((1 + 0), 1) \\
&= 1 \text{ (true)} \\
\text{switch}(103) &= \text{switch}(101) \text{ and switch}(102) \\
&= \min((1 \times 1), 1) \\
&= 1 \text{ (true)}
\end{aligned}$$

Therefore,

$$\overbrace{[\text{switch}(11) \text{ or switch}(12) \text{ or switch}(13)]}^{\text{switch}(101)=\text{true}} \text{ AND } \overbrace{[\text{switch}(14) \text{ or switch}(15)]}^{\text{switch}(102)=\text{true}}$$
$$= \text{switch}(103)$$
$$= \text{true}$$

***SENSOR_SWITCH_SHELL_TO_VENT**_*{OPTION}*

Available options include:

ID

TITLE

OPTION provides a means to specify an airbag ID number and a heading for the airbag.

Purpose: This option will treat the failed shell elements as vent holes for the airbag defined by *AIRBAG_PARTICLE. The mass escaped from the vent will be reported in the abstat_cpm file.

ID Card. Additional card for the ID or TITLE keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	ABID	HEADING						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	ID	ITYPE	C23	AMAX				
Type	I	I	F	F				

Shell Fail Time Cards. Optional Cards for setting time at which shells in a shell list (see *SHELL_LIST) change into vents. This card may be repeated up time 15 times. This input ends at the next keyword ("*") cards.

Optional	1	2	3	4	5	6	7	8
Variable	SSID	FTIME	C23V					
Type	I	F	F					
Default	none	0.	C23					

VARIABLE	DESCRIPTION
ABID	Airbag ID
HEADING	Airbag descriptor
ID	Part set ID/part ID
TYPE	Type for ID: EQ.0: part EQ.1: part set
C23	Vent Coefficient (Default = 0.7) LT.0: User defined load curve ID. The vent coefficient is a function of pressure.
AMAX	Maximum allowable area for failed vent surface area (VA). If the area is bigger than AMAX, C23 will be scaled down by a factor of AMAX/VA. Otherwise, C23 will be used.
SSID	ID of *SET_SHELL_LIST
FTIME	Time to convert shell list to vent
C23V	Vent Coefficient. LT.0: User defined load curve ID. The vent coefficient is a function of pressure.

*SET

The keyword family *SET provides a convenient way of defining groups of nodes, parts, elements, and segments. The sets can be used in the definitions of contact interfaces, loading conditions, boundary conditions, and other inputs. This keyword family also provides a convenient way of defining groups of vibration modes to be used in frequency domain analysis. Each set type must have a unique numeric identification. The keyword control cards in this section are defined as follows:

SET_BEAM_*{OPTION1}_{OPTION2}

*SET_BEAM_ADD

*SET_BEAM_INTERSECT

*SET_BOX

SET_DISCRETE_*{OPTION1}_{OPTION2}

*SET_DISCRETE_ADD

SET_MODE_*{OPTION}

*SET_MULTI

*SET_MULTIMATERIAL_GROUP_LIST

SET_NODE_*{OPTION1}_{OPTION2}

SET_NODE_ADD_*{OPTION}

*SET_NODE_INTERSECT

SET_PART_*{OPTION1}_{OPTION2}

*SET_PART_ADD

*SET_PART_TREE

*SET_PERI_LAMINATE

SET_SEGMENT_*{OPTION1}_{OPTION2}

*SET_SEGMENT_ADD

*SET_SEGMENT_INTERSECT

*SET_SEGMENT_N8

SET_SHELL_*{OPTION1}_{OPTION2}

*SET_SHELL_ADD

*SET_SHELL_INTERSECT

SET_SOLID_*{OPTION1}_{OPTION2}

*SET_SOLID_ADD

*SET_SOLID_INTERSECT

SET_TSHELL_*{OPTION1}_{OPTION2}

SET_2D_SEGMENT_*{OPTION1}_{OPTION2}

An additional keyword option**TITLE** may be appended to all the ***SET** keywords. If this option is used, then an addition line is read for each section in 80a format which can be used to describe the set. At present, the title serves no purpose other than to perhaps lend clarity to input decks.

The **GENERAL** keyword option is available for set definitions. In this option, the commands are executed in the order defined. For example, the delete option cannot delete a node or element unless the node or element was previously added using a command such as BOX or ALL.

The **COLLECT** keyword option allows for the definition of multiple sets that share the same ID and combines them into one large set whenever this option is found. If two or more sets of the same type have definitions that share the same set IDs, they are combined if and only if the **COLLECT** option is specified in each definition. If the **COLLECT** option is not specified for one or more set definitions of the same type that share identical set IDs, an error termination will occur. For include files using ***INCLUDE_TRANSFORM** where set offsets are specified, the offsets are not applied for the case where the **COLLECT** option is present.

***SET_BEAM**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

GENERATE

GENERATE_INCREMENT

GENERAL

For *OPTION2* the available option is:

COLLECT

The GENERATE and GENERATE_INCREMENT options will generate block(s) of beam element ID's between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the set.

Purpose: Define a set of beam elements or a set of seat belt elements (see BSID of *DATABASE_CROSS_SECTION_SET).

Card Summary:

Card 1. This card is required.

SID							

Card 2a. Include this card if the keyword option is unset (<BLANK>). Include as many cards as needed. This input ends at the next keyword ("*") card.

K1	K2	K3	K4	K5	K6	K7	K8

Card 2b. Include this card if the GENERATE option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the GENERATE_INCREMENT option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Card 2d. Include this card if the GENERAL option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SID	Set ID

Beam Element ID Cards. This Card 2 format applies to the case of an unset (<BLANK>) keyword option. Set one value per element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	K1	K2	K3	K4	K5	K6	K7	K8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
K*i*	*i*th beam element (see Remark 1)

Beam Element Range Cards. This Card 2 format applies to the GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of elements. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First beam element ID in block N.

VARIABLE	DESCRIPTION
B[N]END	Last beam element ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the element numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not element IDs.

Beam Element Range with Increment Cards. This Card 2 format applies to the GENERATE_INCREMENT keyword option. For each block of elements add one card to the deck. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First beam element ID in block.
BEND	Last beam element ID in block.
INCR	Beam ID increment. Beam IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Generalized Beam Element Range Cards. This Card 2 format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2d	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below.
E1, …, E7	Specified entity. Each card must have the option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accept up to 7 arguments, but they may take fewer. Values of E*n* left unspecified are ignored.

Note that the order of the selected operations matters. Elements are only excluded from the set if they were previously added to the set. For instance, if you exclude beam element 5 and then include it, element 5 will be included in the set. For the example set below, suppose that part 6 includes beam elements 10, 15, 20, and 32, box 7 contains beam elements 5, 20, and 32, and part 10 includes beam elements 5, 22, and 106. Then, beam element set 1 contains beam elements 5, 10, 15, 22, and 106.

```
*SET_BEAM_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All beam elements will be included in the set.
BOX	Elements inside boxes E1, E2, E3, ... will be included. See *DEFINE_BOX. See Remark 1.
DBOX	Previously added elements that are inside boxes E1, E2, E3, ... will be excluded. See Remark 1.
ELEM	Elements E1, E2, E3, ... will be included. See Remark 1.
DELEM	Elements E1, E2, E3, ... if previously added will be excluded. See Remark 1.
PART	Elements of parts E1, E2, E3, ... will be included. See Remark 1.
DPART	Elements that have been previously added and are of parts E1, E2, E3, ... will be excluded. See Remark 1.
SET	Elements of beam element sets E1, E2, E3, … will be included. See Remark 1.
DSET	Previously added elements that are members of beam element sets E1, E2, E3, … will be excluded. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_BEAM
         1
        10        20        35        60        85
*SET_BEAM_GENERAL
         2
      PART         2       102
```

   They should *not* be input like the following:

```
*SET_BEAM
         1
        10        20         0        35        60        85
*SET_BEAM_GENERAL
         2
      PART         2                 102
```

*SET_BEAM_ADD

Purpose: Define a beam set by combining beam sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Beam Element Set Cards. Each card can be used to specify up to 8 beam element sets. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	BSID1	BSID2	BSID3	BSID4	BSID5	BSID6	BSID7	BSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new beam set. All beam sets should have a unique set ID.
BSID[N]	The N^{th} beam set ID on the card. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of beam sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_BEAM_ADD
         1
        10        20        35        60        85
*SET_BEAM_ADD
         3
         2       102
```

They should *not* be input like the following:

```
*SET_BEAM_ADD
         1
        10        20         0        35        60        85
*SET_BEAM_ADD
         3
         2                 102
```

*SET_BEAM_INTERSECT

Purpose: Define a beam set as the intersection, ∩, of a series of beam sets. The new beam set, SID, contains only the elements common to of all beam sets listed on the cards of format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Beam Set Cards. Each card can be used to specify up to 8 beam sets. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	BSID1	BSID2	BSID3	BSID4	BSID5	BSID6	BSID7	BSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new beam set. All beam sets should have a unique set ID.
BSID[N]	The N^{th} beam set ID on Card 2. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of beam sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_BEAM_INTERSECT
         1
        10        20        35        60        85
*SET_BEAM_INTERSECT
         3
         2       102
```

They should *not* be input like the following:

```
*SET_BEAM_INTERSECT
         1
        10        20         0        35        60        85
*SET_BEAM_INTERSECT
         3
         2               102
```

*SET_BOX

Purpose: Define a set of boxes. See *DEFINE_BOX.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Box Set Cards. Each card can be used to specify up to 8 box IDs. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	BID1	BID2	BID3	BID4	BID5	BID6	BID7	BID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new box set. All box sets should have a unique set ID.
BID[N]	The N^{th} box ID. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of box IDs in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_BOX
         1
        10        20        35        60        85
*SET_BOX
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_BOX
         1
        10        20         0        35        60        85
*SET_BOX
         3
         2               102
```

***SET_DISCRETE**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

GENERATE

GENERAL

For *OPTION2* the available option is:

COLLECT

The GENERATE option will generate a block of discrete element IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the set.

Purpose: Define a set of discrete elements.

Card Summary:

Card 1. This card is required.

SID							

Card 2a. Include this card if the keyword option is unset (<BLANK>). Include as many cards as needed. This input ends at the next keyword ("*") card.

K1	K2	K3	K4	K5	K6	K7	K8

Card 2b. Include this card if the GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the GENERAL keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SID	Set ID

Discrete Element ID Cards. This card applies to the case of an unset (<BLANK>) keyword option. Set one value per element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	K1	K2	K3	K4	K5	K6	K7	K8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
K*i*	i^{th} discrete element (see Remark 1)

Discrete Element Range Cards. This card applies to the GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of elements. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First discrete element ID in block N.

VARIABLE	DESCRIPTION
B[N]END	Last discrete element ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the element numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not element IDs.

Generalized Discrete Element Range Cards. This card applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below.
E1, …, E7	Specified entity. Each card must have the option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" field in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accept up to 7 arguments, but they may take fewer. Values of E*n* left unspecified are ignored.

Note that the order of the selected operations matters. Elements are only excluded from the set if they were previously added. For instance, if you exclude discrete element 5 and then include it, discrete element 5 will be included in the set. For the example set below, suppose that part 6 includes discrete elements 10, 15, 20, and 32, box 7 contains discrete elements 5, 20, and 32, and part 10 includes discrete elements 5, 22, and 106. Then, discrete element set 1 includes discrete elements 5, 10, 15, 22, and 106.

```
*SET_DISCRETE_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All discrete elements will be included in the set.
BOX	Elements inside boxes E1, E2, E3, ... will be included. See *DEFINE_BOX. See Remark 1.
DBOX	Previously added elements that are within boxes E1, E2, E3, ... will be excluded. See Remark 1.
ELEM	Elements E1, E2, E3, ... will be included. See Remark 1.
DELEM	Elements E1, E2, E3, ... if previously added will be excluded. See Remark 1.
PART	Elements of parts E1, E2, E3, ... will be included. See Remark 1.
DPART	Elements that have been previously added and are of parts E1, E2, E3, ... will be excluded. See Remark 1.
SET	Elements of discrete element sets E1, E2, E3, … will be included. See Remark 1.
DSET	Previously added elements that are members of discrete element sets E1, E2, E3, … will be excluded. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_DISCRETE
         1
        10        20        35        60        85
*SET_DISCRETE_GENERAL
         2
      PART         2       102
```

   They should *not* be input like the following:

```
*SET_DISCRETE
         1
        10        20         0        35        60        85
*SET_DISCRETE_GENERAL
         2
      PART         2                 102
```

*SET_DISCRETE_ADD

Purpose: Define a discrete set by combining discrete sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Discrete Element Set Cards. Each card can be used to specify up to 8 discrete element sets. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	DSID1	DSID2	DSID3	DSID4	DSID5	DSID6	DSID7	DSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new discrete element set. All discrete element sets must have a unique ID.
DSID[N]	The N^{th} discrete set ID on Card 2

Remarks:

1. **Tightly packed data.** Lists of discrete element sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_DISCRETE_ADD
         1
        10        20        35        60        85
*SET_DISCRETE_ADD
         3
```

```
         2       102
```

They should *not* be input like the following:

```
*SET_DISCRETE_ADD
         1
        10        20         0        35        60        85
*SET_DISCRETE_ADD
         3
         2               102
```

***SET_IGA_EDGE**_*OPTION1_{OPTION2}_{OPTION3}*

Purpose: Define a set of parametric or physical edges (see *IGA_EDGE_UVW and *IGA_-EDGE_XYZ) with optional attributes.

For *OPTION1* the available options are:

UVW

XYZ

For *OPTION2* the available options are:

<BLANK>

LIST

LIST_GENERATE

LIST_GENERATE_INCREMENT

For *OPTION3* the available option is:

COLLECT

The LIST_GENERATE and LIST_GENERATE_INCREMENT options will generate block(s) of parametric/physical edge IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the parametric/physical edge set.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER		

Card 2a. Include this card if *OPTION2* is left unset or is set to LIST. Include as many cards as needed. This input ends at the next keyword ("*") card.

EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8

Card 2b. Include this card if the GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.	0.	0.	0.	MECH		

VARIABLE	DESCRIPTION
SID	Set ID. A unique number must be chosen.
DA1	First attribute default value; see Remark 1.
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 2.

Parametric/Physical Edge ID Cards. This Card 2 format applies when *OPTION2* is LIST or unset (<BLANK>). Set one value per parametric/physical edge in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
EID*i*	i^{th} parametric/physical edge ID (see Remark 3)

Parametric/Physical Edge ID Range Cards. This Card 2 format applies to the GENERATE keyword option. Set one pair of B[N]BEG and B[N]END values per block of parametric/physical edge IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First parametric/physical edge ID in block N
B[N]END	Last parametric/physical edge ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the parametric/physical edge numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not parametric/physical edge IDs.

Parametric/Physical Edge ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of parametric/physical edges add one card to the deck. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First parametric/physical edge ID in block.
BEND	Last parametric/physical edge ID in block.

VARIABLE	DESCRIPTION
INCR	Parametric/physical edge ID increment. Parametric/physical edge IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Remarks:

1. **Parametric/Physical Edge Attributes.** Parametric/physical edge attributes can be assigned for some input types.

2. **Solvers.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

3. **Tightly packed data.** Lists of edges in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_IGA_EDGE_UVW
         1
        10        20        35        60        85
*SET_IGA_EDGE_UVW
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_IGA_EDGE_UVW
         1
        10        20         0        35        60        85
*SET_IGA_EDGE_UVW
         3
         2                 102
```

***SET_IGA_FACE**_*OPTION1_{OPTION2}_{OPTION3}*

Purpose: Define a set of parametric or physical faces (see *IGA_FACE_UVW and *IGA_-FACE_XYZ) with optional attributes.

For *OPTION1* the available options are:

UVW

XYZ

For *OPTION2* the available options are:

<BLANK>

LIST

LIST_GENERATE

LIST_GENERATE_INCREMENT

For *OPTION3* the available option is:

COLLECT

The LIST_GENERATE and LIST_GENERATE_INCREMENT options will generate block(s) of parametric/physical face IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the parametric/physical face set.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER		

Card 2a. Include this card if *OPTION2* is left unset or is set to LIST. Include as many cards as needed. This input ends at the next keyword ("*") card.

FID1	FID2	FID3	FID4	FID5	FID6	FID7	FID8

Card 2b. Include this card if the GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.	0.	0.	0.	MECH		

VARIABLE	DESCRIPTION
SID	Set ID. A unique number must be chosen.
DA1	First attribute default value; see Remark 1.
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 2.

Parametric/Physical Face ID Cards. This Card 2 format applies to *OPTION2* set to LIST or left unset (<BLANK>). Set one value per parametric/physical face in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	FID1	FID2	FID3	FID4	FID5	FID6	FID7	FID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
FID*i*	i^{th} parametric/physical face ID (see Remark 3)

Parametric/Physical Face ID Range Cards. This Card 2 format applies to the GENERATE keyword option. Set one pair of B[N]BEG and B[N]END values per block of parametric/physical face IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First parametric/physical face ID in block N.
B[N]END	Last parametric/physical face ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the parametric/physical face numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not parametric/physical face IDs.

Parametric/Physical Face ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of parametric/physical faces add one card to the deck. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First parametric/physical face ID in block
BEND	Last parametric/physical face ID in block

VARIABLE	DESCRIPTION
INCR	Parametric/physical face ID increment. Parametric/physical face IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Remarks:

1. **Parametric/physical face attributes.** Parametric/physical face attributes can be assigned for some input types.

2. **Solvers.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

3. **Tightly packed data.** Lists of faces in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_IGA_FACE_UVW
         1
        10        20        35        60        85
*SET_IGA_FACE_UVW
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_IGA_FACE_UVW
         1
        10        20         0        35        60        85
*SET_IGA_FACE_UVW
         3
         2                 102
```

***SET_IGA_POINT_UVW**_*{OPTION1}_{OPTION2}*

Purpose: Define a set of parametric points (see *IGA_POINT_UVW) with optional attributes.

For *OPTION1* the available options are:

<BLANK>

LIST

LIST_GENERATE

LIST_GENERATE_INCREMENT

For *OPTION2* the available option is:

COLLECT

The LIST_GENERATE and LIST_GENERATE_INCREMENT options will generate block(s) of parametric point IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the parametric point set.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER		

Card 2a. Include this card if *OPTION1* is unset or is set to LIST. Include as many cards as needed. This input ends at the next keyword ("*") card.

PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8

Card 2b. Include this card if the GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.	0.	0.	0.	MECH		

VARIABLE	DESCRIPTION
SID	Set ID. A unique number must be chosen.
DA1	First attribute default value; see Remark 1.
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 2.

Parametric Point ID Cards. This Card 2 format applies to *OPTION1* set to list or left unset (<BLANK>). Set one value per parametric point in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
PID*i*	i^{th} parametric point ID

Parametric Point ID Range Cards. This Card 2 format applies to the GENERATE keyword option. Set one pair of B[N]BEG and B[N]END values per block of parametric point IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First parametric point ID in block N
B[N]END	Last parametric point ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the parametric point numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not parametric point IDs.

Parametric Point ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of parametric points add one card to the deck. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First parametric point ID in block
BEND	Last parametric point ID in block
INCR	Parametric point ID increment. Parametric point IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Remarks:

1. **Parametric point attributes.** Parametric point attributes can be assigned for some input types.

2. **Solvers.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

3. **Tightly packed data.** Lists of parametric points in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_IGA_POINT_UVW
         1
        10        20        35        60        85
*SET_IGA_POINT_UVW
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_IGA_POINT_UVW
         1
        10        20         0        35        60        85
*SET_IGA_POINT_UVW
         3
         2                 102
```

***SET_MODE**_*{OPTION1}_{OPTION2}*

For *OPTION1, the* available options are:

<BLANK>

LIST

LIST_GENERATE

For *OPTION2*, the available option is

COLLECT

The option LIST_GENERATE will generate a block of mode IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the set.

Purpose: Define a set of modes.

Card Summary:

Card 1. This card is required.

SID							

Card 2a. Include this card if the keyword option is set to LIST or left unset. Include as many cards as needed. This input ends at the next keyword ("*") card.

MID1	MID2	MID3	MID4	MID5	MID6	MID7	MID8

Card 2b. Include this card if the LIST_GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

M1BEG	M1END	M2BEG	M2END	M3BEG	M3END	M4BEG	M4END

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Mode ID Cards. Include this card when the keyword option is set to LIST *or* left unset (<BLANK>). Set one value per mode in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	MID1	MID2	MID3	MID4	MID5	MID6	MID7	MID8
Type	I	I	I	I	I	I	I	I

Mode Range Cards. This card applies to the LIST_GENERATE keyword option. Set one pair of MNBEG and MNEND values per block of modes. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	M1BEG	M1END	M2BEG	M2END	M3BEG	M3END	M4BEG	M4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set identification. All mode sets should have a unique set ID.
MID[N]	Mode ID N (see Remark 2)
M[N]BEG	First mode ID in block N
M[N]END	Last mode ID in block N. All defined IDs between and including M[N]BEG and M[N]END are added to the set.

Remarks:

1. **Files containing available mode IDs.** The available mode IDs can be found in ASCII file eigout, or binary database d3eigv.

2. **Tightly packed data.** Lists of modes in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_MODE
         1
        10        20        35        60        85
*SET_MODE
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_MODE
         1
        10        20         0        35        60        85
*SET_MODE
         3
         2               102
```

*SET_MULTI

Note that this keyword's name has been shortened. Its older long form, however, is still also valid.

*SET_MULTI-MATERIAL_GROUP_LIST

Purpose: This command defines an ALE multi-material set ID (AMMSID) which contains a collection of one or more ALE multi-material group ID(s) (AMMGID). This provides a means for selecting any specific ALE multi-material(s). Applications include, for example, a selection of any particular fluid(s) to be coupled to a fluid-structure interaction.

Card 1	1	2	3	4	5	6	7	8
Variable	AMMSID							
Type	I							
Default	0							

Multi-Material Group ID Cards. Set one value per element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	AMMGID1	AMMGID2	AMMGID3	AMMGID4	AMMGID5	AMMGID6	AMMGID7	AMMGID8
Type	I/A	I/A	I/A	I/A	I/A	I/A	I/A	I/A
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
AMMSID	An ALE multi-material set ID (AMMSID), which contains a collection of one or more ALE multi-material group ID(s) (AMMGID).
AMMGID*i*	Desired ALE multi-material referenced with its AMMGID for general ALE or with either its AMMGID or AMMG name (AMMGNM) for S-ALE. See Remarks 1 and 2.

Remarks:

1. **AMMGID and AMMG names.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, each AMMG can only be referred to by their AMMGID. The AMMGID for each AMMG is based on the order of appearance of the AMMG in the input deck.

   For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_-MULTI-MATERIAL_GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the field AMMGNM. Each AMMG defined with that keyword can then be referred with either its name or its AMMGID (which is again based on order of appearance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, then the AMMGIDs may change. Then, you must find all those references and change them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

2. **Tightly packed data.** Lists of AMMGs in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_MULTI-MATERIAL_GROUP_LIST
         1
        10        20        35        60        85
*SET_MULTI-MATERIAL_GROUP_LIST
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_MULTI-MATERIAL_GROUP_LIST
         1
        10        20         0        35        60        85
*SET_MULTI-MATERIAL_GROUP_LIST
         3
         2                 102
```

***SET_NODE**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

LIST

COLUMN

LIST_GENERATE

LIST_GENERATE_INCREMENT

GENERAL

LIST_SMOOTH

For *OPTION2* the available option is:

COLLECT

The LIST option generates a set for a list of node IDs. The LIST_GENERATE and LIST_-GENERATE_INCREMENT options will generate block(s) of node IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the node set. The option LIST_SMOOTH is used to define a local region on a distorted tooling mesh to be smoothed. The LIST_SMOOTH option is documented in Remark 10 of the manual page for *INTERFACE_COMPENSATION_3D. The COLUMN option is for setting nodal attributes, which pass data to other keyword cards, on a node-by-node basis.

Purpose: Define a nodal set with some identical or unique attributes.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER	ITS	

Card 2a. Include this card if the keyword option is unset (<BLANK>), LIST, or LIST_-SMOOTH. Include as many cards as needed. This input ends at the next keyword ("*") card.

NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8

Card 2b. Include this card if the keyword option is COLUMN. Include one card per node in the set. This input ends at the next keyword ("*") card

NID	A1	A2	A3	A4			

Card 2c. Include this card if the LIST_GENERATE keyword option is used. Set one pair of BNBEG and BNEND values per block of nodes. This input ends at the next keyword ("*") card

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2d. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. For each block of nodes, add one card to the deck. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Card 2e. Include this card if the GENERAL keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER	ITS	
Type	I	F	F	F	F	A	I	
Default	none	0.	0.	0.	0.	MECH	0	
Remark		1	1	1	1	3		

VARIABLE	DESCRIPTION
SID	Set identification. All node sets should have a unique set ID.
DA1	First nodal attribute default value
DA2	Second nodal attribute default value
DA3	Third nodal attribute default value
DA4	Fourth nodal attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.)

VARIABLE	DESCRIPTION
ITS	Specify coupling type across different scales in two-scale co-simulation. This flag should only be included for node sets that provide coupling information in the input file referred to by *INCLUDE_COSIM. EQ.1: Tied contact coupling EQ.2: Solid-in-shell immersed coupling EQ.3: Solid-in-solid immersed coupling

Node ID Cards. This Card 2 format applies to LIST and LIST_SMOOTH keyword options. Additionally, it applies to the case of an unset (<BLANK>) keyword option. Set one value per node in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
NID*i*	Node ID *I* (see Remark 5)

Node ID with Column Cards. This Card 2 format applies to the COLUMN keyword option. Include one card per node in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	NID	A1	A2	A3	A4			
Type	I	F	F	F	F			

VARIABLE	DESCRIPTION
NID	Nodal ID
A1	First nodal attribute (see Remark 2)
A2	Second nodal attribute (see Remark 2)

VARIABLE	DESCRIPTION
A3	Third nodal attribute (see Remark 2)
A4	Fourth nodal attribute (see Remark 2)

Node ID Range Cards. This Card 2 format applies to the LIST_GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of nodes. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B*n*BEG	First node ID in block *n*.
B*n*END	Last node ID in block *n*. All defined IDs between and including B*n*BEG to B*n*END are added to the set. These sets are generated after all input is read so that gaps in the node numbering are not a problem. B*n*BEG and B*n*END may simply be limits on the IDs and not nodal IDs.

Node ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of nodes add one card to the deck. This input ends at the next keyword ("*") card.

Card 2d	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First node ID in block.
BEND	Last node ID in block.
INCR	Node ID increment. Node IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Generalized Node ID Range Cards. This Card 2 format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2e	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below.
E1, …, E7	Specified entity. Each card must have the option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2e for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2e. Each of the following operations accepts up to 7 arguments, but they may take fewer. Values of "E*n*" left unspecified are ignored.

Note that the order of the selected operations matters. Nodes are only excluded from the set if they were previously added to the set. For instance, if you exclude node 5 and then include it, node 5 will be included in the set. For the example set below, suppose that part 6 includes nodes 10, 15, 20, and 32, box 7 contains nodes 5, 20, and 32, and part 10 includes nodes 5, 22, and 106. Then, node set 1 contains nodes 5, 10, 15, 22, and 106.

```
*SET_NODE_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All nodes will be included in the set.
BRANCH	Nodes inside tree branches E1, E2, E3, … will be included. See *SET_PART_TREE. See Remark 5.
DBRANCH	Previously added nodes that are inside tree branches E1, E2, E3, … will be excluded. See Remark 5.

OPTION	DESCRIPTION
BOX	Nodes inside boxes E1, E2, E3, … will be included. See *DEFINE_BOX. See Remark 5.
DBOX	Previously added nodes that are inside boxes E1, E2, E3, … will be excluded. See Remark 5.
NODE	Nodes E1, E2, E3, … will be included. See Remark 5.
DNODE	Nodes E1, E2, E3, … if previously added will be excluded. See Remark 5.
PART	Nodes of parts E1, E2, E3, … will be included. See Remark 5.
DPART	Nodes that have been previously added and are of parts E1, E2, E3, … will be excluded. See Remark 5.
SALEBOXL	Nodes inside a box in a structured ALE (S-ALE) mesh. E1 here is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX. They are the minimum and the maximum nodal local coordinates, as defined in *ALE_STRUCTURED_MESH_CONTROL_POINTS, along each direction in an S-ALE mesh. This option can only be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALECPT	Nodes inside a box in a structured ALE (S-ALE) mesh. E1 here is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, E7 correspond to IMIN, IMAX, JMIN, JMAX, KMIN, KMAX. They are the minimum and the maximum nodal indices along each direction in S-ALE mesh. To include all nodes in the S-ALE mesh defined by E1, set values E2–E7 to zero or leave them blank. This option can only be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALEFAC	Nodes that are on the face of an S-ALE mesh. E1 gives the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, E7 correspond to $-x$, $+x$, $-y$, $+y$, $-z$, $+z$ faces. Assigning 1, for instance, to these six values would

OPTION	DESCRIPTION
	include nodes belonging to all the surface segments at these faces in the node set. This option is can only be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. For trimmed S-ALE meshes (see *ALE_STRUCTURED_MESH_TRIM), a segment is treated as a surface segment as long as it has no neighboring element along the specified direction. The set, thus, includes the nodes of surface segments belonging to exterior and interior boundaries. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SET_NODE	Nodes from node sets with IDs E1, …, E7 will be included. See Remark 5.
DSET_NODE	Nodes that have been previously added and are from node sets with IDs E1, …, E7 will be excluded. See Remark 5.
SET_*XXXX*	Include nodal points of element sets defined by *SET_*XXXX*, where *XXXX* could be SHELL, SOLID, BEAM, TSHELL, or DISCRETE. See Remark 5.
SET_PART	Include nodal points in part sets E1, …, E7. See Remark 5.
VOL	Nodes inside contact volumes E1, E2, E3, … will be included. See *DEFINE_CONTACT_VOLUME. See Remark 5.
DVOL	Previously added nodes that are inside contact volumes E1, E2, E3, … will be excluded. See Remark 5.

Remarks:

1. **Nodal attributes.** Nodal attributes can be assigned to pass data to other keywords. For example, for contact option, *CONTACT_TIEBREAK_NODES_TO_SURFACE the attributes are:

   DA1 = NFLF ⇒ Normal failure force,

   DA2 = NSFL ⇒ Shear failure force,

   DA3 = NNEN ⇒ Exponent for normal force,

   DA4 = NMES ⇒ Exponent for shear force.

2. **Overriding nodal attributes.** The nodal attributes set in Card 1 can be overridden on a node-by-node basis by using the COLUMN keyword option. Card 2b for this keyword option allows the user to set the attributes for each node. If left unset on Card 2b, the values default to those set in Card 1, that is, A1 = DA1, etc.

3. **Solvers.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics solver's mesh.

4. **LIST_SMOOTH option.** The option *SET_NODE_LIST_SMOOTH is used for localized tooling surface smoothing, and it is used in conjunction with keywords *INTERFACE_COMPENSATION_NEW_LOCAL_SMOOTH, *INCLUDE_COMPENSATION_ORIGINAL_RIGID_TOOL, and *INCLUDE_COMPENSATION_NEW_RIGID_TOOL.

5. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_NODE
         1
        10        20        35        60        85
*SET_NODE_GENERAL
         2
      PART         2       102
```

   They should *not* be input like the following:

```
*SET_NODE
         1
        10        20         0        35        60        85
*SET_NODE_GENERAL
         2
      PART         2                 102
```

***SET_NODE_ADD**_{OPTION}_

Available options include:

<BLANK>

ADVANCED

Purpose: Define a node set by combining node sets or, for the ADVANCED option, by combining NODE, SHELL, SOLID, BEAM, SEGMENT, DISCRETE, and TSHELL sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	none	none	none	none	MECH		

Node Set Cards. Include this card when the keyword option is left unset (<BLANK>). Each card can be used to specify up to 8 node set IDs. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	NSID1	NSID2	NSID3	NSID4	NSID5	NSID6	NSID7	NSID8
Type	I	I	I	I	I	I	I	I

Node Set Advanced Cards. Include this card when the keyword option is set to ADVANCED. Each card can be used to specify up to 4 set IDs (node sets, beam sets, etc.). Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	SID1	TYPE1	SID2	TYPE2	SID3	TYPE3	SID4	TYPE4
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of the new node set. All node sets should have a unique set ID.
DA1	First nodal attribute default value; see Remark 1 below.
DA2	Second nodal attribute default value
DA3	Third nodal attribute default value
DA4	Fourth nodal attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 1.
NSID[N]	The Nth node set ID on Card 2a. See Remark 2.
SID[N]	The Nth set ID on Card 2b. See Remark 2.
TYPE[N]	Type of set for SID[N] (see Remark 2): EQ.1: Node set EQ.2: Shell set EQ.3: Beam set EQ.4: Solid set EQ.5: Segment set EQ.6: Discrete set EQ.7: Thick shell set

Remarks:

1. **Solvers and mesh.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of node sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_NODE_ADD
         1
        10        20        35        60        85
*SET_NODE_ADD
         3
         2       102
```

They should *not* be input like the following:

```
*SET_NODE_ADD
         1
        10        20         0        35        60        85
*SET_NODE_ADD
         3
         2               102
```

*SET_NODE_INTERSECT

Purpose: Define a node set as the intersection, ∩, of a series of node sets. The new node set, NSID, contains all common elements of all node sets listed on all cards in format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.	0.	0.	0.	MECH		
Remarks						1		

Node Set Cards. For each SID in the intersection specify one field. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	NSID1	NSID2	NSID3	NSID4	NSID5	NSID6	NSID7	NSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new node set. All node sets should have a unique set ID.
DA*i*	i^{th} nodal attribute
SOLVER	Name of solver using this set (MECH, CESE, etc.)
NSID[N]	N^{th} node set ID. See Remark 2.

Remarks:

1. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of node sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or

empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_NODE_INTERSECT
         1
        10        20        35        60        85
*SET_NODE_INTERSECT
         3
         2       102
```

They should *not* be input like the following:

```
*SET_NODE_INTERSECT
         1
        10        20         0        35        60        85
*SET_NODE_INTERSECT
         3
         2                 102
```

***SET_PART**_*{OPTION1}_{OPTION2}*

For *OPTION1* available options are:

<BLANK>

COLUMN

GENERAL

LIST

LIST_GENERATE

LIST_GENERATE_INCREMENT

For *OPTION2* the available option is:

COLLECT

The LIST_GENERATE and LIST_GENERATE_INCREMENT options will generate block(s) of part IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the part set.

Purpose: Define a set of parts with optional attributes.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER		

Card 2a. Include this card if the keyword option is left unset or is LIST. Include as many cards as needed. This input ends at the next keyword ("*") card.

PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8

Card 2b. Include this card if the COLUMN keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

PID	A1	A2	A3	A4			

Card 2c. Include this card if the LIST_GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2d. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Card 2e. Include this card if *OPTION1* is GENERAL. Include as many cards as necessary. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.	0.	0.	0.	MECH		

VARIABLE	DESCRIPTION
SID	Set ID. All part sets should have a unique set ID.
DA1	First attribute default value; see Remark 1.
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 3.

Part ID Cards. This Card 2 format applies to the LIST keyword option and the unset (<BLANK>) keyword option. Set one value per part in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
PID*i*	*i*th part ID (see Remark 4)

Part ID with Column Cards. This Card 2 format applies to the COLUMN keyword option. Include one card per part in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	PID	A1	A2	A3	A4			
Type	I	F	F	F	F			
Remark		2	2	2	2			

VARIABLE	DESCRIPTION
PID	Part ID
A1	First part attribute
A2	Second part attribute
A3	Third part attribute
A4	Fourth part attribute

Part ID Range Cards. This Card 2 format applies to the LIST_GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of part IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First part ID in block N.
B[N]END	Last part ID in block N. All defined ID's between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the part numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the ID's and not part ID's.

Part ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of parts add one card to the deck. This input ends at the next keyword ("*") card.

Card 2d	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First part ID in block.
BEND	Last part ID in block.
INCR	Part ID increment. Part IDs BBEG, BBEG+INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Generalized Part ID Range Cards. This Card 2 format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2e	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below
E1, …, E7	Specified entity. Each card must have an option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accept up to 7 arguments, but they may take fewer. Values of "E*n*" left unspecified are ignored.

Note that the order of the selected operations matters. Parts are only excluded from the set if they were previously added to the set. Consider the following example:

```
*SET_PART
1
1,2
*SET_PART
2
2,3
*SET_PART_GENERAL
1001
SET,1
DSET,2
*SET_PART_GENERAL
1002
DSET,2
SET,1
```

Because of the order dependence, part sets 1001 and 1002 are not the same. Part set 1001 includes part 1 while part set 1002 includes parts 1 and 2.

OPTION	DESCRIPTION
ALL	All parts will be included in the set.
MATTYPE	E1, …, E7 refer to material types, such as 1 for *MAT_ELASTIC and 2 for *MAT_ORTHOTROPIC_ELASTIC. For each provided material type, all parts with the specified material type are added to the part set. See Remark 4.
PART	Parts E1, …, E7 will be included. See Remark 4.
DPART	Parts E1, …, E7 if previously added will be excluded. See Remark 4.
SET	Parts of part sets E1, …, E7 will be included. Note that *SET_PART_ADD cannot be included. See Remark 4.
DSET	Previously added parts that are members of part sets E1, …, E7 will be excluded. See Remark 4.

Remarks:

1. **Part attributes.** Part attributes can be assigned for some input types. For example, for airbags, a time delay, DA1 = T1, can be defined before pressure begins to act along with a time delay, DA2 = T2, before full pressure is applied (default T2 = T1). For *CONSTRAINED_RIGID_BODY_STOPPERS one attribute can be defined: DA1 as the closure distance which activates the stopper constraint.

2. **Individual part attributes.** The default part attributes can be overridden on the part cards with the COLUMN keyword option; otherwise, A1 = DA1, etc.

3. **Solvers.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

4. **Tightly packed data.** Lists of parts, sets, materials, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between the needed IDs (or material types in the case of OPTION = MATTYPE). Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_PART
         1
        10        20        35        60        85
*SET_PART_GENERAL
         3
      PART         2       102
```

They should *not* be input like the following:

```
*SET_PART
         1
        10        20         0        35        60        85
*SET_PART_GENERAL
         3
      PART         2               102
```

*SET_PART_ADD

Purpose: Define a part set by combining part sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER		
Type	I	F	F	F	F	A		
Default	none	0.0	0.0	0.0	0.0	MECH		
Remarks		1, 2	1, 2	1, 2	1, 2	3		

Part Set Cards. Each card can be used to specify up to 8 part set IDs. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	PSID1	PSID2	PSID3	PSID4	PSID5	PSID6	PSID7	PSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID. All part sets should have a unique set ID.
DA1	First attribute default value
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value
SOLVER	Name of solver using this set (MECH, CESE, etc.)
PSID[N]	The N^{th} part set ID (see Remark 4) GT.0: PSID[N] is added to SID, LT.0: All part sets with ID between PSID[N-1] and -PSID[N], including PSID[N-1] and -PSID[N], will be added to SID.

VARIABLE	DESCRIPTION
	PSID[N-1] must be > 0 and must have a magnitude smaller or equal to -PSID[N] when PSID[N] < 0.

Remarks:

1. **Part attributes.** Part attributes can be assigned for some input types. For example, for airbags a time delay, DA1 = T1, can be defined before pressure begins to act along with a time delay, DA2 = T2, before full pressure is applied, (default T2 = T1). For the constraint option, *CONSTRAINED_RIGID_BODY_STOPPERS one attribute can be defined: DA1, the closure distance which activates the stopper constraint.

2. **Overriding part attributes.** The default values for the part attributes are given in the contributing *SET_PART_{*OPTION*} commands. Nonzero values of DA1, DA2, DA3, or DA4 in *SET_PART_ADD will override the respective default values.

3. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

4. **Tightly packed data.** Lists of part sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_PART_ADD
         1
        10        20        35        60        85
*SET_PART_ADD
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_PART_ADD
         1
        10        20         0        35        60        85
*SET_PART_ADD
         3
         2                 102
```

*SET_PART_TREE

Purpose: Define a branch in a tree structure. A branch is a part set that can be defined using parts and/or sub-branches. With this keyword, the whole model can be modeled as a hierarchical tree structure.

Card Summary:

Card 1. This card is required.

BRID							

Card 2. This card is required.

HEADING	

Card 3. Include as many cards as needed. This input ends at the next keyword ("*") card.

COMP1	COMP2	COMP3	COMP4	COMP5	COMP6	COMP7	COMP8

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	BRID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
BRID	Branch identification. A unique number must be specified.

Card 2	1	2	3	4	5	6	7	8
Variable	HEADING							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
HEADING	Heading for the branch

Component Data Cards. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	COMP1	COMP2	COMP3	COMP4	COMP5	COMP6	COMP7	COMP8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
COMP*i*	Components of branch BRID (see Remark 2): GT.0: ID of a sub-branch LT.0: ID of a part

Remarks:

1. **Properties of part trees.** Part trees have the following properties:

   a) A branch is basically a part set. A branch that does *not* belong to any other branch is considered to belong to the root of the whole model.

   b) A branch or part cannot belong to more than one branch.

   c) Parts which are never referenced, directly or indirectly, in *SET_PART_-TREE are considered the children of the root of the whole model.

2. **Tightly packed data.** The list of components (parts and sub-branches) of a branch in the data cards must be tightly packed, meaning no zeros or empty

fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_PART_TREE
         1
        10        20        35        60        85
*SET_PART_TREE
         3
         2       102
```

They should *not* be input like the following:

```
*SET_PART_TREE
         1
        10        20         0        35        60        85
*SET_PART_TREE
         3
         2                 102
```

*SET_PERI_LAMINATE

Purpose: Assemble laminae (each lamina must be defined as one individual part) into a laminate and define the fiber angles for laminae.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Repeat this card as many times as needed to specify each lamina of the laminate. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	PID1	A1	T1	PID2	A2	T2		
Type	I	F	F	I	F	F		

VARIABLE	DESCRIPTION
SID	Set ID for the laminate
PID*i*	Part ID of i^{th} lamina
A*i*	Fiber angle of i^{th} lamina. This angle is relative to the direction (V1,V2,V3) specified in *MAT_ELASTIC_PERI_LAMINATE
T*i*	Thickness of i^{th}

*SET_SEGMENT_{OPTION1}_{OPTION2}

For *OPTION1* the available options are:

<BLANK>

GENERAL

For *OPTION2* the available option is

COLLECT

Purpose: Define a set of segments with optional identical or unique attributes. For three-dimensional geometries, a segment can be triangular or quadrilateral. For two-dimensional geometries, a segment is a line defined by two nodes and the GENERAL option does not apply.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4	SOLVER	ITS	

Card 2a. This card is included if *OPTION1* is <BLANK>. Include as many cards as necessary. This input ends at the next keyword ("*") card.

N1	N2	N3	N4	A1	A2	A3	A4

Card 2b. This card is included if *OPTION1* is GENERAL. Include as many cards as necessary. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Cards:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4	SOLVER	ITS	
Type	I	F	F	F	F	A	I	
Default	none	0.	0.	0.	0.	MECH	0	

VARIABLE	DESCRIPTION
SID	Set ID. All segment sets should have a unique set ID.
DA1	First segment attribute default value. See Remark 1.
DA2	Second segment attribute default value. See Remark 1.
DA3	Third segment attribute default value. See Remark 1.
DA4	Fourth segment attribute default value. See Remark 1.
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 3.
ITS	Define coupling type across different scales in two-scale co-simulation. This flag should only be included for segment sets that provide coupling information in the input file referred to by *INCLUDE_COSIM. EQ.1: Tied contact coupling EQ.2: Solid-in-shell immersed coupling

Segment Cards. For each segment in the set include one card of this format unless the GENERAL option is used. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	A1	A2	A3	A4
Type	I	I	I	I	F	F	F	F

VARIABLE	DESCRIPTION
N1	Nodal point n_1
N2	Nodal point n_2
N3	Nodal point n_3
N4	Nodal point n_4. To define a triangular segment, set N4 = N3.
A1	First segment attribute. See Remark 2.
A2	Second segment attribute. See Remark 2.

VARIABLE	DESCRIPTION
A3	Third segment attribute. See Remark 2.
A4	Fourth segment attribute. See Remark 2.

Generalized Part ID Range Cards. This Card 2 format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I or F	I or F	I or F	I or F

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below.
E1, …, E7	Specified entity. Each card must have an option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accept up to 7 arguments, but they may take fewer. Values of "E*n*" left unspecified are ignored.

OPTION	DESCRIPTION
ALL	All exterior segments will be included in the set.
BOX	Generate segments inside boxes having IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. For shell elements, one segment per shell is generated. For solid elements, only those segments wrapping the solid part and pointing outward from the part will be generated.
BOX_SHELL	Generate segments inside boxes having IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. The segments are only

OPTION	DESCRIPTION
	generated for shell elements. One segment per shell is generated.
BOX_SLDIO	Generate segments inside boxes having IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. Both exterior segments and inter-element segments are generated.
BOX_SOLID	Generate segments inside boxes having IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. The segments are only generated for exterior solid elements
BRANCH	Generate segments from tree branches E1, E2, and E3 with attributes E4, E5, E6, and E7. For shell elements, one segment is generated per shell. For solid elements, only those segments that wrap the solid part and point outward from the part will be generated. See *SET_PART_TREE.
BRANCH_IO	Generate segments from tree branches E1, E2, and E3 with attributes E4, E5, E6, and E7. Same as the BRANCH option above, except that inter-element segments inside parts will be generated as well. This option is sometimes useful for preventing negative volumes when using single surface contact between solid elements.
BRSLDF*i*	Generate segments from the *i*th face of solid tree branches E1, E2, and E3 with attributes E4, E5, E6, and E7. See Table 43-1 below for the face definition.
DBOX	Segments inside boxes with IDs E1, …, E7 will be excluded. See Remark 4.
DBOX_SHELL	Shell-related segments inside boxes of IDs E1, …, and E7 will be excluded. See Remark 4.
DBOX_SOLID	Solid-related segments inside boxes of IDs E1, …, and E7 will be excluded. See Remark 4.
DPART	Segments of parts with IDs E1, …, and E7 will be excluded. See Remark 4.
DSEG	Segment with node IDs E1, E2, E3, and E4 will be deleted.
DSET	Segments of segment sets (*SET_SEGMENT) with IDs E1, E2, E3… will be excluded. See Remark 4.
DVOL	Segments inside contact volumes having IDs E1, …, and E7 will be excluded. See Remark 4.

OPTION	DESCRIPTION
DVOL_SHELL	Shell-related segments inside contact volumes having IDs E1, …, and E7 will be excluded. See Remark 4.
DVOL_SOLID	Solid-related segments inside contact volumes having IDs E1, …, and E7 will be excluded. See Remark 4.
PART	Generate segments from parts E1, E2, and E3 with attributes E4, E5, E6, and E7. For shell elements, one segment per shell is generated. For solid elements, only those segments wrapping the solid part and pointing outward from the part will be generated. PART could refer to beam parts when defining 2D segments for traction application.
PART_IO	Generate segments from parts E1, E2, and E3 with attributes E4, E5, E6, and E7. Same as the PART option above except that inter-element segments inside parts will be generated as well. This option is sometimes useful for single surface contact of solid elements to prevent negative volumes.
PSLDF*i*	Generate segments from the i^{th} face of solid parts E1, E2, and E3 with attributes E4, E5, E6, and E7. See Table 43-1 below for face definition.
SALEBOXL	Segments inside a box in a structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX, respectively. They are the minimum and the maximum nodal local coordinates, as defined in *ALE_STRUCTURED_MESH_CONTROL_POINTS, along each direction in S-ALE mesh. This option is only to be used for S-ALE meshes. . It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALECPT	Segments inside a box in a structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to IMIN, IMAX, JMIN, JMAX, KMIN, and KMAX. They are the minimum and the maximum nodal indices along each direction in the S-ALE mesh. To include all segments in the S-ALE mesh defined by E1, set values E2–E7 to zero or leave them blank. This option is only to be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used

OPTION	DESCRIPTION
	with other GENERAL options. Refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALEFAC	Segments on the face of an S-ALE mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, E7 correspond to -X, +X, -Y, +Y, -Z, +Z faces. Assigning 1 to these 6 values would include all the surface segments at these faces in the segment set. This option is only to be used for S-ALE meshes. . It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. For trimmed S-ALE meshes (see *ALE_STRUCTURED_MESH_TRIM), a segment is treated as a surface segment as long as it has no neighboring element along the specified direction. The set thus includes all surface segments facing that direction at the exterior and interior boundaries. Refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SEG	Create a segment with node IDs E1, E2, E3, and E4.
SET	Segments of segment sets (*SET_SEGMENT) with IDs E1, E2, E3… will be included. See Remark 4.
SET_SHELL	Generate segments from shell elements in shell sets created with *SET_SHELL_LIST with IDs E1, E2, and E3. E4, E5, E6, and E7 set the attributes.
SET_SOLID	Generate segments from solid elements in solid sets created with *SET_SOLID with IDs E1, E2, and E3. E4, E5, E6, and E7 set the attributes.
SET_SLDIO	Generate segments for solid elements in solid sets created with *SET_SOLID with IDs E1, E2, and E3. E4, E5, E6, and E7 set the attributes. Both exterior and interior segments are generated.
SET_SLDF*i*	Generate segments from the i^{th} face of solid elements in the solid element sets created with *SET_SOLID with IDs E1, E2, and E3. E4, E5, E6, and E7 set the attributes. See Table 43-1 below for face definitions.

OPTION	DESCRIPTION
SET_TSHELL	Generate segments from thick shell elements in the thick shell sets created with *SET_TSHELL with IDs of E1, E2, and E3. E4, E5, E6, and E7 set the attributes. Only exterior segments are generated.
SET_TSHIO	Generate segments from thick shell elements in the thick shell sets created with *SET_TSHELL with IDs of E1, E2, and E3. E4, E5, E6, and E7 set the attributes. Both exterior and interior segments are generated.
SHELL	Generate segments for shell elements with IDs of E1, E2, and E3 with attributes having values E4, E5, E6, and E7.
VOL	Generate segments inside contact volume IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. See BOX option for other details. See *DEFINE_CONTACT_VOLUME.
VOL_SHELL	Generate segments from shells inside contact volume IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7
VOL_SLDIO	Generate segments from solid elements inside contact volume IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. See BOX_SLDIO for other details.
VOL_SOLID	Generate segments from solid elements inside contact volume IDs E1, E2, and E3 with attributes having values E4, E5, E6, and E7. See BOX_SOLID for other details.

Table 43-1. Face definition of solid elements

Face	Hexahedron	Pentahedron	Tetrahedron
1	N1, N5, N8, N4	N1, N2, N5	N1, N2, N4
2	N2, N3, N7, N6	N4, N6, N3	N2, N3, N4
3	N1, N2, N6, N5	N1, N4, N3, N2	N1, N3, N2
4	N4, N8, N7, N3	N2, N3, N6, N5	N1, N4, N3
5	N1, N4, N3, N2	N1, N5, N6, N4	
6	N5, N6, N7, N8		

Remarks:

1. **Segment attributes.** Segment attributes can be assigned for some input types. For example, for the contact options:

   The attributes for the SURFA surface are:

   DA1 (NFLS) = Normal failure stress, *CONTACT_TIEBREAK_SURFACE_... contact only,

   DA2 (SFLS) = Shear failure stress, *CONTACT_TIEBREAK_SURFACE_... contact only,

   DA3 (FSF) = Coulomb friction scale factor,

   DA4 (VSF) = Viscous friction scale factor,

   and the attributes for the SURFB surface are:

   DA3 (FSF) = Coulomb friction scale factor,

   DA4 (VSF) = Viscous friction scale factor.

   For airbags (see *AIRBAG), a time delay, DA1 = T1, can be defined before pressure begins to act on a segment, along with a time delay, DA2 = T2, before full pressure is applied to the segment (default T2 = T1).

2. **Setting individual attributes.** The default segment attributes can be overridden on Card 2a; otherwise, A1 = DA1, A2 = DA2, etc.

3. **Solvers.** The SOLVER field is used for a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics' mesh.

4. **Tightly packed data.** Lists of data in the data cards for OPTION = DBOX, DBOX_SHELL, DBOX_SOLID, DPART, DSET, DVOL, DVOL_SHELL, DVOL_SOLID, and SET must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_SEGMENT_GNERAL
         1
       SET         20         35         60         85
*SET_SEGMENT_GENERAL
         3
       SET          2        102
```

   They should *not* be input like the following:

```
*SET_SEGMENT_GENERAL
         1
       SET         10         20          0         35         60         85
```

```
*SET_SEGMENT_GENERAL
         3
       SET         2                 102
```

*SET_SEGMENT_ADD

Purpose: Define a segment set by combining segment sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SOLVER						
Type	I	A						
Default	none	MECH						
Remarks		1						

Segment Set Cards. Each card can be used to specify up to 8 segment set IDs. Include as many cards of this kind as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new segment set. All segment sets should have a unique set ID.
SOLVER	Name of solver using this set (MECH, CESE, etc.)
SSID[N]	The N^{th} segment set ID (see Remark 2)

Remarks:

1. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of segment sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable

behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_SEGMENT_ADD
         1
        10        20        35        60        85
*SET_SEGMENT_ADD
         3
         2       102
```

They should *not* be input like the following:

```
*SET_SEGMENT_ADD
         1
        10        20         0        35        60        85
*SET_SEGMENT_ADD
         3
         2                  102
```

*SET_SEGMENT_INTERSECT

Purpose: Define a segment set as the intersection, ∩, of a series of segment sets. The new segment set, SID, contains all segments common to the sets listed on all of the cards in format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SOLVER						
Type	I	A						
Default	none	MECH						
Remarks		1						

Segment Set Cards. For each SID in the intersection specify one field. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new segment set. All segment sets should have a unique set ID.
SOLVER	Name of solver using this set (MECH, CESE, etc.)
SSID[N]	The N^{th} segment set ID (see Remark 2)

Remarks:

1. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of segment sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs.

Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_SEGMENT_INTERSECT
         1
        10        20        35        60        85
*SET_SEGMENT_INTERSECT
         3
         2       102
```

They should *not* be input like the following:

```
*SET_SEGMENT_INTERSECT
         1
        10        20         0        35        60        85
*SET_SEGMENT_INTERSECT
         3
         2                 102
```

*SET_SEGMENT_N8

Purpose: Define an interface region using a set of segments that is used to transfer data in a System Coupling co-simulation. This keyword can only be used for System Coupling. See *COSIM_SYC_CONTROL and *COSIM_SYC_INTERFACE. An N8 segment can have up to eight nodes.

Card Summary:

Card 1. This card is required.

SID							

Card 2. Include as many cards as necessary to define a System Coupling interface region. This input ends at the next keyword ("*") card.

N1	N2	N3	N4	N5	N6	N7	N8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SID	Set ID. All segment sets should have a unique set ID.

Segment Cards. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	N1	N2	N3	N4	N5	N6	N7	N8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
Ni	Nodal point *i*. See Figure 19-26 of *ELEMENT_SHELL for the numbering sequence.

***SET_SHELL**_*{OPTION1}*_*{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

LIST

COLUMN

LIST_GENERATE

LIST_GENERATE_INCREMENT

GENERAL

For *OPTION2* the available option is:

COLLECT

The LIST_GENERATE and LIST_GENERATE_INCREMENT options will generate block(s) of shell element IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the shell element set.

Purpose: Define a set of shell elements with optional identical or unique attributes.

Card Summary:

Card 1. This card is required.

SID	DA1	DA2	DA3	DA4			

Card 2a. Include this card if the keyword option is unused or if the LIST keyword option is used. Set one value per element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8

Card 2b. Include this card if the COLUMN keyword option is used. Include one card per shell element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

EID	A1	A2	A3	A4			

Card 2c. Include this card if the LIST_GENERATE keyword option is used. Set one pair of BNBEG and BNEND values per block of shell element IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2d. Include this card if the LIST_GENERATE_INCREMENT keyword option is used. For each block of shell elements add one card to the deck. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Card 2e. Include this card if the GENERAL keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4			
Type	I	F	F	F	F			
Default	none	0.	0.	0.	0.			
Remarks		1	1	1	1			

VARIABLE	DESCRIPTION
SID	Set ID. All shell sets should have a unique set ID.
DA1	First attribute default value
DA2	Second attribute default value
DA3	Third attribute default value
DA4	Fourth attribute default value

Shell Element ID Cards. This Card 2 format applies to the LIST keyword option. Additionally, it applies to the case of an unset (<blank>) keyword option. Set one value per element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
EID*i*	i^{th} shell element ID (see Remark 3)

Shell Element ID with Column Cards. This Card 2 format applies to the COLUMN keyword option. Include one card per shell element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	EID	A1	A2	A3	A4			
Type	I	F	F	F	F			
Remarks		2	2	2	2			

VARIABLE	DESCRIPTION
EID	Element ID
A1	First attribute
A2	Second attribute
A3	Third attribute
A4	Fourth attribute

Shell Element ID Range Cards. This Card 2 format applies to the LIST_GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of shell element IDs. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First shell ID in shell block N.
B[N]END	Last shell ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the element numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not element IDs.

Shell Element ID Range with Increment Cards. This Card 2 format applies to the LIST_GENERATE_INCREMENT keyword option. For each block of shell elements add one card to the deck. This input ends at the next keyword ("*") card.

Card 2d	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First shell element ID in block.
BEND	Last shell element ID in block.
INCR	Shell element ID increment. Shell element IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Generalized Shell Element ID Range Cards. This Card 2 format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2e	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See table below.
E1, …, E7	Specified entity. Each card must have the option specified. See table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accepts up to 7 arguments, but they may take fewer. Values of "E*n*" left unspecified are ignored.

Note that the order of the selected operations matters. Elements are only excluded from the set if they were previously added to the set. For instance, if you exclude shell element 5 and then include it, element 5 will be included in the set. For the example set below, suppose that part 6 includes shell elements 10, 15, 20, and 32, box 7 contains shell elements 5, 20, and 32, and part 10 includes shell elements 5, 22, and 106. Then, shell element set 1 contains shell elements 5, 10, 15, 22, and 106.

```
*SET_SHELL_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All shell elements will be included in the set.
BOX	Shell elements inside boxes E1, E2, E3, … will be included. See *DEFINE_BOX. See Remark 3.

OPTION	DESCRIPTION
DBOX	Previously added shell elements that are inside boxes E1, E2, E3, … will be excluded. See Remark 3.
ELEM	Shell elements E1, E2, E3, … will be included. See Remark 3.
DELEM	Shell elements E1, E2, E3, … if previously added will be excluded. See Remark 3.
PART	Shell elements of parts E1, E2, E3, … will be included. See Remark 3.
DPART	Shell elements that have been previously added and are of parts E1, E2, E3, … will be excluded. See Remark 3.
SALEBOXL	Elements inside a box for a 2D Structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, and E5 correspond to XMIN, XMAX, YMIN, and YMAX, respectively. They are the minimum and the maximum nodal local coordinates, as defined in *ALE_STRUCTURED_MESH_CONTROL_POINTS, along each direction in S-ALE mesh. This option is only to be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALECPT	Elements inside a box for a 2D Structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, and E5 correspond to IMIN, IMAX, JMIN, and JMAX, respectively. They are the minimum and the maximum nodal indices along each direction in the S-ALE mesh. To include all shells in the S-ALE mesh defined by E1, set values E2–E7 to zero or leave them blank. This option is only to be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALEFAC	Elements on the face of a 2D S-ALE mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, and E5 correspond to the -X, +X, -Y, and +Y faces, respectively. Assigning 1 to these 4 values would include all the boundary elements at these faces in the shell element set. This option is only to be used for S-ALE meshes. It can be used

OPTION	DESCRIPTION
	with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. For trimmed S-ALE meshes (see *ALE_STRUCTURED_MESH_-TRIM), an element is treated as a surface element as long as it has no neighboring element along the specified direction. The set thus includes all surface elements facing that direction at exterior and interior boundaries. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SET	Elements of shell element sets (*SET_SHELL) E1, E2, E3, … will be included. See Remark 3.
DSET	Previously added elements that are members of shell element sets E1, E2, E3, ... will be excluded. See Remark 3.

Remarks:

1. **Shell attributes.** Shell attributes can be assigned for some input types.

   For example, for contact options, the attributes for the SURFA surface are:

   DA1 (NFLS) = Normal failure stress, *CONTACT_TIEBREAK_SUR-FACE_... contact only,

   DA2 (SFLS) = Shear failure stress, *CONTACT_TIEBREAK_SURFACE_... contact only,

   DA3 (FSF) = Coulomb friction scale factor,

   DA4 (VSF) = Viscous friction scale factor,

   and the attributes for the SURFB surface are:

   DA1 (FSF) = Coulomb friction scale factor,

   DA2 (VSF) = Viscous friction scale factor.

2. **Overriding shell attributes.** The default shell attributes for the set can be overridden for individual elements using Card 2b; otherwise, A1 = DA1, etc.

3. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to

unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_SHELL
         1
        10        20        35        60        85
*SET_SHELL_GENERAL
         2
      PART         2       102
```

They should *not* be input like the following:

```
*SET_SHELL
         1
        10        20         0        35        60        85
*SET_SHELL_GENERAL
         2
      PART         2                 102
```

*SET_SHELL_ADD

Purpose: Define a shell set by combining shell sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Shell Element Set Cards. Each card can be used to specify up to 8 shell element set IDs. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new shell set. All shell sets should have a unique set ID.
SSID[N]	The N^{th} shell set ID on Card 2 (see Remark 1)

Remarks:

1. **Tightly packed data.** Lists of shell sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_SHELL_ADD
         1
        10        20        35        60        85
*SET_SHELL_ADD
         3
         2       102
```

   They should *not* be input like the following:

```
*SET_SHELL_ADD
         1
        10        20         0        35        60        85
*SET_SHELL_ADD
         3
         2               102
```

*SET_SHELL_INTERSECT

Purpose: Define a shell set as the intersection, ∩, of a series of shell sets. The new shell set, SID, contains all shells common to all sets on the cards of format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							
Default	none							

Shell Element Set Cards. For each shell element SID in the intersection input one field. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new shell set. All shell sets should have a unique set ID.
SSID[N]	The N^{th} shell set ID (see Remark 1)

Remarks:

1. **Tightly packed data.** Lists of shell sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_SHELL_INTERSECT
         1
        10        20        35        60        85
*SET_SHELL_INTERSECT
         3
         2       102
```

They should *not* be input like the following:

```
*SET_SHELL_INTERSECT
         1
        10        20         0        35        60        85
*SET_SHELL_INTERSECT
         3
         2               102
```

***SET_SOLID**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

GENERATE

GENERATE_INCREMENT

GENERAL

For *OPTION2* the available option is:

COLLECT

The GENERATE and GENERATE_INCREMENT options will generate block(s) of solid element IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the solid element set.

Purpose: Define a set of solid elements.

Card Summary:

Card 1. This card is required.

SID	SOLVER	ITS					

Card 2a. Include this card if the keyword option is left unset (<BLANK>). Set one value per solid element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

K1	K2	K3	K4	K5	K6	K7	K8

Card 2b. Include this card if the keyword option GENERATE is used. Set one pair of BNBEG and BNEND values per block of solid elements. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the keyword option GENERATE_INCREMENT is used. For each block of solid elements add one card to the deck. This input ends at the next keyword ("*") card.

BBEG	BEND	INCR					

Card 2d. Include this card if the keyword option GENERAL is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SOLVER	ITS					
Type	I	A	I					
Default	none	MECH	0					

VARIABLE	DESCRIPTION
SID	Set ID. All solid sets should have a unique set ID.
SOLVER	Name of solver using this set (MECH, CESE, etc.). See Remark 1.
ITS	Define coupling type across different scales in two-scale co-simulation. This flag should only be included for solid sets that provide coupling information in the input file referred to by *INCLUDE_COSIM. EQ.3: Solid-in-solid immersed coupling

Solid Element ID Cards. This card format applies to the case of an unset (<BLANK>) keyword option. Set one value per solid element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	K1	K2	K3	K4	K5	K6	K7	K8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
K*i*	*i*th element ID (see Remark 2)

Solid Element ID Range Cards. This card format applies to the GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of solid elements. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First solid element ID in block N.
B[N]END	Last solid element ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the element numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not element IDs.

Solid Element ID Range with Increment Cards. This card format applies to the GENERATE_INCREMENT keyword option. For each block of solid elements add one card to the deck. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	BBEG	BEND	INCR					
Type	I	I	I					

VARIABLE	DESCRIPTION
BBEG	First solid element ID in block.
BEND	Last solid element ID in block.
INCR	Solid ID increment. Solid IDs BBEG, BBEG + INCR, BBEG + 2 × INCR, and so on through BEND are added to the set.

Generalized Solid Element ID Range Cards. This card format applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2d	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See the table below.
E1, ..., E7	Specified entity. Each card must have the option specified. See the table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2d for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option of Card 2d.

Each of the following operations accepts up to 7 arguments, but they may take fewer. Values of "En" left unspecified are ignored.

Note that the order of the selected operations matters. Elements are only excluded from the set if they were previously added to the set. For instance, if you exclude solid element 5 and then include it, element 5 will be included in the set. For the example set below, suppose that part 6 includes solid elements 10, 15, 20, and 32, box 7 contains solid elements 5, 20, and 32, and part 10 includes solid elements 5, 22, and 106. Then, solid element set 1 contains solid elements 5, 10, 15, 22, and 106.

```
*SET_SOLID_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All solid elements will be included in the set.
BOX	Elements inside boxes E1, E2, E3, ... will be included. See *DEFINE_BOX. See Remark 2.

OPTION	DESCRIPTION
DBOX	Previously added elements that are inside boxes E1, E2, E3, ... will be excluded. See Remark 2.
ELEM	Elements E1, E2, E3, ... will be included. See Remark 2.
DELEM	Elements E1, E2, E3, ... if previously added will be excluded. See Remark 2.
PART	Elements of parts E1, E2, E3, ... will be included. See Remark 2.
DPART	Elements that have been previously added and are of parts E1, E2, E3, ... will be excluded. See Remark 2.
SALEBOXL	Elements inside a box in a structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX, respectively. They are the minimum and the maximum nodal local coordinates, as defined in *ALE_STRUCTURED_MESH_CONTROL_POINTS, along each direction in S-ALE mesh. This option is only to be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALECPT	Elements inside a box in a Structured ALE (S-ALE) mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to IMIN, IMAX, JMIN, JMAX, KMIN, and KMAX. They are the minimum and the maximum nodal indices along each direction in the S-ALE mesh. To include all solids in the S-ALE mesh defined by E1, set values E2–E7 to zero or leave them blank. This option is only to be used for S-ALE meshes. It can be used with options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SALEFAC	Elements on the face of the S-ALE mesh. E1 is the S-ALE mesh ID (MSHID). E2, E3, E4, E5, E6, and E7 correspond to -X, +X, -Y, +Y, -Z, and +Z faces. Assigning 1 to these 6 values would include all the boundary elements at these faces in the solid element set. This option is only to be used for S-ALE mesh. It can be used with

OPTION	DESCRIPTION
	options SALECPT, SALEFAC, and SALEBOXL but should not be used with other GENERAL options. For trimmed S-ALE meshes (see *ALE_STRUCTURED_MESH_-TRIM), an element is treated as a surface element as long as it has no neighboring element along the specified direction. The set thus includes all surface elements facing that direction at the exterior and interior boundaries. Please refer to *ALE_STRUCTURED_MESH_CONTROL_POINTS and *ALE_STRUCTURED_MESH for more details.
SET	Elements of solid element sets E1, E2, E3, … will be included. See Remark 2.
DSET	Previously added elements that are members of solid element sets E1, E2, E3, ... will be excluded. See Remark 2.
VOL	Elements inside contact volumes E1, E2, E3, … will be included. See *DEFINE_CONTACT_VOLUME. See Remark 2.
DVOL	Previously added elements that are inside contact volumes E1, E2, E3, … will be excluded. See Remark 2.

Remarks:

1. **Solvers and mesh.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics' mesh.

2. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_SOLID
         1
        10        20        35        60        85
*SET_SOLID_GENERAL
         2
      PART         2       102
```

   They should *not* be input like the following:

```
*SET_SOLID
         1
```

```
        10        20         0        35        60        85
*SET_SOLID_GENERAL
         2
      PART         2               102
```

*SET_SOLID_ADD

Purpose: Define a solid set by combining solid sets.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SOLVER						
Type	I	A						
Default	none	MECH						
Remarks		1						

Node Set Cards. Each card can be used to specify up to 8 solid set IDs. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new solid set. All solid sets should have a unique set ID.
SOLVER	Name of solver using this set (MECH, CESE, etc.)
SSID[N]	The N^{th} solid set ID.

Remarks:

1. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of solid sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_SOLID_ADD
         1
        10        20        35        60        85
*SET_SOLID_ADD
         3
         2       102
```

They should *not* be input like the following:

```
*SET_SOLID_ADD
         1
        10        20         0        35        60        85
*SET_SOLID_ADD
         3
         2                 102
```

*SET_SOLID_INTERSECT

Purpose: Define a solid set as the intersection, ∩, of a series of solid sets. The new solid set, SID, contains all common elements of all solid sets SSID*n*.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	SOLVER						
Type	I	A						
Default	none	MECH						
Remarks		1						

Solid Element Set Cards. For each solid element SID in the intersection, input one field. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	SSID1	SSID2	SSID3	SSID4	SSID5	SSID6	SSID7	SSID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
SID	Set ID of new solid set. All solid sets should have a unique set ID.
SOLVER	Name of solver using this set (MECH, CESE, etc.)
SSID[N]	The N^{th} solid set ID (see Remark 2)

Remarks:

1. **Solver.** This field is used by a non-mechanics solver to create a set defined on that solver's mesh. By default, the set refers to the mechanics mesh.

2. **Tightly packed data.** Lists of solid sets in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

For example, lists of IDs should be input like the following:

```
*SET_SOLID_INTERSECT
         1
        10        20        35        60        85
*SET_SOLID_INTERSECT
         3
         2       102
```

They should *not* be input like the following:

```
*SET_SOLID_INTERSECT
         1
        10        20         0        35        60        85
*SET_SOLID_INTERSECT
         3
         2                 102
```

***SET_TSHELL**_*{OPTION1}_{OPTION2}*

For *OPTION1* the available options are:

<BLANK>

GENERATE

GENERAL

For *OPTION2* the available option is:

COLLECT

The GENERATE option generates a block of thick shell element IDs between a starting ID and an ending ID. An arbitrary number of blocks can be specified to define the set.

Purpose: Define a set of thick shell elements.

Card Summary:

Card 1. This card is required.

SID							

Card 2a. Include this card if the keyword option is unset (<BLANK>). Include as many cards as needed. This input ends at the next keyword ("*") card.

K1	K2	K3	K4	K5	K6	K7	K8

Card 2b. Include this card if GENERATE keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END

Card 2c. Include this card if the GENERAL keyword option is used. Include as many cards as needed. This input ends at the next keyword ("*") card.

OPTION	E1	E2	E3	E4	E5	E6	E7

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID							
Type	I							

VARIABLE	DESCRIPTION
SID	Set ID. All tshell sets should have a unique set ID.

Thick Shell Element ID Cards. This card applies to the case of an unset (<BLANK>) keyword option. Set one value per thick shell element in the set. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2a	1	2	3	4	5	6	7	8
Variable	K1	K2	K3	K4	K5	K6	K7	K8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
K*i*	i^{th} thick shell element ID (see Remark 1)

Thick Shell Element ID Range Cards. This card applies to the GENERATE keyword option. Set one pair of BNBEG and BNEND values per block of thick shell elements. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2b	1	2	3	4	5	6	7	8
Variable	B1BEG	B1END	B2BEG	B2END	B3BEG	B3END	B4BEG	B4END
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
B[N]BEG	First thick shell element ID in block N.

VARIABLE	DESCRIPTION
B[N]END	Last thick shell element ID in block N. All defined IDs between and including B[N]BEG to B[N]END are added to the set. These sets are generated after all input is read so that gaps in the element numbering are not a problem. B[N]BEG and B[N]END may simply be limits on the IDs and not element IDs.

Generalized Thick Shell Element ID Range Cards. This card applies to the GENERAL keyword option. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 2c	1	2	3	4	5	6	7	8
Variable	OPTION	E1	E2	E3	E4	E5	E6	E7
Type	A	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
OPTION	Option for GENERAL. See the table below.
E1, ..., E7	Specified entity. Each card must have the option specified. See the table below.

The General Option:

The "OPTION" column in the table below enumerates the allowed values for the "OPTION" variable in Card 2 for the GENERAL option. Likewise, the variables E1, …, E7 refer to the GENERAL option Card 2.

Each of the following operations accept up to 7 arguments, but they may take fewer. Values of "En" left unspecified are ignored.

Note that the order of the selected operations matters. Elements are only excluded from the set if they were previously added to the set. For instance, if you exclude tshell element 5 and then include it, element 5 will be included in the set. For the example set below, suppose that part 6 includes tshell elements 10, 15, 20, and 32, box 7 contains tshell elements 5, 20, and 32, and part 10 includes tshell elements 5, 22, and 106. Then, tshell element set 1 contains tshell elements 5, 10, 15, 22, and 106.

```
*SET_TSHELL_GENERAL
1
PART, 6
DBOX, 7
PART, 10
```

OPTION	DESCRIPTION
ALL	All thick shell elements will be included in the set.
BOX	Elements inside boxes E1, E2, E3, ... will be included. See *DEFINE_BOX. See Remark 1.
DBOX	Previously added elements that are inside boxes E1, E2, E3, ... will be excluded. See Remark 1.
ELEM	Elements E1, E2, E3, ... will be included. See Remark 1.
DELEM	Elements E1, E2, E3, ... if previously added will be excluded. See Remark 1.
PART	Elements of parts E1, E2, E3, ... will be included. See Remark 1.
DPART	Elements that have been previously added and are of parts E1, E2, E3, ... will be excluded. See Remark 1.
SET	Elements in thick shell sets E1, E2, E3, … will be included. See Remark 1.
DSET	Previously added elements that are members of shell element sets E1, E2, E3, … will be excluded. See Remark 1.

Remarks:

1. **Tightly packed data.** Lists of elements, parts, sets, boxes, etc., in the data cards must be tightly packed, meaning no zeros or empty fields can appear between needed IDs. Zeros or empty fields in the middle of needed IDs can lead to unpredictable behavior, such as IDs after the zeros or empty fields being ignored or an error termination.

   For example, lists of IDs should be input like the following:

```
*SET_TSHELL
         1
        10        20        35        60        85
*SET_TSHELL_GENERAL
         2
      PART         2       102
```

   They should *not* be input like the following:

```
*SET_TSHELL
         1
        10        20         0        35        60        85
*SET_TSHELL_GENERAL
         2
      PART         2                 102
```

*SET_2D_SEGMENT_{OPTION1}_{OPTION2}

For *OPTION1* the available options are:

<BLANK>

SET

For *OPTION2* the available option is:

COLLECT

Purpose: Define a set of boundary line segments in two-dimensional axisymmetric, plane stress, and plane strain geometries with optional attributes. This command does not apply to beam formulations 7 and 8. This feature is sometimes convenient for two-dimensional parts that are subject to adaptivity because the segments in the set are updated as the geometry adapts.

Card Sets. For each set include a pair of Cards 1 and 2. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	DA1	DA2	DA3	DA4			
Type	I	F	F	F	F			
Default	none	0.	0.	0.	0.			
Remarks		1	1	1	1			

Card 2	1	2	3	4	5	6	7	8
Variable	PID/PSID							
Type	I							
Remarks	2							

VARIABLE	DESCRIPTION
SID	Set ID. All segment sets should have a unique set ID.
DA1	First segment attribute default value, see Remark 1 below.
DA2	Second segment attribute default value
DA3	Third segment attribute default value
DA4	Fourth segment attribute default value
PID/PSID	Part ID or part set ID if SET option is specified.

Remarks:

1. **Axisymmetric Problems.** The boundary along $r = 0$ is not included in axisymmetric problems.

2. **Common Boundary in Part Sets.** The common boundary between parts in the part set PSID is not included in the boundary segments.

*RVE

The *RVE keywords control a Representative Volume Element (RVE) analysis. This method can predict macroscopic material properties for various types of composites from their microstructural information. Using this method, you can conduct virtual testing of numerically re-constructed material samples at their characteristic length scales. The algorithms invoked with *RVE keywords automatically create periodic or linear displacement boundary conditions, perform FEM-based computational homogenization, and predict the homogenized macroscopic constitutive responses as well as the detailed microscopic stress / strain fields.

The only available keyword is:

*RVE_ANALYSIS_FEM

*RVE_ANALYSIS_FEM

Purpose: Predict the macroscopic effective constitutive behaviors of composite materials under the micromechanics-based computational homogenization framework. Composite materials supported for this feature include but are not limited to fiber-reinforced composites, particulate composites, laminar composites, polycrystalline aggregates, and single-phase or multi-phase porous media.

We account for both the geometrical and material nonlinearities of the Representative Volume Element (RVE) with the nonlinear computational homogenization theory. Given the material microstructural information (geometry and constitutive properties for base materials), this keyword causes

1. Automatic creation of periodic displacement boundary conditions (or linear displacement boundary conditions) on the RVE finite element mesh;

2. A nonlinear quasi-static implicit finite element analysis of the RVE under user-defined loading conditions; and

3. The calculation of the macroscopic effective material responses through homogenization of the RVE's microscopic material responses as well as a detailed distribution and evolution of microscopic stress/strain fields within the RVE.

Double precision SMP/MPP versions support this RVE analysis function.

Card Summary:

Card 1. This card is required.

MESHFILE

Card 2. This card is required.

INPT	OUPT	LCID	IDOF	BC	IMATCH	IMAGE	

Card 3. This card is required.

H11	H22	H33	H12	H23	H13		

Card 4. This card is required only if BC = 2.

PX	PY	PZ					

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MESHFILE							
Type	C							
Default	none							

VARIABLE	DESCRIPTION
MESHFILE	Name of an input file that contains the mesh information (nodal coordinates, element connectivity) of the RVE model. Note that this file should be in keyword format [basename].k. The finite element mesh given in this file is for the spatial discretization of the material microstructures. Do not include dummy nodes for specifying control points in boundary conditions. See Remark 1.

Card 2	1	2	3	4	5	6	7	8
Variable	INPT	OUPT	LCID	IDOF	BC	IMATCH	IMAGE	
Type	I	I	I	I	I	I	I	
Default	0	1	none	none	0	1	0	

VARIABLE	DESCRIPTION
INPT	Type of input: EQ.0: RVE boundary conditions are fully defined by two factors: (1) the parameter BC of this input card, and (2) the mesh information in the file specified in MESHFILE. When running an RVE simulation, the solver automatically creates a file named rve_[basename].k. This file contains all the necessary information (e.g., dummy nodes, displacement constraints, etc.) for boundary condition enforcement. EQ.1: You provide a file named rve_[basename].k to specify the boundary condition keywords (e.g. *CONSTRAINED_MULTIPLE_GLOBAL, *BOUNDARY_SPC_NODE, etc.)

VARIABLE	DESCRIPTION
	and dummy nodes for enforcing RVE boundary conditions. We do not recommend this option since it is usually nontrivial to manually define all the keywords for RVE boundary conditions. If the file rve_[basename].k is not found when running RVE simulations, then the solver creates rve_[basename].k based on the parameter BC and the mesh information in the file specified with MESHFILE.
OUPT	Type of output: EQ.1: RVE homogenization results will be written out to a file named rveout. Please refer to the keyword *DATABASE_RVE.
LCID	Load curve ID for specifying loading history (see *DEFINE_CURVE) or flag to indicate that each component of the macroscopic deformation measure (H11, H22, …, H13 on Card 3) is input as a load curve: GT.0: Load curve ID. This curve gives a scale factor for the user-defined macroscopic deformation measure (H11, H22, …, H13 input on Card 3) as a function of loading time. The scale factor is the ordinate (second column) while the loading time is the abscissa (first column). LT.0: Flag to indicate that each component of the macroscopic dispalcement gradient, H_{ij}, is input as a load curve on Card 3.
IDOF	Dimension of the RVE: EQ.2: 2D geometry EQ.3: 3D geometry
BC	Type of RVE boundary condition (see Remark 2 and Figure 44-1): EQ.0: Periodic displacement boundary condition EQ.1: Linear displacement boundary condition EQ.2: Partially Periodic Displacement Boundary Condition (PDBC). Card 4 is required. EQ.3: Periodic displacement boundary condition for RVEs with zero-thickness cohesive interface elements. It is intended for material interface debonding prediction.

VARIABLE	DESCRIPTION
IMATCH	Type of the given RVE mesh (ignored unless BC = 0 or 2; see Remark 3): EQ.0: The mesh is non-matching for periodic displacement boundary condition. EQ.1: The mesh is periodic displacement boundary condition matching.
IMAGE	Create image RVE in specified directions. This feature is only available for IMATCH = 1. See Remark 11 and Figure 44-2. EQ.110: Create image RVE in x direction EQ.120: Create image RVE in y direction EQ.130: Create image RVE in z direction EQ.212: Create image RVE in both the x and y directions EQ.213: Create image RVE in both the x and z directions EQ.232: Create image RVE in both the y and z directions EQ.300: Create image RVE in the x, y and z directions

Card 3	1	2	3	4	5	6	7	8
Variable	H11	H22	H33	H12	H23	H13		
Type	F	F	F	F	F	F		
Default	optional	optional	optional	optional	optional	optional		

VARIABLE	DESCRIPTION
H*IJ*	Component *ij* of the prescribed macroscopic displacement gradient, $\widetilde{\mathbf{H}}$. $\widetilde{\mathbf{H}}$ is assumed to be symmetric (see Remark 4). To not impose the corresponding constraints on the RVE, leave the component H*IJ* empty (instead of setting it to be zero). If LCID < 0 on Card 2, the input value is assumed to be a load curve ID for a load curve giving the component as a function of time. Please refer to Remarks 2 and 8.

PDBC Card. This card is required for BC = 2.

Card 4	1	2	3	4	5	6	7	8
Variable	PX	PY	PZ					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
P*i*	Flag to apply the periodic boundary condition in the i^{th} direction ($i = x$, y, or z) for PDBC: EQ.0: Periodic boundary condition is not applied for this direction. EQ.1: Periodic boundary condition is applied for this direction.

Remarks:

1. **RVE mesh.** The RVE's mesh information (nodal coordinates, element connectivity) should be provided in a separate input file, of which the file name should be given in Card 1 of the keyword *RVE_ANALYSIS_FEM. Usually, a 3D RVE model has a cuboid shape with its edges parallel to the X-, Y-, and Z-axes of the global coordinate system, respectively. A 2D RVE model is rectangular with its edges parallel to the X- and Y- axes of the global coordinate system, respectively. The size of the RVE should be large enough to include sufficient statistical microstructural information (fibers, particles, voids, grains, etc.) of the composite material. It should, however, remain small enough to be considered as a volume element of continuum mechanics. The 3D RVE should be meshed with solid elements (see *ELEMENT_SOLID) while the 2D RVE should be meshed with shell elements (see *ELEMENT_SHELL).

2. **Boundary conditions.** The solver can automatically create boundary conditions on the external boundary $\partial\Omega$ of the RVE. Four types of RVE boundary conditions can be created, depending on the input value of BC:

   a) *Periodic Displacement Boundary Condition (BC = 0 or 3).* When BC = 0 or 3, the solver imposes the following periodic displacement boundary condition:

$$\mathbf{w}_\alpha^+ - \mathbf{w}_\alpha^- = \widetilde{\mathbf{H}}(\mathbf{X}_\alpha^+ - \mathbf{X}_\alpha^-) \ .$$

Here $\mathbf{X}_\alpha^+ \in \partial\Omega_\alpha^+$ and $\mathbf{X}_\alpha^- \in \partial\Omega_\alpha^-$ denote the microscale material points located on a pair of opposite external boundaries $\partial\Omega_\alpha^+$ and $\partial\Omega_\alpha^-$, respectively, in the RVE's initial configuration (See Figure 44-1 for a 2D illustration). $\mathbf{w}_\alpha^+$ and $\mathbf{w}_\alpha^-$ are the microscale displacements of the material points $\mathbf{X}_\alpha^+$ and $\mathbf{X}_\alpha^-$, respectively. The subscript $\alpha = 1,2\ldots,d$ where $d = 2$ for 2D RVE models and $d = 3$ for 3D RVE models. To impose the periodicity constraint, a control point-based method is implemented, which is similar to the keywords *INCLUDE_UNITCELL and *CONSTRAINED_MULTIPLE_GLOBAL. Unlike these other keywords, *RVE_ANALYSIS_FEM automatically creates these control points.

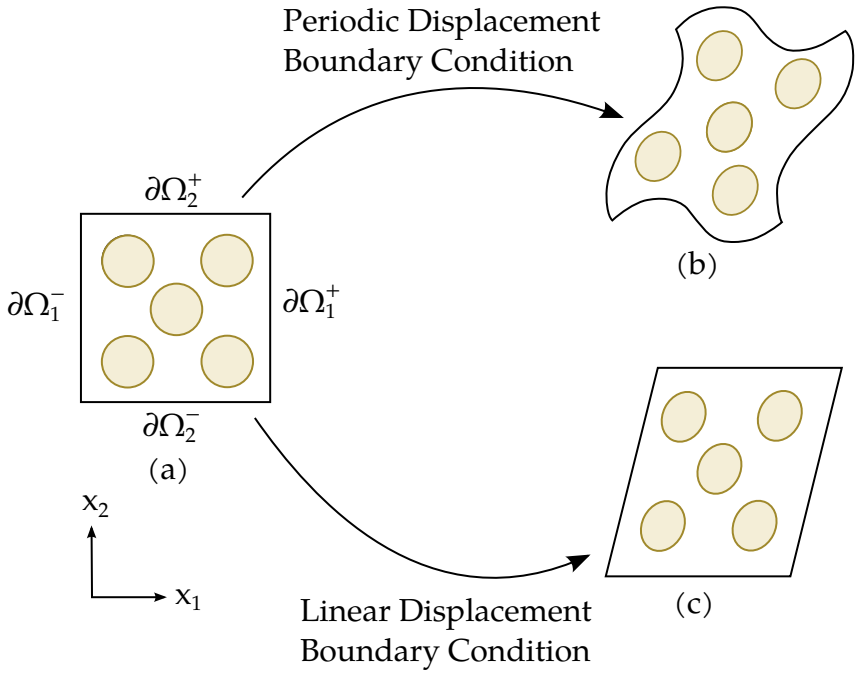


Figure 44-1. Schematic of the two types of displacement boundary conditions for RVE analysis. (a) The initial configuration of RVE occupying the domain Ω, where $\partial\Omega_\alpha^+$ and $\partial\Omega_\alpha^-$ denote a pair of opposite external boundaries. $\alpha = 1,2$ for this 2D model. (b) The current configuration of RVE subjected to periodic displacement boundary conditions. (c) The current configuration of RVE subjected to linear displacement boundary conditions.

b) *Linear Displacement Boundary Condition (BC = 1).* When BC = 1, LS-DYNA imposes the following linear displacement boundary condition:

$$\mathbf{w}_\alpha = \widetilde{\mathbf{H}}\mathbf{X}_\alpha,$$

where $\mathbf{X}_\alpha \in \partial\Omega_\alpha$ denotes any micro-scale material point located on the external boundaries $\partial\Omega_\alpha = \partial\Omega_\alpha^+ \cup \partial\Omega_\alpha^-$ of the RVE. Note that the RVE's complete external boundary can be expressed as $\partial\Omega = \bigcup_{\alpha=1}^{d} \partial\Omega_\alpha$, in which $d = 2$ for 2D RVE models and $d = 3$ for 3D RVE models. It is noteworthy to mention that RVEs with linear displacement boundary conditions usually

appear to be stiffer than RVEs with periodic displacement boundary conditions. When the size of RVE is large enough, however, the influence of different types of boundary conditions on the homogenized material properties becomes negligibly small.

c) *Partially Periodic Displacement Boundary Condition (BC = 2)*, When BC = 2, LS-DYNA enforces periodicity only along selected directions of the RVE, while the remaining direction(s) are treated as non-periodic (meaning free or subject to other specified boundary conditions). The periodic directions are defined in Card 4. For example, in a textile RVE, periodic constraints may be applied in the x and y directions, while the z direction remains non-periodic. In this case, the displacement field satisfies periodicity on boundary pairs corresponding to the periodic directions, whereas the non-periodic direction is left unconstrained or governed by independent boundary conditions. Accordingly, PX = PY = 1 and PZ = 0 are specified in Card 4.

The assignment of a zero value to any component of the macroscopic displacement gradient $\widetilde{\mathbf{H}}$ indicates the imposition of the boundary constraint based on the above constraint equations. If you do not want to impose constraints in certain directions (i.e., allow the associated RVE boundaries to deform freely), then you should leave the corresponding component of $\widetilde{\mathbf{H}}$ empty in the input card. Please refer to Remark 8 for an application scenario where most components of $\widetilde{\mathbf{H}}$ should be set empty.

3. **IMATCH for periodic displacement boundary conditions.** When the mesh is periodic displacement boundary condition matching, the nodal distributions on the RVE's opposite sides match well with each other. For instance, let us consider two opposite surfaces, surface A and surface B, that are both perpendicular to the X-axis. For any FEM node on surface A, if we draw a straight line that is parallel to the X-axis, then the intersection point of this line with surface B must also be an FEM node. For such periodic displacement boundary condition matching meshes, an efficient direct nearest neighbor search algorithm imposes the boundary condition. Thus, a matching mesh is preferred for imposing the periodic displacement boundary conditions. However, if complex material microstructures exist, it is not always straightforward to create matching meshes. In this case, set IMATCH to zero which calls a projection-based constraint imposition method.

4. **Prescribed macroscopic displacement gradient.** The prescribed macroscopic displacement gradient is given as:

$$\widetilde{\mathbf{H}} \equiv \nabla_{\tilde{\mathbf{X}}} \tilde{\mathbf{u}} = \tilde{\mathbf{F}} - \mathbf{I} \ ,$$

where $\nabla_{\tilde{\mathbf{X}}}$ is the gradient operator with respect to the macroscale, $\tilde{\mathbf{u}}$ is the macroscopic displacement field, $\tilde{\mathbf{F}}$ is the macroscopic deformation gradient, and $\mathbf{I}$ is the identity tensor. Generally, $\tilde{\mathbf{F}}$ is not symmetric. Thus, $\widetilde{\mathbf{H}}$ is not usually

symmetric. Through polar decomposition of the deformation gradient, $\tilde{\mathbf{F}} = \tilde{\mathbf{R}}\tilde{\mathbf{U}}$. $\tilde{\mathbf{R}}$ represents the macroscopic rigid body rotation, and $\tilde{\mathbf{U}}$ is a symmetric stretch tensor describing the pure material deformation. Since material constitutive behaviors are not affected by macroscopic rigid body rotations, we will assume $\tilde{\mathbf{R}} = \mathbf{I}$ in the RVE analysis. Under this condition, both the macroscopic deformation gradient $\tilde{\mathbf{F}}$ and macroscopic displacement gradient $\tilde{\mathbf{H}}$ become symmetric.

Because $\tilde{\mathbf{H}}$ is symmetric, only six components of $\tilde{\mathbf{H}}$ are needed to prescribe boundary conditions for 3D RVE analysis. For 2D RVE problems, the inputs for H33, H23, and H13 are ignored, and only the inputs for H11, H22, and H12 are adopted for enforcing the boundary condition. Note that you should leave the component H*IJ* empty (instead of setting it to be zero) if you do not want to impose the corresponding constraints on the RVE. Please refer to Remarks 2 and 8.

5. **Constitutive models for base materials.** Depending on the actual morphology of material microstructures, an RVE finite element model can consist of many parts. Each part can be assigned a unique constitutive law to describe the behaviors of the corresponding base material (e.g., fiber, particle, matrix, grain, etc.). Any constitutive model (linear/nonlinear, isotropic/anisotropic, etc.) can be selected for the base materials as long as the model is supported for implicit analysis. The part and material information should be contained in the main input file, not in MESHFILE. See the example below.

6. **Accuracy control for the implicit calculation.** To ensure high accuracy of the nonlinear implicit finite element simulation, we recommend using the 2nd order objective stress update scheme specified with OSU = 1 on *CONTROL_ACCURACY. We also suggest specifying other control parameters for the implicit solver in relevant keywords, such as *CONTROL_IMPLICIT_GENERAL, *CONTROL_IMPLICIT_SOLUTION, and *CONTROL_IMPLICIT_SOLVER.

7. **Homogenization results.** A file named rveout contains the homogenized material responses. In this file, the macroscopic deformation gradient $\tilde{\mathbf{F}}^t$, Green strain $\tilde{\mathbf{E}}^t$, Cauchy stress $\tilde{\boldsymbol{\tau}}^t$, and the 1st Piola-Kirchhoff stress $\tilde{\mathbf{P}}^t$, are recorded at each output time t. Different measures for the macroscale stress and deformation can be obtained based on their relationships with the results in rveout. For instance, the 2nd Piola-Kirchhoff stress $\tilde{\mathbf{S}}$ is defined as $\tilde{\mathbf{S}} = \tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-T}$. It can be easily calculated since $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{F}}$ are available in rveout.

   Note that, as discussed in Remark 4, the macroscale rigid body rotations are excluded from the macroscopic displacement gradient. Accordingly, the macroscopic deformation gradient $\tilde{\mathbf{F}}^t$ and 1st Piola-Kirchoff stress $\tilde{\mathbf{P}}^t$ obtained from RVE analysis are both symmetric, so only six components of each symmetric tensor are written to rveout.

8. **Degeneration to Small Strain Analysis**. Although a nonlinear computational homogenization formulation is implemented to capture the RVE's finite deformation effects, the homogenized quantities will be close to those calculated by small strain homogenization theories when the actual macroscopic deformation is small. In this situation, the macroscopic Green strain $\tilde{\mathbf{E}}$ given in the rveout file will be approximately equal to the infinitesimal strain, and different stress measures ($\tilde{\mathbf{P}}$, $\tilde{\mathbf{S}}$, and $\tilde{\boldsymbol{\tau}}$) will have an identical magnitude.

   For small strain linear analysis, if we conduct 3D RVE simulations with six orthogonal loading conditions (e.g., three uniaxial tensile loadings and three pure shear loadings), respectively, then we will obtain the full macroscopic elasticity tensor for the composite material. Recall that the macroscale linear elastic constitutive relationship can be expressed as follows:

$$\tilde{\varepsilon} = \tilde{C}\tilde{\sigma} \quad .$$

   $\tilde{C}$ is the 6 × 6 macroscopic material compliance matrix:

$$\tilde{C} = \begin{bmatrix} \tilde{C}_{11} & \tilde{C}_{12} & \tilde{C}_{13} & \tilde{C}_{14} & \tilde{C}_{15} & \tilde{C}_{16} \\ \tilde{C}_{21} & \tilde{C}_{22} & \tilde{C}_{23} & \tilde{C}_{24} & \tilde{C}_{25} & \tilde{C}_{26} \\ \tilde{C}_{31} & \tilde{C}_{32} & \tilde{C}_{33} & \tilde{C}_{34} & \tilde{C}_{35} & \tilde{C}_{36} \\ \tilde{C}_{41} & \tilde{C}_{42} & \tilde{C}_{43} & \tilde{C}_{44} & \tilde{C}_{45} & \tilde{C}_{46} \\ \tilde{C}_{51} & \tilde{C}_{52} & \tilde{C}_{53} & \tilde{C}_{54} & \tilde{C}_{55} & \tilde{C}_{56} \\ \tilde{C}_{61} & \tilde{C}_{62} & \tilde{C}_{63} & \tilde{C}_{64} & \tilde{C}_{65} & \tilde{C}_{66} \end{bmatrix},$$

   The vectors $\tilde{\varepsilon} = [\tilde{\varepsilon}_{11} \quad \tilde{\varepsilon}_{22} \quad \tilde{\varepsilon}_{33} \quad \tilde{\varepsilon}_{12} \quad \tilde{\varepsilon}_{23} \quad \tilde{\varepsilon}_{31}]^T$ and $\tilde{\sigma} = [\tilde{\sigma}_{11} \quad \tilde{\sigma}_{22} \quad \tilde{\sigma}_{33} \quad \tilde{\sigma}_{12} \quad \tilde{\sigma}_{23} \quad \tilde{\sigma}_{31}]^T$ contain six macroscopic strain and stress components, respectively.

   If we specify uniaxial tensile loading (H11 = ϵ, where ϵ is a non-zero small number, and all other components of the macroscopic displacement gradient are left empty) in Card 3, then the finite element simulation of RVE will yield a homogenized stress vector $\tilde{\sigma} = [\tilde{\sigma}_1 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0]^T$ and a homogenized strain vector $\tilde{\varepsilon} = [\tilde{\varepsilon}_1 \quad \tilde{\varepsilon}_2 \quad \tilde{\varepsilon}_3 \quad \tilde{\varepsilon}_4 \quad \tilde{\varepsilon}_5 \quad \tilde{\varepsilon}_6]^T$. These results are available in rveout. We can then calculate the first column of the macroscopic material compliance matrix as follows:

$$\tilde{C}_{11} = \tilde{\varepsilon}_1/\tilde{\sigma}_1, \quad \tilde{C}_{21} = \tilde{\varepsilon}_2/\tilde{\sigma}_1, \quad \tilde{C}_{31} = \tilde{\varepsilon}_3/\tilde{\sigma}_1,$$
$$\tilde{C}_{41} = \tilde{\varepsilon}_4/\tilde{\sigma}_1, \quad \tilde{C}_{51} = \tilde{\varepsilon}_5/\tilde{\sigma}_1, \quad \tilde{C}_{61} = \tilde{\varepsilon}_6/\tilde{\sigma}_1.$$

   Similarly, all the other macroscopic compliance coefficients can be computed by applying different uniaxial tensile and pure shear loading conditions. By inverting the compliance matrix $\tilde{C}$, we can obtain the macroscopic material stiffness matrix.

9. **Calibration of macroscale constitutive laws.** If a functional form of the macroscopic constitutive equation is available, then a series of numerical material

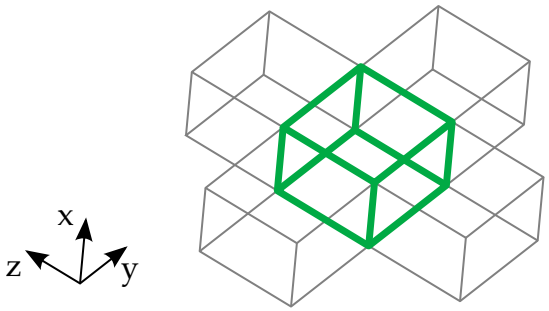


Figure 44-2. Example of image RVEs in both the y and z directions. The green lines outline the real RVE while the grey lines outline the image RVEs.

tests can be properly designed and conducted on the RVE model to identify the material parameters for the assumed constitutive model. In other words, by treating RVEs as virtual material samples, we can fit macroscale material model parameters while reducing the amount of expensive (or unfeasible) physical experiments for composite materials.

10. **De-homogenization/localization analysis.** After performing a standard macroscale structural finite element analysis in LS-DYNA, we can obtain the macroscale deformation history at any element or integration point of the macrostructure. If we convert such information to the macroscopic displacement gradient $\widetilde{\mathbf{H}} \equiv \nabla_{\mathbf{X}}\tilde{\mathbf{u}}$, we can then apply this *RVE_ANALYSIS_FEM keyword to perform the RVE localization analysis (also called de-homogenization analysis) to evaluate the actual micro-scale material responses, including the evolution and distribution of microscopic stress/strain fields within the heterogeneous RVE. These detailed microscale material responses are available in the standard LS-DYNA output files (e.g., d3plot). The multiscale simulation can provide useful guidance to both microscopic material design and macroscopic structural analysis.

11. **Image RVE.** The RVE feature in LS-DYNA generally imposes periodic boundary conditions on the RVE boundary but does not impose periodic contact constraints. For some special RVEs, such as textile RVEs, no matrix exists and the yarns in the RVE tend to untangle from other yarns during deformation. To resolve this issue, we developed the image RVE feature. When IMAGE is nonzero, image RVEs are generated automatically in the specified directions. The motion of the image RVE will follow the motion of the real RVE but with an offset distance (equal to the displacement of the control nodes in the specified direction). Specifically, the offset is defined as

$$u_I = u_R + \alpha u_c \ ,$$

where u_I is the displacement of the nodes in the image RVE, u_R is the displacement of the nodes in the real RVE, u_C is the displacement of the control nodes in a certain direction, and α is the coefficient relating the position of the image RVE. The value of α will be 1 or -1. Figure 44-2 gives an example of a real RVE

(outlined with green lines) and image RVEs in both the y and z directions (outlined with grey lines).

Example:

```
Main_RVE_input.k:

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  *RVE_ANALYSIS_FEM
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$   * This example presents a 3D RVE model of a composite material
$   * The composite material consists of two base materials
$   * Finite element mesh is provided in a separate file named “Mesh_RVE.k”
$   * PDBC with a macroscopic displacement gradient H12=0.001 is specified
$   * Macroscopic results are written out to the “rveout” file
$   * Microscopic results are written out to the “d3plot” file
$
*KEYWORD
$
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
*CONTROL_TERMINATION
$   ENDTIM    ENDCYC     DTMIN    ENDENG    ENDMAS     NOSOL
      1e-3
*CONTROL_ACCURACY
$      OSU       INN
         1         1
*CONTROL_IMPLICIT_GENERAL
$   IMFLAG       DT0
         1      1e-4
*CONTROL_IMPLICIT_SOLUTION
$   NSOLVR    ILIMIT    MAXREF     DCTOL     ECTOL     RCTOL     LSTOL    ABSTOL
                                    1e-6     1e-10                         1e-30
*CONTROL_IMPLICIT_SOLVER
$   LSOLVR    LPRINT     NEGEV     ORDER      DRCM    DRCPRM   AUTOSPC   AUTOTOL
         6         0
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
*RVE_ANALYSIS_FEM
$ MESHFILE
Mesh_RVE.k
$     INPT      OUPT      LCID      IDOF        BC    IMATCH
         0         1         1         3         0         0
$      H11       H22       H33       H12       H23       H13
                                    1e-3
$...>....1....>....2....>....3....>....4....>....5....>....6....>....7....>....8
*DEFINE_CURVE
         1
       0.0       0.0
      1e-3       1.0
*DATABASE_RVE
      1e-4
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*part
         1         6        21
*part
         2         6        22
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*MAT_ELASTIC
        21      1e-3      10.2       0.3
*MAT_ELASTIC
        22      1e-3       5.8       0.4
```

```
$---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*section_solid
$#   secid    elform
        6        10
$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Mesh_RVE.k:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$$$$  This is the mesh file accompanying *RVE_ANALYSIS_FEM
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$  * The finite element mesh represents the composite material micro-structures.
$  * For 3D models, this file should contain keywords *ELEMENT_SOLID & *NODE
$  * For 2D models, this file should contain keywords *ELEMENT_SHELL & *NODE
$  * Only a reduced input is given below
$
*KEYWORD
$
*NODE
…

*ELEMENT_SOLID
…

$
*END
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

*TERMINATION

The keyword *TERMINATION provides an alternative way of stopping the calculation before the termination time is reached. The termination time is specified on the *CONTROL_TERMINATION input and will terminate the calculation whether or not the options available in this section are active. Different types of termination may be defined:

*TERMINATION_BODY

*TERMINATION_CONTACT

*TERMINATION_CURVE

*TERMINATION_DELETED_SHELLS

*TERMINATION_DELETED_SOLIDS

*TERMINATION_FATIGUE

*TERMINATION_NODE

*TERMINATION_SENSOR

*TERMINATION_BODY

Purpose: Terminate calculation based on rigid body displacements. For *TERMINATION_BODY the analysis terminates when the center of mass displacement of the rigid body specified reaches either the maximum or minimum value (stops 1, 2 or 3) or the displacement magnitude of the center of mass is exceeded (stop 4). If more than one condition is input, the analysis stops when any of the conditions is satisfied. Termination by other means than *TERMINATION input is controlled by the *CONTROL_TERMINATION control card. Note that this type of termination is not active during dynamic relaxation.

Part Cards. Add one card for each part having termination criterion. Include as many cards as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	STOP	MAXC	MINC				
Type	I	I	F	F				
Default	none	none	-	-				

VARIABLE	DESCRIPTION
PID	Part ID of rigid body, see *PART_*OPTION*.
STOP	Stop criterion: EQ.1: global x direction, EQ.2: global y direction, EQ.3: global z direction, EQ.4: stop if displacement magnitude is exceeded.
MAXC	Maximum (most positive) displacement, options 1, 2, 3 and 4: EQ.0.0: MAXC set to 1.0e21.
MINC	Minimum (most negative) displacement, options 1, 2 and 3 above only: EQ.0.0: MINC set to -1.0e21.

*TERMINATION_CONTACT

Purpose: The analysis terminates when the magnitude of the contact interface resultant force is either zero or less than or equal to a threshold value. If more than one contact condition is input, the analysis stops when any of the conditions is satisfied. Termination by means other than *TERMINATION input is controlled by the *CONTROL_TERMINATION control card. Note that this type of termination is not active during dynamic relaxation and does not apply to 2D contact types.

Contact ID Cards. Add one card for each contact ID having a termination criterion. Include as many cards as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	CID	ACTIM	DUR	THRES	DOF			
Type	I	F	F	F	I			
Default	none	none	↓	0.0	0			

VARIABLE	DESCRIPTION
CID	Contact ID. The contact ID is defined by the ordering of the contact input unless the TITLE option which allows the CID to be defined is used in the *CONTACT section.
ACTIM	Activation time
DUR	Time duration of null resultant force prior to termination. This time is tracked only after the activation time is reached and the contact resultant forces are zero. EQ.0.0: Immediate termination after null force is detected.
THRES	Any measured force magnitude below or equal to this specified threshold is taken as a null force.
DOF	Option to consider only the force magnitude in the x, y, or z global directions corresponding to DOF = 1, 2, and 3, respectively.

*TERMINATION_CURVE

Purpose: Terminate the calculation when the load curve value returns to zero. This termination can be used with the contact option *CONTACT_AUTO_MOVE. With this option, the load curve is modified to account for the movement of the SURFB surface.

Load Curve Card. For each load curve used as a termination criterion add a card. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID	ATIME						
Type	I	F						
Default	none	↓						

VARIABLE	DESCRIPTION
LCID	Load curve ID governing termination.
ATIME	Activation time. After this time the load curve is checked. EQ.0.0: termination will occur after the load curve value becomes nonzero and then returns to zero

*TERMINATION_DELETED_SHELLS_{OPTION}

Available options include:

<BLANK>

SET

Purpose: Terminate the calculation when the number of deleted shells for a specified part ID exceeds the value defined here. This input has no effect for a part ID that is left undefined. Generally, this option should be used with the NFAIL1 and NFAIL4 parameters that are defined in the *CONTROL_SHELL control information.

When using the SET option, termination will occur when NDS elements are deleted in any one of the parts in the part set PSID.

Part (set) Cards. Include one card for each part having a termination criterion based on shell deletion. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	NDS						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
PID / PSID	Part ID or if option SET is active, part set ID.
NDS	Number of elements that must be deleted for the specified part ID's, before an error termination occurs.

***TERMINATION_DELETED_SOLIDS**_*{OPTION}*

Available options include:

<BLANK>

SET

Purpose: Terminate the calculation when the number of deleted solids for a specified part ID exceeds the value defined here. This input has no effect for a part ID that is left undefined.

When using the SET option, termination will occur when NDS elements are deleted in any one of the parts in the part set PSID.

Part (set) Cards. Include one card for each part having a termination criterion based on solid element deletion. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID/PSID	NDS						
Type	I	I						
Default	none	1						

VARIABLE	DESCRIPTION
PID/PSID	Part ID or if option SET is active, part set ID.
NDS	Number of elements that must be deleted for the specified part ID's, before an error termination occurs.

*TERMINATION_FATIGUE

Purpose: Terminate the analysis when the fatigue damage ratio reaches a predefined threshold value. Termination by means other than *TERMINATION input is controlled by *CONTROL_TERMINATION.

Part ID Cards. Add one card for each part ID having a termination criterion. Include as many cards as necessary. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	DMG						
Type	I	F						
Default	0	none						

VARIABLE	DESCRIPTION
PID	Part ID
DMG	Threshold fatigue damage ratio to cause termination

Remarks:

PID. PID is optional. When a PID is specified, the threshold fatigue damage ratio defined in this card applies only to the corresponding part. If multiple cards with nonzero PIDs are defined, the analysis terminates as soon as any one part reaches its threshold fatigue damage ratio. If PID is left blank, the threshold fatigue damage ratio applies to the entire model.

*TERMINATION_NODE

Purpose: Terminate calculation based on nodal point *coordinates*. The analysis terminates for *TERMINATION_NODE when the current position of the node specified reaches either the maximum or minimum value (stops 1, 2 or 3), or picks up force from any contact surface (stops 4). Termination by other means than *TERMINATION is controlled by the *CONTROL_TERMINATION control card. Note that this type of termination is not active during dynamic relaxation.

Node Cards. Include one card for each node having a termination criterion. Include as many cards as desired. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	STOP	MAXC	MINC				
Type	I	I	F	F				
Default	none	none	↓	↓				

VARIABLE	DESCRIPTION
NID	Node ID, see *NODE_*OPTION*.
STOP	Stop criterion: EQ.1: global *x*-direction, EQ.2: global *y*-direction, EQ.3: global *z*-direction, EQ.4: stop if node touches contact surface.
MAXC	Maximum (most positive) coordinate (options 1, 2 and 3) above only.
MINC	Minimum (most negative) coordinate (options 1, 2 and 3) above only.

*TERMINATION_SENSOR

Purpose: Terminates the calculation when the switch condition defined in *SENSOR_-SWITCH is met.

Card 1	1	2	3	4	5	6	7	8
Variable	SWID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
SWID	ID of *SENSOR_SWITCH which will terminate the calculation when its condition is met. Only one *TERMINATION_SENSOR is allowed. If more than one *TERMINATION_SENSOR is defined; only the last one is effective.

Remarks:

An example allowing more than one sensor_switch to terminate calculation:

```
*SENSOR_DEFINE_ELEMENT
$ Axial force of beam element 1
44,BEAM,1,AXIAL,FORCE
*SENSOR_DEFINE_ELEMENT
$ Axial force of beam element 2
55,BEAM,21,AXIAL,FORCE
*SENSOR_SWITCH
$a switch condition is  met when the axial force of beam-1 >5.0
11,SENSOR,44,GT,5.
*SENSOR_SWITCH
$a switch condition is  met when the axial force of beam-2 >10.0
22,SENSOR,55,GT,10.
*SENSOR_SWITCH
$ a switch condition is  met when time >50.
33,TIME, , 50
*SENSOR_SWITCH_CALC-LOGIC
$ a switch condition is met if both conditions
$ of switch-11 and switch-33 are met, I.e.,
$ axial force of beam-1>5.0 and time>50
44,11,33
*SENSOR_SWITCH_CALC-LOGIC
$ a switch condition is met if both conditions
$ of switch-22 and switch-33 are met, I.e.,
$ axial force of beam-2>10.0 and time>50
55,33,22
*SENSOR_SWITCH_CALC-LOGIC
$ a switch condition is met if the conditions
```

```
$ of switch-44 or switch-55 is met, I.e.,
$ axial force of beam-1>5.0 and time>50 or
$ axial force of beam-2>10.0 and time>50
66,44,-55
*TERMINATION_SENSOR
$ job will be terminated when the switch condition of switch-66 is met, I.e.,
$ axial force of beam-1>5.0 and time>50 or
$ axial force of beam-2>10.0 and time>50
66
```

*TITLE

Purpose: Define job title.

Card 1	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	C							
Default	LS-DYNA USER INPUT							

VARIABLE	DESCRIPTION
TITLE	Heading to appear on output and in output files.

*UNIT

The keyword *UNIT provides a coherent way to specify units in an LS-DYNA problem: Initially, it will be used to define the default units of a problem, and then the units in any LS-DYNA solver or keyword card that implements this capability. Its initial use is with the *DUALCESE solver. See Vol. III for details about those solvers. *DUALCESE uses this unit information to convert to and from external physical databases, such as the EOS libraries REFPROP and COOLPROP. Later, this information will also be used for connecting *DUALCESE with Chemkin-type chemistry databases.

*UNIT_AMOUNT

*UNIT_ANGLE

*UNIT_DEFAULTS

*UNIT_DERIVED

*UNIT_ELECTRIC_CURRENT

*UNIT_LENGTH

*UNIT_LUMINOUS_INTENSITY

*UNIT_MASS

*UNIT_SYSTEM

*UNIT_TEMPERATURE

*UNIT_TIME

WARNING: Including these cards in the same input deck with *INCLUDE_TRANSFORM with non-unity values for FCTMAS, FCTLEN, or FCTTIM is undefined. The same applies to the case of *INCLUDE_TRANSFORM with a non-blank specification for FCTTEM.

*UNIT_DEFAULTS

Purpose: Specify the user units for the current keyword input deck. These *UNIT cards are designed for when a solver and an external database to which the solver is connecting are in different units. It is also intended to be used for converting the units of data exchanged between solvers. Other applications are being planned.

Only one *UNIT_DEFAULTS card is allowed per problem.

Card 1	1	2	3	4	5	6	7	8
Variable	LENGTH	MASS	TIME	TEMP	CURRENT	AMOUNT	LUMIN	ANGLE
Type	A	A	A	A	A	A	A	A
Default	m	kg	sec	K	ampere	mole	candela	radian

VARIABLE	DESCRIPTION	
LENGTH	Length units:	
	EQ.M:	meter (default)
	EQ.KM:	kilometer
	EQ.MM:	millimeter
	EQ.CM:	centimeter
	EQ.DM:	decimeter
	EQ.UM:	micron
	EQ.NM:	nanometer
	EQ.MIL:	mil
	EQ.IN:	inch
	EQ.FT:	foot
	EQ.YD:	yard
	EQ.A:	Angstrom
	EQ.MILE:	mile
	EQ.ROD:	rod
	EQ.LIGHTYEAR:	lightyear

VARIABLE	DESCRIPTION	
MASS	Mass units:	
	EQ.KG:	kilogram (default)
	EQ.G:	gram
	EQ.CENTIG:	centigram
	EQ.MG:	milligram
	EQ.MICROG:	microgram
	EQ.OUNCE:	ounce
	EQ.POUND:	pound
	EQ.POUNDAL:	poundal
	EQ.TON:	ton
	EQ.METRICTON:	metric ton
	EQ.SHORTTON:	short ton
	EQ.SLUG:	pound × second2 / foot
	EQ.AMU:	amu
TIME	Time units:	
	EQ.SEC:	second (default)
	EQ.MILLISEC:	millisecond
	EQ.MICROSEC:	microsecond
	EQ.NANOSEC:	nanosecond
	EQ.PICOSEC:	picosecond
	EQ.MIN:	minute
	EQ.HOUR:	hour
	EQ.DAY:	day
	EQ.WEEK:	week
	EQ.MONTH:	month
	EQ.YEAR:	year
TEMP	Temperature units:	
	EQ.K:	Kelvin (default)
	EQ.C:	Celsius

VARIABLE	DESCRIPTION
	EQ.F: Fahrenheit
	EQ.R: Rankine
CURRENT	Electric current units:
	EQ.AMPERE: ampere (default)
	EQ.MA: milliampere
	EQ.ABAMP: abampere
	EQ.STATSMP: statampere
	EQ.BIOT: Biot (same as abampere)
AMOUNT	Amount units:
	EQ.MOLE: moles (default)
LUMIN	Luminosity units:
	EQ.CANDELA: candela (default)
	EQ.VIOLLE: violle
	EQ.CANDLE_GER: Candle Germany
ANGLE	Angle units:
	EQ.RADIAN: radians (default)
	EQ.DEGREE: degrees
	EQ.DEGREE_MIN: degree minute
	EQ.DEGREE_SEC: degree second
	EQ.REV: revolutions

*UNIT_DERIVED

Purpose: Derive a new unit based upon products of powers of *UNIT default units: m, kg, sec, K, ampere, mole, candela, and radian. This derived unit is then made available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	UNITNAME							
Type	A							
Remarks	2, 4							

Card 2	1	2	3	4	5	6	7	8
Variable	EXP_LENG	EXPMASS	EXP_TIME	EXPTEMP	EXPCURR	EXPAMT	EXPLUMIN	EXPANGLE
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Remarks	1	1	1	1, 3	1	1	1	1

VARIABLE	DESCRIPTION
UNITNAME	Name of derived unit
EXP_LENG	Power of length unit
EXPMASS	Power of mass unit
EXP_TIME	Power of time unit
EXPTEMP	Power of temperature unit
EXPCURR	Power of electric current unit
EXPAMT	Power of amount unit
EXPLUMIN	Power of luminosity unit

VARIABLE	DESCRIPTION
EXPANGLE	Power of angle unit

Remarks:

1. **Excluding Fundamental Unit.** If the exponent of a given fundamental unit is zero, then that unit is not part of the derived unit.

2. **Validity.** In order to define a valid derived unit, at least one of the fundamental units must have its exponent be nonzero.

3. **Temperature in Derived Unit.** If temperature is involved in deriving a new unit, it must be converting from the Kelvin (K) temperature unit (without a shift).

4. **Availability.** This derived unit is made available for use with all unit systems setup by the user.

*UNIT_AMOUNT

Purpose: Specify a new user amount unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	AMOUNT_UNIT		AMOUNT_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
AMOUNT_UNIT	Name of new amount unit
AMOUNT_SCALE	Scale factor to convert from this new unit amount to moles

Remarks:

1. **Conversion Factor.** The scale factor should be such that:

$$[\text{mole}] = \text{AMOUNT_SCALE} \times \left[\text{AMOUNT_UNIT}\right] .$$

*UNIT_ANGLE

Purpose: Specify a new user angle unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	ANGLE_UNIT		ANGLE_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
ANGLE_UNIT	Name of new angle unit
ANGLE_SCALE	Scale factor to convert from this new angle unit to radians

Remarks:

1. **Conversion Factor.** The scale factor should be such that:

$$[\text{radians}] = \text{ANGLE_SCALE} \times \left[\text{ANGLE_UNIT}\right].$$

*UNIT_ELECTRIC_CURRENT

Purpose: Specify a new user electric current unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	EC_UNIT		EC_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
EC_UNIT	Name of new electric current unit
EC_SCALE	Scale factor to convert from this new electric current unit to amperes

Remarks:

1. **Unit Conversion.** The scale factor should be such that:

$$[\text{ampere}] = \text{EC_SCALE} \times \left[\text{EC_UNIT}\right] .$$

*UNIT_LENGTH

Purpose: Specify a new user length unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	LENGTH_UNIT		LENGTH_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
LENGTH_UNIT	Name of new length unit
LENGTH_SCALE	Scale factor to convert from this new length unit to meters

Remarks:

1. **Conversion Factor.** The scale factor should be such that:

$$[\mathrm{m}] = \mathrm{LENGTH_SCALE} \times \left[\mathrm{LENGTH_UNIT}\right] .$$

*UNIT_LUMINOUS_INTENSITY

Purpose: Specify a new user luminous intensity unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	LI_UNIT		LI_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
LI_UNIT	Name of new luminous intensity unit
LI_SCALE	Scale factor to convert from this new luminous intensity unit to candela

Remarks:

1. **Unit Conversion.** The scale factor should be such that:

$$[\text{candela}] = \text{LI_SCALE} \times \left[\text{LI_UNIT}\right] .$$

*UNIT_MASS

Purpose: Specify a new user mass unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	MASS_UNIT		MASS_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
MASS_UNIT	Name of new mass unit
MASS_SCALE	Scale factor to convert from this new mass unit to kilograms

Remarks:

1. **Conversion Factor.** The scale factor should be such that:

$$[\text{kg}] = \text{MASS_SCALE} \times \left[\text{MASS_UNIT}\right] .$$

*UNIT_SYSTEM

Purpose: Specify the user units for the current keyword input deck. These *UNIT cards are designed for when a solver and an external database to which the solver is connecting are in different units. It is also intended to be used for converting the units of data exchanged between solvers. Other applications are being planned.

For the built-in options for each type of unit, refer to *UNIT_DEFAULTS.

Card 1	1	2	3	4	5	6	7	8
Variable	UNITSYS							
Type	A							

Card 2	1	2	3	4	5	6	7	8
Variable	LENGTH	MASS	TIME	TEMP	CURRENT	AMOUNT	LUMIN	ANGLE
Type	A	A	A	A	A	A	A	A
Default	m	kg	sec	K	ampere	mole	candela	radian

VARIABLE	DESCRIPTION
UNITSYS	Name of new unit system
LENGTH	Length units
MASS	Mass units
TIME	Time units
TEMP	Temperature units
CURRENT	Electric current units
AMOUNT	Amount units
LUMIN	Luminosity units
ANGLE	Angle units

*UNIT_TEMPERATURE

Purpose: Specify a new user temperature unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	TEMP_UNIT		TEMP_SCALE		TEMP_SHIFT			
Type	A		F		F			
Default	none		1.0		0.0			

VARIABLE	DESCRIPTION
TEMP_UNIT	Name of new temperature unit
TEMP_SCALE	Scale factor to convert from this new temperature unit to degrees Kelvin
TEMP_SHIFT	Shift to apply to a temperature in this new temperature unit before applying the TEMP_SCALE scaling to obtain degrees Kelvin

Remarks:

1. **Unit Conversion.** As the only unit specification that can include a shift, the new temperature unit must satisfy this relation, where [K] represents degrees Kelvin:

$$\left[\mathrm{K}\right] = \mathrm{TEMP_SCALE} \times \left(\left[\mathrm{TEMP_UNIT}\right] + \mathrm{TEMP_SHIFT} \right)$$

*UNIT_TIME

Purpose: Specify a new user time unit to make it available for use in the current keyword input deck.

Card 1	1	2	3	4	5	6	7	8
Variable	TIME_UNIT		TIME_SCALE					
Type	A		F					
Default	none		1.0					

VARIABLE	DESCRIPTION
TIME_UNIT	Name of new time unit
TIME_SCALE	Scale factor to convert from this new time unit to seconds

Remarks:

1. **Unit Conversion.** The scale factor should be such that:

$$[\text{sec}] = \text{TIME_SCALE} \times \left[\text{TIME_UNIT}\right] .$$

*USER

The *USER keyword cards are used to specify input for the user-defined subroutines, including contact and boundary conditions. They are also used to smooth history data. The available keywords in this section are given in alphabetical order:

*USER_GET_HISTORY

USER_INTERFACE_*OPTION

*USER_LOADING

*USER_LOADING_SET

*USER_NONLOCAL_SEARCH

*USER_GET_HISTORY

Purpose: Interface for gathering history data of specified parts with the user subroutine `usr_get_hsv`. The interface works for SMP/MPP and for solids, shells, and beams. It is also element agnostic, meaning any type of element can be accessed from any type of element.

Card Summary:

Card 1. This card is required.

SID	STYPE	TID	TTYPE	HTYPE			

Card 2. Include as many of this card as needed. 96 parameters can be defined (12 cards) at most.

H1	H2	H3	H4	H5	H6	H7	H8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	TID	TTYPE	HTYPE			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
SID	Source part (set) ID. History variables should be accessed from this part or part set.
STYPE	ID type of SID: EQ.0: Part ID EQ.1: Part set ID
TID	Target part (set) ID. The history variables should be used for this part or part set.

VARIABLE	DESCRIPTION
TTYPE	ID type of TID: EQ.0: Part ID EQ.1: Part set ID
HTYPE	List or range of history variables on Card 2 and onwards: EQ.0: Access history variable number H1, H2, H3, …, Hn EQ.1: Access history variables in the range H1 to H2. H3 and onwards are ignored.

Material History Variable Location Card. Specify the locations of the material history variables to be gathered. Define 8 locations per card. Include as many cards as desired, up to 12 cards. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	H1	H2	H3	H4	H5	H6	H7	H8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
H[N]	Location of N^{th} material history variable to be gathered. History includes stress and effective plastic strain in positions 1-6 and 7, respectively. If HTYPE = 1, only H1 and H2 are needed. All history variables in the range H1-H2 are then gathered.

Remarks:

*USER_GET_HISTORY activates the user subroutine `usr_get_hsv`. The subroutine can be called from any element in the target part(s) TID and can access any element in the source part(s) SID. See the subroutines `umat44` and `usergethsv_test` in file dyn21umats.F for example usage.

Call the subroutine with `call usr_get_hsv(elmid,hsvid,nip,hsvnip)`, where `elmid`, `hsvid`, and `nip` are integers and `hsvnip` is an array of reals. For a specified element number `elmid` and history variable number `hsvid`, the subroutine returns the number of integration points `nip` and the corresponding array of history data `hsvnip`. Note that the accessible history includes stress and effective plastic strain, which are stored in the first seven positions. For example, `hsvid = 7` indicates the effective plastic strain while `hsvid = 8` is first history variable in the material.

The material subroutines set the history data in each cycle, after which data is synchronized. Thus, if a user material subroutine calls `usr_get_hsv`, it returns the synchronized data from the previous cycle. This synchronization works in the same way for both MPP and SMP for consistency.

USER_INTERFACE_*OPTION

Available options include:

CONTROL

FRICTION

FORCES

CONDUCTIVITY

Purpose: Define user-defined input and allocate storage for user-defined subroutines for the contact algorithms. See also *CONTROL_CONTACT.

The CONTROL option above allows the user to take information from the contact interface for further action, such as stopping the analysis. A sample user subroutine is provided in Appendix F.

The FRICTION option may be used to modify the Coulomb friction coefficients for contact types 3, a 3, m 3, 5, a 5, m 5, 10, a 10, and m 10 (that is, *CONTACT_(AUTOMATIC/FORMING)_SURFACE_TO_SURFACE, *CONTACT_(AUTOMATIC/FORMING)_NODES_TO_SURFACE, and *CONTACT_(AUTOMATIC/FORMING)_ONE_WAY_SURFACE_TO_SURFACE) and for all Mortar contacts. Note that for the supported non-Mortar contact types, segment-to-segment contacts (meaning SOFT = 2 on Optional Card A of *CONTACT_...) are *not* supported as discussed in Remark 2. The FRICTION option is also *not* supported for 2D contacts, except in the case of Mortar contacts. A sample user-defined friction subroutine is provided in Appendix G. For the subroutine to be called, the static friction coefficient FS on Card 2 of *CONTACT must be any nonzero value, and shell thickness offsets must be invoked in the contact by setting SHLTHK to 1 or 2 using *CONTROL_CONTACT or Optional Card B in *CONTACT_.... The array length USRFRC in *CONTROL_CONTACT should be set to a value no less than the sum of the number of history variables NOC and the number of user-defined input parameters in *USER_INTERFACE_FRICTION. For Mortar contacts, the subroutine to call is `mortar_usrfrc`, which is found among the source routines in an object version.

The CONDUCTIVITY option is for defining heat transfer contact conductance properties for thermal contacts.

The FORCES option invokes collecting contact nodal forces from the specified contact ID list for user subroutines.

Card Summary:

Card 1a. Include this card if the keyword option is CONTROL, FRICTION, or CONDUCTION.

IFID	NOC	NOCI	NHSV	NEHIS	MHSV		

Card 1a.1. Use as many cards as necessary to set NOCI variables.

UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8

Card 1b. Include this card if the keyword option is FORCES.

NCONT							

Card 1b.1. Use as many cards as necessary to set NCONT variables.

CID1	CID2	CID3	CID4	CID5	CID6	CID7	CID8

Data Cards:

Include this card if the keyword option is CONTROL, FRICTION, or CONDUCTION

Card 1a	1	2	3	4	5	6	7	8
Variable	IFID	NOC	NOCI	NHSV	NEHIS	MHSV		
Type	I	I	I	I	I	I		
Default	none	none	none	0	0	0		

Initialization Cards. Use as many cards as necessary to set NOCI variables.

Card 1a.1	1	2	3	4	5	6	7	8
Variable	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
IFID	Interface number

VARIABLE	DESCRIPTION
NOC	Number of history variables for interface. The number should not exceed the length of the array defined on *CONTROL_CONTACT. See Remark 1.
NOCI	Initialize the first NOCI history variables in the input. NOCI must be smaller or equal to NOC.
NHSV	Number of history variables per interface node (only for friction and conductivity interface). For Mortar contact it is the number of history variables per SURFA segment.
NEHIS	Flag for element (material) history data provided as input quantities in subroutine `usrfrc` (see comments there). EQ.0: A special choice of element history variables is provided, namely plastic strain, yield stress, and material directions. GT.0: Plastic strain and element history variables up to NEHIS-1 are provided in original order.
MHSV	Number of history variables per SURFB segment for Mortar contact, ignored for single surface Mortar contact
UC1	First user defined input parameter
UC2	Second user defined input parameter
⋮	⋮
UC[N]	Last user defined input parameter, where N = NOCI

FORCES Card. Include this card if using the FORCES keyword option.

Card 1b	1	2	3	4	5	6	7	8
Variable	NCONT							
Type	I							
Default	none							

Contact ID Cards. Use as many cards as necessary to set NCONT contact IDs.

Card 1b.1	1	2	3	4	5	6	7	8
Variable	CID1	CID2	CID3	CID4	CID5	CID6	CID7	CID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
NCONT	Number of contact IDs. LE.0: All contacts will be used.
CID1	First contact user ID
CID2	Second contact user ID
⋮	⋮
CID[N]	Last contact user ID, where N = NCONT.

Remarks:

1. **Interface variables.** The (NOC) interface variables (of which NOCI are initialized) are passed as arguments to the user-defined subroutine. See Appendix G for the full list of arguments passed to the subroutine.

2. **Segment-to-segment contact.** This keyword is not supported by segment-to-segment contact which is invoked by setting SOFT = 2 on Optional Card A of the *CONTACT_.... It is, however, supported for Mortar contact.

*USER_LOADING

Purpose: Provide a means of applying pressure and force boundary conditions. The keyword *USER_LOADING activates this option. Input here is optional with the input being read until the next "*" keyword appears. The data read here is to be stored in a common block provided in the user subroutine, LOADUD. This data is stored and retrieved from the restart files.

Parameter Cards. Add one card for each input parameter. Include as many cards as needed. This input ends at the next keyword ("*") card.

Card 1…	1	2	3	4	5	6	7	8
Variable	PARM1	PARM2	PARM3	PARM4	PARM5	PARM6	PARM7	PARM8
Type	F	F	F	F	F	F	F	F
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
PARM[N]	This is the N^{th} user input parameter.

*USER_LOADING_SET

Purpose: Provides a means to apply user-defined loading to a set of nodes or segments. Loading could be nodal force, body force, temperature distribution, and pressure on segment or beam.

Set Cards. Add a card for each set to which a load is applied. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	LTYPE	LCID	CID	SF1	SF2	SF3	IDULS
Type	I	A	I	I	F	F	F	I
Default	none	none	none	global	none	none	none	Seq. #

VARIABLE	DESCRIPTION
SID	ID of the set to which user-defined loading will be applied. Set type depends on the type of loading, see LTYPE.
LTYPE	Loading type: EQ."FORCEN": Force a will be applied to node set SID. The load is to be given in units of force. EQ."BODYFN": Body force density will be applied to node set SID. The load is to be given in units of force per volume. EQ."TEMPTN": Temperature will be assigned to node set SID. This option cannot be coexist with *LOAD_THERMAL_VARIABLE. In other word, users can only use either this option or *LOAD_THERMAL_VARIABLE to specify temperature distribution, not both of them, EQ."PRESSS": Pressure will be applied to segment set SID. The load is to be given in units of force per area. EQ."PRESSB": Pressure in units of force per length will be applied to beam set SID.
LCID	Load curve, a function of time. Its current value, crv, is passed to user subroutine LOADSETUD.

VARIABLE	DESCRIPTION
CID	Optional coordinate system along which scale factors SF_i is defined. Global system is the default system.
SF[i]	Scale factor of loading magnitude, when LTYPE
	LTYPE.EQ."FORCEN": SFi is the factor along i^{th} direction of CID. For example, set SF1 = 1. and the others to zero if the load is to be applied in the positive x-direction. This applies whether the global or a local coordinate system is used.
	LTYPE.EQ."BODYFN": See "EQ.FORCEN"
	LTYPE.EQ."PRESSS": SF1 is used as the scale factor, SF2 and SF3 are ignored,
	LTYPE.EQ."PRESSB": Scale factor along r, s, t axis of beam.
IDULS	Each USER_LOADING_SET can be assigned a unique ID, which is passed to user subroutine LOADSETUD and allows multiple loading definitions by using a single user subroutine, LOADSETUD. If no value is input, LS-DYNA will assign a sequence number to each USER_LOADING_SET based on its definition sequence.

Remarks:

*USER_LOADING_SET activates the loading defined in user subroutine LOADSETUD, part of dyn21.F. When both *USER_LOADING_SET and *USER_LOADING are defined, *USER_LOADING is only used to define user-defined parameters, PARM*n*; not to activate user subroutine LOADUD. Therefore only loading defined in LOADSETUD will be applied.

More than one loading definitions can be defined and assigned a unique ID, that enables multiple loading to be taken care of by a single subroutine, LOADSETUD, as shown below:

```
subroutine loadsetud(time,lft,llt,crv,iduls,parm)
c
c     Input (not modifiable)
c       x   : coordinate of node or element center
c       d   : displacement of node or element center
c       v   : velocity of node or element center
c       temp: temperature of node or element center
c       crv : value of LCID at current time
c     isuls : id of user_loading_set
c     parm: parameters defined in *USER_LOADING
c     Output (defined by user)
```

```
c       udl : user-defined load value
      include 'nlqparm'
C_TASKCOMMON (aux8loc)
      common/aux8loc/
     & x1(nlq),x2(nlq),x3(nlq),v1(nlq),v2(nlq),v3(nlq),
     & d1(nlq),d2(nlq),d3(nlq),temp(nlq),udl(nlq),tmp(nlq,12)
c
c    sample code
c    if (iduls.eq.100) then
c      do i=lft,llt
c       your code here
c        udl(i)=.........
c      enddo
c    elseif (iduls.eq.200) then
c      do i=lft,llt
c        udl(i)=.........
c      enddo
c    endif
      return
      end
```

*USER_NONLOCAL_SEARCH

Purpose: Interface for gathering the history data of specified elements that surround an element ("averaged" element) to average (or smooth) the history data of that element. The surrounding elements are determined using a user defined strategy. The type of averaging is also user specified. This keyword only works for solid elements. Note that all the averaged elements *must* be the *same material* and all the surrounding elements *must* also be the *same material*, but the averaged elements and the surrounding elements may be *different materials* from each other.

Card Summary:

Card 1. This card is required.

SSID	ASID	STYPE	ATYPE	R	SF1	SF2	SF3

Card 2. This card is required.

NFREQ	VOLTYPE	UTYPE	NUCONST	NUELHSV			

Card 2.1. Define ceil(NUCONST/8) of this card. NUCONST can not exceed 48, so only a maximum of 6 of this card can be defined.

P1	P2	P3	P4	P5	P6	P7	P8

Card 2.2. Define ceil(NUELHSV/8) of this card.

U1	U2	U3	U4	U5	U6	U7	U8

Card 3. Include this card as many times as needed. The next keyword ("*") card ends this input.

H1	H2	H3	H4	H5	H6	H7	H8

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	SSID	ASID	STYPE	ATYPE	R	SF1	SF2	SF3
Type	I	I	I	I	F	F	F	F
Default	none	none	none	none	none	1.0	1.0	1.0

VARIABLE	DESCRIPTION
SSID	Surrounding elements set ID (note that the surrounding elements can include averaged elements)
ASID	Averaged elements set ID
STYPE	ID type of SSID: EQ.0: Part set ID EQ.1: Part ID
ATYPE	ID type of ASID: EQ.0: Part set ID EQ.1: Part ID EQ.2: Solid set ID EQ.3: Solid ID
R	Search distance from the center of the averaged solid element
SF*i*	Scale factor for each search (**a**, **b**, **c**) direction (Default = 1.0).

Card 2	1	2	3	4	5	6	7	8
Variable	NFREQ	VOLTYPE	UTYPE	NUCONST	NUELHSV			
Type	I	I	I	I	I			
Default	1	1	none	none	none			

VARIABLE	DESCRIPTION
NFREQ	Number of cycles for collecting the history data (Default = 1).
VOLTYPE	Search geometry: EQ.1: Global axes (Default), EQ.2: Material axes (*ELEMENT_SOLID_ORTHO only)
UTYPE	User specified function type

VARIABLE	DESCRIPTION
NUCONST	Number of user defined parameters. NUCONST is limited to a maximum of 48.
NUELHSV	Number of user element history variables to be gathered. This is only needed if applied to user defined elements.

User Define Parameter Card. Define NUCONST parameters with this card. Include ceil(NUCONST/8) of this card. Only 48 parameters at most can be defined (6 cards).

Card 2.1	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
P[N]	N^{th} user defined parameter (48 maximum)

User Element History Variable Location Card. Specify the locations of the user element history variables to be gathered. This card is only needed if applied to user defined elements. Define NUELHSV locations with this card. Include ceil(NUELHSV/8) of this card.

Card 2.2	1	2	3	4	5	6	7	8
Variable	U1	U2	U3	U4	U5	U6	U7	U8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
U[N]	Location of N^{th} user element history variable to be gathered

Material History Variable Location Card. Specify the locations of the material history variables to be gathered. Define 8 locations per card. Include as many cards as desired. This input ends at the next keyword ("*") card.

Card 3	1	2	3	4	5	6	7	8
Variable	H1	H2	H3	H4	H5	H6	H7	H8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
H[N]	Location of N^{th} material history variable to be gathered

Remarks:

*USER_NONLOCAL_SEARCH activates the user subroutines USER_NUNONL and USER_NUNONL_SMOOTH which are a part of dyn21.F. This keyword can be defined more than once. Each definition is assigned a unique ID which enables multiple instances of nonlocal data to be taken care of by a single user interface (see below). Each instance can use different history variables and have different averaging functions. The subroutine below indicates how this keyword interacts with the user subroutines.

```
      subroutine user_nunonl(nnon,ictl,rctl,auxvec,lochvh,
     . ixh,nwcon,x,iuhhv,uehh)

c nnon       : number of user defined *user_nonlocal_search cards
c
c Element information (I indicates integers and H memory pointers):
c ictl(1)    : I, set ID
c ictl(2)    : I, number of averaged solid elements
c ictl(3)    : H, 1:nelems                    , solid internal sorted ID
c                 nelems+1:nelems+1+(nelems+1), range of neighorbing elements
c ictl(14)   : H, neighboring element list in the local list
c ictl(15)   : H, number of surrounding solid elements under this card and its
c                 solid internal ID
c ictl(20)   : I, location of last local solid element
c ictl(25)   : H, user ID of the surrounding element (>0 exist on current processor)
c                                                    (<0 on remote processor)
c ictl(27)   : I, nip
c ictl(28)   : H, element fail flag (0 = deleted)
c ictl(29)   : H, remote element connectivity ixhrmt(8,*)
c ictl(32)   : H, remote node UID
c ictl(33)   : H, remote xyz

c Subroutine parameters:
c ictl(8)    : I, user defined function flag
c ictl(9)    : H, user defined constants (max 48), -1 means none

c Material history variables:
c ictl(4)    : I, number of requested material history variables, nhisv
c ictl(5)    : H, location list of requested material history variables
c ictl(26)   : I, total no.  of user material history variables available
```

```
c ictl(16)   : H, array of requested material history variables for surrounding
elements
c ictl(17)   : H, temporary working array (length nip*nhisv)
c
c User element history variables:
c ictl(34)    : I, number of requested user element history variables, nhisvue
c ictl(35)    : H, location list of requested user element history variables
c ictl(36)    : I, total no.  of user element history variables available
c ictl(37)    : H, array of requested user element history variables for surrounding
elements
c ictl(38)    : H, temporary working array (length nhisvue)

      do ii=1,nnon
        nfreq  = ictl(6,ii)
        iufnc  = ictl(8,ii)
        if(mod(ncycle,nfreq).ne.0) cycle

c       Retrieve user defined constants
        if(ictl(9,ii).ge.0) then
          ncnst = memh_length(ictl(9,ii))
          pcnst = memh_ptr(ictl(9,ii))
        endif

        ids=ictl(1,ii)
        nelems = ictl(2,ii)
        peid   = memh_ptr(ictl(3,ii))
        ped1   = peid + nelems
        nhisv  = ictl(4,ii)
        nhisvue  = ictl(34,ii)
        plst   = memh_ptr(ictl(5,ii))
        plstue = memh_ptr(ictl(35,ii))

        pelst  = memh_ptr(ictl(14,ii))
        peslt  = memh_ptr(ictl(15,ii))
        nsld   = memh_length(ictl(15,ii))
        phst   = memh_ptr(ictl(16,ii))
        phstue = memh_ptr(ictl(37,ii))
        ptmp   = memh_ptr(ictl(17,ii))
        ptmpue = memh_ptr(ictl(38,ii))
        pseid  = memh_ptr(ictl(25,ii))
        pfail  = memh_ptr(ictl(28,ii))

        max_lo = ictl(20,ii)
        pixh   = memh_ptr(ictl(29,ii))
        pnid   = memh_ptr(ictl(32,ii))
        pxrmt  = memh_ptr(ictl(33,ii))

        nmtcon = ictl(26,ii)
        nmtconue = ictl(36,ii)
        nip    = ictl(27,ii)

c       Collect all the history values into local sorted storage
        call user_nunonl_smooth(nelems,i_mem(peid),i_mem(ped1), nhisv,
     $       nhisvue,i_mem(pelst),i_mem(peslt),i_mem(phst),i_mem(phstue)
     $       ,i_mem(plst),i_mem(plstue), i_mem(ptmp),i_mem(ptmpue)
     $       ,i_mem(pseid) ,auxvec,lochvh,nmtcon ,nmtconue ,nip,
     $       i_mem(pfail),ixh ,nwcon,x,max_lo ,i_mem(pixh) ,i_mem(pnid),
     $       i_mem(pxrmt),iuhhv,uehh)
      enddo

      return
      end

      subroutine user_nunonl_smooth(nelems,neid,nrang, nhisv,nhisvue
     $     ,nlist,nsrt,histv,histvue,list,listue,htmp,htmpue,uid,auxvec
     $     ,lochvh ,nmtcon,nmtconue ,nip, ifail,ixh,nwcon,x,lstsld,ixhr
```

```
     $     ,nid,xrmt ,iuhhv ,uehh)


c  Element data:
c  nelems: number of averaged solid elements
c  neid  : the isolid internal sorted ID
c          (lqfinvf(ie,2) to get solid user ID)
c  nrang : nrang(ii):nrang(ii+1)-1 range of neighboring element for ii_th element
c  nlist : surrounding element in packed sorted list for this group
c  nsrt  : surrounding element array to convert "nlist" to internal sorted element ID

c  Material history variables for surrounding elements:
c  list  : list of requested material history variables
c  histv : array of requested material history vars.  of all surrounding elements
c  htmp  : working array for material history
c
c  User element history variables for surrounding elements:
c  listue  : list of user requested user element history variables
c  histvue : array of requested user element history vars.  of all surrounding
elements
c  htmpue  : working array for user element history
c
c  Material history variables for averaged elements:
c  auxvec: ls-dyna history variables storage
c          1-7: sxx, syy, szz, sxy, syz, sxz, plastic strain
c  lochvh: starting point of the history variable for ie_th element
c
c  User element history variables for averaged elements:
c  iuhhv: starting point of the history variable for ie_th element
c  uehh:  ls-dyna ueser element history variables storage
c
c  Conversion from internal to User number:
c  lqfinvf : convert internal sorted element ID to user ID
c  lqfmiv  : convert internal part ID to user ID
c
c  Element connectivity and nodal coordinates:
c  local  element uid(je)>0:  ixh (2:9,jje), x   (1:3,ixh(2:9,jje));  jje=nsrt(je)
c  remote element uid(je)<0:  ixhr(1:8,lje), xrmt(1:3,ixhr(1:8,lje)); lje=je-lstsld

      DO II=1,NELEMS

c       Averaged eid
        ie = neid(ii)
        if(ixh(1,ie).eq.0) cycle

c       Reset working arrays
        do ip=1,nip
          do k=1,nhisv
            htmp(k,ip) = 0.
          enddo
        enddo
        do k=1,nhisvue
          htmpue(k) = 0.
        enddo

c       ----------------------------------
c       Begin operations of neighboring data

c       Range of elements
        nstr = nrang(ii)
        nend = nrang(ii+1)-1
        do j=nstr,nend
c         Element internal sorted for this group:
c           je sorted ID for this group
c           jje internal sorted ID for this element
          je=nlist(j)
```

```
          if(ifail(je).eq.0) cycle
          jje=nsrt(je)
          do ip=1,nip

c           Get material history values
            do k=1,nhisv
c             Example: sum for average
              htmp(k,ip) = htmp(k,ip) + histv(k,ip,je)
            enddo

          enddo

c         Get user element history values
          do k=1,nhisvue
c            Example: sum for average
             htmpue(k) = htmpue(k)+histvue(k,je)
          enddo

        enddo

c       End operations of neighboring data
c       -----------------------------------

c       -----------------------------------
c       Begin operations on target element

c       Set material history values
        lav=lochvh(ie)-1
        do ip=1,nip
          do k=1,nhisv
            ipos = list(k)+7+(ip-1)*nmtcon
c           Example: average over both averaged elements and surrounding elements
            auxvec(lav+ipos) = (auxvec(lav+ipos) + htmp(k,ip))/(nend
     $           -nstr+2)
          enddo
        enddo

c       Set user element history values
        lav = iuhhv(ie)-1
        do k=1,nhisvue
           ipos = listue(k)
c          Example: average over both averaged elements and surrounding elements
           uehh(lav+ipos) = (uehh(lav+ipos) + htmpue(k))/(nend
     $           -nstr+2)
        enddo

c       End operations on target element
c       -----------------------------------

      ENDDO

      return
      end
```

Restart Input Data

In general three categories of restart actions are possible with LS-DYNA and are outlined in the following discussion:

1. A simple restart occurs when LS-DYNA was interactively stopped before reaching the termination time. Then, by specifying the R=**rtf** command line option on the execution line, LS-DYNA restarts the calculation from the termination point. The calculation will pick up at the specified termination time. See Execution Syntax in the Getting Started section. No additional input deck is required.

2. For small modifications of the input deck during the restart run, LS-DYNA offers a "small restart" capability which can

   a) reset termination time,

   b) reset output printing interval,

   c) reset output plotting interval,

   d) delete contact surfaces,

   e) delete elements and parts,

   f) switch deformable bodies to rigid,

   g) switch rigid bodies to deformable,

   h) change damping options.

   All modifications to the problem made with the restart input deck will be reflected in subsequent restart dumps. All the members of the file families are consecutively numbered beginning from the last member prior to termination.

   For a small restart run a *small input deck* replaces the standard input deck on the execution line which must have at least the following command line arguments:

   LS-DYNA I=**restartinput** R=**D3DUMP*nn***

   where **D3DUMP*nn*** (or whatever name is chosen for the family member) is the *n*th restart file from the last run where the data is taken. LS-DYNA automatically detects that a small input deck is used since the I=**restartinput** file may contain *only* the *restart* keywords (excluding *STRESS_INITIALIZATION):

   CHANGE_*OPTION

*CONTROL_DYNAMIC_RELAXATION

*CONTROL_SHELL

*CONTROL_TERMINATION

*CONTROL_TIMESTEP

*DAMPING_GLOBAL

DATABASE_*OPTION

DATABASE_BINARY_*OPTION

DELETE_*OPTION

*INTERFACE_SPRINGBACK_LSDYNA

RIGID_DEFORMABLE_*OPTION

STRESS_INITIALIZATION_*{OPTION} (full restart only)

TERMINATION_*OPTION

*TITLE

*KEYWORD

*CONTROL_CPU

DEFINE_*OPTION

SET_*OPTION

You must take care to avoid nonphysical modifications to the input deck; otherwise, complete nonsense may be the result.

3. If many modifications are desired, a *full restart* may be the appropriate choice. A full restart is selected by including a full model along with a *STRESS_INITIALIZATION keyword card and possibly other restart cards. As mentioned in the Restart Analysis subsection of the Getting Started portion of the manual, either all parts or some subset of parts can be made for the stress initialization.

Remarks:

1. In a full restart, only those nodes and elements defined in the full restart deck will be present in the analysis after the full restart is initiated. But as a

convenience, any of those nodes or elements can be deleted using the *DELETE command.

2. In a small restart, velocities of nodes come from the dump file by default, but those velocities can be changed using *CHANGE_VELOCITY_....

3. In a full restart, velocities of pre-existing nodes come from the dump file by default, but those velocities can be changed using *CHANGE_VELOCITY_.... To set the starting velocities for new nodes in a full restart, use *INITIAL_VELOCITY_....

4. Pre-existing contacts, in general, carry forward seamlessly using data from the d3dump (or d3full if MPP) database. It is important that the contact ID(s) in the full restart input deck match the contact ID(s) in the original input deck if the intent is for the contacts to be initialized using data from the d3dump/d3full database. EXCEPTION: In the special case of MPP, a *CONTACT_AUTOMATIC_GENERAL contact in the full restart input deck is treated as a brand new contact and is not initialized using data from d3full.

5. Parameters used in a restart input deck must be defined in that same restart input deck using *PARAMETER.

6. Only sets using element IDs and node IDs are permitted in a small restart deck; part IDs are not recognized. Sets referenced by other commands in a small restart deck must be defined in the small restart deck.

***CHANGE**_*OPTION*

Purpose: Change solution options.

Available options include:

BOUNDARY_CONDITION

CONTACT_SMALL_PENETRATION

CURVE_DEFINITION

OUTPUT

RIGIDWALL_GEOMETRIC

RIGIDWALL_PLANAR

RIGID_BODY_CONSTRAINT

RIGID_BODY_INERTIA

RIGID_BODY_STOPPERS

STATUS_REPORT_FREQUENCY

THERMAL_PARAMETERS

VELOCITY

VELOCITY_GENERATION

VELOCITY_NODE

VELOCITY_RIGID_BODY

VELOCITY_ZERO

Boundary Condition Cards. This card 1 format is for the ***BOUNDARY_CONDITION*** keyword option. Add one card for each boundary condition. This card imposes *additional* boundary conditions. It does not remove previously imposed conditions (for example, this option will not free a fixed node). This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	BCC						
Type	I	I						

VARIABLE	DESCRIPTION
NID	Nodal point ID, see also *NODE.
BCC	New translational boundary condition code: EQ.1: Constrained x displacement, EQ.2: Constrained y displacement, EQ.3: Constrained z displacement, EQ.4: Constrained x and y displacements, EQ.5: Constrained y and z displacements, EQ.6: Constrained z and x displacements, EQ.7: Constrained x, y, and z displacements.

Small Penetration Check Cards. This Card 1 format is for the ***CONTACT_SMALL_PENETRATION*** keyword option. Set one value for each contact surface ID where the small penetration check is to be turned on. The input terminates at the next keyword ("*") card. See the PENCHK variable in *CONTACT.

Card 1	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
IDn	Contact ID for surface number n

Load Curve Redefinition Cards. This Card 1 format is for the ***CURVE_DEFINITION*** keyword option. *The new load curve must contain the same number of points as the curve it replaces.* The curve should be defined according to the *DEFINE_CURVE section of the manual. This input terminates at the next keyword ("*") card. Offsets and scale factors are ignored.

Card 1	1	2	3	4	5	6	7	8
Variable	LCID							
Type	I							

VARIABLE	DESCRIPTION
LCID	Load curve ID

ASCII Output Overwrite Card. This format applies to the ***OUTPUT*** keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	IASCII							
Type	I							

VARIABLE	DESCRIPTION
IASCII	Flag to control manner of outputting ASCII data requested by *DATABASE_*OPTION* commands in a full restart deck: EQ.0: Full restart overwrites existing ASCII output (default), EQ.1: Full restart appends to existing ASCII output.

Rigidwall Modification. The format for the ***RIGIDWALL_GEOMETRIC*** and ***RIGIDWALL_PLANAR*** cards is identical to the original cards (see *RIGIDWALL_GEOMETRIC and *RIGIDWALL_PLANAR), however there are restrictions on the entries that may be changed: only those entries that define the size and orientation of the rigid walls may be changed, but not any of the others (e.g., the type). A rigidwall may only be modified when doing a full restart.

Rigid Body Constraint Modification Cards. This format for Card 1 applies to the ***RIGID_BODY_CONSTRAINT*** keyword option. This option can change translation and rotational boundary condition on a rigid body. This input ends at the next keyword ("*") card. See CONSTRAINTED_RIGID_BODIES.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	TC	RC					
Type	I	I	I					

VARIABLE	DESCRIPTION
PID	Part ID, see *PART.

VARIABLE	DESCRIPTION
TC	Translational constraint: EQ.0: No constraints, EQ.1: Constrained x displacement, EQ.2: Constrained y displacement, EQ.3: Constrained z displacement, EQ.4: Constrained x and y displacements, EQ.5: Constrained y and z displacements, EQ.6: Constrained z and x displacements, EQ.7: Constrained x, y, and z displacements.
RC	Rotational constraint: EQ.0: No constraints, EQ.1: Constrained x rotation, EQ.2: Constrained y rotation, EQ.3: Constrained z rotation, EQ.4: Constrained x and y rotations, EQ.5: Constrained y and z rotations, EQ.6: Constrained z and x rotations, EQ.7: Constrained x, y, and z rotations.

Card sets for *RIGID_BODY_INERTIA* keyword option. This option supports changing the mass and inertia properties of a rigid body. Include as many pairs of the following two cards as necessary. This input ends at the next keyword ("*") card. The inertia tensor is specified relative to the coordinate system set in *MAT_RIGID at the start of the calculation, which is fixed in the rigid body and tracks the rigid body rotation.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	PID	TM					
Type	I	I	F					

Card 2	1	2	3	4	5	6	7	8
Variable	IXX	IXY	IXZ	IYY	IYZ	IZZ		
Type	F	F	F	F	F	F		

VARIABLE	DESCRIPTION
ID	ID for this change inertia input.
PID	Part ID, see *PART.
TM	Translational mass.
IXX	I_{xx}, xx component of inertia tensor.
IXY	I_{xy}
IXZ	I_{xz}
IYY	I_{yy}
IYZ	I_{yz}
IZZ	I_{zz}

Card sets for the *RIGID_BODY_STOPPERS* keyword option. This option is for redefining existing stoppers. Include as many pairs of cards as necessary. This input terminates when the next keyword ("*") card is encountered. See *CONSTRAINED_RIGID_-BODY_STOPPERS. Note that new stopper definitions cannot be introduced in this section. Existing stoppers can be modified.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LCMAX	LCMIN	PSIDMX	PSIDMN	LCVMNX	DIR	VID
Type	I	I	I	I	I	I	I	I
Default	required	0	0	0	0	0	required	0

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH						
Type	F	F						
Default	0.0	10^{28}						

VARIABLE	DESCRIPTION
PID	Part ID of lead rigid body, see *PART.
LCMAX	Load curve ID defining the maximum coordinate as a function of time: EQ.0: No limitation of the maximum displacement. New curves can be defined by the *DEFINE_CURVE within the present restart deck. (Not applicable for small deck restart).
LCMIN	Load curve ID defining the minimum coordinate as a function of time: EQ.0: No limitation of the minimum displacement. New curves can be defined by the *DEFINE_CURVE within the present restart deck. (Not applicable for small deck restart).
PSIDMX	Optional part set ID of rigid bodies that are constrained in the maximum coordinate direction to the lead rigid body. This option requires additional input by the *SET_PART definition.
PSIDMN	Optional part set ID of rigid bodies that are constrained in the minimum coordinate direction to the lead rigid body. This option requires additional input by the *SET_PART definition.
LCVMNX	Load curve ID which defines the maximum absolute value of the velocity that is allowed within the stopper: EQ.0: No limitation of the maximum velocity
DIR	Direction stopper acts in: EQ.1: x-translation EQ.2: y-translation EQ.3: z-translation

VARIABLE	DESCRIPTION
	EQ.4: arbitrary, defined by vector VID
	EQ.5: x-axis rotation
	EQ.6: y-axis rotation
	EQ.7: z-axis rotation
	EQ.8: Arbitrary, defined by vector VID
VID	Vector for arbitrary orientation of stopper. The vector must be defined by a *DEFINE_VECTOR within the present restart deck.
BIRTH	Time at which stopper is activated
DEATH	Time at which stopper is deactivated

Remarks:

The optional definition of part sets in minimum or maximum coordinate directions allows the motion to be controlled in an arbitrary direction.

D3HSP Interval Change Card. This card format applies to the ***STATUS_REPORT_FREQUENCY*** keyword option.

Card 1	1	2	3	4	5	6	7	8
Variable	IKEDIT							
Type	I							

VARIABLE	DESCRIPTION
IKEDIT	Problem status report interval steps in the D3HSP output file:
	EQ.0: Interval remains unchanged.

Card set for the *THERMAL_PARAMETERS* keyword option. This option is for changing the parameters used by a thermal or coupled structural/thermal analysis. See *CONTROL_THERMAL. Add the two following cards to the deck (they do not repeat).

Card 1	1	2	3	4	5	6	7	8
Variable	TS	DT	TMIN	TMAX	DTEMP	TSCP		
Type	I	F	F	F	F	F		

Card 2	1	2	3	4	5	6	7	8
Variable	REFMAX	TOL						
Type	I	F						

VARIABLE	DESCRIPTION
TS	Thermal time step code: EQ.0: No change, EQ.1: Fixed time step, EQ.2: Variable time step.
DT	Thermal time step on restart: EQ.0: No change.
TMIN	Minimum thermal time step: EQ.0: No change.
TMAX	Maximum thermal time step: EQ.0: No change.
DTEMP	Maximum temperature change in a thermal time step: EQ.0: No change.
TSCP	Time step control parameter (0.0 < TSCP < 1.0): EQ.0: No change.

VARIABLE	DESCRIPTION
REFMAX	Maximum number of reformations per thermal time step: EQ.0: No change.
TOL	Non-linear convergence tolerance: EQ.0: No change.

Node Set Velocity Card Sets. The formats for Cards 1 and 2 apply to the ***VELOCITY*** and ***VELOCITY_ONLY*** keyword options. These options are for setting velocity fields on node sets at restart. For each node set add one pair of the following cards. This input ends at the next keyword ("*") card. Undefined nodes (not listed on a set velocity card) will have their nodal velocities reset to zero if a *CHANGE_VELOCITY definition is encountered in the restart deck. However, if any of the *CHANGE_VELOCITY definitions have the keyword option ONLY appended, then only the specified nodes will have their nodal velocities modified.

Card 1	1	2	3	4	5	6	7	8
Variable	NSID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	VX	VY	VZ	VXR	VYR	VZR		
Type	F	F	F	F	F	F		
Default	0.	0.	0.	0.	0.	0.		

VARIABLE	DESCRIPTION
NSID	Nodal set ID containing nodes for initial velocity (see Remark 1)
VX	Velocity in x-direction

VARIABLE	DESCRIPTION
VY	Velocity in y-direction
VZ	Velocity in z-direction
VXR	Rotational velocity about the x-axis
VYR	Rotational velocity about the y-axis
VZR	Rotational velocity about the z-axis

Remarks:

1. **Multiple Velocity Initializations for a Node.** If a node is initialized on more than one input card set, then the last set input will determine its velocity, unless it is specified on a *CHANGE_VELOCITY_NODE card.

2. **Undefined Nodes.** Undefined nodes will have their nodal velocities set to zero if a *CHANGE_VELOCITY definition is encountered in the restart deck.

3. **Zero Velocity.** If both *CHANGE_VELOCITY and *CHANGE_VELOCITY_ZERO cards are defined, then all velocities will be reset to zero.

Velocity Generation Cards. The velocity generation cards for the ***VELOCITY_GENERATION*** option are identical to the standard velocity generation cards (see *INITIAL_VELOCITY_GENERATION), and all parameters may be changed.

Nodal Point Velocity Cards. This format applies to the ***VELOCITY_NODE*** and ***VELOCITY_NODE_ONLY*** keyword options. These option support changing nodal velocities. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	NID	VX	VY	VZ	VXR	VYR	VZR	
Type	I	F	F	F	F	F	F	
Default	none	0.	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
NID	Node ID

VARIABLE	DESCRIPTION
VX	Translational velocity in x-direction
VY	Translational velocity in y-direction
VZ	Translational velocity in z-direction
VXR	Rotational velocity about the x-axis
VYR	Rotational velocity about the y-axis
VZR	Rotational velocity about the z-axis

Remarks:

1. **Undefined Nodes.** Undefined nodes (not listed on a point velocity card) will have their nodal velocities reset to zero if a *CHANGE_VELOCITY_NODE definition is encountered in the restart deck. However, if any of the *CHANGE_-VELOCITY or CHANGE_VELOCITY_NODE definitions have ONLY appended, then only the specified nodes will have their nodal velocities modified.

2. **Multiple Velocity Initializations for a Node.** If a node is initialized on more than one input card set, then the last set input will determine its velocity, unless it is specified on a *CHANGE_VELOCITY_NODE card.

3. **Zero Velocity.** If both *CHANGE_VELOCITY and *CHANGE_VELOCITY_ZERO cards are defined, then all velocities will be reset to zero.

Rigid Body Velocity Cards. This Card 1 format applies to the ***VELOCITY_RIGID_BODY*** keyword option. This option allows for setting the velocity components of a rigid body at restart. Include as many of these cards as desired. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	VX	VY	VZ	VXR	VYR	VZR	
Type	I	F	F	F	F	F	F	
Default	none	0.	0.	0.	0.	0.	0.	

VARIABLE	DESCRIPTION
PID	Part ID of rigid body
VX	Translational velocity in x-direction
VY	Translational velocity in y-direction
VZ	Translational velocity in z-direction
VXR	Rotational velocity about the x-axis
VYR	Rotational velocity about the y-axis
VZR	Rotational velocity about the z-axis

Remarks:

1. **Rotational Velocities.** Rotational velocities are defined about the center of mass of the rigid body.

2. **Undefined Rigid Bodies.** Rigid bodies not defined in this section will not have their velocities modified.

Restarting the Model at Rest. The ***VELOCITY_ZERO*** option resets the velocities to zero at the start of the restart. There are no data cards associated with *CHANGE_VELOCITY_ZERO.

*CONTROL_DYNAMIC_RELAXATION

Purpose: Define controls for dynamic relaxation.

Card 1	1	2	3	4	5	6	7	8
Variable	NRCYCK	DRTOL	DRFCTR	DRTERM	TSSFDR	IRELAL	EDTTL	IDRFLG
Type	I	F	F	F	F	I	F	I
Default	250	0.001	0.995	∞	TSSFAC	0	0.0	0
Remarks	1	1	1	1	1			1

VARIABLE	DESCRIPTION
NRCYCK	Number of iterations between convergence checks, for dynamic relaxation option
DRTOL	Convergence tolerance for dynamic relaxation option
DRFCTR	Dynamic relaxation factor
DRTERM	Optional termination time for dynamic relaxation. Termination occurs at this time or when convergence is attained.
TSSFDR	Scale factor for computed time step during dynamic relaxation. If zero, the value is set to TSSFAC defined on *CONTROL_TERMINATION. After converging, the scale factor is reset to TSSFAC.
IRELAL	Automatic control for dynamic relaxation option based on algorithm of Papadrakakis [1981]: EQ.0: not active, EQ.1: active.
EDTTL	Convergence tolerance on automatic control of dynamic relaxation.
IDRFLG	Dynamic relaxation flag for stress initialization: EQ.0: not active, EQ.1: dynamic relaxation is activated.

Remarks:

1. **Restart before Dynamic Relaxation Convergence.** If a dynamic relaxation relaxation analysis is being restarted at a point before convergence was obtained, then NRCYCK, DRTOL, DRFCTR, DRTERM and TSSFDR will default to their previous values, and IDRFLG will be set to 1.

2. **Restart from Normal Transient Analysis.** If dynamic relaxation is activated after a restart from a normal transient analysis, LS-DYNA continues the output of data as it would without the dynamic relaxation being active. This is unlike the dynamic relaxation phase at the beginning of the calculation when a separate database is not used. Only load curves that are flagged for dynamic relaxation are applied after restarting.

*CONTROL_SHELL

Purpose: Change failure parameters NFAIL1 and NFAIL4 if necessary. These parameters must be nonzero in the initial run.

Card 1	1	2	3	4	5	6	7	8
Variable								
Type								

Card 2	1	2	3	4	5	6	7	8
Variable						NFAIL1	NFAIL4	PSNFAIL
Type						I	I	I

VARIABLE	DESCRIPTION
NFAIL1	Flag to check for highly distorted under-integrated shell elements, print a message, and delete the element or terminate. Generally, this flag is not needed for one point elements that do not use the warping stiffness. A distorted element is one where a negative jacobian exists within the domain of the shell, not just at integration points. The checks are made away from the integration points to enable the bad elements to be deleted before an instability leading to an error termination occurs. This test will increase CPU requirements for one point elements. EQ.1: Print message and delete element. EQ.2: Print message, write d3dump file, and terminate GT.2: Print message and delete element. When NFAIL1 elements are deleted, LS-Dyna writes the d3dump file and terminates. These NFAIL1 failed elements also include all shell elements that failed for other reasons than distortion. Before the d3dump file is written, NFAIL1 is doubled, so the run can immediately be continued if desired.
NFAIL4	Flag to check for highly distorted fully-integrated shell elements, print a message, and delete the element or terminate. Generally, this flag is recommended. A distorted element is one where a

VARIABLE	DESCRIPTION
	negative jacobian exists within the domain of the shell, not just at integration points. The checks are made away from the integration points to enable the bad elements to be deleted before an instability leading to an error termination occurs. EQ.1: Print message and delete element. EQ.2: Print message, write d3dump file, and terminate GT.2: Print message and delete element. When NFAIL4 elements are deleted, LS-Dyna writes the d3dump file and terminates. These NFAIL4 failed elements also include all shell elements that failed for other reasons than distortion. Before the d3dump file is written, NFAIL4 is doubled, so the run can immediately be continued if desired.
PSNFAIL	Optional shell part set ID specifying which part IDs are checked by the NFAIL1, NFAIL4, and W-MODE options. If zero, all shell part IDs are included.

*CONTROL_TERMINATION

Purpose: Stop the job.

Card	1	2	3	4	5	6	7	8
Variable	ENDTIM	ENDCYC						
Type	F	I						

VARIABLE	DESCRIPTION
ENDTIM	Termination time: EQ.0.0: Termination time remains unchanged.
ENDCYC	Termination cycle. The termination cycle is optional and will be used if the specified cycle is reached before the termination time. EQ.0.0: Termination cycle remains unchanged.

Remarks:

This is a reduced version of the *CONTROL_TERMINATION card used in the initial input deck.

*CONTROL_TIMESTEP

Purpose: Set time step size control using different options.

Card	1	2	3	4	5	6	7	8
Variable	DUMMY	TSSFAC	ISDO	DUMMY	DT2MS	LCTM		
Type	F	F	I	F	F	I		

VARIABLE	DESCRIPTION
DUMMY	Dummy field, see Remark 1 below.
TSSFAC	Scale factor for computed time step. EQ.0.0: TSSFAC remains unchanged.
ISDO	Basis of time size calculation for 4-node shell elements. 3-node shells use the shortest altitude for options 0 and 1 and the shortest side for option 2. This field has no relevance to solid elements, which use a length based on the element volume divided by the largest surface area. EQ.0: characteristic length is given by $\frac{\text{area}}{\min(\text{longest side},\ \text{longest diagonal})}$. EQ.1: characteristic length is given by $\frac{\text{area}}{\text{longest diagonal}}$. EQ.2: based on bar wave speed and, $\max\left[\text{shortest side}, \frac{\text{area}}{\min(\text{longest side},\ \text{longest diagonal})}\right]$. WARNING: Option 2 can give a much larger time step size that can lead to instabilities in some applications, especially when triangular elements are used.
DUMMY	Dummy field, see Remark 1 below.

VARIABLE	DESCRIPTION
DT2MS	New time step for mass scaled calculations. Mass scaling must be active in the time zero analysis. EQ.0.0: DT2MS remains unchanged.
LCTM	Load curve ID that limits maximum time step size: EQ.0: LCTM remains unchanged.

Remarks:

1. **Dummy Fields.** This a reduced version of the *CONTROL_TIMESTEP used in the initial analysis. The dummy fields are included to maintain compatibility. If using free format input then a 0.0 should be entered for the dummy values.

*DAMPING_GLOBAL

Purpose: Define mass weighted nodal damping that applies globally to the deformable nodes.

Card	1	2	3	4	5	6	7	8
Variable	LCID	VALDMP						
Type	I	F						
Default	0	0.0						

VARIABLE	DESCRIPTION
LCID	Load curve ID (see *DEFINE_CURVE) which specifies the system damping constant vs. time: EQ.0: a constant damping factor as defined by VALDMP is used, GT.0: system damping is given by load curve LCID (which must be an integer). The damping force applied to each node is $f = -d(t)mv$, where $d(t)$ is defined by load curve LCID.
VALDMP	System damping constant, d (this option is bypassed if the load curve number defined above is nonzero).

***DATABASE**_*OPTION*

Purpose: Change the output interval for ASCII files. If a file is not specified in the restart deck, then the output interval for the file will remain unchanged.

Options for ASCII files include:

SECFORC	Cross section forces.
RWFORC	Wall forces.
NODOUT	Nodal point data.
ELOUT	Element data.
GLSTAT	Global data.
DEFORC	Discrete elements.
MATSUM	Material energies.
NCFORC	Nodal interface forces.
RCFORC	Resultant interface forces.
DEFGEO	Deformed geometry file.
SPCFORC	Set DT for SPC reaction forces.
SWFORC	Nodal constraint reaction forces (spot welds and rivets).
ABSTAT	Set DT for airbag statistics.
NODFOR	Set DT for nodal force groups.
BNDOUT	Boundary condition forces and energy
RBDOUT	Set DT for rigid body data.
GCEOUT	Set DT for geometric contact entities.
SLEOUT	Set DT for sliding interface energy.
JNTFORC	Set DT for joint force file.
SBTOUT	Set DT for seat belt output file.
AVSFLT	Set DT for AVS database.
MOVIE	Set DT for MOVIE.
MPGS	Set DT for MPGS.
TPRINT	Set DT for thermal file.

Card	1	2	3	4	5	6	7	8
Variable	DT							
Type	F							

VARIABLE	DESCRIPTION
DT	Time interval between outputs: EQ.0.0: Output interval is unchanged.

Remarks:

1. **File Output Termination.** To terminate output to a particular file, set DT to a high value.

2. **NODOUT Results.** If IACCOP = 2 was specified in *CONTROL_OUTPUT, the best results are obtained in the NODOUT file by keeping the same DT on restart. When DT is changed for NODOUT, oscillations may occur around the restart time. If DT is larger than initially specified in the original input file, more memory is required to store the time states for the averaging than was originally allocated. A warning message is printed, and the filtering is applied using the available memory. When DT is smaller than initially specified, more oscillations may appear in the output than earlier in the calculation because the frequency content of the averaged output increases as DT decreases.

***DATABASE_BINARY**_*OPTION*

Options for binary output files with the default names given include:

D3PLOT DT for complete output states.

D3THDT DT for time history data for element subsets.

D3DUMP Binary output restart files. Define output frequency in cycles

RUNRSF Binary output restart file. Define output frequency in cycles.

INTFOR DT for contact surface interface database.

Card	1	2	3	4	5	6	7	8
Variable	DT/CYCL							
Type	F							

VARIABLE	DESCRIPTION
DT	Time interval between outputs. EQ.0.0: Time interval remains unchanged.
CYCL	Output interval in time steps. EQ.0.0: Output interval remains unchanged.

***DELETE**_*OPTION1_OPTION2*

Available options for OPTION1 are:

ALECPL

CONTACT

CONTACT_2DAUTO

ENTITY

PART

ELEMENT_BEAM

ELEMENT_SHELL

ELEMENT_SOLID

ELEMENT_TSHELL

FSI

OPTION2 only works with *DELETE_PART. This option causes LS-DYNA to exclude the geometric and other element/node information of deleted parts in d3plot files after re-start which significantly reduces d3plot file size.

COMP

Purpose: Delete contact surfaces, ALE FSI couplings, parts, or elements by a list of IDs. There are two contact algorithms for two-dimensional problems: the line-to-line contact and the automatic contact defined by part IDs. Each uses their own numbering.

ID Cards. This card applies to the ALECPL, CONTACT, CONTACT_2DAUTO, ENTITY, FSI and PART options. Include as many cards as necessary to input desired IDs. This input ends at the next keyword ("*") card.

Card 1a	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I

VARIABLE	DESCRIPTION
ID*n*	Contact ID/Coupling ID/Part ID. See Remarks.

Element Set Cards. This card applies to the four ELEMENT options. This input ends at the next keyword ("*") card.

Card 1b	1	2	3	4	5	6	7	8
Variable	ESID							
Type	I							

VARIABLE	DESCRIPTION
ESID	Element set ID, see *SET_SOLID, *SET_BEAM, *SET_SHELL, *SET_TSHELL.

Remarks:

The FSI option corresponds to ALE couplings defined with *CONSTRAINED_LAGRANGE_IN_SOLID. The ALECPL option corresponds to ALE couplings defined with *ALE_COUPLING_NODAL_*OPTION*. For CONTACT, FSI, and ALECPL options, a negative ID implies that the absolute value gives the contact surface/FSI/ALECPL coupling which is to be activated.

*INTERFACE_SPRINGBACK_LSDYNA

Purpose: Define a material subset for output to a stress initialization file dynain. The dynain file contains keyword commands that can be included in a subsequent input deck to initialize deformation, stress, and strain in parts. This file can be used, for example, to do an implicit springback analysis after an explicit forming analysis.

Part Set ID Cards.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	NSHV						
Type	I	I						

Constraint Cards. Optional cards that list of nodal points that are constrained in the dynain file. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	NID	TC	RC					
Type	I	F	F					
Default	none	0.	0.					

VARIABLE	DESCRIPTION
PSID	Part set ID for springback, see *SET_PART.
NSHV	Number of shell or solid history variables (beyond the six stresses and effective plastic strain) to be initialized in the interface file. For solids, one additional state variable (initial volume) is also written. If NSHV is nonzero, the element formulations, calculational units, and constitutive models should not change between runs. If NHSV exceeds the number of integration point history variables required by the constitutive model, only the number required is written; therefore, if in doubt, set NHSV to a large number.
NID	Node ID

VARIABLE	DESCRIPTION
TC	Translational constraint: EQ.0: no constraints, EQ.1: constrained x displacement, EQ.2: constrained y displacement, EQ.3: constrained z displacement, EQ.4: constrained x and y displacements, EQ.5: constrained y and z displacements, EQ.6: constrained z and x displacements, EQ.7: constrained x, y, and z displacements.
RC	Rotational constraint: EQ.0: no constraints, EQ.1: constrained x rotation, EQ.2: constrained y rotation, EQ.3: constrained z rotation, EQ.4: constrained x and y rotations, EQ.5: constrained y and z rotations, EQ.6: constrained z and x rotations, EQ.7: constrained x, y, and z rotations.

***RIGID_DEFORMABLE**_*OPTION*

Available options include:

CONTROL	
D2R	(Deformable to rigid part switch)
R2D	(Rigid to deformable part switch)

Purpose: Define parts to be switched from rigid to deformable and deformable to rigid in a restart. It is only possible to switch parts on a restart if part switching was activated in the time zero analysis. See *DEFORMABLE_TO_RIGID for details of part switching.

*RIGID_DEFORMABLE_CONTROL

Card	1	2	3	4	5	6	7	8
Variable	NRBF	NCSF	RWF	DTMAX				
Type	I	I	I	F				
Default	0	0	0	none				

VARIABLE	DESCRIPTION
NRBF	Flag to delete or activate nodal rigid bodies (see Remark 1): EQ.0: no change, EQ.1: delete, EQ.2: activate.
NCSF	Flag to delete or activate nodal constraint set (see Remark 2): EQ.0: no change, EQ.1: delete, EQ.2: activate.
RWF	Flag to delete or activate rigid walls: EQ.0: no change, EQ.1: delete, EQ.2: activate.
DTMAX	Maximum permitted time step size after restart.

Remarks:

1. **Nodal Rigid Bodies.** If nodal rigid bodies or generalized, weld definitions are active in the deformable bodies that are switched to rigid, then the definitions should be deleted to avoid instabilities.

2. **Nodal Constraint Set.** If nodal constraint/spot weld definitions are active in the deformable bodies that are switched to rigid, then the definitions should be deleted to avoid instabilities.

*RIGID_DEFORMABLE_D2R

Part ID Cards. Include one card for each part. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	LRB						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
PID	Part ID of the part which is switched to a rigid material.
LRB	Part ID of the lead rigid body to which the part is merged. If zero, the part becomes either an independent or lead rigid body.

***RIGID_DEFORMABLE_R2D**

Termination of this input is when the next "*" card is read.

Part ID Cards. Include one card for each part. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID							
Type	I							
Default	none							

VARIABLE	DESCRIPTION
PID	Part ID of the part which is switched to a deformable material.

*STRESS_INITIALIZATION_{OPTION}

This keyword causes a full deck restart. For a full deck restart the input deck must contain the full model. The stress initialization feature allows all or selected parts to be initialized from the previous calculation using data from the d3dump or runrsf databases.

The options that are available with this keyword are:

<BLANK>

DISCRETE

SEATBELT

Optional Part Cards. If no part cards are included in the deck then all parts, seatbelts and discrete parts in the new input deck that existed in the previous input deck (with or without the same part IDs) are initialized from the d3dump or runrsf database. Otherwise for each part to be initialized from the restart data include an addition card in format 1. This input terminates at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	PID0	PIDN						
Type	I	I						
Default	none	PIDO						

VARIABLE	DESCRIPTION
PIDO	Old part ID, see *PART.
PIDN	New part ID, see *PART: EQ.0: New part ID is the same as the old part ID.

Remarks:

If one or more of the above cards are defined, then discrete and seatbelt elements will not be initialized unless the additional option cards *STRESS_INITIALIZATION_DISCRETE and *STRESS_INITIALIZATION_SEATBELT are defined.

*STRESS_INITIALIZATION_DISCRETE

Initialize all discrete parts from the old parts. No further input is required with this card. This card is not required if *STRESS_INITIALIZATION is specified without further input.

*STRESS_INITIALIZATION_SEATBELT

Initialize all seatbelt parts from the old parts. No further input is required with this card. This card is not required if *STRESS_INITIALIZATION is specified without further input.

TERMINATION_*OPTION

Purpose: Stops the job depending on some displacement conditions.

Available options include:

NODE

BODY

Caution: The inputs are different for the nodal and rigid body stop conditions. The nodal stop condition works on the global coordinate position, while the body stop condition works on the relative global translation. The number of termination conditions cannot exceed the maximum of 10 or the number specified in the original analysis.

The analysis terminates for *TERMINATION_NODE when the current position of the node specified reaches either the maximum or minimum value (stops 1, 2 or 3), or picks up force from any contact surface (stop 4). For *TERMINATION_BODY the analysis terminates when the center of mass displacement of the rigid body specified reaches either the maximum or minimum value (stops 1, 2 or 3) or the displacement magnitude of the center of mass is exceeded (stop 4). If more than one condition is input, the analysis stops when any of the conditions is satisfied.

> **NOTE:** This input completely overrides the existing termination conditions defined in the time zero run.

Termination by other means is controlled by the *CONTROL_TERMINATION control card.

Card Summary:

Card 1a. This card is included if and only if the NODE keyword option is used.

NID	STOP	MAXC	MINC				

Card 1b. This card is included if and only if the BODY keyword option is used.

PID	STOP	MAXC	MINC				

Data Card Descriptions:

Node Cards. Include an additional card for each node with a termination criterion.

Card 1a	1	2	3	4	5	6	7	8
Variable	NID/PID	STOP	MAXC	MINC				
Type	I	I	F	F				
Default	none	none	↓	↓				

VARIABLE	DESCRIPTION
NID	Node ID
STOP	Stop criterion: EQ.1: global *x* direction, EQ.2: global *y* direction, EQ.3: global *z* direction, EQ.4: stop if node touches contact surface.
MAXC	Maximum (most positive) coordinate, options 1, 2 and 3 above only.
MINC	Minimum (most negative) coordinate, options 1, 2 and 3 above only.

Part Cards. Include an additional card for each part with a termination criterion.

Card 1b	1	2	3	4	5	6	7	8
Variable	NID/PID	STOP	MAXC	MINC				
Type	I	I	F	F				
Default	none	none	↓	↓				

VARIABLE	DESCRIPTION
PID	Part ID of rigid body

VARIABLE	DESCRIPTION
STOP	Stop criterion: EQ.1: global *x*-direction, EQ.2: global *y*-direction, EQ.3: global *z*-direction, EQ.4: stop if displacement magnitude is exceeded.
MAXC	Maximum (most positive) displacement, options 1, 2, 3 and 4: EQ.0.0: MAXC set to 10^{21}
MINC	Minimum (most negative) displacement, options 1, 2 and 3 above only: EQ.0.0: MINC set to -10^{21}

***TITLE**

Purpose: Define job title.

Card	1	2	3	4	5	6	7	8
Variable	TITLE							
Type	C							
Default	LS-DYNA USER INPUT							

VARIABLE	DESCRIPTION
TITLE	Heading to appear in a small restart's d3hsp file. This heading is not written to any other output file.

REFERENCES

Abbo, A.J., and S.W. Sloan, "A Smooth Hyperbolic Approximation to the Mohr-Coulomb Yield Criterioin," *Computers and Structures*, Vol. 54, No. 1, (1995).

Alameda, A., Dominguez, A., Martin-Santos, E., Miura, T.M., "Enhanced Material Model for Composite Crash Performance at Ply Level - *MAT_262 Evolution -", *Proceedings 2024 JSAE Annual Congress (Spring)*, No. 30-34, May (2024).

Allen, D.J., Rule, W.K., Jones, S.E., "Optimizing Material Strength Constants Numerically Extracted from Taylor Impact Data", *Experimental Mechanics*, Vol. 37, No. 3, September (1997).

Allen, E.J., "Approximate ballistics formulas for spherical pellets in free flight", *Defence Technology*, Vol. 14, No. 1, pp. 1-11, (2018).

Allman, D.J., "A Compatible Triangular Element Including Vertex Rotations for Plane Elasticity Analysis," *Computers and Structures*, 19, 1-8, (1984).

Anagonye, A.U. and J.T. Wang, "A Semi-Empirical Method for Estimating the Effective Leak and Vent Areas of an Airbag", AMD-Vol. 237/BED-Vol. 45, pp. 195-217, (1999).

Anand, L. and M.E. Gurtin, "A theory of amorphous solids undergoing large deformations, with application to polymeric glasses," *International Journal of Solids and Structures*, 40, pp. 1465-1487 (2003)

Andrade, F.X.C., Feucht, M., Haufe, A., "On the Prediction of Material Failure in LS-DY-NA: A Comparison between GISSMO and DIEM", 13th International LS-DYNA Users Conference, Dearborn, MI, June 8-10, (2014).

Andrade, F.X.C., Feucht, M., Haufe, A., "An incremental stress state dependent damage model for ductile failure prediction", *International Journal of Fracture*, 200, 127-150 (2016).

Aretz, H. "Applications of a New Plane Stress Yield Function to Orthotropic Steel and Aluminum Sheet Metals," *Modeling and Simulation in Materials Science and Engineering*, 12, 491-509 (2004).

Aretz, H., Barlat, F. "General orthotropic yield function based on linear stress deviator transformations," in: S. Gosh, G.C. Castro, J.K. Lee (eds.) *Materials processing and design: Modelling, simulation and applications*, Proceedings of the NUMIFORM 2004 Conference, Columbus, Ohio, 147-151 (2004).

REFERENCES

Aretz, H., Aegerter, J. and Engler, O., "Analysis of earing in deep drawn cups", in: F. Barlat, Y.H. Moon, M.G. Lee (eds.), Proceedings of the NUMIFORM 2010 Conference, 1252 (1), 417-424 (2010).

Argon, AS., "A theory for the low-temperature plastic deformation of glassy polymers", *Philosophical Magazine*, 28, 839-865 (1973).

Armstrong, P.J., and Frederick, C.O., "A Mathematical Representation of the Multiaxial Bauschinger Effect," CEGB Report, RD/B/N731, Berkeley Nuclear Laboratories (1966).

Arruda, E. and M. Boyce, "A Three-Dimensional Constitutive Model for the Large Stretch Behavior of Rubber Elastic Materials," *Journal of the Mechanics and Physics of Solids*, Vol. 41, No. 2, pp. 389-412, (1993).

Auricchio, F., R.L. Taylor and J. Lubliner, "Shape-memory alloys: macromodeling and numerical simulations of the superelastic behavior", *Computer Methods in Applied Mechanics and Engineering*, vol. 146, pp. 281-312, (1997).

Auricchio, F. and R.L. Taylor, "Shape-memory alloys: modeling and numerical simulations of the finite-strain superelastic behavior", *Computer Methods in Applied Mechanics and Engineering*, vol. 143, pp. 175-194, (1997).

Bahler AS: The series elastic element of mammalian skeletal muscle. *Am J Physiol* 213:1560-1564, (1967).

Baker, E.L., "An Explosives Products Thermodynamic Equation of State Appropriate for Material Acceleration and Overdriven Detonation: Theoretical Background and Fourmulation," Technical Report ARAED-TR-911013, U.S. Army Armament Research, Development and Engineering Center, Picatinney Arsenal, New Jersey, 1991).

Baker, E.L. and J. Orosz, J., "Advanced Warheads Concepts: An Advanced Equation of State for Overdriven Detonation," Technical Report ARAED-TR-911007, U.S. Army Armament Research, Development and Engineering Center, Picatinney Arsenal, New Jersey, (1991).

Baker, E.L. and L.I. Stiel, "Improved Quantitative Explosive Performance Prediction Using Jaguar," 1997 Insensitive Munitions and Energetic Materials Technology Symposium, Tampa, FL, (1997).

Bammann, D.J. and E.C. Aifantis, "A Model for Finite-Deformation Plasticity," *Acta Mechanica*, 70, 1-13 (1987).

Bammann, D.J. and G. Johnson, "On the Kinematics of Finite-Deformation Plasticity," *Acta Mechanica, 69,* 97-117 (1987).

Bammann, D.J., "Modeling the Temperature and Strain Rate Dependent Large Deformation of Metals," Proceedings of the 11th US National Congress of Applied Mechanics, Tuscon, AZ, (1989).

Bammann, D.J., M.L. Chiesa, A. McDonald, W.A. Kawahara, J.J. Dike, and V.D. Revelli, "Predictions of Ductile Failure in Metal Structures," in AMD-Vol. 107, Failure Criteria and Analysis in Dynamic Response, Edited by. H.E. Lindberg, 7-12, (1990).

Bandak, F.A., private communications, U.S. Dept. of Trans., Division of Biomechanics Research, 400 7th St., S.W. Washington, D.C. 20590 (1991).

Banks, J., Henshaw, W., Schwendeman, D., and A. Kapila, "A study of detonation propagation and diffraction with compliant confinement," Combust. Theory Model. 12 (4) (2008) 769–808.

Bansal, S., Mobasher, B., Rajan, S., and Vintilescu, I., "Development of Fabric Constitutive Behavior for Use in Modeling Engine Fan Blade-Out Events." J. Aerosp. Eng. 22, SPECIAL ISSUE: Ballistic Impact and Crashworthiness Response of Aerospace Structures, 249–259, (2009).

Barlat, F. and J. Lian, "Plastic Behavior and Stretchability of Sheet Metals. Part I: A Yield Function for Orthotropic Sheets Under Plane Stress Conditions," *Int. J. of Plasticity*, Vol. 5, pp. 51-66 (1989).

Barlat, F., D.J. Lege, and J.C. Brem, "A Six-Component Yield Function for Anisotropic Materials," *Int. J. of Plasticity*, 7, 693-712, (1991).

Barlat, F., Y. Maeda, K. Chung, M. Yanagawa, J.C. Brem, Y. Hayashida, D.J. Lege, K. Matsui, S.J. Murtha, S. Hattori, R.C. Becker, and S. Makosey, "Yield Function Development for Aluminum Alloy Sheets", J. Mech. Phys. Solids, Vol. 45, No. 11-12, 1727-1763, (1997).

Barlat, F., Brem, J.C., Yoon, J.W., Chung, K., Dick, R.E., Lege, D.J., Pourboghrat, F., Choi, S.H., Chu, E., "Plane Stress Yield Function for Aluminum Alloy Sheets – Part 1: Theory, Int. J. Plast. 19, 1-23, (2003).

Barlat, F., Aretz, H., Yoon, J.W., Karabin, M.E., Brem, J.C., Dick, R.E. "Linear transformation-based anisotropic yield functions", *International Journal of Plasticity* 21, 1009-1039 (2005).

Basu, U., "Explicit finite element perfectly matched layer for transient three-dimensional elastic waves," *International Journal for Numerical Methods in Engineering*, vol. 77, pp. 151–176, (2009).

REFERENCES

Basu, U. and Chopra, A.K., "Perfectly matched layers for time-harmonic elastodynamics of unbounded domains theory and finite-element implementation," *Computer Methods in Applied Mechanics and Engineering*, vol. 192, pp. 1337–1375, (2003).

Basu, U. and Chopra, A.K., "Perfectly matched layers for transient elastodynamics of unbounded domains," *International Journal for Numerical Methods in Engineering*, vol. 59, pp. 1039–1074, (2004). Erratum: Ibid. vol. 61, pp. 156–157, (2004).

Bathe, K.-J. Conserving energy and momentum in nonlinear dynamics: A simple implicit time integration scheme, *Computers and Structures*, 85, 437-445 (2007).

Bathe, K.-J. and Dvorkin, E.N. A four node plate bending element based on Mindlin-Reissner plate theory and a mixed interpolation, *Int. J. Num. Meth. Eng.*, 21, 367-383 (1985).

Bathe, K-J. and Noh, G. Insight into an implicit time integration scheme for structural dynamics, *Computers and Structures*, 98/99, 1-6 (2012).

Batoz, J.L. and Ben Tahar, M. Evaluation of a new quadrilateral thin plate bending element, Int. *J. Num. Meth. Eng.*, 18, 1644-1677 (1982).

Batoz, J.-L. and M. Ben Tahar, Evaluation of a new quadrilateral thin plate bending element, *International Journal for Numerical Methods in Engineering*, 18, (1982), 1655-1677.

Bayer, J.D., Blake, R. C., Plank, G., and Trayanova, N. A., "A novel rule-based algorithm for assigning myocardial fiber orientation to computational heart models," *Annals of biomedical engineering*, 40(10), 2243-2254 (2012).

Bazeley, G.P., W.K. Cheung, R.M. Irons, and O.C. Zienkiewicz, "Triangular Elements in Plate Bending-Confirming and Nonconforming Solutions in Matrix Methods and Structural Mechanics," Proc. Conf. on Matrix Methods in Structural Analysis, Rept. AFFDL-R-66-80, Wright Patterson AFB, 547-576 (1965).

Belytschko, T. and Bindeman, L. P. "Assumed Strain Stabilization of the Eight Node Hexahedral Element," Comp. Meth. Appl. Mech. Eng. **105**, 225-260 (1993).

Belytschko, T.B. and A.H. Marchertas, "Nonlinear Finite Element Method for Plates and its Application to the Dynamic Response of Reactor Fuel Subassemblies," *Trans, ASME J. Pressure Vessel Tech.*, 251-257 (1974).

Belytschko, T.B. and C.S. Tsay, "Explicit Algorithms for Nonlinear Dynamics of Shells," AMD-Vol.48, ASME, 209-231 (1981).

Belytschko, T.B. and C.S. Tsay, "Explicit Algorithms for Nonlinear Dynamics of Shells," *Comp. Meth. Appl. Mech. Eng.*, 43, 251-276, (1984).

Belytschko, T.B. and C.S. Tsay, "A Stabilization Procedure for the Quadrilateral Plate Element with One-Point Quadrature," *Int. J. Num. Method. Eng.*, 19, 405-419 (1983).

Belytschko, T.B., H. Stolarski, and N. Carpenter, "A C° Triangular Plate Element with One-Point Quadrature," *Int. J. Num. Meth. Eng.*, 20, 787-802 (1984).

Belytschko, T.B., I. Yeh, "The Splitting Pinball Method for Contact-Impact Problems," *Comp. Meth. Appl. Mech. Eng.*, 105, 375-393, (1993).

Belytschko, T.B., L. Schwer, and M.J. Klein, "Large Displacement Transient Analysis of Space Frames," *Int. J. Num. Eng.*, 11, 65-84 (1977).

Benson, D.J. and J.O. Hallquist, "A Simple Rigid Body Algorithm for Structural Dynamics Programs," *Int. J. Numer. Meth. Eng.*, 22, (1986).

Benson, D.J. and J.O. Hallquist, "A Single Surface Contact Algorithm for the Postbuckling Analysis of Shell Structures," Comp. Meths. Appl. Mech. Eng., 78, 141-163 (1990).

Benzeggagh, M.L. and Kenane, M., "Measurement of Mixed-mode Delamination Fracture Toughness of Unidirectional Glass/Epoxy Composites with Mixed-mode Bending Apparatus," Composites Science and Technology, 56, 439-449 (1996).

Berstad, T., "Material Modeling of Aluminium for Crashworthiness Analysis", Dr.Ing. Dissertation, Department of Structural Engineering, Norwegian University of Science and Technology, Trondheim, Norway, (1996).

Berstad, T., Hopperstad, O.S., Lademo, O.-G. and Malo, K.A., "Computational Model of Ductile Damage and Fracture in Shell Analysis", Second European LS-DYNA Conference, Gothenburg, Sweden, (1999).

Berstad, T., Lademo, O.-G., Pedersen, K.O. and Hopperstad, O.S., "*Formability modeling with LS-DYNA"*, 8th International LS-DYNA User's Conference, Detroit, May 3-5, 2004.

Berstad, T., Langseth, M. and Hopperstad, O.S., "Elasto-viscoplastic Constitutive Models in the Explicit Finite Element Code LS-DYNA3D," Second International LS-DYNA3D conference, San Francisco, (1994).

Bergström, J.S. and M.C. Boyce, "Constitutive modeling of the large strain time-dependent behavior of elastomers" *J. Mech. Phys. Solids*, 46, 931-954 (1998).

Bielak, J. and Christiano, P., "On the effective seismic input for non-linear soil-structure interaction systems," *Earthquake Engineering and Structural Dynamics*, vol. 12, pp. 107–119, (1984).

REFERENCES

Bier, M., Sommer, S., "Simplified modeling of self-piercing riveted joints for crash simulation with a modified version of *CONSTRAINED_INTERPOLATION_-SPOTWELD". 9th European LS-DYNA Conference, Manchester, (2013).

Bilkhu, S.S., M. Founas, and G.S. Nasholtz, "Material Modeling of Structural Foams in Finite Element Analysis Using Compressive Uniaxial and Triaxial Data," SAE (Nat. Conf.) Detroit 1993, pp. 4-34.

Blatz, P.J., and Ko, W.L., "Application of Finite Element Theory to the Deformation of Rubbery Materials," *Trans. Soc. of Rheology*, 6, 223-251 (1962).

Borrvall T., Bhalsod D., Hallquist J.O., and Wainscott B., "Current Status of Subcycling and Multiscale Simulations in LS-DYNA". 13th International LS-DYNA User's Conference, Dearborn MI, (2014).

Boyce, M.C., Parks, D.M., and Argon, A.S., "Large inelastic deformation of glassy polymers. Part I: Rate dependent constitutive model". *Mechanics of Materials*, 7, 15-33 (1988).

Boyce, M.C., Socrate, C. and Llana, P.G., "Constitutive model for the finite deformation stress-strain behavior of poly(ethylene terephthalate) above the glass transition". *Polymer*, 41, 2183-2201 (2000).

Brandt, J., On Constitutive Modelling of the Compaction and Sintering of Cemented Carbides, Linköping Studies in Science and Technology, Dissertations 515, 1998.

Brekelmans, W.A.M., Scheurs,P.J.G., and de Vree, J.H.P., 1991, "Continuum damage mechanics for softening of brittle materials", *Acta Mechanica*, vol 93, pp 133-143

Broadhouse, B.J., "The Winfrith Concrete Model in LS-DYNA3D," Report: SPD/D(95)363, Structural Performance Department, AEA Technology, Winfrith Technology Centre, U.K. (1995).

Broadhouse, B.J. and Neilson, A.J., "Modelling Reinforced Concrete Structures in DY-NA3D", Safety and Engineering Division, United Kingdom Atomic Energy Authority, Winfrith, AEEW-M 2465, 1987.

Brode, H.L., "Height of Burst Effects at High Overpressure," RAND, RM-6301-DASA, DASA 2506, (1970).

Brown, B.E. and J.O. Hallquist, "TAURUS: An Interactive Post-Processor for the Analysis Codes NIKE3D, DYNA3D, TACO3D, and GEMINI," University of California, Lawrence Livermore National Laboratory, Rept. UCID-19392 (1982) Rev. 1 (1984).

Bruneau, M., Uang, C.M., Whittaker, A., Ductile Design of Steel Structures, McGraw Hill, (1998).

Burton, D.E. et al. "Physics and Numerics of the TENSOR Code," Lawrence Livermore National Laboratory, Internal Document UCID-19428, (July 1982).

Carney, K.S., S.A. Howard, B.A. Miller, and D.J. Benson. "New Representation of Bearings in LS-DYNA," 13th International LS-DYNA Users Conference, Dearborn, MI, June 8-10, 2014.

CEB Code 1993, Comite euro-international du beton, *CEB-FIP Model Code 1990*, Thomas Telford, London, (1993).

Chang, F.K. and K.Y. Chang, "A Progressive Damage Model for Laminated Composites Containing Stress Concentration," *J. of Composite Materials*, 21, 834-855 (1987a).

Chang, F.K. and K.Y. Chang, "Post-Failure Analysis of Bolted Composite Joints in Tension or Shear-Out Mode Failure," *J. of Composite Materials*, 21 809-823 (1987b).

Chang, F.S., "Constitutive Equation Development of Foam Materials," Ph.D. Dissertation, submitted to the Graduate School, Wayne State University, Detroit, Michigan (1995).

Chao, Y. J., Wang, K, Miller, K. W. and Zhu, X. K. "Dynamic Separation of Resistance Spot Welded joints: Part I – Experiments", *Exp Mech*, Vol. 50, Issue 7, pp 889-900 (2010).

Chen, W.F., and Baladi, G.Y., Soil Plasticity: Theory and Implementation, Elesvier, New York, (1985).

Cheng, H., Obergefell, L.A., and Rizer, A., March 1994, "Generator of Body (GEBOD) Manual," Report No. AL/CF-TR-1994-0051.

Chowdhury, S.R. and Narasimhan R., "A Cohesive Finite Element Formulation for Modeling Fracture and Delamination in Solids," *Sadhana*, 25(6), 561-587, (2000).

Christensen, R.M. "A Nonlinear Theory of Viscoelasticity for Application to Elastomers," *Journal of Applied Mechanics,* Volume 47, American Society of Mechanical Engineers, pages 762-768, December 1980.

Chu, C.C. and A. Needleman, "Void Nucleation Effects in Biaxially Stretched Sheets", ASME Journal of Engineering Materials and Technology, 102, 249-256 (1980).

Chung, K. and K. Shah, "Finite Element Simulation of Sheet Metal Forming for Planar Anisotropic Metals," *Int. J. of Plasticity*, 8, 453-476, (1992).

Ciambella, J. and G. Saccomandi, "A continuum hyperelastic model for auxetic materials," *Proceedings of the Royal Society A*, 470:20130691, (2014).

REFERENCES

Cochran, S.G. and J. Chan, "Shock Initiation and Detonation Models in One and Two Dimensions," University of California, Lawrence Livermore National Laboratory, Rept. UCID-18024 (1979).

M. G. Cockcroft and D. J. Latham, "Ductility and the Workability of Metals," *Journal Institute of Metals*, Vol. 96, 1968, pp. 33-39.

Cook, R. D., Concepts and Applications of Finite Element Analysis, John Wiley and Sons, Inc. (1974).

Costas M., Morin D., Hopperstad O.S., Børvik T., and Langseth M., "A through-thickness damage regularisation scheme for shell elements subjected to severe bending and membrane deformations", submitted to *Journal of the Mechanics and Physics of Solids*, (2018)

Couch, R., E. Albright, and N. Alexander, The Joy Computer Code, Lawrence Livermore National Laboratory, Internal Document Rept. UCID-19688, (January, 1983).

Couque, H., "The use of the direct impact Hopkinson pressure bar technique to describe thermally activated and viscous regimes of metallic materials", *Phil. Trans. R. Soc. A*, 372: 20130218, (2014).

Cowper, G.R. and P.S. Symonds, Strain Hardening and Strain Rate Effects in the Impact Loading of Cantilever Beams, Brown University, Applied Mathematics Report, 1958.

CRAY-1 Computer System CFT Reference Manual, Cray Research Incorporated, Bloomington, NM., Publication No. 2240009 (1978).

Dafalias,Y. F. and Manzari, M.T. "Simple plasticity sand model accounting for fabric change effects." *Journal of Engineering Mechanics*, 130(6), 622–634 (2004).

Dafalias, Y. F., Papadimitriou, A. G., and Li, X. S. "Sand plasticity model accounting for inherent fabric anisotropy." *Journal of Engineering Mechanics*,130(11), 1319–1333 (2004).

Dal, H. and M. Kaliske, "Bergström-Boyce model for nonlinear finite rubber viscoelasticity: theoretical aspects and algorithmic treatment for the FE method" *Computational Mechanics*, 44(6), 809-823, (2009).

Davaze, V., Vallino, N., Langrand, B., Besson, J., Feld-Payet, S., "A non-local damage approach compatible with dynamic explicit simulations and parallel computing" *International Journal of Solids and Structures*, Volume 228, Article 110999, (2021).

Davaze, V., Feld-Payet, S., Vallino, N., Langrand, B., Besson, J., "A non-local approach for Reissner-Mindlin shell elements in dynamic simulations: Application with a

Gurson model" *Computer Methods in Applied Mechanics and Engineering*, Vol. 415, Article 116142, (2023).

DeRuntz, J.A. Jr., "Reference Material for USA, The Underwater Shock Analysis Code, USA-STAGS, and USA-STAGS-CFA," Report LMSC-P032568, Computational Mechanics Laboratory, Lockheed Palo Alto Research Laboratory, Palo Alto, CA. (1993).

Desai, C.S., and H.J. Siriwardane, Constitutive Laws for Engineering Materials with Emphasis On Geologic Materials, Prentice-Hall, Chapter 10, (1984).

Deshpande, V.S. and N.A. Fleck, "Isotropic Models for Metallic Foams," *Journal of the Mechanics and Physics of Solids*, 48, 1253-1283, (2000).

Dick, R.E., and W.H. Harris, "Full Automated Rezoning of Evolving Geometry Problems," Numerical Methods in Industrial Forming Processes, Chenot, Wood, and Zienkiewicz, Editors, Bulkema, Rotterdam, 243-248, (1992).

Dilger, W.H., R. Koch, and R. Kowalczyk, "Ductility of Plain and Confined Concrete Under Different Strain Rates," *ACI Journal,* January-February, (1984).

Dobratz, B.M., "LLNL Explosives Handbook, Properties of Chemical Explosives and Explosive Simulants," University of California, Lawrence Livermore National Laboratory, Rept. UCRL-52997 (1981).

Du Bois, P.A., "Numerical Simulation of Strandfoam" Daimler-Chrysler AG Abt. EP/CSV, Report (2001).

Dufailly, J., and Lemaitre, J., "Modeling very low cycle fatigue", *International Journal of damage mechanics*, 4, pp. 153-170 (1995).

Dunand, M., Maertens, A.P., Luo, M., Mohr, D., "Experiments and modeling of anisotropic aluminum extrusions under multi-axial loading – Part I: Plasticity," *Int. J. of Plasticity*, Vol. 36, pp. 34-49 (2012).

Effelsberg, J., Haufe, A., Feucht, M., Neukamm, F., DuBois, P., "On the Parameter Identification for the GISSMO Damage Model", 12th International LS-DYNA Users Conference, Dearborn, MI, (2012).

Ellison, K., Soga, K., & Simpson, B., "A strain space soil with evolving stiffness memory", Géotechnique, Vol. 62, No. 7, 627-641 (2012).

Erhart, T., Andrade, F., Du Bois, P., "Short Introductionof a New Generalized Damage Model", 11th European LS-DYNA Conference, Salzburg, Austria, May 2017.

Erhart, T., "A New Feature to Model Shell-Like Structures with Stacked Elements", 10th European LS-DYNA Conference, Würzburg, Germany, June 2015.

REFERENCES

Erhart, T., Borrvall, T., "Drilling rotation constraint for shell elements in implicit and explicit analyses", 9th European LS-DYNA Conference, Manchester, UK, June 2013.

Erhart, T., "An Overview of User Defined Interfaces in LS-DYNA", LS-DYNA Forum, Bamberg, Germany, 2010.

Englemann, B. E., R.G. Whirley, and G.L. Goudreau, "A Simple Shell Element Formulation for Large-Scale Elastoplastic Analysis," CED-Vol. 3. Analytical and Computational Models of Shells, A.K. Noor, T. Belytschko, and J.C. Simo, Editors, 1989, pp. 399-416.

Faßnacht, W., "Simulation der Rißbildung in Aluminiumgußbauteilen," Dissertation, Technishe Universität Darmstadt, (1999).

Fan, H.F, Zhu, X.H., Zhang, L., and Xiao, Y.Z., Improvement of Mesh Fusion in LS-DYNA, FEA Information Engineering Journal, Volume 6, Issue Q2, June 2017, ISSN 2167-1273.

Fedoroff, A., Calonius, K., "Using the Abaqus CDP model in impact simulations", *Journal of Structural Mechanics*, Vol. 53, No. 3, 180-207 (2020).

Feng, W.W. and Hallquist, J.O., "On Constitutive Equations for Elastomers and Elastomeric Foams", The 4th European LS-DYNA Conference, D-II-15, Ulm, Germany, May 2003.

Feucht, M., "Ein gradientenabhängiges Gursonmodell zur Beshreibung duktiler Schädigung mit Entfestigung," Dissertation, Technishe Universität Darmstadt, (1998).

Fiolka, M. and Matzenmiller, A., "Delaminationsberechnung von Faserverbundstrukturen", *PAMM Proc. Appl. Math. Mech.* 5, S.393-394 (2005).

Flanagan, D.P. and T. Belytschko, "A Uniform Strain Hexahedron and Quadrilateral and Orthogonal Hourglass Control," *Int. J. Numer. Meths. Eng.*, 17, 679-706 (1981).

Fleischer, M., Borrvall, T. and Bletzinger, K-U., "Experience from using recently implemented enhancements for Material 36 in LS-DYNA 971 performing a virtual tensile test", 6th European LS-DYNA Users Conference, Gothenburg, 2007.

Forghani A., "A Non-Local Approach to Simulation of Damage in Composite Structures", *PhD Thesis, Department of Civil Engineering, The University of British Columbia,, Vancouver, Canada,* (2011).

Forghani A., Zobeiry N., Vaziri R., Poursartip A., and Ellyin F., "A Non-Local Approach to Simulation of Damage in Laminated Composites." *Proc., ASC/CANCOM Conference,* Montreal, Canada (2011b).

Forghani A., Zobeiry N., Poursartip A., and Vaziri R., "A Structural Modeling Framework for Prediction of Damage Development and Failure of Composite Laminates". Accepted for publication in *Composites Sci. Technol*.

Freed AD., Einstein DR. and Vesely I., "Invariant Formulation for Dispersed Transverse Isotropy in Aortic Heart Valves – An Efficient Means for Modeling Fiber Splay", *Biomechan Model Mechanobiol*, 4, 100-117 (2005)

Fung, Y.C., Biomechanics, Springer, New York, 1993.

Fung, Y.C., Foundations of Solid Mechanics, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1965.

Gerlach, S., Fiolka M. and Matzenmiller, A., Modelling and analysis of adhesively bonded joints with interface elements for crash analysis, 4. LS-DYNA Forum, 20-21, (2005) Bamberg, DYNAmore GmbH, Stuttgart.

Ginsberg, M. and J. Johnson, "Benchmarking the Performance of Physical Impact Simulation Software on Vector and Parallel Computers," Applications Track of Supercomputing, IEEE monograph, Computer Society Press, March, 1989.

Giroux, E.D. HEMP User's Manual, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-51079 (1973).

Goldberg, R., and D. Stouffer, "High Strain Rate Dependent Modeling Polymer Matrix Composites," NASA/TM-1999-209433 (1999).

Goudreau, G.L. and J.O. Hallquist, "Recent Developments in Large Scale Finite Element Lagrangian Hydrocode Technology," *J. Comp. Meths. Appl. Mechs. Eng.*, 30 (1982).

Goldak, J., Chakravarti, A., and Bibby, M., "A New Finite Element Model for Welding Heat Sources," *Metallurgical Transactions B*, vol. 15B, pp. 299-305, June, 1984.

Govindjee, S., Kay, J.G., and Simo, J.C. [1994], Anisotropic Modeling and Numerical Simulation of Brittle Damage in Concrete, Report No. UCB/SEMM-94/18, Department of Civil Engineering, University of California, Berkeley, CA 94720.

Govindjee, S., Kay, J.G., and Simo, J.C. [1995], "Anisotropic Modeling and Numerical Simulation of Brittle Damage in Concrete," *Int. J. Numer. Meth. Engng*, **38**, 3611-3633.

Graefe, H., W. Krummheuer, and V. Siejak, "Computer Simulation of Static Deployment Tests for Airbags, Air Permeability of Uncoated Fabrics and Steady State Measurements of the Rate of Volume Flow Through Airbags," SAE Technical Paper Series, 901750, Passenger Car Meeting and Expositition, Dearborn, Michigan, September 17-20, 1990.

REFERENCES

Gran, J.K. and P.E. Senseny, "Compression Bending of Scale-Model Reinforced-Concrete Walls," *ASCE Journal of Engineering Mechanics*, Volume 122, Number 7, pages 660-668, July (1996).

Grassl, P., U. Nyström, R. Rempling, and K. Gylltoft, "A Damage-Plasticity Model for the Dynamic Failure of Concrete", 8th International Conference on Structural Dynamics, Leuven, Belgium, March (2011).

Grassl, P., D. Xenos, U. Nyström, R. Rempling, and K. Gylltoft, "CDPM2: A damage-plasticity approach to modelling the failure of concrete", *International Journal of Solids and Structures*, Vol. 50, Issue 24, pp. 3805-3816, (2013).

Grassl, P. and M. Jirásek, "Damage-Plastic Model for Concrete Failure", *International Journal of Solids and Structures*, Vol. 43, Issues 22-23, pp. 7166-7196, November (2006).

Gruben, G., Hopperstad O.S., and Børvik T., "Evaluation of uncoupled ductile fracture criteria for the dual-phase steel Docol 600DL". *International Journal of Mechanical Sciences*, Volume 62, pages 133–146, (2012)

Guccione, J., A. McCulloch, and L. Waldman, "Passive Material Properties of Intact Ventricular Myocardium Determined from a Cylindrical Model", *ASME Journal of Biomechanical Engineering*, Vol. 113, pages 42-55, (1991).

Guccione JM, Waldman LK, McCulloch AD., "Mechanics of Active Contraction in Cardiac Muscle: Part II – Cylindrical Models of the Systolic Left Ventricle", J. Bio Mech, 115, 82-90, (1993).

Gurson, A.L., Plastic Flow and Fracture Behavior of Ductile Materials Incorporating Void Nucleation, Growth, and Interaction, Ph.D. Thesis, Brown University, (1975).

Gurson, A.L., "Continuum Theory of Ductile Rupture by Void Nucleation and Growth: Part I - Yield Criteria and Flow Rules for Porous Ductile Media", *J. of Eng. Materials and Technology*, (1977).

Hallquist, J.O., Preliminary User's Manuals for DYNA3D and DYNAP (Nonlinear Dynamic Analysis of Solids in Three Dimension), University of California, Lawrence Livermore National Laboratory, Rept. UCID-17268 (1976) and Rev. 1 (1979).[**a**]

Hallquist, J.O., A Procedure for the Solution of Finite Deformation Contact-Impact Problems by the Finite Element Method, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-52066 (1976).

Hallquist, J.O., "A Numerical Procedure for Three-Dimensional Impact Problems," *American Society of Civil Engineering*, Preprint 2956 (1977).

Hallquist, J.O., "A Numerical Treatment of Sliding Interfaces and Impact," in: K.C. Park and D.K. Gartling (eds.) *Computational Techniques for Interface Problems*, AMD Vol. 30, ASME, New York (1978).

Hallquist, J.O., NIKE2D: An Implicit, Finite-Element Code for Analyzing the Static and Dynamic Response of Two-Dimensional Solids, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-52678 (1979).**[b]**

Hallquist, J.O., User's Manual for DYNA2D – An Explicit Two-Dimensional Hydrodynamic Finite Element Code with Interactive Rezoning, University of California, Lawrence Livermore National Laboratory, Rept. UCID-18756 (1980).

Hallquist, J.O., User's Manual for DYNA3D and DYNAP (Nonlinear Dynamic Analysis of Solids in Three Dimensions), University of California, Lawrence Livermore National Laboratory, Rept. UCID-19156 (1981).**[a]**

Hallquist, J. O., NIKE3D: An Implicit, Finite-Deformation, Finite-Element Code for Analyzing the Static and Dynamic Response of Three-Dimensional Solids, University of California, Lawrence Livermore National Laboratory, Rept. UCID-18822 (1981).**[b]**

Hallquist, J.O., DYNA3D User's Manual (Nonlinear Dynamic Analysis of Solids in Three Dimensions), University of California, Lawrence Livermore National Laboratory, Rept. UCID-19156 (1982; Rev. 1: 1984; Rev. 2: 1986).

Hallquist, J.O., Theoretical Manual for DYNA3D, University of California, Lawrence Livermore National Laboratory, Rept. UCID-19501 (March, 1983).

Hallquist, J.O., DYNA3D User's Manual (Nonlinear Dynamic Analysis of Solids in Three Dimensions), University of California, Lawrence Livermore National Laboratory, Rept. UCID-19156 (1988, Rev. 4).

Hallquist, J.O., LS-DYNA User's Manual (Nonlinear Dynamic Analysis of Solids in Three Dimensions), Livermore Software Technology Corporation, Rept. 1007 (1990).

Hallquist, J.O., D.J. Benson, and G.L. Goudreau, "Implementation of a Modified Hughes-Liu Shell into a Fully Vectorized Explicit Finite Element Code," Proceedings of the International Symposium on Finite Element Methods for Nonlinear Problems, University of Trondheim, Trondheim, Norway (1985).

Hallquist, J.O. and D.J. Benson, "A Comparison of an Implicit and Explicit Implementation of the Hughes-Liu Shell," Finite Element Methods for Plate and Shell Structures, T.J.R. Hughes and E. Hinton, Editors, 394-431, Pineridge Press Int., Swanea, U.K. (1986).

REFERENCES

Hallquist, J.O. and D.J. Benson, DYNA3D User's Manual (Nonlinear Dynamic Analysis of Solids in Three Dimensions), University of California, Lawrence Livermore National Laboratory, Rept. UCID-19156 (Rev. 2: 1986; Rev. 3: 1987).

Hallquist, J.O., D.W. Stillman, T.J.R. Hughes, C. and Tarver,"Modeling of Airbags Using MVMA/DYNA3D," LSTC Report (1990).

Hashin, Z, "Failure Criteria for Unidirectional Fiber Composites," *Journal of Applied Mechanics*, 47, 329 (1980).

Hatchard, T.D., D. D. MacNeil, A. Basu, and J. R. Dahn, "Thermal Model of Cylindrical and Prismatic Lithium-Ion Cells", *J. of the Electrochemical Society*, 2001

Haßler, M., Schweizerhof, K., "On the influence of fluid-structure-interaction on the static stability of thin walled shell structures", *International Journal of Structural Stability*, Vol. 7, pp. 313–335, (2007).

Haßler, M., Schweizerhof, K., "On the static interaction of fluid and gas loaded multi-chamber systems in a large deformation finite element analysis", *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, pp. 1725–1749, (2008).

Hänsel, C., P. Hora, and J. Reissner, "Model for the Kinetics of Strain-Induced Martensitic Phase Transformation at Isothermal Conditions for the Simulation of Sheet Metal Forming Processes with Metastable Austenitic Steels," Simulation of Materials Processing: Theory, Methods, and Applications, Huétink and Baaijens (eds), Balkema, Rotterdam, (1998).

Haward, R.N., and Thackray, G., "The use of a mathematical model to describe isothermal stress-strain curves in glassy thermoplastics". *Proc Roy Soc A*, 302, 453-472 (1968).

Herrmann, L.R. and F.E. Peterson, "A Numerical Procedure for Viscoelastic Stress Analysis," Seventh Meeting of ICRPG Mechanical Behavior Working Group, Orlando, FL, CPIA Publication No. 177, 1968.

Hill A.V., "The heat of shortening and the dynamic constants of muscle," *Proc Roy Soc* B126:136-195, (1938).

Hill, R., "A Theory of the Yielding and Plastic Flow of Anisotropic Metals," *Proceedings of the Royal Society of London*, Series A., Vol. 193, pp. 281-197 (1948).

Hill, R., "Aspects of Invariance in Solid Mechanics," *Advances in Applied Mechanics*, Vol. 18, pp. 1-75 (1979).

Hill, R., "Constitutive Modeling of Orthotropic Plasticity in Sheet Metals," *J. Mech. Phys. Solids*, Vol. 38, No. 3, 1989, pp. 405-417.

Hippchen, P., Merklein, M., Lipp, A., Fleischer, M., Grass, H., Craighero, P., "Modelling kinetics of phase transformation for the indirect hot stamping process", *Key Engineering Materials*, Vol. 549, pages 108-116, (2013).

Hippchen, P., "Simulative Prognose der Geometrie indirekt pressgehärteter Karosseriebauteile für die industrielle Anwendung", Dissertation, Technische Fakultät der Friedrich-Alexander Universität Erlangen-Nürnberg, Meisenbach Verlag Bamberg, Band 249, (2014).

Hirth, A., P. Du Bois, and K. Weimar, "Improvement of LS-DYNA Material Law 83 (Fu Chang) for the Industrial Simulation of Reversible Energy-Absorbing Foams," CAD-FEM User's Meeting, Bad Neuenahr - Ahrweiler, Germany, October 7-9, Paper 2-40, (1998).

Hollenstein M., M. Jabareen, and M.B. Rubin, "Modelling a smooth elastic-inelastic transition with a strongly objective numerical integrator needing no iteration", Comput. Mech., Vol. 52, pp. 649-667, DOI 10.1007/s00466-013-0838-7 (2013).

Hollenstein M., M. Jabareen, and M.B. Rubin, "Erratum to: Modelling a smooth elastic-inelastic transition with a strongly objective numerical integrator needing no iteration", Comput. Mech., DOI 10.1007/s00466-014-1099-9 (2014).

Holmquist, T.J., G.R. Johnson, and W.H. Cook, "A Computational Constitutive Model for Concrete Subjected to Large Strains, High Strain Rates, and High Pressures", Proceedings 14th International Symposium on Ballistics, Quebec, Canada, pp. 591-600, (1993).

Hopperstad, O.S. and Remseth, S.," A return Mapping Algorithm for a Class of Cyclic Plasticity Models", *International Journal for Numerical Methods in Engineering*, Vol. 38, pp. 549-564, (1995).

Huang, Yuli, "Simulating the Inelastic Seismic Behavior of Steel Braced Frames Including the Effects of Low-Cycle Fatigue", Doctoral Dissertation, University of California, Berkeley (2009). Permalink: https://escholarship.org/uc/item/3dr054cx.

Hudson, C.C., "Sound Pulse Approximatons to Blast Loading (with comments on transient drag)", Rept. SC-TM-191-55-51, Sandia Corporation (1955).

Hughes, T.J.R. and E. Carnoy, "Nonlinear Finite Element Shell Formulation Accounting for Large Membrane strains," *AMD*-Vol.48, ASME, 193-208 (1981).

Hughes, T.J.R. and W.K. Liu, "Nonlinear Finite Element Analysis of Shells: Part I. Three-Dimensional Shells." *Comp. Meths. Appl. Mechs.*, 27, 331-362 (1981a).

Hughes, T.J.R. and W.K. Liu, "Nonlinear Finite Element Analysis of Shells: Part II. Two-Dimensional Shells." *Comp. Meths. Appl. Mechs.*, 27, 167-181 (1981b).

REFERENCES

Hughes, T.J.R., W.K. Liu, and I. Levit, "Nonlinear Dynamics Finite Element Analysis of Shells." Nonlinear Finite Element Analysis in Struct. Mech., Eds. W. Wunderlich, E. Stein, and K.J. Bathe, Springer-Verlag, Berlin, 151- 168 (1981c).

Huh, H., Chung, K., Han, S.S., and Chung, W.J., "The NUMISHEET 2011 Benchmark Study of the 8th International Conference and Workshop on Numerical Simulation of 3D Sheet Metal Forming Processes, Part C Benchmark Problems and Results", p 171-228, Seoul, Korea, August, 2011.

Huh, H. and Kang, W.J., "Crash-Worthiness Assessment of Thin-Walled Structures with the High-Strength Steel Sheet", *Int. Journal of Vehicle Design*, Vol. 30, Nos. 1/2 (2002).

Ibrahimbegovic, A. and Wilson, E.L. "A unified formulation for triangular and quadrilateral flat shell finite elements with six nodal degrees of freedom", *Comm. Applied Num. Meth*, 7, 1-9 (1991).

Isenberg, J., Vaughan, D.K., Sandler, I.S., Nonlinear Soil-Structure Interaction, Electric Power Research Institute report EPRI NP-945, Weidlinger Associates (1978).

Ivanov, I. and A. Tabiei, "Flexible Woven Fabric Micromechanical Material Model with Fiber Reorientation," Mechanics of Advanced Materials and Structures, Vol. 9, January 2002.

Ivanov, I., and A. Tabiei, "Loosely Woven Fabric Model With Viscoelastic Crimped Fibers for Ballistic Impact Simulations", IJNME, 57, (2004).

Jabareen, M., "Strongly objective numerical implementation and generalization of a unified large inelastic deformation model with a smooth elastic-inelastic transition", submitted to Int. J. Solids and Struct. (2015).

Jabareen, M., and Rubin, M.B., A Generalized Cosserat Point Element (CPE) for Isotropic Nonlinear Elastic Materials including Irregular 3-D Brick and Thin Structures, J. Mech. Mat. And Struct., Vol 3-8, 1465-1498 (2008).

Jabareen, M., Hanukah, E. and Rubin, M.B., A Ten Node Tetrahedral Cosserat Point Element (CPE) for Nonlinear Isotropic Elastic Materials, J. Comput. Mech. 52, 257-285 (2013).

Johnson, G.C. and D.J. Bammann, "A discussion of stress rates in finite deformation problems," *Int. J. Solids Struct*, 20, 725-737 (1984).

Johnson, G.R. and W.H. Cook, "A Constitutive Model and Data for Metals Subjected to Large Strains, High Strain Rates and High Temperatures." Presented at the Seventh International Symposium on Ballistics, The Hague, The Netherlands, April 1983.

Johnson, G.R. and T.J. Holmquist, "An Improved Computational Model for Brittle Materials" in High-Pressure Science and Technology - 1993 American Institute of Physics Conference Proceedings 309 (c 1994) pp.981-984 ISBN 1-56396-219-5.

Jones, R.M., Mechanics of Composite Materials, Hemisphere Publishing Corporation, New York, (1975).

Kanok-Nukulchai, W., "A Simple and Efficient Finite Element for General Shell Analysis", International Journal for Numerical Methods in Engineering, Vol. 14, pp. 179-200 (1979).

Kelly A., Stebner A.P. and Bhattacharya K., "A micromechanics-inspired constitutive model for shape-memory alloys that accounts for initiation and saturation of phase transformation," Journal of the Mechanics and Physics of Solids, 97, 197–224 (2016).

Kenchington, G.J., "A Non-Linear Elastic Material Model for DYNA3D," Proceedings of the DYNA3D Users Group Conference, published by Boeing Computer Services (Europe) Limited (1988).

Key, S.W. HONDO – A Finite Element Computer Program for the Large Deformation Dynamic Response of Axisymmetric Solids, Sandia National Laboratories, Albuquerque, N.M., Rept. 74-0039 (1974).

Kim, G., Pesaran, A., and Spotnitz, R., "A Three-dimensional thermal abuse model for lithium-ion cells", *J. of Power Resources*, 2007.

Kolling, S., Haufe, A., Feucht, M., DuBois, P. A. "SAMP-1: A Semi-Analytical Model for the Simulation of Polymers", 4. LS-DYNA Anwenderforum, October 20-21, Bamberg, Germany, (2005).

Kolling, S., Hirth, A., Erhart, and Du Bois P.A., Private Communication, Livermore, California (2006).

Krieg, R.D.,A Simple Constitutive Description for Cellular Concrete, Sandia National Laboratories, Albuquerque, NM, Rept. SC-DR-72-0883 (1972).

Krieg, R.D. and S.W. Key, "Implementation of a Time Dependent Plasticity Theory into Structural Computer Programs," Vol. 20 of Constitutive Equations in Viscoplasticity: Computational and Engineering Aspects (American Society of Mechanical Engineers, New York, N.Y., pp. 125-137 (1976).

Lademo, O.G., Berstad, T., Tryland, T., Furu, T., Hopperstad, O.S. and Langseth, M., "*A model for process-based crash simulation*", 8th International LS-DYNA User's Conference, Detroit, May 3-5, 2004.

REFERENCES

Lademo, O.G., Hopperstad, O.S., Berstad, T. and Langseth M., "Prediction of Plastic Instability in Extruded Aluminum Alloys Using Shell Analysis and a Coupled Model of Elasto-plasticity and Damage," Journal of Materials Processing Technology, 2002 (Article in Press).

Lademo, O.G., Hopperstad, O.S., Malo, K.A. and Pedersen, K.O., "Modelling of Plastic Anisotropy in Heat-Treated Aluminum Extrusions", Journal of Materials Processing Technology **125-126**, pp. 84-88 (2002).

Lee, E.L. and C.M. Tarver, "Phenomenological Model of Shock Initiation in Heterogenous Explosives," PHYS. Fluids, Vol. 23, p. 2362 (1980).

Lee, Y. L. and S Balur of Chrysler Group LLC, "Method of predicting spot weld failure" (Attorney Docket No 708494US2), filed on December 22, 2011 and assigned US Serial No. 13/334,701.

Lehane, B. & Simpson, B., "Modelling glacial till conditions using a Brick soil model", Can. Geotech. J. Vol. 37, No. 5, 1078–1088 (2000).

Lemaitre, J., A Course on Damage Mechanics, Springer-Verlag, (1992).

Lemaitre, J., and Chaboche, J.L., Mechanics of Solid Materials, Cambridge University Press, (1990).

Lemmen, P. P. M. and Meijer, G. J., "Failure Prediction Tool Theory and User Manual," TNO Report 2000-CMC-R0018, (2001).

Lewis, B.A., "Developing and Implementing a Road Side Safety Soil Model into LS-DY-NA," FHWA Research and Development Turner-Fairbank Highway Research Center, (1999).

Li, Y.H. and Sellars, C.M., "Modeling Deformation Behavior of Oxide Scales and their Effects on Interfacial Heat Transfer and Friction during Hot Steel Rolling", Proc. Of the 2nd Int. Conf. Modeling of Metals Rolling Processes, The Insitute of Materials, Londong, UK, 192-201 (1996).

Lian, W., personal communication: "LS-DYNA Airbag Module Improvement Request", General Motors Corporation (2000).

Lindström P.R.M, "DNV Platform of Computational Welding Mechanics", Proc. of Int. Inst. Welding 66th Annual Assembly (2013).

Lindström, P., "Improved CWM platform for modelling welding procedures and their effects on structural behaviour", PhD Thesis, Production Technology, University West, Trollhättan, Sweden (2015).

Liu, Z. and C.T. Wu, "Exploring the 3D architectures of deep material network in data-driven multiscale mechanics," Journal of the Mechanics and Physics of Solids, Vol. 127, pp. 20-46, (2019).

Liu, Z., C.T. Wu, and M. Koishi, "A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials," Computer Methods in Applied Mechanics and Engineering, Vol. 345, pp. 1138-1168, (2019).

Liu, Z., C.T. Wu, and M. Koishi, "Transfer learning of deep material network for seamless structure-property predictions," Computer Mechanics, Vol. 64, pp. 451-465, (2019).

Liu, Z., H. Wei, T. Huang, and C.T. Wu, "Intelligent multiscale simulation based on process-guided composite database," (2020). https://arxiv.org/abs/2003.09491

Løkke, A., and A. Chopra, "Direct-Finite-Element Method for Nonlinear Earthquake Analysis of Concrete Dams Including Dam–Water–Foundation Rock Interaction," Pacific Earthquake Engineering Research Center (PEER), Report No. 2019/02, (2019). https://doi.org/10.55461/CRJY2161

Luo, Y. "An Efficient 3D Timoshenko Beam Element with Consistent Shape Functions" *Adv. Theor. Appl. Mech.*, 1(3), 95-106, (2008).

MacNeil, D.D. and J. R. Dahn, J.R., "Test of Reaction Kinetics Using Both Differential Scanning and Accelerating Rate Calorimetries as Applied to the Reaction of LixCoO2 in Non-aqueous Electrolyte", *J. Phys Chem.*, 2001.

MADYMO3D USER'S MANUAL, Version 4.3, TNO Road-Vehicles Research Institute, Department of Injury Prevention, The Hague, The Netherlands, (1990).

Maimí, P., Camanho, P.P., Mayugo, J.A., Dávila, C.G., "A continuum damage model for composite laminates: Part I – Constitutive model", *Mechanics of Materials*, Vol. 39, pp. 897–908, (2007).

Maimí, P., Camanho, P.P., Mayugo, J.A., Dávila, C.G., "A continuum damage model for composite laminates: Part II – Computational implementation and validation", *Mechanics of Materials*, Vol. 39, pp. 909–919, (2007).

Maker, B.N., Private communication Lawrence Livermore National Laboratory, Dr. Maker programmed and implemented the compressible Mooney Rivlin rubber model (1987).

Makris N. and Zhang, J., "Time-domain visco-elastic analysis of earth structures," *Earthquake Engineering and Structural Dynamics*, vol. 29, pp. 745–768, (2000).

REFERENCES

Malvar, L.J., Crawford, J.E., Morrill, K.B., K&C Concrete Material Model Release III — Automated Generation of Material Model Input, K&C Technical Report TR-99-24-B1, 18 August 2000 (*Limited Distribution*).

Malvar, L.J., Crawford, J.E., Wesevich, J.W., Simons, D., "A Plasticity Concrete Material Model for DYNA3D," *International Journal of Impact Engineering*, Volume 19, Numbers 9/10, pages 847-873, December 1997.

Malvar, L.J., and Ross, C.A., "Review of Static and Dynamic Properties of Concrete in Tension," *ACI Materials Journal*, Volume 95, Number 6, pages 735-739, November-December 1998.

Malvar, L.J., and Simons,D., "Concrete Material Modeling in Explicit Computations," Proceedings, Workshop on Recent Advances in Computational Structural Dynamics and High Performance Computing, USAE Waterways Experiment Station, Vicksburg, MS, pages 165-194, April 1996. (LST may provide this reference upon request.)

Malvar, H.S., Sullivan, G.S., and Wornell, G.W., "Lapped Orthogonal Vector Quantization", in Proc. Data Compression Conference, Snowbird, Utah, 1996.

Marin, E.B., Bamman, D.J., Regueiro, R.A., and Johnson, G.C., "On the Formulation, Parameter Identification, and Numerical Integration of the EMMI Model: Plasticity and Isotropic Damage," Sandia Report, Sand 2006-0200, Sandia National Laboratory, CA (2006).

Matzenmiller, A. and Burbulla, F., " Robustheit und Zuverlässigkeit der Berechnungsmethoden von Klebverbindungen mit hochfesten Stahlblechen unter Crashbedingungen" (2013), http://www.ifm.maschinenbau.uni-kassel.de/~amat/publikationen/p75_p828-modellerweiterung.pdf

Matzenmiller, A., Lubliner, J., and Taylor, R.L., "A Constitutive Model for Anisotropic Damage in Fiber-Composites," Mechanics of Materials, Vol. 20, pp. 125-152 (1995).

Matzenmiller, A. and J. K. Schweizerhof, "Crashworthiness Considerations of Composite Structures – A First Step with Explicit Time Integration in Nonlinear Computational Mechanics–State-of-the-Art," Ed. P. Wriggers, W. Wagner, Springer Verlay, (1991).

Mauldin, P.J., R.F. Davidson, and R.J. Henninger, "Implementation and Assessment of the Mechanical-Threshold-Stress Model Using the EPIC2 and PINON Computer Codes," Report LA-11895-MS, Los Alamos National Laboratory (1990).

Maurer, A., Gebhardt, M., Schweizerhof, K., "Computation of fluid and/or gas filled inflatable dams", LS-Dyna Forum, Bamberg, Germany, (2010).

McCormick, P.G., "Theory of flow localization due to dynamic strain ageing," *Acta Metallurgica*, 36, 3061-3067 (1988).

Mi Y., Crisfield, M.A., Davies, A.O. Progressive delamination using interface elements. *J Compos Mater*, 32(14)1246-72 (1998).

Michael, L. and N. Nikiforakis, "A hybrid formulation for the numerical simulation of condensed phase explosives," J. Comp. Phys. 316 (2016) 193-217.

Moran, B., Ortiz, M. and Shih, C.F., "Formulation of implicit finite element methods for multiplicative finite deformation plasticity". *Int J for Num Methods in Engineering*, 29, 483-514 (1990).

Morrison, F. A., An Introduction to Fluid Mechanics. Cambridge University Press, New York, 2013. This correlation appears in Figure 8.13 on page 625.

de Moura MFSF, Gonçalves, J.P., Marques, A.T., and de Castro, P.T., Elemento finito isoparamétrico de interface para problemas tridimensionais. *Revista Internacional de Métodos Numéricos Para Cálculo e Diseño en Ingeniería*, 14:447-66 (1996).

Murray, Y.D., Users Manual for Transversely Isotropic Wood Model APTEK, Inc., Technical Report to the FHWA (to be published) (2002).

Murray, Y.D. and Lewis, B.A., Numerical Simulation of Damage in Concrete APTEK, Inc., Technical Report DNA-TR-94-190, Contract DNA 001-91-C-0075, Defense Nuclear Agency, Alexandria VA 22310.

Murray, Y.D., Users Manual for LS-DYNA Concrete Material Model 159, Report No. FH-WA-HRT-05-062, Federal Highway Administration, (2007).

Murray, Y.D., A. Abu-Odeh, and R. Bligh, Evaluation of Concrete Material Model 159, Report No. FHWA-HRT-05-063, Federal Highway Administration, (2007).

Muscolini, G., Palmeri, A. and Ricciardelli, F., "Time-domain response of linear hysteretic systems to deterministic and random excitations," *Earthquake Engineering and Stuctrual Dynamics*, vol. 34, pp. 1129–1147, (2005).

Nagararaiah, Reinhorn, & Constantinou, "Nonlinear Dynamic Analysis of 3-D Base-Isolated Structures", *Jounal of Structural Engineering* Vol 117, No 7, (1991).

Nahshon, K. and Hutchinson, J.W., "Modification of the Gurson Model for shear failure", *European Journal of Mechanics A/Solids*, Vol. 27, 1-17, (2008).

Naik D., Sankaran S., Mobasher D., Rajan S.D., Pereira J.M., "Development of reliable modeling methodologies for fan blade out containment analysis – Part I: Experimental studies", International Journal of Impact Engineering, Volume 36, Issue 1, Pages 1-11, (2009)

REFERENCES

Neal, M.O., C-H Lin, and J. T. Wang, "Aliasing Effects on Nodal Acceleration Output from Nonlinear Finite Element Simulations," ASME 2000 International Mechanical Engineering Congress and Exposition, Orlando, Florida, November 5-10, (2000).

Neilsen, M.K., H.S. Morgan, and R.D. Krieg, "A Phenomenological Constitutive Model for Low Density Polyurethane Foams," Rept. SAND86-2927, Sandia National Laboratories, Albuquerque, N.M., (1987).

Neukamm, F., Feucht, M., Haufe, A., Roll, K., "On Closing the Constitutive Gap between Forming and Crash Simulation", 10th International LS-DYNA Users Conference, Dearborn, MI, (2008).

Nie, J., R. Morante, M. Mirand, and J. Braverman, "On the Correct Application of the 100-40-40 Rule for Combining Responses Due to Three Directions of Earthquake Loading", Proceedings of the ASME 2010 Pressure Vessels & Piping Division / K-PVP Conference, Bellevue, WA (2010).

Nusholtz, G., W. Fong, and J. Wu, "Air Bag Wind Blast Phenomena Evaluation," *Experimental Techniques*, Nov.-Dec. (1991).

Nusholtz, G., D. Wang, and E.B. Wylie, "Air Bag Momentum Force Including Aspiration," Preprint, Chrysler Corporation, (1996).

Nusholz, private communication, (1996).

Nygards, M., M. Just, and J. Tryding, "Experimental and numerical studies of creasing of paperboard," *Int. J. Solids and Struct.*, 46, 2493-2505 (2009).

Özen, K., "Finite Element Analysis of Prestressed Concrete Slabs Under Impact Loading", Master's Thesis, Aalto University, Finland (2021).

Ogden, R.W., Non-Linear Elastic Deformations, Ellis Horwood Ltd., Chichester, Great Britian (1984).

Oliver, J., "A Consistent Characteristic Length of Smeared Cracking Models," *International Journal for Numerical Methods in Engineering*, **28**, 461-474 (1989).

Ortiz, M. and Pandolfi, A., 1999. "Finite-deformation irreversible cohesive elements for three-dimensional crack-propagation analysis," *International Journal for Numerical Methods in Engineering*, 44(9), pp.1267-1282.

Ottosen N.S., "A failure criterion for concrete," *Journal of the Engineering Mechanics Division*, 103(4):527-35, 1977.

Pack, K., Mohr, D., "Combined necking & fracture model to predict ductile failure with shell finite elements", *Engineering Fracture Mechanics*, Vol. 182, pp. 32-51 (2017).

Papadrakakis, M., "A Method for the Automatic Evaluation of the Dynamic Relaxation Parameters," *Comp. Meth. Appl. Mech. Eng.*, Vol. 25, pp. 35-48 (1981).

Park, R. and Paulay, T., (1975) Reinforced Concrete Structures, J. Wiley and Sons, New York.

Park, Y.J., Wen, Y.K, and Ang, A.H-S, "Random Vibration of Hysteretic Systems Under Bi-directional Ground Motions", *Earthquake Engineering and Structural Dynamics*, Vol. 14, pp. 543-557 (1986).

Penelis, G.G. and Kappos, A.J., Earthquake-Resistant Concrete Structures, E&FN Spon., (1997).

Perzyna, P., "Fundamental problems in viscoplasticity", *Rec. Adv. Appl. Mech.*, Vol. 9, 243-377 (1966).

Pijaudier-Cabot, G., and Bazant, Z.P., "Nonlocal Damage Theory," *Journal of Engineering Mechanics*, ASCE, Vol. 113, No. 10, 1512-1533 (1987).

Pinho, S.T., Iannucci, L., Robinson, R., "Physically-based failure models and criteria for laminated fibre-reinforced composites with emphasis on fibre kinking: Part I: Development", *Composties Part A*, Vol. 37, 63-73 (2006).

Pinho, S.T., Iannucci, L., Robinson, R., "Physically-based failure models and criteria for laminated fibre-reinforced composites with emphasis on fibre kinking: Part II: FE implementation", *Composties Part A*, Vol. 37, 766-777 (2006).

Porcaro, R., A.G. Hanssen, A. Aalberg and M. Langseth, "The behaviour of aself-piercing riveted connection under quasi-static loading conditions," *Int. J. Solids and Structures,* Vol. 43/17, pp. 5110-5131 (2006).

Porcaro, R., A.G. Hanssen, A. Aalberg and M. Langseth, "Self-piercing riveting process, an experimental and numerical investigation," *Journal of Materials processing Technology*, Vol. 171/1, pp. 10-20 (2006).

Porcaro, R., M. Langseth, A.G. Hanssen, H. Zhao, S. Weyer and H. Hooputra, "Crashworthiness of self-piercing riveted connections," *International Journal of Impact Engineering*, In press, Accepted manuscript (2007).

Puck, A., Kopp, J., Knops, M., "Guidelines for the determination of the parameters in Puck's action plane strength criterion", *Composites Science and Technology*, Vol. 62, 371-378 (2002).

Puso, M.A., "A Highly Efficient Enhanced Assumed Strain Physically Stabilized Hexahedral Element", *Int. J. Numer. Meth. Eng.*, Vol. 49, 1029-1064 (2000).

REFERENCES

Puso, M.A., and Laursen, T.A., "A Mortar segment-to-segment contact method for large deformation solid mechanics", *Comput. Methods Appl. Mech. Engrg*. 193 (2004) 601-629.

Puso, M.A., and Laursen, T.A., "A Mortar segment-to-segment frictional contact method for large deformations", *Comput. Methods Appl. Mech. Engrg*. 193 (2004) 4891-4913.

Puso, M.A. and Weiss, J.A., "Finite Element Implementation of Anisotropic Quasilinear Viscoelasticity Using a Discrete Spectrum Approximation", *ASME J. Biomech. Engng*., 120, 62-70 (1998).

Pelessone, D., Private communication, GA Technologies, P.O. Box 85608, San Diego, CA., Telephone No. 619-455-2501 (1986).

Quapp, K.M. and Weiss, J.A., "Material Characterization of Human Medial Collateral Ligament", *ASME J. Biomech Engng.*, 120, 757-763 (1998).

Reyes, A., O.S. Hopperstad, T. Berstad, and M. Langseth, Implementation of a Material Model for Aluminium Foam in LS-DYNA, Report R-01-02, Restricted, Department of Structural Engineering, Norwegian University of Science and Technology, (2002).

Randers-Pehrson, G. and K. A. Bannister, Airblast Loading Model for DYNA2D and DYNA3D, Army Research Laboratory, Rept. ARL-TR-1310, publicly released with unlimited distribution, (1997).

Rasilo, P., U. Aydin, F. Martin, A. Belahcen, R. Kouhia, and L. Daniel, "Equivalent Strain and Stress Models for the Effect of Mechanical Loading on the Permeability of Ferromagnetic Materials," *IEEE Trans. Magn*, vol. 55, No. 6, pp. 1-4 June 2019.

Richards, G.T., Derivation of a Generalized Von Neuman Psuedo-Viscosity with Directional Properties, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-14244 (1965).

Riedel W., Thoma K., Hiermaier S. and Schmolinske E., "Penetration of reinforced concrete by BETA-B-500", in Proc. 9. ISIEMS, Berlin Strausberg, Mai (1999).

Riedel W., "Beton unter Dynamischen Lasten – Meso- und Makromechanische Modelle" In: Ernst-Mach-Institut, editor. Freiburg: Fraunhofer IRB, ISBN 3-8167-6340-5; 2004.

Ritto-Corrêa M. and Camotim D., "On the arc-length and other quadratic control methods: Established, less known and new implementation procedures", *Comput. Struct.*, 86, pp. 1353-1368 (2008).

Roussis, P.C., and Constantinou, M.C., "Uplift-restraining Friction Pendulum seismic isolation system", *Earthquake Engineering and Structural Dynamics*, 35 (5), 577-593, (2006).

Rudshaug, J., Aasen, K.O., Hopperstad, O.S., Børvik, T., "A physically based strength prediction model for glass" , *International Journal of Solids and Structures*, 285, (2023).

Rumpel, T., Schweizerhof, K., "Volume-dependent pressure loading and its influence on the stability of structures", *International Journal for Numerical Methods in Engineering*, Vol. 56, pp. 211-238, (2003).

Rumpel, T., Schweizerhof, K., "Hydrostatic fluid loading in non-linear finite element analysis", *International Journal for Numerical Methods in Engineering*, Vol. 59, pp. 849–870, (2004).

Rupp, A., Grubisic, V., and Buxbaum, O., Ermittlung ertragbarer Beanspruchungen am Schweisspunkt auf Basis der ubertragbaren Schnittgrossen, FAT Schriftenreihe 111, Frankfurt (1994).

Russell, A.J. and Street, K.N, "Predicting interlaminar fatigue crack growth rates in compressively loaded laminates," Delamination and Debonding of Materials, ASTM STP 1012. Philadelphia: American Society for Testing and Materials, 1989. p. 162-178.

Sala, M.O. Neal, and J.T. Wang, Private Communication, General Motors, May, 2004.

Sackett, S.J., "Geological/Concrete Model Development," Private Communication (1987).

Sahraei, E., Bosco, E., Dixon, B., Lai, B., "Microscale failure mechanisms leading to internal short circuit in Li-ion batteries under complex loading scenarios", *Journal of Power Sources*, Vol. 319, pp. 56-65, (2016).

Sandler, I.S. and D. Rubin, "An Algorithm and a Modular Subroutine for the Cap Model," *Int. J. Numer. Analy. Meth. Geomech.*, 3, pp. 173-186 (1979).

Schedin, E., Prentzas, L. and Hilding D., "Finite Element Simulation of the TRIP-effect in Austenitic Stainless Steel," presented at SAE 2004, SAE Technical paper 2004-01-0885, (2004).

Schulte-Frankenfeld N., Deiters, T., "General Introduction to FATXML – data format", https://www.vda.de/dam/vda/publications/FATXML-Format%20Version%20V1.1/1305643077_de_2066216985.zip, (2016).

Schweizerhof, K. and E. Ramm, "Displacement Dependent Pressure Loads in Nonlinear Finite Element Analyses," *Comput. Struct.*, 18, pp. 1099-1114 (1984).

REFERENCES

Schwer, L.E., "A Viscoplastic Augmentation of the Smooth Cap Model," *Nuclear Engineering and Design*, Vol. 150, pp. 215-223, (1994).

Schwer, L.E., "Demonstration of the Continuous Surface Cap Model with Damage: Concrete Unconfined Compression Test Calibration," LS-DYNA Geomaterial Modeling Short Course Notes, July (2001).

Schwer, L.E., W. Cheva, and J.O. Hallquist, "A Simple Viscoelastic Model for Energy Absorbers Used in Vehicle-Barrier Impact," in Computation Aspects of Contact, Impact, and Penetration, Edited by R.F. Kulak and L.E. Schwer, Elmepress International, Lausanne, Switzerland, pp. 99-117 (1991).

Schwer, L.E. and Y.D. Murray, "A Three-Invariant Smooth Cap Model with Mixed Hardening", *International Journal for Numerical and Analytical Methods in Geomechanics*, Volume 18, pp. 657-688, (1994).

Seeger, F., M. Feucht, T. Frank (DaimlerChrysler AG), and A. Haufe, B. Keding (DY-NAmore GmbH), "An Investigation on Spotweld Modeling for Crash Simulation with LS-DYNA", 4th LS-DYNA-Forum, Bamburg, Germany, October (2005), ISBN 3-9809901-1-7.

Shi, M.F., and Gelisse, S., "Issues on the AHSS Forming Limit Determination", International Deep Drawing Research Group (IDDRG), Porto, Portugal, June, 2006.

Shi, M.F., Zhu, X.H., Xia, C., and Stoughton T., "Determination of Nonlinear Isotropic/Kinematic Hardening Constitutive Parameter for AHSS using Tension and Compression Tests", p. 137-142, Proceedings of the 7th International Conference and Workshop on Numerical Simulation of 3D Sheet Metal Forming Processes (NUMISHEET 2008), Interlaken, Switzerland, September, 2008.

Sheppard, S.D., "Estimations of Fatigue Propagation Life in Resistance Spot Welds", ASTM STP 1211, pp. 169-185, (1993).

Sheppard, T. and Wright, D.S., "Determination of flow stress: Part 1 constitutive equation for aluminum alloys at elevated temperatures", *Metals Technology*, p. 215, June 1979.

Shvets, I.T. and Dyban, E., P., "Contact Heat Transfer between Plane Metal Surfaces", *Int. Chem. Eng.*, Vol. 4, No. 4, 621 (1964).

Simo, J.C., J.W. Ju, K.S. Pister, and R.L. Taylor, "An Assessment of the Cap Model: Consistent Return Algorithms and Rate-Dependent Extension", *J. Eng. Mech.*, Vol. 114, No. 2, 191-218 (1988a).

Simo, J.C., J.W. Ju, K.S. Pister, and R.L. Taylor, "Softening Response, Completeness Condition, and Numerical Algorithms for the Cap Model", *Int. J. Numer. Analy. Meth. Eng.*, (in press) (1988b).

Simo, J. C., J.W. Ju, K.S. Pister, and R.L. Taylor, "Softening Response, Completeness Condition, and Numerical Algorithms for the Cap Model", *Int. J. Numer. Analy. Meth. Eng.* (1990).

Simo, J. C. and M.S. Rafai, "A Class of Mixed Assumed Strain Methods and the Method of Incompatible Modes", *Int. J. Numer. Meth. Eng.,* Vol 29, 1595-1638 (1990).

Simo, J.C., F. Armero and R.L. Taylor, "Improved Versions of Assumed Strain Tri-Linear Elements for 3D Finite Deformation Problems", *Comput. Methods in Appl. Mech. Eng.*, Vol 110, 359-386 (1993).

Simpson, B., "Retaining structures: displacement and design", Géotechnique, Vol. 42, No. 4, 539-576, (1992).

Solberg, J.M., and C.M. Noble, "Contact Algorithm for Small-Scale Surface Features with Application to Finite Element Analysis of Concrete Arch Dams with Beveled Contraction Joints", Lawrence Livermore National Laboratory (2002).

Souli, M., R. Messahe, B. Cohen, N. Aquelet, "Numerical Investigation of Phase Change and Cavitation Effects in Nuclear Power Plant Pipes", 13th International LS-DYNA Conference, Dearborn MI, (2014).

Spanos, P.D. and Tsavachidis, S., "Deterministic and stochastic analyses of a nonlinear system with a Biot visco-elastic element", *Earthquake Engineering and Structural Dynamics*, vol. 30, pp. 595–612, (2001).

Steinberg, D.J. and M.W. Guinan, A High-Strain-Rate Constitutive Model for Metals, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-80465 (1978).

Steinberg, D.J. and C.M. Lund, "A Constitutive Model for Strain Rates form 10^{-4} to 10^{6} S^{-1}," *J. Appl. Phys.*, 65, p. 1528 (1989).

Stillman, D.W. and J.O. Hallquist, INGRID: A Three-Dimensional Mesh Generator for Modeling Nonlinear Systems, University of California, Lawrence Livermore National Laboratory, Rept. UCID-20506. (1985).

Stojko, S., privated communication, NNC Limited, Engineering Development Center (1990).

Stahlecker Z., Mobasher B., Rajan S.D., Pereira J.M., "Development of reliable modeling methodologies for engine fan blade out containment analysis. Part II: Finite

element analysis", International Journal of Impact Engineering, Volume 36, Issue 3, Pages 447-459, (2009)

Storakers, B., "On material representation and constitutive branching in finite compressible elasticity", J. Mech. Phy. Solids, 34 No. 2, 125-145 (1986).

Stouffer and Dame, Inelastic Deformation of Metals, Wiley, (1996).

Stout, M.G., D.E. Helling, T.L. Martin, and G.R. Canova, *Int. J. Plasticity*, Vol. 1, pp. 163-174, (1985).

Structural Engineers Association of California, Tentative Lateral Force Requirements, Seismology Committee, SEAOC, 1974, 1990, 1996.

Sturt, R., Montalbini, G. & Jung, H-I., "Developments in *MAT_WINFRITH_CONCRETE and Application to Modelling Tunnel Linings", 14th European LS-DYNA Conference, 2023.

Sussman, T. and Bathe, K.J., "A Finite Element Formulation for Nonlinear Incompressible Elastic and Inelastic Analysis," *Computers & Structures*, **26**, Number 1/2, 357-409 (1987).

Tabiei, A. and G. Nilikantan, "Ballistic Impact of Dry Woven Fabric Composites: A Review," Applied Mechanics Review, January 2008, Vol. 61, Issue 1, p. 10801.

Tabiei, A. and I. Ivanov, "Computational micro-mechanical Model of Flexible Woven Fabric for Finite Element Impact Simulation," IJNME, 53, (6), 1259-1276, (2002).

Tabiei, A., Development and Implementation of *MAT_278 into LS-DYNA, Private Communications, 2016.

Tabiei, A., Implementation of elastic/viscoelastic layered *MAT_076 into LS-DYNA, Private Communications, 2000.

Tabiei, A., Implementation of LAMSHL into *CONTROL_SHELL, Private Communications, 2001.

Tabiei, A., Implementation of *MAT_141 into LS-DYNA, Private Communications, 2005.

Taerwe L, Matthys S., "Fib model code for concrete structures", CEB-FIP, 2010.

Tahoe User Guide, Sandia National Laboratory, can be downloaded from: www.sandia.gov, Input version 3.4.1, (2003).

Taiebat M. and Dafalias, Y. F. "SANISAND: simple anisotropic sand plasticity model." *International Journal for Numerical and Analytical Methods in Geomechanics*, 32(8), 915–948 (2008).

Tanov, R. and A. Tabiei, "Adding transverse normal stresses to layered shell finite elements for the analysis of composite structures," Composite Structures, Vol. 76, Issue 4, December 2006.

Tao, C., Mukhopadhyay, S., Zhang, B., Kawashita, L.F., Qiu, J., and Hallett, S.R. (2018), "An improved delamination fatigue cohesive interface model for complex 3D multi-interface cases," Compos Part A 2018, 107: 633–46.

Tarver, C.M., "Ignition and Growth modeling of LX-17 hockey puck experiments," Propellants Explos. Pyrotech. 30 (2) (2005) 109–117.

Tarver, C.M. and Hallquist, J.O., "Modeling two-dimensional shock initiation and detonation-wave phenomena in PBX 9404 and LX-17," Proceedings of the 7th Symposium on Detonation, p. 488, June 1981.

Tarver, C. M. and P. Urtiew, "Theory and modeling of liquid explosive detonation," J. Energ. Mater. 28 (4) (2010) 299–317.

Taylor, L.M. and D.P. Flanagan, PRONTO3D A Three-Dimensional Transient Solid Dynamics Program, Sandia Report: SAND87-1912, UC-32, (1989).

Taylor, R.L. Finite element analysis of linear shell problems, *in Whiteman, J.R. (ed.),* Proceedings of the Mathematics in Finite Elements and Applications, Academic Press, New York, 191-203, (1987).

Taylor, R.L. and Simo, J.C. Bending and membrane elements for the analysis of thick and thin shells, *Proc. of NUMETA Conference*, Swansea (1985).

Tomek J., Bueno-Orovio A., Passini E., Zhou X., Minchole A., Britton O., Bartolucci C., Severi S., Shrier A., Virag L., Varro A., and Rodriguez B., "Development, calibration, and validation of a novel human ventricular myocyte model in health, disease, and drug block," Elife (2019).

Trovato C, Passini E, Nagy N, Varró A, Abi-Gerges N, Severi S, Rodriguez B., "Human Purkinje in silico model enables mechanistic investigations into automaticity and pro-arrhythmic abnormalities," *J Mol Cell Cardiol*. 2020 May;142:24-38.

Tsai, S.W. and E.M. Wu, "A General Theory of Strength for Anisotropic Materials," J. Composite Materials, 5, pp. 73-96 (1971).

Tuler, F.R. and B.M. Butcher, "A Criterion for the Time Dependence of Dynamic Fracture," *The International Journal of Fracture Mechanics*, Vol. 4, No. 4, (1968).

Tvergaard, V. and J.W. Hutchinson, "The relation between crack growth resistance and fracture process parameters in elastic-plastic solids," *J. of the Mech. And Phy. of Solids*, 40, pp1377-1397, (1992)

REFERENCES

Tvergaard, V. and Needleman, A., "Analysis of the cup-cone fracture in a round tensile bar", *Acta Metallurgica*, 32, 157-169 (1984).

Vawter, D., "A Finite Element Model for Macroscopic Deformation of the Lung," published in the *Journal of Biomechanical Engineering*, Vol.102, pp. 1-7 (1980).

VDA Richtlinier (Surface Interfaces), Version 20, Verband der Automobilindustrie e.v., Frankfurt, Main, Germany, (1987).

Vecchio, F.J. and Collins, M.P., "The Modified Compression-Field Theory for Reinforced Concrete Elements Subjected to Shear", ACI J 83(2) (1986) pp219-231, 1986.

Vegter, H., Horn, C.H.L.J. ten, An, Y., Atzema, E.H., Pijlman, H.H., Boogaard, A.H. van den, Huetink, H., "Characterisation and Modelling of the Plastic Material Behaviour and its Application in Sheet Metal Forming Simulation", *Proceedings of 7th International Conference on Computational Plasticity COMPLAS VII*, Barcelona (2003).

Vegter, H., and Boogaard, A.H. van den, "A plane stress yield function for anisotropic sheet material by interpolation of biaxial stress states", *International Journal of Plasticity* 22, 557-580 (2006).

Walker, J.C., Ratcliffe M.B., Zhang P., Wallace A.W., Fata, B., Hsu E., Saloner D., and Guccione J.M. "MRI-based finite-element analysis of left ventricular aneurysm", Am J Physiol Heart Circ Physiol 289(2): H692:700 (2005).

Wang, J.T. and O.J. Nefske, "A New CAL3D Airbag Inflation Model," SAE paper 880654, 1988.

Wang, J.T., "An Analytical Model for an Airbag with a Hybrid Inflator", Publication R&D 8332, General Motors Development Center, Warren, Mi. (1995).

Wang, J.T., "An Analytical Model for an Airbag with a Hybrid Inflator", *AMD*-Vol. 210, *BED*-Vol. 30, ASME, pp 467-497, (1995).

Wang, K., Y. J. Chao, Y. J., X. Zhu., and K.W. Miller "Dynamic Separation of Resistance Spot Welded Joints: Part II—Analysis of Test Results and a Model" *Exp Mech* Vol 50, Issue 7, pp 901-913 (2010).

Wei, H., C.T. Wu, D. Lyu, W. Hu, F.H. Rouet, K. Zhang, P. Ho, H. Oura, M. Nishi, T. Naito, and L. Shen, "Multiscale simulation of short-fiber-reinforced composites from computational homogenization to mechanistic machine learning in LS-DYNA," 13th European LS-DYNA Conference, Ulm, Germany (2021). Vol. 64, pp. 451-465, (2019). https://www.dynalook.com/conferences/13th-european-ls-dyna-conference-2021.

Wei, H., Wu, C. T., Hu, W., Su, T. H., Oura, H., Nishi, M., Naito, T., Chung, S., Shen, L., "LS-DYNA Machine Learning–Based Multiscale Method for Nonlinear Modeling of Short Fiber–Reinforced Composites." Journal of Engineering Mechanics, 149(3), 04023003, (2023).

Weiss, J.A., Maker, B.N. and Govindjee, S., "Finite Element Implementation of Incompressible, Transversely Isotropic Hyperelasticity", *Comp. Meth. Appl. Mech. Eng.*, 135, 107-128 (1996).

Wen, T.K. "Method for Random Vibration of Hysteretic Systems", *J. Engrg. Mech., ASCE*, Vol. 102, No. EM2, Proc. Paper 12073, pp.249-263 (1976).

Whirley, R. G., and J. O. Hallquist, DYNA3D, A Nonlinear, Explicit, Three-Dimensional Finite Element Code for Solid and Structural Mechanics-Users Manual, Report No.UCRL-MA-107254 , Lawrence Livermore National Laboratory, (1991).

Whirley, R. G., and G.A. Henshall, "Creep Deformation Structural Analysis Using An Efficient Numerical Algorithm," *IJNME*, Vol. 35, pp. 1427-1442, (1992).

Wilkins, M.L., "Calculations of Elastic Plastic Flow," *Meth. Comp. Phys.*, **3**, (Academic Press), 211-263 (1964).

Wilkins, M.L., Calculation of Elastic-Plastic Flow, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-7322, Rev. I (1969).

Wilkins, M.L., The Use of Artificial Viscosity in Multidimensional Fluid Dynamics Calculations, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-78348 (1976)

Wilkins, M.L., R.E. Blum, E. Cronshagen, and P. Grantham, A Method for Computer Simulation of Problems in Solid Mechanics and Gas Dynamics in Three Dimensions and Time, University of California, Lawrence Livermore National Laboratory, Rept. UCRL-51574 (1974).

Wilkins, M.L., J.E. Reaugh, B. Moran, J.K. Scudder, D.F. Quinones, M.E. Prado, Fundamental Study of Crack Initiation and Propagation Annual Progress Report, Report UCRL-52296, Lawrence Livermore National Laboratory, Livermore, CA. (1977).

Wilkins, M.L, Streit, R.D, and Reaugh, J.E. Cumulative-Strain-Damage Model of Ductile Fracture: Simulation and Prediction of Engineering Fracture Tests, Report UCRL-53058, Lawrence Livermore National Laboritory, Livermore, CA (1980).

Williams K. V., Vaziri R., Poursartip A., "A Physically Based Continuum Damage Mechanics Model for Thin Laminated Composite Structures." *Int J Solids Struct*, Vol 40(9), 2267-2300

REFERENCES

Wilson, E.L. *Three Dimensional Static and Dynamic Analysis of Structures,* Computers and Structures, Inc., Berkeley CA, (2000).

Winters, J.M., "Hill-based muscle models: A systems engineering perspective," In Multiple Muscle Systems: Biomechanics and Movement Organization, JM Winters and SL-Y Woo eds, Springer-Verlag (1990).

Winters J.M. and Stark L., "Estimated mechanical properties of synergistic muscles involved in movements of a variety of human joints," *J Biomechanics* 21:1027-1042, (1988).

Woodruff, J.P., KOVEC User's Manual, University of California, Lawrence Livermore National Laboratory, Report UCRL-51079, (1973).

Worswick, M.J., and Xavier Lalbin, Private communication, Livermore, California, (1999).

Wu Y., John E. Crawford, Shengrui Lan, and Joseph M., "Validation studies for concrete constitutive models with blast test data", 13th International LS-DYNA User's Conference, Dearborn MI, (2014).

Xia, Q.S., M.C. Boyce, and D.M. Parks, "A constitutive model for the anisotropic elasticplastic deformation of paper and paperboard," *Int. J. Solids and Struct.*, 39, 4053-4071 (2002).

Yamasaki, H., M. Ogura, R. Nishimura, and K. Nakamura, "Development of Material Model for Crack Propagation of Casting Aluminum", Presented at the 2006 JSAE Annual Congress, Paper Number 20065077, (2006).

Yen, C.F., "Ballistic Impact Modeling of Composite Materials", Proceedings of the 7th International LS-DYNA Users Conference, Dearborn, MI, May 19-21, 2002, 6.15-6.25.

Yoshida, F. and Uemori, T., "A Model of Large-Strain Cyclic Plasticity Describing the Bauschinger Effect and Work Hardening Stagnation", International Journal of Plasticity 18, 661-686 (2002).

Yoshida, F. and Uemori, T., "A Model of Large-Strain Cyclic Plasticity and its Application to Springback Simulation", *International Journal of Mechanical Sciences*, Vol. 45, 1687-1702, (2003).

Zajac F.E., "Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control", *CRC Critical Reviews in Biomedical Engineering* 17(4):359-411, (1989).

Zayas, V.A., Low, S.S. and Mahin, S.A., "A Simple Pendulum Technique for Achieving Seismic Isolation", *J. Earthquake Spectra*, Vol. 6, No. 2, pp. 317-334 (1990).

Zhang, S., Approximate Stress Intensity Factors and Notch Stresses for Common Spot-Welded Specimens, Welding Research Supplement, pp. 173s-179s, (1999).

Zhang, S., McCormick, P.G., Estrin, Y., "The morphology of Portevin-Le Chatelier bands: Finite element simulation for Al-Mg-Si", *Acta Materialia* 49, 1087-1094, (2001).

Zhu, H., Zhu, X., "A Mixed-Mode Fracture Criterion for AHSS Cracking Prediction at Large Strains", SAE Technical paper 2011-01-0007, (2011).

Zhu, X.H., Fan, H.F., Zhang, L., and Xiao Y.Z., Improvement of Sandwich Structure Part Adaptivity in LS-DYNA, 2017 3rd China LS-DYNA Users' Conference, Shanghai, China.

APPENDIX A: User-Defined Materials

GETTING STARTED WITH USER-DEFINED FEATURES

We begin with a general introduction to user-defined features. This section is valid for any user-defined feature described in the remaining appendices and serves as a practical guide to enable you to begin coding. We will specifically discuss the different methods for linking your user-defined subroutines to LS-DYNA. For a comprehensive overview of user-defined features and their applications, please refer to Erhart [2010], which can be seen as a complement to the present text.

Download

First, download a *usermat version* of LS-DYNA for the computer architecture/platform of interest. This version is a compressed package (.tgz, .tar.gz, .zip) provided by Ansys or your local LS-DYNA distributor. The unpacked content contains a usermat directory, possibly including

- precompiled static object libraries (.a, .lib)
- Fortran source code files (.f, .f90)
- include files (.inc)
- makefile
- a shared library binary LS-DYNA executable

Choosing a version to download that is compatible with your computer environment in terms of architecture, operating system, and possible MPI implementation for MPP is essential. Selecting an appropriate version may not be obvious. For questions regarding the version, contact your LS-DYNA distributor. A usermat version does not require a special license but falls under the general LS-DYNA license agreement.

Static or dynamic linking

To get something that is runnable, it is necessary to *compile* the source code files and link the resulting object files either to a binary executable (see Figure 51-1) or a *shared object* (see Figure 51-2). The latter option requires a shared library version of the usermat package and a sharelib binary, while the former assumes a statically linked usermat package. Thus, this choice needs to be made before retrieving the package. Working with shared objects is more flexible because a binary executable can be installed once and for all, and the (small) shared object is dynamically linked at runtime. Upon execution, LS-DYNA will look for the shared object file in directories specified by a library path that is set either by the shell variable LD_LIBRARY_PATH, on the command line with module=, or by the *MODULE_LOAD keyword. The shared object is easily substituted, facilitating portability when working on large projects. A statically linked version, on the other hand,

requires generating the entire binary executable during compilation. This option may be preferred if working on small projects or during the development phase.

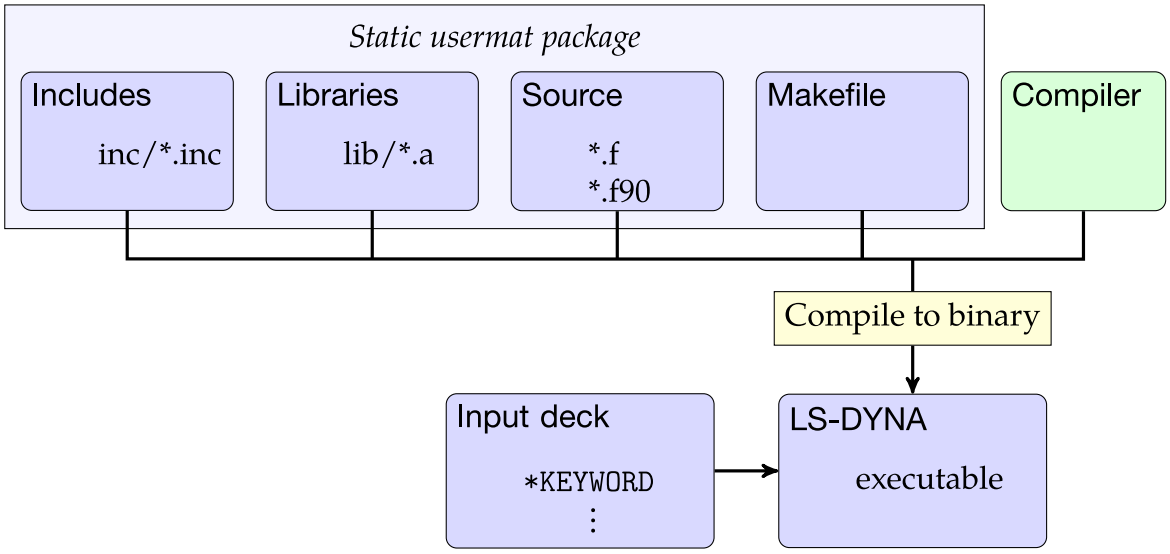


Figure 51-1. Statically linked usermat package

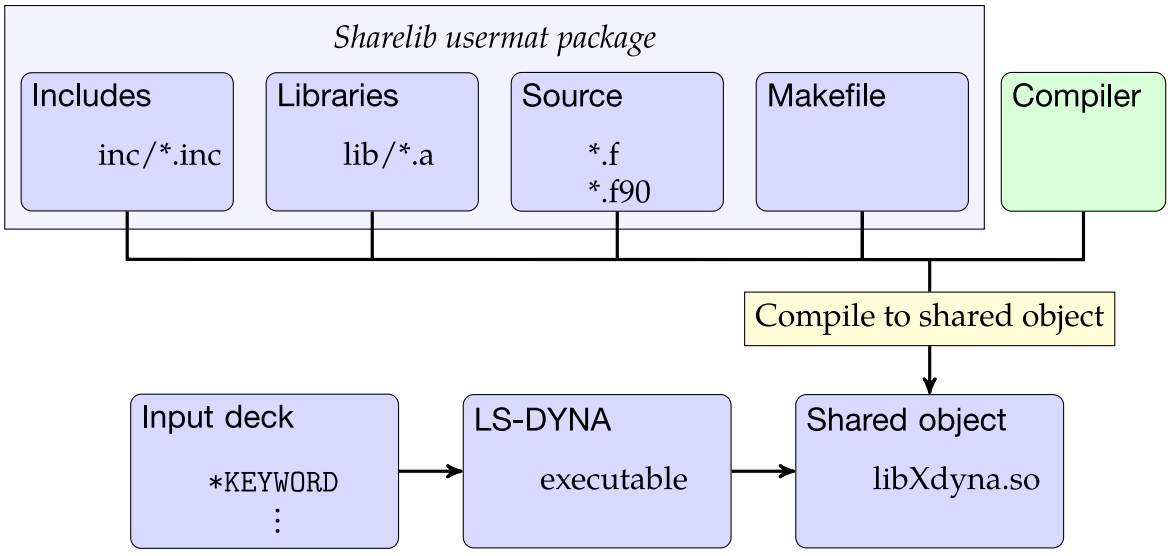


Figure 51-2. Dynamically linked usermat package

Compiler and compiling

The compilation is usually performed in the usermat (working) directory using a terminal window. The usermat package does not contain a compiler; it is assumed that the appropriate compiler is installed on your system and accessible from your working directory. To render compatibility between the pre-compiled libraries or binary executable and your compiled object files, the appropriate compiler is the (by LS-DYNA team) designated compiler with which the pre-compiled libraries or binary are readily compiled.

For example, on Linux, this is typically either the Intel Fortran Compiler (IFC) on Red Hat or CentOS or Portland Groups Compiler (PGF) on SUSE; on Windows, it is the Intel Fortran Compiler (IFC) in combination with Microsoft Visual Studio (MSVS). For MPP, you also need a wrapper for whatever MPI implementation you have installed, but again, consult your distributor for detailed information.

To compile, execute '`make clean; make`' in the terminal window. In most cases, these commands will generate a shared object or binary executable depending on your type of LS-DYNA usermat version. If not, it comes down to interpreting error messages. A possible reason for failure is that the included makefile does not contain appropriate compiler directives and requires some editing. This could range from trivial tasks like altering the path used to find the compiler on your system to more complex endeavors such as adding or changing compiler flags. An even worse scenario is if your operating system is not up-to-date and may require updating your computer's library content. Preferably, your computer administrator resolves such scenarios in collaboration with your distributor.

Execution

When compiling a virgin instance of a usermat version, the generated executable (for static usermat) or the shared object combined with a sharelib executable (for sharelib usermat) form an exact replica of a regular executable. A customized version only generates when the source code is modified. The binary and/or shared object may be left in the working directory, moved to some other location on your system, or properly installed for execution on clusters by a queuing system. While both the executable and shared object may be renamed, a shared object should not be renamed if you use LD_LIBARY_PATH to point to its location. During the development phase, it may be convenient to leave the binary in its place to facilitate debugging but move the shared object to the execution directory to not have to edit the library path for the executable to find it. Two Linux examples on how to run an input file are given below, one assuming an SMP static object version has been downloaded and the other an MPP shared object version. Let `/path_to_my_source/usermat` be the complete path to the working directory and in.k be the name of the keyword input file. If located in the *execution* directory, meaning the directory containing in.k, the problem is run as

```
/path_to_my_source/usermat/lsdyna i=in.k
```

in the SMP case. For the MPP shared object case, you may copy the shared object file to the execution directory, the default directory where executables can look for dynamic object files. Assuming the MPI software is platform-MPI, the input file could be run on two cores as

```
/path_to_platform-mpi/bin/mpirun -np 2  /path_to_my_source/usermat/mppdyna i=in.k
```

These two examples are only intended to indicate how to treat the compiled files, and changes in details thereof are to be expected.

APPENDIX A

Coding

All subroutines for the user-defined features are collected in the files dyn21*.f and are ready for editing using your favorite text editor. Mechanical user materials are contained in dyn21umat*.f; user-defined loads in dyn21.f; user-defined contacts, wear, and friction in dyn21cnt.f; user-defined elements in dyn21usld.f and dyn21ushl.f; and thermal user materials are in dyn21tumat.f, just to give a few examples. The prevailing programming language is Fortran 77, but many compilers support Fortran 90. Writing C code may also be possible, but using C requires some manipulation of the makefile and function interfaces. Each feature is connected to one or a few keywords to properly take advantage of innovative coding. For user materials, this keyword is *MAT_USER_DEFINED_MATERIAL_MODELS and is described in detail below, and for user-defined loads, the corresponding keyword is *USER_LOADING. We refer to the keyword manual pages for details on their respective usage.

Module concept

Here, we delve further into the dynamically linked, shared object/library usermat version and the module concept. A share object usermat package only consists of a makefile and a few source code files. In principle, all you need to do is to (*i*) implement the feature of interest in the source code files, (*ii*) compile to a shared object using the makefile, and (*iii*) use the keyword input file to access the generated object.

There are several advantages to using shared objects:

1. The content and size of a shared object usermat package is significantly reduced compared to a statically compiled usermat. Also, the size of a shared object is small.

2. Several shared objects can be used simultaneously, expanding the number of available user features in a single model.

3. Shared objects can be delivered to or by third parties.

We will illustrate how LS-DYNA, in practice, accesses shared objects and, at the same time, addresses items 2 and 3 above with an example. Due to the modular nature of shared objects, they can also be called *modules*. We will describe how to access these using the *MODULE keyword.

In a shared library version, the keywords

*MODULE_PATH

*MODULE_LOAD

*MODULE_USE

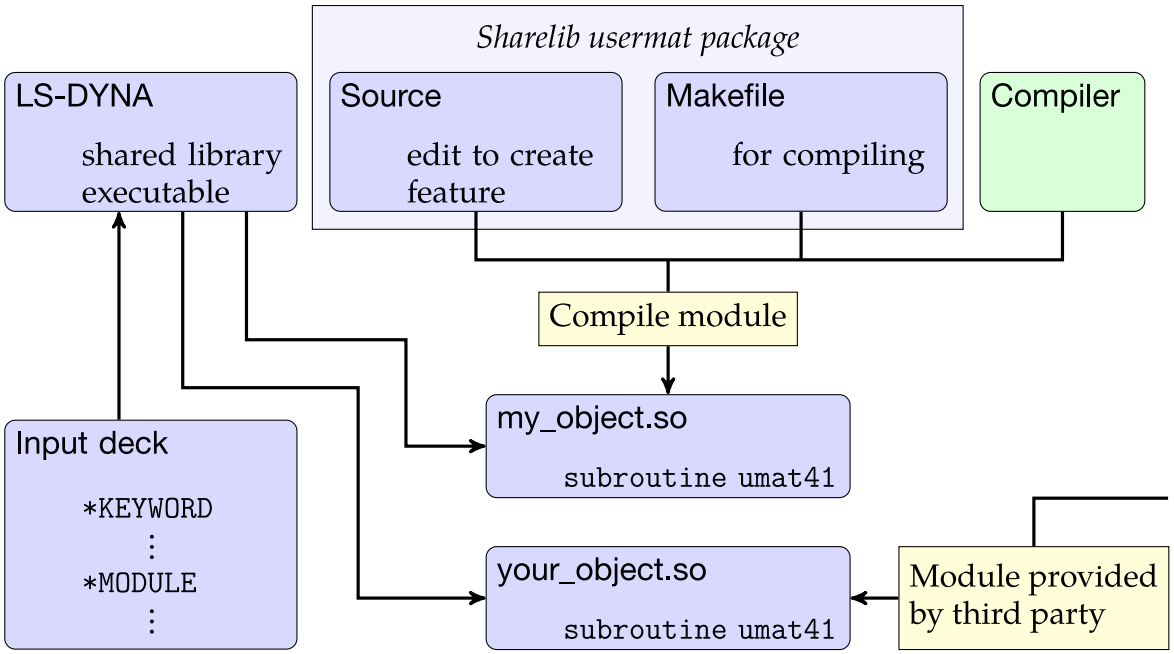


Figure 51-3. Dynamic linking with modules

are available. See the section on *MODULE for more detailed explanations of these keywords. With reference to Figure 51-3, we assume that my_object.so and your_object.so are two independently generated modules, and both are located in directory `/path_to_modules`. We also assume that these two modules contain different implementations for user material 41 (`subroutine umat41`). Without the present approach, executing both implementations with the same executable and keyword input would be at least intricate. A simple way of dealing with this using modules is to use the following set of keywords:

```
*MODULE_PATH
/path_to_modules
*MODULE_LOAD
$ MDLID               TITLE
myid                  my library
$ FILENAME
my_object.so
*MODULE_LOAD
$ MDLID               TITLE
yourid                your library
$ FILENAME
your_object.so
*MODULE_USE
$ MDLID
myid
$ TYPE                PARAM1                PARAM2
UMAT                  1001                  41
*MODULE_USE
$ MDLID
```

```
yourid
$ TYPE                PARAM1                PARAM2
UMAT                  1002                  41
*MAT_USER_DEFINED_MATERIAL_MODELS
$       MID         RO          MT
          1    7.85e-9       1001
…
*MAT_USER_DEFINED_MATERIAL_MODELS
$       MID         RO          MT
          2    7.85e-9       1002
…
*PART
first part
$       PID      SECID        MID
          1          1          1
*PART
second part
$       PID      SECID        MID
          2          2          2
```

Here, *MODULE_PATH lists the path(s) to the shared objects to be loaded, *MODULE_-LOAD loads the dynamic libraries, and *MODULE_USE tells the executable how to access routines in the shared object. In this particular example, a user material (*MAT_-USER_...) with MT set to 1001 (because PARAM1 = 1001) will execute `subroutine umat41` (because PARAM2 = 41) from the module with ID myid (because MDLID=myid). Similarly, user material with MT set to 1002 will also execute subroutine umat41 but now from the module with ID yourid. Hence, we have made parts 1 and 2 execute the *same* subroutine (by name) but from *different* modules. This generalizes to any number of modules by analog. See *MODULE_USE for more details.

GENERAL OVERVIEW OF USER-DEFINED MATERIALS

We now turn to the specific discussion of user-defined materials. You can simultaneously implement up to ten user subroutines to update the stresses in solids, shells, beams, discrete beams, and truss beams. This section serves as an introductory guide to implementing such a model. Note that the names of variables and subroutines below may differ from the actual ones depending on the platform and version of LS-DYNA.

When the keyword *MAT_USER_DEFINED_MATERIAL_MODELS is defined for a part in the keyword deck, LS-DYNA calls the subroutine `usrmat` with appropriate input data for the constitutive update. This routine calls `urmathn` for 2D and 3D solid elements, `urmats` for 2D plane stress and 3D shell elements, `urmatb` for beam elements, `urmatd` for discrete beam elements, and `urmatt` for truss beam elements. You may modify these routines if necessary. These routines initialize the following data structures:

`sig(6)` – stresses in the previous time step
`eps(6)` – strain increments

`epsp`	–	effective plastic strain in the previous time step
`hsv(*)`	–	history variables in the previous time step, excluding plastic strain
`dt1`	–	current time step size
`temper`	-	current temperature
`failel`	–	flag indicating failure of element

A specific *scalar* material subroutine receives these data structures. Suppose you activate the vectorization flag (IVECT = 1 on the material card). In that case, the program stores variables in vector blocks of length `nlq`, with vector indexes ranging from `lft` to `llt`. This storage method allows for more efficient execution of the material routine. As an example, the data structures mentioned above are exchanged in the vectorized case for

`sig`*`X`*`(nlq)`	–	stresses in the previous time step
`d`*`X`*`(nlq)`	–	strain increments
`epsps(nlq)`	–	effective plastic strains in the previous time step
`hsvs(nlq,*)`	–	history variables in the previous time step
`dt1siz(nlq)`	-	current time step sizes
`temps(nlq)`	–	current temperatures
`failels(nlq)`	–	flags indicating failure of elements

where *X* ranges from 1 to 6 for the different components. Each entry in a vector block is associated with an element in the finite element mesh for a fixed integration point.

The number of entries in the history variables array (indicated by `*` above) matches the number of history variables requested on the material card (NHV). Hence, the number NHV should equal the number of history variables excluding the effective plastic strain. We give effective plastic strain special treatment. All history variables, including the effective plastic strain, are initially zero. Furthermore, all user-defined material models require bulk and shear moduli for transmitting boundaries, contact interfaces, rigid body constraints, and time-step calculations. Thus, the length of the material constants array, `LMC`, must, generally, be increased by two to store these parameters. In addition to the variables mentioned above, user-material routines may receive the following data, regardless of whether vectorization is used or not:

`cm(*)`	–	material constants array
`capa`	–	transverse shear correction factor for shell elements
`tt`	–	current time
`crv(lq1,2,*)`	–	array representation of curves defined in the keyword deck

Next, the previously mentioned element-specific routine calls a specific material routine, `umat`*`XX`* in the scalar case or `umat`*`XX`*`v` in the vector case. *XX* is a number between 41 and 50 and matches MT on the material card. You write the specific material subroutine. This subroutine should update the stresses and history variables to the current time. For shells and beams, your subroutine must also determine the strain increments in the directions of constrained zero stress. To be able to write different stress updates for distinct element types, your subroutine receives the following character string:

etype – character string that equals `solid`, `shell`, `beam`, `dbeam`, or `tbeam`

A sample user subroutine of a hypoelastic material in the scalar case is provided below. This sample and the others below are from the dyn21.F file distributed with version R6.1.

Sample user subroutine 41

```
      subroutine umat41 (cm,eps,sig,epsp,hsv,dt1,capa,etype,tt,
     1 temper,failel,crv,nnpcrv,cma,qmat,elsiz,idele,reject)
c
c************************************************************************
c|  Livermore Software Technology Corporation  (LSTC)                    |
c|  ------------------------------------------------------------------  |
c|  Copyright 1987-2008 Livermore Software Tech.  Corp                   |
c|  All rights reserved                                                  |
c************************************************************************
c
c     isotropic elastic material (sample user subroutine)
c
c     Variables
c
c     cm(1)=first material constant, here Young's modulus
c     cm(2)=second material constant, here Poisson's ratio
c        .
c        .
c        .
c     cm(n)=nth material constant
c
c     eps(1)=local x  strain increment
c     eps(2)=local y  strain increment
c     eps(3)=local z  strain increment
c     eps(4)=local xy strain increment
c     eps(5)=local yz strain increment
c     eps(6)=local zx strain increment
c
c     sig(1)=local x  stress
c     sig(2)=local y  stress
c     sig(3)=local z  stress
c     sig(4)=local xy stress
c     sig(5)=local yz stress
c     sig(6)=local zx stress
c
c     hsv(1)=1st history variable
c     hsv(2)=2nd history variable
c        .
c        .
c        .
c        .
c     hsv(n)=nth history variable
c
c     dt1=current time step size
c     capa=reduction factor for transverse shear
c     etype:
c        eq."solid" for solid elements
c        eq."sld2d" for shell forms 13, 14, and 15 (2D solids)
c        eq."shl_t" for shell forms 25, 26, and 27 (shells with thickness
c         stretch)
c        eq."shell" for all other shell elements plus thick shell forms 1
c         and 2
c        eq."tshel" for thick shell forms 3 and 5
c        eq."hbeam" for beam element forms 1 and 11
```

```
c        eq."tbeam" for beam element form 3 (truss)
c        eq."dbeam" for beam element form 6 (discrete)
c        eq."beam " for all other beam elements
c
c     tt=current problem time.
c
c     temper=current temperature
c
c     failel=flag for failure, set to .true.  to fail an integration point,
c            if .true.  on input, the integration point has failed earlier
c
c     crv=array representation of curves in keyword deck
c
c     nnpcrv=# of discretization points per crv()
c
c     cma=additional memory for material data defined by LMCA at
c       6th field of 2nd card of *DATA_USER_DEFINED
c
c     elsiz=characteristic element size
c
c     idele=element id
c
c     reject (implicit only) = set to .true.  if this implicit iterate is
c                              to be rejected for some reason
c
c     All transformations into the element's local system are
c     performed before entering this subroutine.  Transformations
c     back to the global system are performed after exiting this
c     routine.
c
c     All history variables are initialized to zero in the input
c     phase.  Initialization of history variables to nonzero values
c     may be done during the first call to this subroutine for each
c     element.
c
c     Energy calculations for the dyna3d energy balance are done
c     outside this subroutine.
c
      include 'nlqparm'
      include 'bk06.inc'
      include 'iounits.inc'
      dimension cm(*),eps(*),sig(*),hsv(*),crv(lq1,2,*),cma(*)
      integer nnpcrv(*)
      logical failel,reject
      character*5 etype
c
      if (ncycle.eq.1) then
        if (cm(16).ne.1234567) then
          call usermsg('mat41')
        endif
      endif
c
c     compute shear modulus, g
c
      g2 =abs(cm(1))/(1.+cm(2))
      g  =.5*g2
c
      if (etype.eq.'solid'.or.etype.eq.'shl_t'.or.
     1     etype.eq.'sld2d'.or.etype.eq.'tshel') then
        if (cm(16).eq.1234567) then
          call mitfail3d(cm,eps,sig,epsp,hsv,dt1,capa,failel,tt,crv)
        else
          if (.not.failel) then
          davg=(-eps(1)-eps(2)-eps(3))/3.
          p=-davg*abs(cm(1))/(1.-2.*cm(2))
          sig(1)=sig(1)+p+g2*(eps(1)+davg)
```

```
          sig(2)=sig(2)+p+g2*(eps(2)+davg)
          sig(3)=sig(3)+p+g2*(eps(3)+davg)
          sig(4)=sig(4)+g*eps(4)
          sig(5)=sig(5)+g*eps(5)
          sig(6)=sig(6)+g*eps(6)
          if (cm(1).lt.0.) then
            if (sig(1).gt.cm(5)) failel=.true.
          endif
          endif
        end if
c
      else if (etype.eq.'shell') then
        if (cm(16).eq.1234567) then
          call mitfailure(cm,eps,sig,epsp,hsv,dt1,capa,failel,tt,crv)
        else
          if (.not.failel) then
          gc    =capa*g
          q1    =abs(cm(1))*cm(2)/((1.0+cm(2))*(1.0-2.0*cm(2)))
          q3    =1./(q1+g2)
          eps(3)=-q1*(eps(1)+eps(2))*q3
          davg  =(-eps(1)-eps(2)-eps(3))/3.
          p     =-davg*abs(cm(1))/(1.-2.*cm(2))
          sig(1)=sig(1)+p+g2*(eps(1)+davg)
          sig(2)=sig(2)+p+g2*(eps(2)+davg)
          sig(3)=0.0
          sig(4)=sig(4)+g *eps(4)
          sig(5)=sig(5)+gc*eps(5)
          sig(6)=sig(6)+gc*eps(6)
          if (cm(1).lt.0.) then
            if (sig(1).gt.cm(5)) failel=.true.
          endif
          endif
        end if
      elseif (etype.eq.'beam ' ) then
          q1    =cm(1)*cm(2)/((1.0+cm(2))*(1.0-2.0*cm(2)))
          q3    =q1+2.0*g
          gc    =capa*g
          deti  =1./(q3*q3-q1*q1)
          c22i  = q3*deti
          c23i  =-q1*deti
          fac   =(c22i+c23i)*q1
          eps(2)=-eps(1)*fac-sig(2)*c22i-sig(3)*c23i
          eps(3)=-eps(1)*fac-sig(2)*c23i-sig(3)*c22i
          davg  =(-eps(1)-eps(2)-eps(3))/3.
          p     =-davg*cm(1)/(1.-2.*cm(2))
          sig(1)=sig(1)+p+g2*(eps(1)+davg)
          sig(2)=0.0
          sig(3)=0.0
          sig(4)=sig(4)+gc*eps(4)
          sig(5)=0.0
          sig(6)=sig(6)+gc*eps(6)

c
      elseif (etype.eq.'tbeam') then
        q1    =cm(1)*cm(2)/((1.0+cm(2))*(1.0-2.0*cm(2)))
        q3    =q1+2.0*g
        deti  =1./(q3*q3-q1*q1)
        c22i  = q3*deti
        c23i  =-q1*deti
        fac   =(c22i+c23i)*q1
        eps(2)=-eps(1)*fac
        eps(3)=-eps(1)*fac
        davg  =(-eps(1)-eps(2)-eps(3))/3.
        p     =-davg*cm(1)/(1.-2.*cm(2))
        sig(1)=sig(1)+p+g2*(eps(1)+davg)
        sig(2)=0.0
```

```
        sig(3)=0.0
c
      else
c       write(iotty,10) etype
c       write(iohsp,10) etype
c       write(iomsg,10) etype
c       call adios(TC_ERROR)
        cerdat(1)=etype
        call lsmsg(3,MSG_SOL+1150,ioall,ierdat,rerdat,cerdat,0)
      endif
c
c10   format(/
c    1 ' *** Error element type ',a,' can not be',
c    2 '           run with the current material model.')
      return
      end
```

Based on the subroutine `umat41` shown above, the following material input:

```
*MAT_USER_DEFINED_MATERIAL_MODELS
$#     mid        ro        mt       lmc       nhv    iortho     ibulk        ig
         1 7.8300E-6        41         4         0         0         3         4
$#   ivect     ifail    itherm    ihyper      ieos
         0         0         0         0         0
$        E        PR      BULK         G
$#      p1        p2        p3        p4        p5        p6        p7        p8
  2.000000  0.300000  1.667000  0.769200     0.000     0.000     0.000     0.000
```

is functionally equivalent to:

```
*MAT_ELASTIC
$#     mid        ro         e        pr        da        db  not used
         1 7.8300E-6  2.000000  0.300000     0.000     0.000         0
```

ADDITIONAL FEATURES

Load curves and tables

If the material of interest requires load curves, predefined routines easily obtain curve and table data. For instance, your material implementation may need a curve defining yield stress as a function of effective plastic strain. For curves, call

```
subroutine curveval(eid,xval,yval,slope)
```

or

```
subroutine curveval_v(eid,xval,yval,slope,lft,llt)
```

where the former routine works for scalar routines and the latter for vectorized. These subroutines have the following arguments:

`eid` - external load curve ID, meaning the load curve ID taken from the keyword deck

GT.0: Use a rediscretized representation of the curve

LT.0: Use exact representation of the curve (with ID –eid)

xval - abscissa value
yval - ordinate value (output from routine)
slope - slope of curve (output from routine)
lft - first index of vector
llt - final index of vector

where xval, yval, and slope are scalars in the scalar routine and vectors of length nlq in the vectorized routine. Note that eid should be passed as a float. A positive value for eid causes the routine to use a rediscretized representation of the curve. In contrast, a negative value causes the extraction to be made on the curve as it is defined in the keyword input deck.

For tables, we have two subroutines available for extracting values. The scalar version is

```
subroutine tableval(eid,dxval,yval,dslope,xval,slope)
```

and the vector version is

```
      subroutine
     1 tableval_v(eid,dxval,yval,dslope,lft,llt,xval,slope)
```

where

eid - external table ID (data type real), meaning table ID taken from the keyword deck
GT.0: Use a rediscretized representation of the curve
LT.0: Use exact representation of curve (with ID –eid)
dxval - abscissa value (x_2-axis)
yval - ordinate value (y-axis, output from routine)
dslope - slope of curve (dy/dx_2, output from routine)
xval - abscissa value (x_1-axis)
slope - slope of curve (dy/dx_1, output from routine)
lft - vector index
llt - vector index

In the scalar routine, dxval, yval, dslope, xval, and slope are all scalars, whereas, in the vector routine, they are vectors of length nlq. A positive and negative value of eid works in the same way for tables as it does for curves.

Local coordinate system

Invoke the local coordinate system option if the material model has directional properties, such as composite and anisotropic plasticity models. To do this, set IORTHO to 1 on the material card. A local coordinate requires two additional cards for the material with values for forming and updating the coordinate system. When invoked, the constitutive routine, umat*XX* or umat*XX*v, receives all the data in the local system. You do not transform the data to the global system since another routine performs the transformation

unless your model requires the deformation gradient. See Deformation gradient for more details.

Temperature

Setting ITHERMAL to 1 on the material card for a material with thermal properties makes temperature available to your subroutine. The `temper` variable for a scalar and `temps` array for the vectorized implementation contain this data. For a coupled thermal-structural analysis, LS-DYNA solves the thermal problem first, making temperatures at the current time available in the user-defined subroutine. Note that LS-DYNA calculates the dissipated heat in the presence of plastic deformation. Thus, you do not include it in your subroutine. If the stress update requires the time derivative of the temperature, request a history variable that contains the temperature in the previous time step. Adding a backward finite difference estimate to your subroutine gives the time derivative.

Failure

Your material model may include failure, resulting in deleting elements that fulfill a failure criterion. To accomplish this, set IFAIL to 1 or a negative number on the material card. For a scalar implementation, the variable `failel` is set to `.true.` when a failure criterion is met. For a vectorized implementation, the corresponding entry in the `failels` array is set to `.true.`

Deformation gradient

For some material models, the stresses depend on the deformation gradient, **F**, instead of incremental strains. For instance, this dependency applies to hyperelastic and hyperplastic materials. To make the deformation gradient available for bricks and shells in the user-defined material subroutines, set IHYPER to 1 on the material card. LS-DYNA puts the deformation gradient components F_{11}, F_{21}, F_{31}, F_{12}, F_{22}, F_{32}, F_{13}, F_{23}, and F_{33} in the history variables array in positions NHV+1 to NHV+9, that is, the positions right after the requested number of history variables.

For shell elements, the components of the deformation gradient are with respect to the co-rotational system for the element. Note that the user-defined material routine needs to properly update the third row of the deformation gradient, components F_{31}, F_{32}, and F_{33}. These components depend on the thickness strain increment, which must be determined so that the normal stress in the shell vanishes. For a given thickness strain increment `d3`, the following subroutines can determine these three components, `f31`, `f32` and `f33`:

```
subroutine compute_f3s(f31,f32,f33,d3)
```

for a scalar implementation and

```
subroutine compute_f3(f31,f32,f33,d3,lft,llt)
```

for a vector implementation. The first four arguments are scalars for the scalar routine and arrays of length `nlq` for the vector routine.

For hyperelastic materials, the user-defined subroutines can call push-forward operations. These routines are

```
subroutine push_forward_2(sig1,sig2,sig3,sig4,sig5,sig6,
      f11,f21,f31,f12,f22,f32,f13,f23,f33,lft,llt)
```

and

```
subroutine push_forward_2s(sig1,sig2,sig3,sig4,sig5,sig6,
      f11,f21,f31,f12,f22,f32,f13,f23,f33)
```

which perform push-forward operations on the stress tensor for the vectorized and scalar routines, respectively. In the latter subroutine, all arguments are scalars, whereas the corresponding entries in the vectorized routine are vectors of length `nlq`. The `sig1` to `sig6` are components of the stress tensor, and `f11` to `f33` are components of the deformation gradient.

Suppose you invoke the local coordinate system option (IORTHO = 1). In that case, the program transforms the deformation gradient into the local system with the following expression before passing it to the user-defined subroutine:

$$\bar{F}_{ij} = Q^s_{ki} F_{kj} \quad .$$

Here Q^s_{ij} refers to a transformation between the current global and material frames. For IORTHO equal to 1, setting IHYPER equal to –1 results in the following transformation of the deformation gradient:

$$\bar{F}_{ij} = F_{ik} Q^r_{kj} \quad .$$

Q^r_{ij} is the transformation between the reference global and material frames. For this latter option the spatial frame remains the global one, so the user-defined routine must express the stresses in this frame of reference upon exiting the routine. The suitable choice of IHYPER depends on the formulation of the material model.

For shells only, a particular option invoked by setting IHYPER to 3 causes LS-DYNA to compute the deformation gradient from the nodal coordinates in the global coordinate system. With this option, you must calculate the stress in the local system of interest. LS-DYNA passes a transformation matrix between the global and this local system to the user material routines (`qmat`). The columns in this matrix correspond to local basis vectors expressed in global coordinates. Your subroutine must compute the stress in this system. Note that the resulting deformation gradient may not be consistent with the theory for the element for your material because of how LS-DYNA calculates it. To account for thickness changes due to membrane straining, we provide the following subroutines:

```
subroutine usrshl_updatfs(f,t,s,e)
```

and

```
subroutine usrshl_updatfv(f,t,s,e,lft,llt)
```

for the scalar and vector versions. The subroutine recomputes the deformation gradient based on the thickness strain increment, `e`, and the nodal thicknesses, `t`. The history variables array stores the current nodal thicknesses immediately after the deformation gradient. Your subroutine must call this routine with these four values as `t`. This subroutine produces a deformation, `f`, and the new thicknesses, `s`. Your subroutine finds the strain increment, `e`, giving zero thickness stress through this subroutine. After obtaining zero thickness stress, your subroutine needs to store the new thicknesses, `s`, in the history variables array. To achieve this, copy the new thicknesses, `s`, to the location for the nodal thicknesses. We provide sample code in the object library.

Sample user subroutine 45

Here we provide an example of a user-defined material that requires the deformation gradient. Our user-defined subroutine models a Neo-Hookean material. With λ and μ being the Lame parameters in the linearized theory, the following gives the strain energy density for this material:

$$\psi = \frac{1}{2}\lambda(\ln(\det \mathbf{F}))^2 - \mu \ln(\det \mathbf{F}) + \frac{1}{2}\mu\left(\mathrm{tr}\left(\mathbf{F}^T\mathbf{F}\right) - 3\right) \ .$$

Thus, we can express the Cauchy stress as

$$\sigma = \frac{1}{\det \mathbf{F}}\left(\lambda \ln(\det \mathbf{F})\,\mathbf{I} + \mu\left(\mathbf{F}\mathbf{F}^T - \mathbf{I}\right)\right) \ .$$

```
      subroutine umat45 (cm,eps,sig,epsp,hsv,dt1,capa,
     . etype,time,temp,failel,crv,nnpcrv,cma,qmat,elsiz,idele,reject)
c
c************************************************************
c|  Livermore Software Technology Corporation  (LSTC)        |
c|  ------------------------------------------------------  |
c|  Copyright 1987-2008 Livermore Software Tech.  Corp       |
c|  All rights reserved                                      |
c************************************************************
c
c     Neo-Hookean material (sample user subroutine)
c
c     Variables
c
c     cm(1)=first material constant, here young's modulus
c     cm(2)=second material constant, here poisson's ratio
c        .
c        .
c        .
c     cm(n)=nth material constant
c
c     eps(1)=local x  strain increment
c     eps(2)=local y  strain increment
c     eps(3)=local z  strain increment
c     eps(4)=local xy strain increment
c     eps(5)=local yz strain increment
c     eps(6)=local zx strain increment
c
c     sig(1)=local x  stress
c     sig(2)=local y  stress
c     sig(3)=local z  stress
c     sig(4)=local xy stress
c     sig(5)=local yz stress
```

APPENDIX A

```
c     sig(6)=local zx stress
c
c     hsv(1)=1st history variable
c     hsv(2)=2nd history variable
c        .
c        .
c        .
c        .
c     hsv(n)=nth history variable
c
c     dt1=current time step size
c     capa=reduction factor for transverse shear
c     etype:
c       eq."solid" for solid elements
c       eq."sld2d" for shell forms 13, 14, and 15 (2D solids)
c       eq."shl_t" for shell forms 25, 26, and 27 (shells with thickness
c        stretch)
c       eq."shell" for all other shell elements plus thick shell forms 1
c        and 2
c       eq."tshel" for thick shell forms 3 and 5
c       eq."hbeam" for beam element forms 1 and 11
c       eq."tbeam" for beam element form 3 (truss)
c       eq."dbeam" for beam element form 6 (discrete)
c       eq."beam " for all other beam elements
c
c     time=current problem time.
c
c     temp=current temperature
c
c     failel=flag for failure, set to .true.  to fail an integration point,
c            if .true.  on input the integration point has failed earlier
c
c     crv=array representation of curves in keyword deck
c
c     nnpcrv=# of discretization points per crv()
c
c     cma=additional memory for material data defined by LMCA at
c       6th field of 2nd crad of *DATA_USER_DEFINED
c
c     elsiz=characteristic element size
c
c     idele=element id
c
c     reject (implicit only) = set to .true.  if this implicit iterate is
c                              to be rejected for some reason
c
c     All transformations into the element local system are
c     performed prior to entering this subroutine.  Transformations
c     back to the global system are performed after exiting this
c     routine.
c
c     All history variables are initialized to zero in the input
c     phase.   Initialization of history variables to nonzero values
c     may be done during the first call to this subroutine for each
c     element.
c
c     Energy calculations for the dyna3d energy balance are done
c     outside this subroutine.
c
      include 'nlqparm'
      include 'iounits.inc'
      include 'bk06.inc'
      character*5 etype
      dimension cm(*),eps(*),sig(*),hsv(*),crv(lq1,2,*),cma(*)
      logical failel
c
```

```
      if (ncycle.eq.1) then
        call usermsg('mat45')
      endif
c
c     compute lame parameters
c
      xlambda=cm(1)*cm(2)/((1.+cm(2))*(1.-2.*cm(2)))
      xmu=.5*cm(1)/(1.+cm(2))
c
      if (etype.eq.'solid'.or.etype.eq.'shl_t'.or.
     1     etype.eq.'sld2d'.or.etype.eq.'tshel') then
c
c       deformation gradient stored in hsv(1),...,hsv(9)
c
c       compute jacobian
c
        detf=hsv(1)*(hsv(5)*hsv(9)-hsv(6)*hsv(8))
     1      -hsv(2)*(hsv(4)*hsv(9)-hsv(6)*hsv(7))
     2      +hsv(3)*(hsv(4)*hsv(8)-hsv(5)*hsv(7))
c
c       compute left cauchy-green tensor
c
        b1=hsv(1)*hsv(1)+hsv(4)*hsv(4)+hsv(7)*hsv(7)
        b2=hsv(2)*hsv(2)+hsv(5)*hsv(5)+hsv(8)*hsv(8)
        b3=hsv(3)*hsv(3)+hsv(6)*hsv(6)+hsv(9)*hsv(9)
        b4=hsv(1)*hsv(2)+hsv(4)*hsv(5)+hsv(7)*hsv(8)
        b5=hsv(2)*hsv(3)+hsv(5)*hsv(6)+hsv(8)*hsv(9)
        b6=hsv(1)*hsv(3)+hsv(4)*hsv(6)+hsv(7)*hsv(9)
c
c       compute cauchy stress
c
        detfinv=1./detf
        dmu=xmu-xlambda*log(detf)
        sig(1)=detfinv*(xmu*b1-dmu)
        sig(2)=detfinv*(xmu*b2-dmu)
        sig(3)=detfinv*(xmu*b3-dmu)
        sig(4)=detfinv*xmu*b4
        sig(5)=detfinv*xmu*b5
        sig(6)=detfinv*xmu*b6
c
      else if (etype.eq.'shell') then
c
c       deformation gradient stored in hsv(1),...,hsv(9)
c
c       compute part of left cauchy-green tensor
c       independent of thickness strain increment
c
        b1=hsv(1)*hsv(1)+hsv(4)*hsv(4)+hsv(7)*hsv(7)
        b2=hsv(2)*hsv(2)+hsv(5)*hsv(5)+hsv(8)*hsv(8)
        b4=hsv(1)*hsv(2)+hsv(4)*hsv(5)+hsv(7)*hsv(8)
c
c       secant iterations for zero normal stress
c
        do iter=1,5
c
c         first thickness strain increment initial guess
c         assuming Poisson's ratio different from zero
c
          if (iter.eq.1) then
            eps(3)=-xlambda*(eps(1)+eps(2))/(xlambda+2.*xmu)
c
c         second thickness strain increment initial guess
c
          else if (iter.eq.2) then
            sigold=sig(3)
            epsold=eps(3)
```

```
            eps(3)=0.
c
c         secant update of thickness strain increment
c
          else if (abs(sig(3)-sigold).gt.0.0) then
            deps=-(eps(3)-epsold)/(sig(3)-sigold)*sig(3)
            sigold=sig(3)
            epsold=eps(3)
            eps(3)=eps(3)+deps
          endif
c
c         compute last row of deformation gradient
c
          call compute_f3s(hsv(3),hsv(6),hsv(9),eps(3))
c
c         compute jacobian
c
          detf=hsv(1)*(hsv(5)*hsv(9)-hsv(6)*hsv(8))
     1         -hsv(2)*(hsv(4)*hsv(9)-hsv(6)*hsv(7))
     2         +hsv(3)*(hsv(4)*hsv(8)-hsv(5)*hsv(7))
c
c         compute normal component of left cauchy-green tensor
c
          b3=hsv(3)*hsv(3)+hsv(6)*hsv(6)+hsv(9)*hsv(9)
c
c         compute normal stress
c
          detfinv=1./detf
          dmu=xmu-xlambda*log(detf)
          sig(1)=detfinv*(xmu*b1-dmu)
          sig(2)=detfinv*(xmu*b2-dmu)
          sig(3)=detfinv*(xmu*b3-dmu)
          sig(4)=detfinv*xmu*b4
c
c         exit if normal stress is sufficiently small
c
          if (abs(sig(3)).le.1.e-5*
     1     (abs(sig(1))+abs(sig(2))+abs(sig(4)))) goto 10
         enddo
c
c       compute remaining components of left cauchy-green tensor
c
 10     b5=hsv(2)*hsv(3)+hsv(5)*hsv(6)+hsv(8)*hsv(9)
        b6=hsv(1)*hsv(3)+hsv(4)*hsv(6)+hsv(7)*hsv(9)
c
c       compute remaining stress components
c
        sig(5)=detfinv*xmu*b5
        sig(6)=detfinv*xmu*b6
c
c     material model only available for solids and shells
c
      else
        cerdat(1)=etype
        call lsmsg(3,MSG_SOL+1151,ioall,ierdat,rerdat,cerdat,0)
      endif
      return
      end
```

Implicit analysis

We also support user-defined material models with implicit analysis for solid, shell, thick shell, and Hughes-Liu beam elements. When you request implicit analysis in the input

deck, LS-DYNA calls the subroutine `urtanh` for solids (and thick shell types 3, 5, and 7), `urtans` for shells (and thick shell types 1, 2, and 6), and `urtanb` for beams with appropriate input data for calculating the material tangent modulus. This subroutine then calls a user-defined subroutine for calculating the tangent modulus for your material: `utan`*XX* for the scalar implementation or `utan`*XX*`v` for the vectorized implementation. Again, *XX* is the number that matches MT on the material card. `utan`*XX* includes

`es(6,6)` – material tangent modulus

as an argument, while `utan`*XX* has the corresponding vector block

`dsave(nlq,6,6)` – material tangent modulus

as an argument. Your user-defined subroutine builds the tangent modulus matrix needed for assembling the tangent stiffness matrix. This matrix equals the zero matrix when entering the user-defined subroutine. The matrix must be symmetric. If you invoke the local coordinate system option for solids, it should be in this local system. For shell elements, it should be in the co-rotational system defined for the current shell element. All transformations back to the global system are made after exiting the user-defined subroutine.

The user-defined material routine (`umat`*XX* or `umat`*XX*`v`) includes the argument `reject` to improve convergence characteristics. To use this parameter, have your user-defined subroutine set it to `.true.` when a criterion declares the iteration unacceptable. For example, the plastic strain could increase too much in one step. If your material subroutine rejects the iteration, LS-DYNA prints a warning message that says, 'Material model rejected current iterate.' LS-DYNA then retries the step with a smaller time step. If chosen carefully (by way of experimenting), this feature may result in a good trade-off between the number of implicit iterations per step and the step size for overall speed.

If the material is hyperelastic, you can apply push-forward operations on the tangent modulus tensor with

```
subroutine push_forward_4(dsave,
      f11,f21,f31,f12,f22,f32,f13,f23,f33,lft,llt)
```

or

```
subroutine push_forward_4s(es,
      f11,f21,f31,f12,f22,f32,f13,f23,f33)
```

for the vector and scalar implementations, respectively. In the latter subroutine, all arguments are scalars, whereas the corresponding entries in the vectorized routine are vectors of length `nlq`. `f11` to `f33` are components of the deformation gradient.

Sample user-subroutine 42, tangent modulus

The following sample user subroutine illustrates how to implement the tangent stiffness modulus for the Neo-Hookean material covered in the Sample user subroutine 45 subsection of the Deformation gradient section. The material tangent modulus is for this material given by

APPENDIX A

$$\mathbf{C} = \frac{1}{\det \mathbf{F}} \left(\lambda \mathbf{I} \otimes \mathbf{I} + 2(\mu - \lambda \ln(\det \mathbf{F})\, \mathbf{I}) \right) \ .$$

```
      subroutine utan42(cm,eps,sig,epsp,hsv,dt1,capa,
     .     etype,tt,temper,es,crv)
c************************************************************************
c|  livermore software technology corporation  (lstc)                    |
c|  ------------------------------------------------------------------   |
c|  copyright 1987-1999                                                  |
c|  all rights reserved                                                  |
c************************************************************************
c
c     Neo-Hookean material tangent modulus (sample user subroutine)
c
c     Variables
c
c     cm(1)=first material constant, here young's modulus
c     cm(2)=second material constant, here poisson's ratio
c        .
c        .
c        .
c      cm(n)=nth material constant
c
c     eps(1)=local x  strain increment
c     eps(2)=local y  strain increment
c     eps(3)=local z  strain increment
c     eps(4)=local xy strain increment
c     eps(5)=local yz strain increment
c     eps(6)=local zx strain increment
c
c     sig(1)=local x  stress
c     sig(2)=local y  stress
c     sig(3)=local z  stress
c     sig(4)=local xy stress
c     sig(5)=local yz stress
c     sig(6)=local zx stress
c
c      epsp=effective plastic strain
c
c     hsv(1)=1st history variable
c     hsv(2)=2nd history variable
c        .
c        .
c        .
c        .
c     hsv(n)=nth history variable
c
c     dt1=current time step size
c     capa=reduction factor for transverse shear
c     etype:
c        eq."brick" for solid elements
c        eq."shell" for all shell elements
c        eq."beam"  for all beam elements
c        eq."dbeam"  for all discrete beam elements
c
c     tt=current problem time.
c
c     temper=current temperature
c
c     es=material tangent modulus
c
c      crv=array representation of curves in keyword deck
c
```

```
c      The material tangent modulus is set to 0 before entering
c      this routine.  It should be expressed in the local system
c      upon exiting this routine.  All transformations back to the
c      global system are made outside this routine.
       include 'nlqparm'
       character*(*) etype
       dimension cm(*),eps(*),sig(*),hsv(*),crv(lq1,2,*)
       dimension es(6,*)
c
c      no history variables, NHV=0
c      deformation gradient stored in hsv(1),...,hsv(9)
c
c      compute jacobian
c
       detf=hsv(1)*(hsv(5)*hsv(9)-hsv(6)*hsv(8))
     1      -hsv(2)*(hsv(4)*hsv(9)-hsv(6)*hsv(7))
     2      +hsv(3)*(hsv(4)*hsv(8)-hsv(5)*hsv(7))
c
c      compute lame parameters
c
       xlambda=cm(1)*cm(2)/((1.+cm(2))*(1.-2.*cm(2)))
       xmu=.5*cm(1)/(1.+cm(2))
c
c      compute tangent stiffness
c      same for both shells and bricks
c
       detfinv=1./detf
       dmu=xmu-xlambda*log(detf)
       es(1,1)=detfinv*(xlambda+2.*dmu)
       es(2,2)=detfinv*(xlambda+2.*dmu)
       es(3,3)=detfinv*(xlambda+2.*dmu)
       es(4,4)=detfinv*dmu
       es(5,5)=detfinv*dmu
       es(6,6)=detfinv*dmu
       es(2,1)=detfinv*xlambda
       es(3,2)=detfinv*xlambda
       es(3,1)=detfinv*xlambda
       es(1,2)=es(2,1)
       es(2,3)=es(3,2)
       es(1,3)=es(3,1)
c
       return
       end
```

User-defined materials with equations-of-state

The following example user-defined subroutine uses an equation-of-state (EOS). Unlike standard models, it updates only the deviatoric stress and assigns a value to PC, the pressure cut-off. The pressure cut-off limits the amount of hydrostatic pressure that can be carried in tension (that is, when the pressure is negative). The default value is zero. A large negative number will allow the material to carry an unlimited pressure load in tension. Typically, the pressure cut-off is a function of the current state of the material and varies with time. Thus, the material model subroutine needs to calculate it. We made the pressure cut-off a constant value for simplicity in this example. The named common block `eosdloc` stores the pressure cut-off array. Depending on the computing environment, compiler directives may be required (such as the task common directive in the example) for correct SMP execution.

In addition, the number of history variables, NHV, must be increased by 4 in the input file to allocate the extra storage required for the EOS. The first 4 variables in `hsvs` contain the storage. The user-defined material model must not alter it.

```
      subroutine umat44v(cm,d1,d2,d3,d4,d5,d6,sig1,sig2,
     . sig3,sig4,sig5,sig6,eps,hsvs,lft,llt,dt1siz,capa,
     . etype,tt,temps,failels,nlqa,crv)
      parameter (third=1.0/3.0)
      include 'nlqparm'
c
c***  isotropic plasticity with linear hardening
c
c***  updates only the deviatoric stress so that it can be used with
c     an equation of state
c
      character*5 etype
      logical failels
c
C_TASKCOMMON (eosdloc)
      common/eosdloc/pc(nlq)
c
      dimension cm(*),d1(*),d2(*),d3(*),d4(*),d5(*),d6(*),
     & sig1(*),sig2(*),sig3(*),sig4(*),sig5(*),sig6(*),
     & eps(*),hsvs(nlqa,*),dt1siz(*),temps(*),crv(lq1,2,*),
     & failels(*)
c
c***  shear modulus, initial yield stress, hardening, and pressure cut-off
      g   =cm(1)
      sy0 =cm(2)
      h   =cm(3)
      pcut=cm(4)
c
      ofac=1.0/(3.0*g+h)
      twog=2.0*g
c
      do i=lft,llt
c
c***    trial elastic deviatoric stress
        davg=third*(d1(i)+d2(i)+d3(i))
        savg=third*(sig1(i)+sig2(i)+sig3(i))
        sig1(i)=sig1(i)-savg+twog*(d1(i)-davg)
        sig2(i)=sig2(i)-savg+twog*(d2(i)-davg)
        sig3(i)=sig3(i)-savg+twog*(d3(i)-davg)
        sig4(i)=sig4(i)+g*d4(i)
        sig5(i)=sig5(i)+g*d5(i)
        sig6(i)=sig6(i)+g*d6(i)
c
c***    radial return
        aj2=sqrt(1.5*(sig1(i)**2+sig2(i)**2+sig3(i)**2)+
     &           3.0*(sig4(i)**2+sig5(i)**2+sig6(i)**2))
        sy=sy0+h*eps(i)
        eps(i)=eps(i)+ofac*max(0.0,aj2-sy)
        synew=sy0+h*eps(i)
        scale=synew/max(synew,aj2)
c
c***    scaling for radial return.  note that the stress is now deviatoric.
        sig1(i)=scale*sig1(i)
        sig2(i)=scale*sig2(i)
        sig3(i)=scale*sig3(i)
        sig4(i)=scale*sig4(i)
        sig5(i)=scale*sig5(i)
        sig6(i)=scale*sig6(i)
c
c***    set pressure cut-off
        pc(i)=pcut
```

```
c
      enddo
c
      return
      end
```

Post-processing a user-defined material

Post-processing a user-defined material is very similar to post-processing a regular LS-DYNA material. However, we would like to stress some aspects of post-processing, all dealing with how to post-process history variables.

First, LS-DYNA always writes the effective plastic strain to the d3plot database and thus need not be requested by you. LS-PREPOST treats it the same as for the result from any other LS-DYNA material.

NEIPH and NEIPS on *DATABASE_EXTENT_BINARY specify the number of additional history variables written to the d3plot database for solids and shells, respectively. For instance, if NEIPH (NEIPS) equals 2, LS-DYNA writes the first two history variables in the history variables array as history var#1 and history var#2 in the d3plot database. By setting NEIPH (NEIPS) equal to NHV, LS-DYNA outputs all history variables to the d3plot database. Furthermore, if the material model uses the deformation gradient (IHYPER = 1), you must request an additional nine variables to make it available for post-processing, meaning set NEIPH (NEIPS) equal to NHV+9. The deformation gradient becomes available in the d3plot database as history variables NHV+1 to NHV+9. Note that LS-DYNA outputs the deformation gradient in the co-rotational system for shells. If using the local coordinate system option (IORTHO = 1), LS-DYNA expresses the deformation gradient in this local system. Set NEIPH (NEIPS) equal to NHV+9+9(=NHV+18) to output the deformation gradient in the global system for bricks and the co-rotational system for shells. LS-DYNA outputs these as history variables NHV+10 to NHV+18.

APPENDIX B: User-Defined Equation-of-State

You can supply your own subroutines to define equation-of-state (EOS) models in LS-DYNA. To invoke a user-defined EOS, you must

1. Write a user EOS subroutine called by the LS-DYNA user-defined EOS interface.
2. Create a custom executable that includes the EOS subroutine.
3. Invoke that subroutine by defining a part in the keyword input deck that uses *EOS_USER_DEFINED with the appropriate input parameters.

We provide subroutine `ueoslib` and sample subroutines `ueos21s` and `ueos21v` in Fortran source files. This text serves as an introductory guide to implementing such a model. Note that the names of variables and subroutines below may differ from the actual ones depending on the platform and the current version of LS-DYNA.

General overview

When you define *EOS_USER_DEFINED for a part, LS-DYNA calls the subroutine `ueoslib` with the appropriate input data for the EOS update. LS-DYNA calls this subroutine twice for each integration point in each element. The first call requires the EOS to calculate the bulk modulus, and the second call updates the pressure and internal energy. You may modify this routine if necessary. This routine initializes the following data structures used by a specific *scalar* material subroutine:

`iflag` – mode flag
- EQ.-1: for initializing EOS constants
- EQ.0: for calculating the bulk modulus
- EQ.1: for the pressure and energy update

`cb` – bulk modulus
`pnew` – the new pressure
`rho0` – reference density
`hist` – array of user-defined history variables, NHV in length
`specen` – internal energy per unit reference volume
`df` – volume ratio, V/V_0
`v0` – the initial volume
`dvol` – volume increment
`pc` – pressure cut-off

If you activate the *vectorization* flag (IVECT = 1) on the EOS card, this routine generally stores these variables in vector blocks of length `nlq`, with vector indices ranging from

lft to llt, which allows for more efficient execution of the EOS routine. The data structures mentioned above for the vectorized case are:

cb(nlq)	–	bulk modulus
pnew(nlq)	–	the new pressure
hist(nlq,*)	–	array of user-defined history variables with NHV columns
specen(nlq)	–	internal energy per unit reference volume
df(nlq)	–	volume ratio, V/V0
v0(nlq)	–	the initial volume
dvol(nlq)	–	volume increment
pc(nlq)	–	pressure cut-off

The value of nlq is set as a parameter in the include file nlqparm, included at the top of the subroutine, and varies between machines and operating systems. Each entry in a vector block is associated with an element in the finite element mesh for a fixed integration point. The number of entries in the history variables array (indicated by * above) matches the number of history variables requested on the material card (NHV). All history variables are initially zero and are initialized within the user-define EOS subroutine on the first time step, when the logical variable first, passed through the argument list, is .TRUE. Furthermore, all user-defined EOS models require a bulk modulus, cb, for transmitting boundaries, contact interfaces, rigid body constraints, and time step calculations. In addition to the variables mentioned above, the following data can be supplied to the user-defined material routines, regardless of whether vectorization is used or not:

eosp(*)	–	array of material constants from the input file
tt	–	current time
crv(lq1,2,*)	–	array representation of curves defined in the keyword deck.

A user-defined EOS subroutine, ueos*XX*s in the scalar case or ueos*XX*v in the vector case, will be called for parts that point to *EOS_USER_DEFINED in the input deck. The letters *XX* represent a number between 21 and 30 that matches the input variable EOST in the *EOS_USER_DEFINED keyword. During the initialization phase, LS-DYNA calls the user-define EOS subroutine with iflag set to -1 to permit the initialization of constants in the user-defined EOS subroutine. Although fewer than 48 constants may be read into the array eosp during the input, you may use all 48 within the EOS subroutines. The user-defined subroutine should calculate the bulk modulus when iflag = 0 and update the pressure, internal energy, and history variables when iflag = 1.

See Appendix A for a discussion about using curves (*DEFINE_CURVE).

Example and Discussion

Here we provide a sample scalar user-defined subroutine for a Gruneisen EOS. Its vector counterpart immediately follows it. We close this section with a discussion about implementation and common pitfalls.

Scalar subroutine

```
      subroutine ueos21s(iflag,cb,pnew,hist,rho0,eosp,specen,
     &                   df,dvol,v0,pc,dt,tt,crv,nnpcrv,first)
      include 'nlqparm'
c
c***  example scalar user implementation of the Gruneisen EOS
c
c***  variables
c          iflag ----- =0 calculate bulk modulus
c                      =1 update pressure and energy
c          cb -------- bulk modulus
c          pnew ------ new pressure
c          hist ------ history variables
c          rho0 ------ reference density
c          eosp ------ EOS constants
c          specen ---- energy/reference volume
c          df -------- volume ratio, v/v0 = rho0/rho
c          dvol ------ change in volume over a time step
c          v0 -------- reference volume
c          pc -------- pressure cut-off
c          dt -------- time step size
c          tt -------- current time
c          crv ------- curve array
c          nnpcrv ---- number of points in each curve
c          first ----- logical .true.  for tt,crv,first time step
c                      (for initialization of the history variables)
c
      logical first
c
      dimension hist(*),eosp(*),crv(lq1,2,*)
      integer nnpcrv(*)
c
      c  =eosp(1)
      s1 =eosp(2)
      s2 =eosp(3)
      s3 =eosp(4)
      g0 =eosp(5)
      sa =eosp(6)
      s11=s1-1.
      s22=2.*s2
      s33=3.*s3
      s32=2.*s3
      sad2=.5*sa
      g0d2=1.-.5*g0
      roc2=rho0*c**2
c
c***  calculate the bulk modulus for the EOS contribution to the sound speed
      if (iflag.eq.0) then
        xmu=1.0/df-1.
        dfmu=df*xmu
        facp=.5*(1.+sign(1.,xmu))
        facn=1.-facp
        xnum=1.+xmu*(+g0d2-sad2*xmu)
        xdem=1.-xmu*(s11+dfmu*(s2+s3*dfmu))
        tmp=facp/(xdem*xdem)
        a=roc2*xmu*(facn+tmp*xnum)
        b=g0+sa*xmu
        pnum=roc2*(facn+facp*(xnum+xmu*(g0d2-sa*xmu)))
        pden=2.*xdem*(-s11 +dfmu*(-s22+dfmu*(s2-s33+s32*dfmu)))
        cb=pnum*(facn+tmp)-tmp*a*pden+sa*specen+
     &        b*df**2*max(pc,(a+b*specen))
c
c***  update the pressure and internal energy
      else
```

```
        xmu=1.0/df-1.
        dfmu=df*xmu
        facp=.5*(1.+sign(1.,xmu))
        facn=1.-facp
        xnum=1.+xmu*(+g0d2-sad2*xmu)
        xdem=1.-xmu*(s11+dfmu*(s2+s3*dfmu))
        tmp=facp/(xdem*xdem)
        a=roc2*xmu*(facn+tmp*xnum)
        b=g0+sa*xmu
        dvov0=0.5*dvol/v0
        denom=1.+ b*dvov0
        pnew=(a+specen*b)/max(1.e-6,denom)
        pnew=max(pnew,pc)
        specen=specen-pnew*dvov0
      endif
c
      return
      end
```

Vectorized routine

```
      subroutine ueos21v(lft,llt,iflag,cb,pnew,hist,rho0,eosp,specen,
     &                   df,dvol,v0,pc,dt,tt,crv,nnpcrv,first)
      include 'nlqparm'
c
c***  example vectorized user implementation of the Gruneisen EOS
c
c***  variables
c          lft,llt --- tt,crv,first and last indices into arrays
c          iflag ----- =0 calculate bulk modulus
c                      =1 update pressure and energy
c          cb -------- bulk modulus
c          pnew ------ new pressure
c          hist ------ history variables
c          rho0 ------ reference density
c          eosp ------ EOS constants
c          specen ---- energy/reference volume
c          df -------- volume ratio, v/v0 = rho0/rho
c          dvol ------ change in volume over a time step
c          v0 -------- reference volume
c          pc -------- pressure cut-off
c          dt -------- time step size
c          tt -------- current time
c          crv ------- curve array
c          nnpcrv ---- number of points in each curve
c          first ----- logical .true.  for tt,crv,first time step
c                      (for initialization of the history variables)
c
      logical first
c
      dimension cb(*),pnew(*),hist(nlq,*),eosp(*),
     &          specen(*),df(*),dvol(*),pc(*),v0(*)
      dimension crv(lq1,2,*)
      integer nnpcrv(*)
c
      c  =eosp(1)
      s1 =eosp(2)
      s2 =eosp(3)
      s3 =eosp(4)
      g0 =eosp(5)
      sa =eosp(6)
      s11=s1-1.
      s22=2.*s2
      s33=3.*s3
      s32=2.*s3
```

```
      sad2=.5*sa
      g0d2=1.-.5*g0
      roc2=rho0*c**2
c
c***  calculate the bulk modulus for the EOS contribution to the sound speed
      if (iflag.eq.0) then
        do i=lft,llt
          xmu=1.0/df(i)-1.
          dfmu=df(i)*xmu
          facp=.5*(1.+sign(1.,xmu))
          facn=1.-facp
          xnum=1.+xmu*(+g0d2-sad2*xmu)
          xdem=1.-xmu*(s11+dfmu*(s2+s3*dfmu))
          tmp=facp/(xdem*xdem)
          a=roc2*xmu*(facn+tmp*xnum)
          b=g0+sa*xmu
          pnum=roc2*(facn+facp*(xnum+xmu*(g0d2-sa*xmu)))
          pden=2.*xdem*(-s11 +dfmu*(-s22+dfmu*(s2-s33+s32*dfmu)))
          cb(i)=pnum*(facn+tmp)-tmp*a*pden+sa*specen(i)+
     &          b*df(i)**2*max(pc(i),(a+b*specen(i)))
        enddo
c
c***  update the pressure and internal energy
      else
        do i=lft,llt
          xmu=1.0/df(i)-1.
          dfmu=df(i)*xmu
          facp=.5*(1.+sign(1.,xmu))
          facn=1.-facp
          xnum=1.+xmu*(+g0d2-sad2*xmu)
          xdem=1.-xmu*(s11+dfmu*(s2+s3*dfmu))
          tmp=facp/(xdem*xdem)
          a=roc2*xmu*(facn+tmp*xnum)
          b=g0+sa*xmu
          dvov0=0.5*dvol(i)/v0(i)
          denom=1.+b*dvov0
          pnew(i)=(a+specen(i)*b)/max(1.e-6,denom)
          pnew(i)=max(pnew(i),pc(i))
          specen(i)=specen(i)-pnew(i)*dvov0
        enddo
      endif
c
      return
      end
```

Implementation discussion

The Gruneisen EOS implemented in the example subroutines has the same form as *EOS_GRUNEISEN, EOS form 4. Its update of the pressure and the internal energy is typical for an EOS that is linear in the internal energy,

$$P = A(\rho) + B(\rho)E\ ,$$

where A and B correspond to the variables `a` and `b` in the example subroutines, and E is `specen`. Integrating the energy equation with the trapezoidal rule gives

$$E^{n+1} = E^n + \frac{1}{2}(\sigma'^n + \sigma'^{n+1})\Delta\varepsilon - \frac{1}{2}(P^n + q^n + P^{n+1} + q^{n+1})\frac{\Delta V}{V_0}$$

where the superscripts refer to the time step, ΔV is the change in the volume associated with the Gauss point and V_0 is the reference volume. Collecting all the energy

contributions on the right-hand side except for the contribution from the new pressure gives a simple linear relationship between the new internal energy and pressure,

$$E^{n+1} = \tilde{E} - \frac{P^{n+1}\Delta V}{2V_0}.$$

The value of `specen` passed to `ueosXX` for the pressure and energy update corresponds to $\tilde{E}$. Substituting this relation into the EOS and solving for the new pressure gives:

$$P^{n+1} = \frac{A\rho^{n+1} + B\rho^{n+1}\tilde{E}}{1 + \frac{B\Delta V}{2V_0}} .$$

The final update of the new energy is calculated using the updated pressure. For a more general EOS, the nonlinear equation for the new pressure,

$$P^{n+1} = P\left(\rho^{n+1}, \tilde{E} - \frac{P^{n+1}\Delta V}{2V_0}\right)$$

is solved iteratively using Newton iteration or successive substitution.

The pressure cut-off, `pc`, limits the amount of pressure that can be generated by tensile loading, `pnew=max(pnew,pc)`. Its value is usually specified in the *MAT input, such as *MAT_JOHNSON_COOK. It is not enforced outside the EOS subroutines. Thus, you determine whether to enforce the pressure cut-off in `ueosXX`. To impose the pressure cut-off, apply it before the final update to the internal energy; otherwise, the energy will be incorrect.

Calculating the bulk modulus is similar to updating the pressure and energy. Since the bulk modulus calculation always precedes the pressure update, the values may be saved in a common block during the bulk modulus calculation to reduce the cost of the pressure update. The arrays that store the values in the vectorized subroutines should be dimensioned by `nlq`.

A common pitfall is using the wrong internal energy when implementing an EOS from a paper or book. Three internal energies are commonly used: the energy per unit mass, e_M, the energy per unit current volume, e_V, and the energy per unit reference volume, E. LS-DYNA always uses the energy per unit reference volume. Some helpful relations for converting between the EOS in the literature and the variables in LS-DYNA are

$$e_V = E\frac{V_0}{V} = \frac{\text{specen}}{\text{df}}$$
$$e_M = E\frac{V_0}{M} = \frac{\text{specen}}{\text{rho0}}$$
$$\rho = \rho_0\frac{V_0}{V} = \frac{\text{rho0}}{\text{df}}$$

APPENDIX C: User-Defined Element Interface for Solids and Shells

This appendix describes the user-defined element interface for solids and shells. The interface can accommodate either an integrated or a resultant element. For the integrated element, you supply two matrices defining the kinematical properties of the element. You also determine whether to use standard LS-DYNA hourglass stabilization, a user-defined stabilization, or no stabilization in the absence of zero-energy modes (see IHGF on *SECTION_SHELL/SOLID). The number and locations of the integration points are arbitrary, meaning you define them. For the resultant/discrete element formulations, you implement the force and stiffness assembly. For both element formulations, history variables can be associated with the elements. In both cases, if desired, the element may have more than the conventional 3 (for bricks) and 6 (for shells) degrees-of-freedom per node (see Extra Degrees of Freedom).

Algorithm Outline

We implemented user-defined, integrated elements in the same way as standard LS-DYNA elements aside from additionally calling user routines for setting up the matrices of interest. In the end, LS-DYNA constructs the gradient-displacement matrix, B_{ijkK}, with the property that

$$B_{ijkK}u_{kK} = \frac{\partial v_i}{\partial x_j} \ ,$$

where

$u_{kK} =$ Vector of velocity nodal degrees-of-freedom

$\frac{\partial v_i}{\partial x_i} =$ Velocity gradient

Moreover, the numerical integration requires the input of J, the determinant of the Jacobian matrix that determines the mapping from the isoparametric to the physical domain. Your subroutine determines the gradient-displacement matrix and either the Jacobian matrix or the determinant of the Jacobian matrix, depending on the definition of ITAJ. From these expressions, the symmetric part of the velocity gradient determines the strain, and the corresponding antisymmetric part determines the spin. LS-DYNA, then, evaluates the stresses with a specified constitutive model. From the stresses, it obtains the internal forces with

$$f_{kK} = \int \sigma_{ij} B_{ijkK} dV \ ,$$

where σ_{ij} are the stresses. LS-DYNA calculates the geometric and material tangent stiffnesses through

APPENDIX C

$$K_{iIjJ}^{\text{mat}} = \int C_{klmn} B_{kliI} B_{mnjJ} dV$$

and

$$K_{iIjJ}^{\text{geo}} = \int \sigma_{mn} B_{kmiI} B_{knjJ} dV$$

where C_{klmn} is the tangent modulus for the material. LS-DYNA evaluates the integrals using user-defined quadrature and J.

For user-defined hourglass control, you must provide the corresponding internal force and stiffness contribution in a separate user routine.

With resultant elements, you provide the force and stiffness matrix directly for the entire element.

Invoking User-Defined Elements

To invoke a user-defined element, you must do the following:

1. Write a user element subroutine that defines the kinematics or kinetics of the element.

2. Create a custom executable that includes these subroutines.

3. Invoke the element by specifying this subroutine on the corresponding *SECTION card with ELFORM (101 through 105).

We provide you with the dummy subroutines for the user-defined elements in a FORTRAN source file for you to modify, along with the necessary object files to compile a new executable. Contact Ansys or your local distributor for information about obtaining these files and what compiler/version to use for your specific platform. Up to five user elements can simultaneously be used for bricks and shells (i.e., ten). This text is an introductory guide to implementing such an element.

Integrated Elements
To activate a user-defined element, set ELFORM to a number between 101 and 105 and NIPP/NIP to a value greater than 0 on Card 5/3 of the *SECTION_SHELL/SOLID definition. By doing so, LS-DYNA calls the following user subroutine to determine the kinematics of the elements:

```
subroutine uXXX_bYYY(bmtrx,gmtrx,gjac,...
⋮
dimension bmtrx(nlq,3,3,*),gmtrx(nlq,3,3),gjac(*)
```

where `XXX` is substituted for `shl` for a shell section and `sld` for a solid section, and `YYY` is the value of ELFORM in *SECTION_SHELL/SOLID.

Your choice of ITAJ on *SECTION_SHELL/SOLID determines how to set up these matrices in your subroutine. If ITAJ = 0, set the isoparametric gradient-displacement matrix and the Jacobian matrix. The user subroutine represents these matrices with the arrays `bmtrx` and `gmtrx`, respectively. Here, the first index corresponds to the LS-DYNA block loop index, where `nlq` is the block size. For a more convenient notation in this discussion, we assign a correspondence between the arrays `gmtrx` and `bmtrx` in the subroutines to matrices/tensors as:

`gmtrx(*,i,j)` - g_{ij}
`bmtrx(*,i,j,k)` - b_{ijk}

Your subroutine should determine these matrices should so that at the current integration point:

$$g_{ij} = \frac{\partial x_i}{\partial \xi_j}$$

$$b_{ijk} u_k = \frac{\partial v_i}{\partial \xi_j} \Delta t$$

In the above, we assume summation over repeated indices. We use the following notation:

$x_i(\xi_1, \xi_2, \xi_3, t)$ = i^{th} component of the current position vector at the isoparametric coordinate (ξ_1, ξ_2, ξ_3) and time t.

$v_i(\xi_1, \xi_2, \xi_3, t)$ = i^{th} component of the velocity vector at the isoparametric coordinate (ξ_1, ξ_2, ξ_3) and time t.

Δt = current time step

u_k = k^{th} component of generalized local displacements

ξ_i = i^{th} component of the isoparametric coordinate ranging from -1.0 to 1.0

For shells, ILOC on *SECTION_SHELL specifies the coordinate system for the variables. ILOC = 0 sets the coordinate system to the LS-DYNA local coordinate system (`ILOC=0`), while ILOC = 1 sets it to the global coordinate system. LS-DYNA passes the coordinate system transformation matrix to the user routines, where the columns represent the local unit base vectors. The resulting strains must always be in the local coordinate system for the constitutive evaluations.

For no extra degrees-of-freedom (see Extra Degrees of Freedom), the following formula gives the index k in the displacement expression:

$$k = n(m-1) + d \ ,$$

where $n = 3$ if only translational degrees-of-freedom are present (typical for solids) and $n = 6$ if rotational degrees-of-freedom are present (typical for shells), m is the local node

number ($m = 1,2,\ldots$), and d is the degree of freedom. The translational degrees-of-freedom correspond to $d \leq 3$ and the rotational degrees-of-freedom to $4 \leq d \leq 6$.

If ITAJ = 1, your subroutine sets up the physical gradient-displacement matrix and the Jacobian determinant. The user subroutines represent these by the arrays `bmtrx` and `gjac`, respectively. Again, we assign a correspondence between the arrays `gjac` and `bmtrx` in the subroutines to matrices/tensors as follows

`gjac(*)`	-	J
`bmtrx(*,i,j,k)`	-	b_{ijk}

Your subroutine should determine these matrices should so that at the current integration point:

$$J = \det \frac{\partial x_i}{\partial \xi_j}$$

$$b_{ijk} u_k = \frac{\partial v_i}{\partial x_j} \Delta t$$

To set up these matrices, LS-DYNA passes a set of additional auxiliary variables to the user-defined element subroutines. These variables include the isoparametric coordinates, the element thickness, the shape function values, and the derivatives of the shape functions. Again, for shells, LS-DYNA expresses these variables in either the local or global coordinate system depending on your choice of ILOC. For more information on these variables, see the comments in the subroutines.

The integrated elements can use up to 100 integration points (in the plane for shells) at arbitrary locations. You must specify these integration points in terms of isoparametric coordinates and weights (see Card 5.1 for *SECTION_SHELL and Card 4 for *SECTION_SHELL). The isoparametric coordinates should range from –1 to 1, and the weights should sum up to 4 for shells and 8 for solids.

You may need to incorporate hourglass stabilization to suppress zero energy modes. To do this, set IHGF > 0 in *SECTION_SHELL/SOLID. IHGF = 1 sets automatically using the LS-DYNA hourglass routines. For IHGF = 2 or 3, you must provide hourglass force and stiffness in a specific user-defined routine. If IHGF = 3, physical stabilization becomes available since LS-DYNA passes the resultant material tangent moduli to the hourglass routine to provide the material's current membrane, bending, and coupled membrane-bending stiffness. With C_{ij} denoting the material tangent modulus in matrix form, we express the resultant tangent moduli as

$$\bar{C}^0_{ij} = \int C_{ij}\, dV \qquad \text{(membrane)}$$

$$\bar{C}^1_{ij} = \int z^1 C_{ij}\, dV \qquad \text{(membrane} - \text{bending)}$$

$$\bar{C}^2_{ij} = \int z^2 C_{ij}\, dV \qquad \text{(bending)}$$

where z is the thickness coordinate for shells. For solids, LS-DYNA only passes the first resultant modulus. In this case, the array has 21 entries corresponding to the subdiagonal terms of the 6 by 6 resultant matrix. For the matrix, index (i,j) in the material tangent modulus matrix, where $i \geq j$, the following gives the index I of the array passed to the routine:

$$I = i(i-1)/2 + j$$

In other words, the subdiagonal terms are stored row-wise in the array. For shells, LS-DYNA passes all three moduli in the local coordinate system, where each array has 15 entries corresponding to the subdiagonal terms of the 5 by 5 resultant matrices. We eliminated the through-thickness direction by assuming plane stress. The formula for the array indices transformation above holds.

The subroutine for user-defined hourglass control is called

```
subroutine uXXX_eYYY(force,stiff,ndtot,...
⋮
dimension force(nlq,*),stiff(nlq,ndtot,*)
```

where again `XXX` is substituted for `shl` for a shell section and `sld` for a soli section, and `YYY` is the value of ELFORM in *SECTION_SHELL/SOLID. The variables in the subroutine correspond to the force and stiffness as

`force(*,i)` - f_i

`stiff(*,i,j)` - K_{ij}

The indices correspond to node and degree of freedom numbers in the same way as for the displacements. For shells, ILOC on *SECTION_SHELL specifies the coordinate system for the force and stiffness. The variable `ndtot` gives the total number of degrees-of-freedom for the element ($4 \times (6 + n_{\text{xdof}})$ for shells and $8 \times (3 + n_{\text{xdof}})$ for solids). This subroutine has the property parameters and history variables associated with the element as input. You specify the property parameters on *SECTION_SHELL/SOLID. Each user-defined element can have no more than 40 property parameters and 100 history variables. You must update the history variables in this routine.

Resultant/Discrete Elements

By setting NIPP/NIP equal to 0 in *SECTION_SHELL/SOLID, LS-DYNA assumes a resultant/discrete element formulation. For this option, you provide force and stiffness in the same user-defined routine as for the user-defined hourglass control, that is, subroutine `uXXX_eYYY` where `XXX` is substituted for `shl` for a shell-section and `sld` for a solid-section, and `YYY` is the value of ELFORM in *SECTION_SHELL/SOLID. Thus, IHGF > 0 on *SECTION_SHELL/SOLID (hourglass stabilization) is incompatible with this option. Because you provide the force and stiffness, LS-DYNA does not call a material subroutine to update stresses and history variables. Instead, this user-defined element subroutine updates the stresses and history variables.

Nevertheless, you need to define *MAT_ELASTIC as the material for the corresponding part with suitable values of Young's modulus and Poisson's ratio. LS-DYNA calculates

the time step and determines the contact stiffness with these material properties. Again, LS-DYNA passes the property parameters and history variables to the routine. It also passes the thickness of the elements for shells. For the shell thickness update option (ISTUPD > 0 on *CONTROL_SHELL), you need to update the thicknesses in this routine.

The stiffness matrix assembled in the element routines can be input as nonsymmetric if and only if LCPACK = 3 on *CONTROL_IMPLICIT_SOLVER, that is if LS-DYNA uses the nonsymmetric solver to update the Newton iterates.

Additional Features of User-Defined Elements

Here we will provide a short description of the additional features associated with user-defined elements.

Nodal Fiber Vectors
If a user-defined shell element formulation requires the nodal fiber vectors, set IUNF to 1 on the *SECTION_SHELL card. With this option, the element routines process the nodal fiber vectors, which can be used as input to the user-defined routines for determining the b_{ijk}, g_{ij}/J, f_i and K_{ij} tensors/matrices. If IUNF ≠ 1, LS-DYNA assumes that the fiber direction is always normal to the plane of the shell. LS-DYNA expresses these nodal fiber vectors in either the local or global system, depending on your choice of ILOC on *SECTION_SHELL. See comments in the user subroutines for more information.

Extra Degrees of Freedom
Exotic element formulations may require extra degrees of freedom per node besides the translational (and rotational) degrees of freedom. You can specify up to 15 extra degrees-of-freedom per node for user-defined elements. Note that if your element requires more than three extra degrees-of-freedom per node, you must increase NXDOFUE in the file nhisparm.inc to the corresponding value. This parameter is, by default, set to 3. We chose this value to limit the default sizes of common blocks and heaps. Note that NXDOFUE does not set the number of extra degrees-of-freedom per node for your element. It just sets the limit for allocation. To specify the number, you set NXDOF on *SECTION_SHELL/SOLID. We recommend setting NXDOFUE equal to NXDOF if possible.

We implemented the extra degrees of freedom per node so that LS-DYNA creates *a scalar node for every three extra degrees of freedom* associated with an actual node. Scalar nodes are dummy nodes linked to a node on an element. In other words, LS-DYNA automatically creates ceil(NXDOF/3) scalar nodes. With *NODE_SCALAR_VALUE and *ELEMENT_SOLID/SHELL_DOF, you can associate some of the extra degrees-of-freedom with a scalar node ID of your choice. This association is optional, as LS-DYNA will create the scalar nodes regardless of using these keywords. It helps with setting boundary conditions and can be used to specify initial conditions, which will be discussed in more depth below. With *NODE_SCALAR_VALUE, you select the node ID associated with up to three degrees-of-freedom and then associate this scalar node with the real node using *ELE-

MENT_SOLID/SHELL_DOF. However, using this method, only one related is associated with each real node. Thus, if you have more than three degrees of freedom, you cannot associate a scalar node for all the degrees of freedom.

We have implemented two methods for setting initial conditions. With the first method, you set the initial conditions in the first step of the user subroutine, while with the second method, you use *NODE_SCALAR_VALUE and *ELEMENT_SOLID/SHELL_DOF. We generally recommend the first method. The second method, however, is helpful if you want a node number associated with the extra degrees of freedom and want to apply boundary conditions to a degree of freedom. With the second method, you can set the initial values of up to three degrees-of-freedom per scalar node. Note that this method can only initialize up to three extra degrees-of-freedom per node because you can only associate one scalar node to each actual node. As an example:

```
*NODE_SCALAR_VALUE
$     NID               V1              V2              V3      NDF
       11              1.0                                       1
       12              1.0                                       1
       13              1.0                                       1
       14              1.0                                       1
*ELEMENT_SHELL_DOF
$     EID      PID       N1      N2      N3      N4
        1        1        1       2       3       4
$                      NS1     NS2     NS3     NS4
                        11      12      13      14
```

defines an element with one additional degree of freedom per node. The initial value of the corresponding degree of freedom is 1.0, and it is unconstrained. Recall that NXDOF on *SECTION_SHELL specifies the actual number of extra degrees-of-freedom. Thus, if you set NXDOF to 4 and use *NODE_SCALAR_VALUE as in the previous example, you would still need to initialize three more degrees-of-freedom in the user subroutine.

The array `xdof` contains the current values of these extra variables. LS-DYNA passes `xdof` to the user routines for setting up the correct kinematical properties. See comments in the routines for more information. The formula for the displacement index changes to

$$k = (n + n_{\text{xdof}})(m - 1) + d$$

where n_{xdof} is the number of extra degrees-of-freedom, n is the number of normal degrees-of-freedom, and m is an index for the node. The extra degrees-of-freedom for each node corresponds to $n + 1 \leq d \leq n + n_{\text{xdof}}$.

For dynamic simulations, the mass corresponding to these extra degrees-of-freedom may be defined by either:

- *ELEMENT_INERTIA or *ELEMENT_MASS, or
- the subroutine `uXXX_mYYY` for `XXX` equal to `sld` or `shl` and `YYY` between 101 and 105.

For solid elements with the subroutine method for specifying mass, XNOD on *SECTION_SOLID indicates the interpretation of the input mass. By default (XNOD = 0), the

mass is interpreted as per extra degree of freedom. By setting XNOD to 1, the mass is interpreted as nodal, which means you can specify the mass for the scalar node. The mass for the scalar node is then divided equally among the degrees-of-freedom associated with that scalar node.

Section Keywords

The following is a list of keywords that apply to the user-defined elements.

The *SECTION_SHELL Card
Add Card 5 with the accompanying optional cards, Cards 5.1 and 5.2, of *SECTION_-SHELL to the keyword input deck if you invoke the user-defined shell element option.

Additional Card for ELFORM = 101,102,103,104 or 105

Card 5	1	2	3	4	5	6	7	8
Variable	NIPP	NXDOF	IUNF	IHGF	ITAJ	LMC	NHSV	ILOC
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Include NIPP cards according to the following format.

Card 5.1	1	2	3	4	5	6	7	8
Variable	XI	ETA	WGT					
Type	F	F	F					

Define LMC property parameters using 8 parameters per card.

Card 5.2	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
ELFORM	GT.100.AND.LT.106: User-defined shell
NIPP	Number of in-plane integration points for user-defined shell (0 if resultant element). Up to 100 allowed.
NXDOF	Number of extra degrees-of-freedom per node for user-defined shell
IUNF	Flag for using nodal fiber vectors in user-defined shell EQ.0: Nodal fiber vectors are not used. EQ.1: Nodal fiber vectors are used
IHGF	Flag for using hourglass stabilization (NIPP is greater than 0): EQ.0: Hourglass stabilization is not used. EQ.1: LS-DYNA hourglass stabilization is used. EQ.2: User-defined hourglass stabilization is used. EQ.3: Same as 2, but the resultant material tangent moduli are passed.
ITAJ	Flag for setting up finite element matrices (NIPP is greater than 0) EQ.0: Set up matrices with respect to the isoparametric domain EQ.1: Set up matrices with respect to the physical domain
LMC	Number of property parameters
NHSV	Number of history variables
ILOC	Coordinate system option: EQ.0: Pass all variables in the LS-DYNA local coordinate system EQ.1: Pass all variables in the global coordinate system
XI	First isoparametric coordinate
ETA	Second isoparametric coordinate
WGT	Isoparametric weight
P*i*	i^{th} property parameter

For more information on the variables, consult the previous sections in this appendix.

The *SECTION_SOLID Card

Add Card 3 with the accompanying optional cards, Cards 4 and 5, of *SECTION_SOLID to the input deck if you invoke the user-defined solid element option.

Additional card for ELFORM = 101,102,103,104 or 105

Card 3	1	2	3	4	5	6	7	8
Variable	NIP	NXDOF	IHGF	ITAJ	LMC	NHSV	XNOD	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Include NIP cards according to the following format.

Card 4	1	2	3	4	5	6	7	8
Variable	XI	ETA	ZETA	WGT				
Type	F	F	F	F				

Define LMC property parameters using 8 parameters per card.

Card 5	1	2	3	4	5	6	7	8
Variable	P1	P2	P3	P4	P5	P6	P7	P8
Type	F	F	F	F	F	F	F	F

VARIABLE	DESCRIPTION
ELFORM	GT.100.AND.LT.106: User-defined solid
NIP	Number of integration points for user-defined solid (0 if resultant element)
NXDOF	Number of extra degrees-of-freedom per node for user-defined solid

VARIABLE	DESCRIPTION
IHGF	Flag for using hourglass stabilization (NIP > 0): EQ.0: Hourglass stabilization is not used. EQ.1: LS-DYNA hourglass stabilization is used. EQ.2: User-defined hourglass stabilization is used. EQ.3: Same as 2, but the resultant material tangent moduli are passed.
ITAJ	Flag for setting up finite element matrices (NIP > 0): EQ.0: Set up matrices with respect to the isoparametric domain. EQ.1: Set up matrices with respect to the physical domain.
LMC	Number of property parameters
NHSV	Number of history variables
XNOD	Controls how the mass of extra degrees-of-freedom is interpreted when defined using user subroutine. EQ.0: The mass is defined per degree of freedom. EQ.1: The mass is defined as a nodal mass, where each scalar node has three extra degrees-of-freedom. See Extra Degrees of Freedom.
XI	First isoparametric coordinate
ETA	Second isoparametric coordinate
ZETA	Third isoparametric coordinate
WGT	Isoparametric weight
P*i*	i^{th} property parameter

For more information on the variables, consult the previous sections in this appendix.

Sample User Shell Element 101 (Belytschko-Tsay shell)

We can write the geometry of the Belytschko-Tsay element in local coordinates as:

$$x_i = (x_{iI} + \frac{t}{2}\xi_3\delta_{i3})N_I(\xi_1,\xi_2)$$
$$v_i = (v_{iI} + \frac{t}{2}\xi_3 e_{ij3}\omega_{jI})N_I(\xi_1,\xi_2)$$

where

$x_{iI} =$ i^{th} component of the coordinate of node I

$v_{iI} =$ i^{th} component of the translational velocity of node I

$\omega_{jI} =$ j^{th} component of the rotational velocity of node I

$t =$ thickness of the element

$e_{ijk} =$ permutation tensor

$N_I =$ shape function localized at node I

$\delta_{i3} =$ Kronecker delta

Taking the derivative of these expressions with respect to the isoparametric coordinate yields:

$$\frac{\partial x_i}{\partial \xi_1} = (x_{iI} + \frac{t}{2}\xi_3\delta_{i3})\frac{\partial N_I}{\partial \xi_1}$$
$$\frac{\partial x_i}{\partial \xi_2} = (x_{iI} + \frac{t}{2}\xi_3\delta_{i3})\frac{\partial N_I}{\partial \xi_2}$$
$$\frac{\partial x_i}{\partial \xi_3} = \frac{t}{2}\delta_{i3}$$

and

$$\frac{\partial v_i}{\partial \xi_1} = (v_{iI} + \frac{t}{2}\xi_3 e_{ij3}\omega_{jI})\frac{\partial N_I}{\partial \xi_1}$$
$$\frac{\partial v_i}{\partial \xi_2} = (v_{iI} + \frac{t}{2}\xi_3 e_{ij3}\omega_{jI})\frac{\partial N_I}{\partial \xi_2}$$
$$\frac{\partial v_i}{\partial \xi_3} = \frac{t}{2}e_{ij3}\omega_{jI}N_I$$

respectively. Using these expressions, we can implement the element as a user-defined shell as follows.

```
      subroutine ushl_b101(bmtrx,gmtrx,gjac,
     1     xi,eta,zeta,
     2     n1,n2,n3,n4,
     3     dn1dxi,dn2dxi,dn3dxi,dn4dxi,
     4     dn1deta,dn2deta,dn3deta,dn4deta,
     5     x1,x2,x3,x4,y1,y2,y3,y4,z1,z2,z3,z4,
     6     xdof,
     7     thick,thck1,thck2,thck3,thck4,
     8     fx1,fx2,fx3,fx4,
     9     fy1,fy2,fy3,fy4,
     .     fz1,fz2,fz3,fz4,
     .     gl11,gl21,gl31,gl12,gl22,gl32,gl13,gl23,gl33,
     .     lft,llt)
```

```
      include 'nlqparm'
c
c     Compute b and g matrix for user-defined shell 101
c
      dimension bmtrx(nlq,3,3,*),gmtrx(nlq,3,3),gjac(nlq)
      REAL n1,n2,n3,n4
      dimension x1(nlq),x2(nlq),x3(nlq),x4(nlq)
      dimension y1(nlq),y2(nlq),y3(nlq),y4(nlq)
      dimension z1(nlq),z2(nlq),z3(nlq),z4(nlq)
      dimension thick(nlq)
      dimension thck1(nlq),thck2(nlq),thck3(nlq),thck4(nlq)
      dimension xdof(nlq,8,3)
      dimension fx1(nlq),fx2(nlq),fx3(nlq),fx4(nlq)
      dimension fy1(nlq),fy2(nlq),fy3(nlq),fy4(nlq)
      dimension fz1(nlq),fz2(nlq),fz3(nlq),fz4(nlq)
      dimension gl11(nlq),gl21(nlq),gl31(nlq),
      gl12(nlq),gl22(nlq),gl32(nlq),
      gl13(nlq),gl23(nlq),gl33(nlq)
c
      do i=lft,llt
c
         gmtrx(i,1,1)=
     1        x1(i)*dn1dxi+x2(i)*dn2dxi+
     2        x3(i)*dn3dxi+x4(i)*dn4dxi
         gmtrx(i,2,1)=
     1        y1(i)*dn1dxi+y2(i)*dn2dxi+
     2        y3(i)*dn3dxi+y4(i)*dn4dxi
         gmtrx(i,3,1)=
     1        0.
         gmtrx(i,1,2)=
     1        x1(i)*dn1deta+x2(i)*dn2deta+
     2        x3(i)*dn3deta+x4(i)*dn4deta
         gmtrx(i,2,2)=
     1        y1(i)*dn1deta+y2(i)*dn2deta+
     2        y3(i)*dn3deta+y4(i)*dn4deta
         gmtrx(i,3,2)=
     1        0.
         gmtrx(i,1,3)=
     1        0.
         gmtrx(i,2,3)=
     1        0.
         gmtrx(i,3,3)=
     1        .5*thick(i)
c
         coef=.5*thick(i)*zeta
c
         bmtrx(i,1,1,1) =dn1dxi
         bmtrx(i,1,1,7) =dn2dxi
         bmtrx(i,1,1,13)=dn3dxi
         bmtrx(i,1,1,19)=dn4dxi
c
         bmtrx(i,1,1,5) =coef*dn1dxi
         bmtrx(i,1,1,11)=coef*dn2dxi
         bmtrx(i,1,1,17)=coef*dn3dxi
         bmtrx(i,1,1,23)=coef*dn4dxi
c
         bmtrx(i,1,2,1) =dn1deta
         bmtrx(i,1,2,7) =dn2deta
         bmtrx(i,1,2,13)=dn3deta
         bmtrx(i,1,2,19)=dn4deta
c
         bmtrx(i,1,2,5) =coef*dn1deta
         bmtrx(i,1,2,11)=coef*dn2deta
         bmtrx(i,1,2,17)=coef*dn3deta
         bmtrx(i,1,2,23)=coef*dn4deta
c
```

```
         bmtrx(i,2,1,2) =dn1dxi
         bmtrx(i,2,1,8) =dn2dxi
         bmtrx(i,2,1,14)=dn3dxi
         bmtrx(i,2,1,20)=dn4dxi
c
         bmtrx(i,2,1,4) =-coef*dn1dxi
         bmtrx(i,2,1,10)=-coef*dn2dxi
         bmtrx(i,2,1,16)=-coef*dn3dxi
         bmtrx(i,2,1,22)=-coef*dn4dxi
c
         bmtrx(i,1,3,5) =.5*thick(i)*n1
         bmtrx(i,1,3,11)=.5*thick(i)*n2
         bmtrx(i,1,3,17)=.5*thick(i)*n3
         bmtrx(i,1,3,23)=.5*thick(i)*n4
c
         bmtrx(i,3,1,3) =dn1dxi
         bmtrx(i,3,1,9) =dn2dxi
         bmtrx(i,3,1,15)=dn3dxi
         bmtrx(i,3,1,21)=dn4dxi
c
         bmtrx(i,2,2,2) =dn1deta
         bmtrx(i,2,2,8) =dn2deta
         bmtrx(i,2,2,14)=dn3deta
         bmtrx(i,2,2,20)=dn4deta
c
         bmtrx(i,2,2,4) =-coef*dn1deta
         bmtrx(i,2,2,10)=-coef*dn2deta
         bmtrx(i,2,2,16)=-coef*dn3deta
         bmtrx(i,2,2,22)=-coef*dn4deta
c
         bmtrx(i,2,3,4) =-.5*thick(i)*n1
         bmtrx(i,2,3,10)=-.5*thick(i)*n2
         bmtrx(i,2,3,16)=-.5*thick(i)*n3
         bmtrx(i,2,3,22)=-.5*thick(i)*n4
c
         bmtrx(i,3,2,3) =dn1deta
         bmtrx(i,3,2,9) =dn2deta
         bmtrx(i,3,2,15)=dn3deta
         bmtrx(i,3,2,21)=dn4deta
c
      enddo
c
      return
      end
```

To use the element for a part, we add the following section input to the keyword deck:

```
*SECTION_SHELL
$    SECID    ELFORM
         1       101
$       T1        T2        T3        T4

$     NIPP     NXDOF      IUNF      IHGF
         1         0         0         1
$       XI       ETA       WGT
        0.        0.        4.
```

Sample User Solid Element 101 (constant stress solid)

We can express the geometry for the constant stress solid as:

$$x_i = x_{iI} N_I(\xi_1, \xi_2, \xi_3)$$
$$v_i = v_{iI} N_I(\xi_1, \xi_2, \xi_3)$$

where

$x_{iI} =$ i^{th} component of the coordinate of node I

$v_{iI} =$ i^{th} component of the translational velocity of node I

$N_I =$ shape function localized at node I

Taking the derivative of these expressions with respect to the isoparametric coordinate yields:

$$\frac{\partial x_i}{\partial \xi_1} = x_{iI} \frac{\partial N_I}{\partial \xi_1}$$
$$\frac{\partial x_i}{\partial \xi_2} = x_{iI} \frac{\partial N_I}{\partial \xi_2}$$
$$\frac{\partial x_i}{\partial \xi_3} = x_{iI} \frac{\partial N_I}{\partial \xi_3}$$

and

$$\frac{\partial v_i}{\partial \xi_1} = v_{iI} \frac{\partial N_I}{\partial \xi_1}$$
$$\frac{\partial v_i}{\partial \xi_2} = v_{iI} \frac{\partial N_I}{\partial \xi_2}$$
$$\frac{\partial v_i}{\partial \xi_3} = v_{iI} \frac{\partial N_I}{\partial \xi_3}$$

respectively. Using these expressions, we can implement the element as a user-defined solid as follows.

```
      subroutine usld_b101(bmtrx,gmtrx,gjac,
     1     xi,eta,zeta,
     2     n1,n2,n3,n4,n5,n6,n7,n8,
     3     dn1dxi,dn2dxi,dn3dxi,dn4dxi,
     4     dn5dxi,dn6dxi,dn7dxi,dn8dxi,
     5     dn1deta,dn2deta,dn3deta,dn4deta,
     6     dn5deta,dn6deta,dn7deta,dn8deta,
     7     dn1dzeta,dn2dzeta,dn3dzeta,dn4dzeta,
     8     dn5dzeta,dn6dzeta,dn7dzeta,dn8dzeta,
     9     x1,x2,x3,x4,x5,x6,x7,x8,
     .     y1,y2,y3,y4,y5,y6,y7,y8,
     .     z1,z2,z3,z4,z5,z6,z7,z8,
     .     xdof,
     .     lft,llt)
      include 'nlqparm'
c
c     Compute b and g matrix for user-defined solid 101
c
      dimension bmtrx(nlq,3,3,*),gmtrx(nlq,3,3),gjac(nlq)
      REAL n1,n2,n3,n4,n5,n6,n7,n8
      dimension x1(nlq),x2(nlq),x3(nlq),x4(nlq)
      dimension x5(nlq),x6(nlq),x7(nlq),x8(nlq)
      dimension y1(nlq),y2(nlq),y3(nlq),y4(nlq)
      dimension y5(nlq),y6(nlq),y7(nlq),y8(nlq)
      dimension z1(nlq),z2(nlq),z3(nlq),z4(nlq)
      dimension z5(nlq),z6(nlq),z7(nlq),z8(nlq)
```

APPENDIX C

```
      dimension xdof(nlq,8,3)
c
      do i=lft,llt
c
         gmtrx(i,1,1)=x1(i)*dn1dxi+x2(i)*dn2dxi+
     1        x3(i)*dn3dxi+x4(i)*dn4dxi+
     2        x5(i)*dn5dxi+x6(i)*dn6dxi+
     3        x7(i)*dn7dxi+x8(i)*dn8dxi
         gmtrx(i,2,1)=y1(i)*dn1dxi+y2(i)*dn2dxi+
     1        y3(i)*dn3dxi+y4(i)*dn4dxi+
     2        y5(i)*dn5dxi+y6(i)*dn6dxi+
     3        y7(i)*dn7dxi+y8(i)*dn8dxi
         gmtrx(i,3,1)=z1(i)*dn1dxi+z2(i)*dn2dxi+
     1        z3(i)*dn3dxi+z4(i)*dn4dxi+
     2        z5(i)*dn5dxi+z6(i)*dn6dxi+
     3        z7(i)*dn7dxi+z8(i)*dn8dxi
         gmtrx(i,1,2)=x1(i)*dn1deta+x2(i)*dn2deta+
     1        x3(i)*dn3deta+x4(i)*dn4deta+
     2        x5(i)*dn5deta+x6(i)*dn6deta+
     3        x7(i)*dn7deta+x8(i)*dn8deta
         gmtrx(i,2,2)=y1(i)*dn1deta+y2(i)*dn2deta+
     1        y3(i)*dn3deta+y4(i)*dn4deta+
     2        y5(i)*dn5deta+y6(i)*dn6deta+
     3        y7(i)*dn7deta+y8(i)*dn8deta
         gmtrx(i,3,2)=z1(i)*dn1deta+z2(i)*dn2deta+
     1        z3(i)*dn3deta+z4(i)*dn4deta+
     2        z5(i)*dn5deta+z6(i)*dn6deta+
     3        z7(i)*dn7deta+z8(i)*dn8deta
         gmtrx(i,1,3)=x1(i)*dn1dzeta+x2(i)*dn2dzeta+
     1        x3(i)*dn3dzeta+x4(i)*dn4dzeta+
     2        x5(i)*dn5dzeta+x6(i)*dn6dzeta+
     3        x7(i)*dn7dzeta+x8(i)*dn8dzeta
         gmtrx(i,2,3)=y1(i)*dn1dzeta+y2(i)*dn2dzeta+
     1        y3(i)*dn3dzeta+y4(i)*dn4dzeta+
     2        y5(i)*dn5dzeta+y6(i)*dn6dzeta+
     3        y7(i)*dn7dzeta+y8(i)*dn8dzeta
         gmtrx(i,3,3)=z1(i)*dn1dzeta+z2(i)*dn2dzeta+
     1        z3(i)*dn3dzeta+z4(i)*dn4dzeta+
     2        z5(i)*dn5dzeta+z6(i)*dn6dzeta+
     3        z7(i)*dn7dzeta+z8(i)*dn8dzeta
c
         bmtrx(i,1,1,1) =dn1dxi
         bmtrx(i,1,1,4) =dn2dxi
         bmtrx(i,1,1,7) =dn3dxi
         bmtrx(i,1,1,10)=dn4dxi
         bmtrx(i,1,1,13)=dn5dxi
         bmtrx(i,1,1,16)=dn6dxi
         bmtrx(i,1,1,19)=dn7dxi
         bmtrx(i,1,1,22)=dn8dxi
c
         bmtrx(i,2,1,2) =dn1dxi
         bmtrx(i,2,1,5) =dn2dxi
         bmtrx(i,2,1,8) =dn3dxi
         bmtrx(i,2,1,11)=dn4dxi
         bmtrx(i,2,1,14)=dn5dxi
         bmtrx(i,2,1,17)=dn6dxi
         bmtrx(i,2,1,20)=dn7dxi
         bmtrx(i,2,1,23)=dn8dxi
c
         bmtrx(i,3,1,3) =dn1dxi
         bmtrx(i,3,1,6) =dn2dxi
         bmtrx(i,3,1,9) =dn3dxi
         bmtrx(i,3,1,12)=dn4dxi
         bmtrx(i,3,1,15)=dn5dxi
         bmtrx(i,3,1,18)=dn6dxi
         bmtrx(i,3,1,21)=dn7dxi
```

```
          bmtrx(i,3,1,24)=dn8dxi
c
          bmtrx(i,1,2,1) =dn1deta
          bmtrx(i,1,2,4) =dn2deta
          bmtrx(i,1,2,7) =dn3deta
          bmtrx(i,1,2,10)=dn4deta
          bmtrx(i,1,2,13)=dn5deta
          bmtrx(i,1,2,16)=dn6deta
          bmtrx(i,1,2,19)=dn7deta
          bmtrx(i,1,2,22)=dn8deta
c
          bmtrx(i,2,2,2) =dn1deta
          bmtrx(i,2,2,5) =dn2deta
          bmtrx(i,2,2,8) =dn3deta
          bmtrx(i,2,2,11)=dn4deta
          bmtrx(i,2,2,14)=dn5deta
          bmtrx(i,2,2,17)=dn6deta
          bmtrx(i,2,2,20)=dn7deta
          bmtrx(i,2,2,23)=dn8deta
c
          bmtrx(i,3,2,3) =dn1deta
          bmtrx(i,3,2,6) =dn2deta
          bmtrx(i,3,2,9) =dn3deta
          bmtrx(i,3,2,12)=dn4deta
          bmtrx(i,3,2,15)=dn5deta
          bmtrx(i,3,2,18)=dn6deta
          bmtrx(i,3,2,21)=dn7deta
          bmtrx(i,3,2,24)=dn8deta
c
          bmtrx(i,1,3,1) =dn1dzeta
          bmtrx(i,1,3,4) =dn2dzeta
          bmtrx(i,1,3,7) =dn3dzeta
          bmtrx(i,1,3,10)=dn4dzeta
          bmtrx(i,1,3,13)=dn5dzeta
          bmtrx(i,1,3,16)=dn6dzeta
          bmtrx(i,1,3,19)=dn7dzeta
          bmtrx(i,1,3,22)=dn8dzeta
c
          bmtrx(i,2,3,2) =dn1dzeta
          bmtrx(i,2,3,5) =dn2dzeta
          bmtrx(i,2,3,8) =dn3dzeta
          bmtrx(i,2,3,11)=dn4dzeta
          bmtrx(i,2,3,14)=dn5dzeta
          bmtrx(i,2,3,17)=dn6dzeta
          bmtrx(i,2,3,20)=dn7dzeta
          bmtrx(i,2,3,23)=dn8dzeta
c
          bmtrx(i,3,3,3) =dn1dzeta
          bmtrx(i,3,3,6) =dn2dzeta
          bmtrx(i,3,3,9) =dn3dzeta
          bmtrx(i,3,3,12)=dn4dzeta
          bmtrx(i,3,3,15)=dn5dzeta
          bmtrx(i,3,3,18)=dn6dzeta
          bmtrx(i,3,3,21)=dn7dzeta
          bmtrx(i,3,3,24)=dn8dzeta
c
      enddo
c
      return
      end
```

To use the element for a part, we add the following section input to the keyword deck

```
*SECTION_SOLID
$    SECID    ELFORM
```

```
         1       101
$       NIP     NXDOF      IHGF
         1         0         1
$        XI       ETA      ZETA       WGT
         0.        0.        0.       8.0
```

Examples

We present three test examples. Our first example is a simple tension-compression test of a solid cylinder. Figure 53-1 shows the geometry. We used the sample implementations of user elements and compared the results and performance with standard LS-DYNA elements. As for the computational efficiency, we note that the performance is worse. We expected this decrease in performance because we sought a user-friendly interface that came at the expense of performance. The implicit performance compares well with the other elements in LS-DYNA.

The second example is a combined bending and stretching simulation with the geometry shown in Figure 53-2. We ran the problem with the user element implementations and compared the results and performance with standard LS-DYNA elements. We saw the same tendencies as for the solid elements.

The third example is an impact between a solid bar and a shell beam. Both parts are modeled with user-defined elements. The results were similar to the ones obtained by substituting the sections for standard LS-DYNA sections, but the simulation time was about 3-4 times longer.

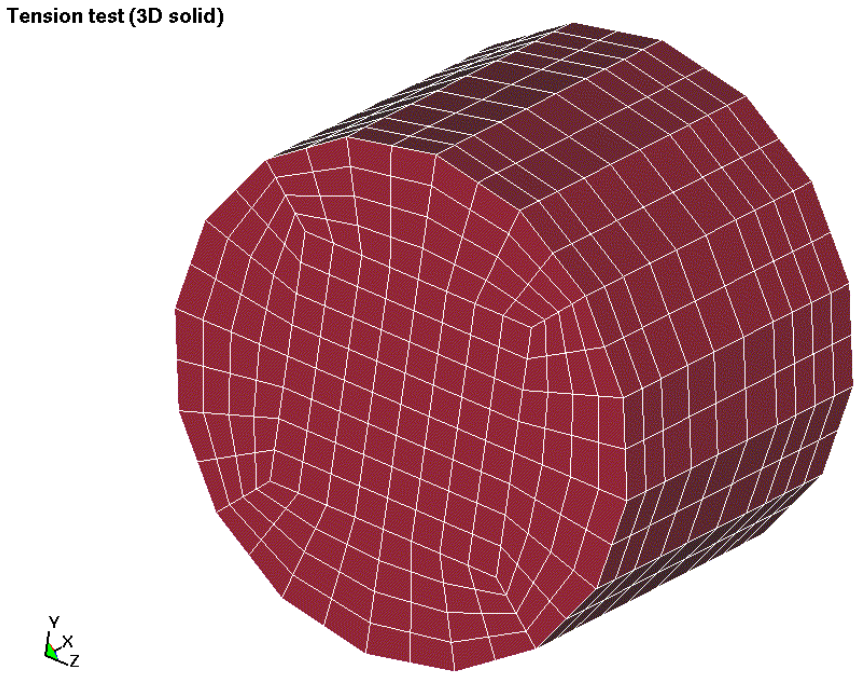


Figure 53-1. Solid mesh for user element test.



Figure 53-2. Shell mesh for the user element test.

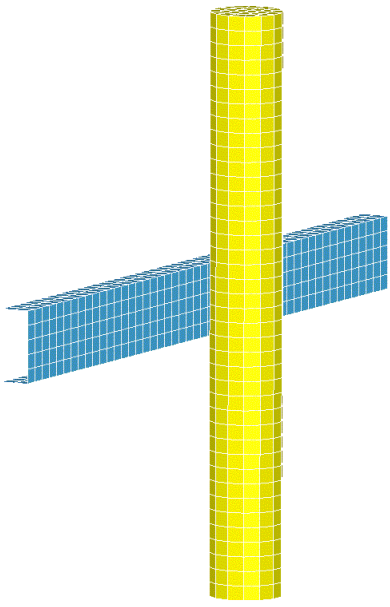


Figure 53-3. Impact between a user-defined shell and user-defined solid part.

APPENDIX D: User Defined Airbag Sensor

The addition of a user sensor subroutine into LS-DYNA is relatively simple. The sensor is mounted on a rigid body which is attached to the structure. The motion of the sensor is provided in the local coordinate system defined for the rigid body in the definition of material model 20–the rigid material. When the user defined criterion is met for the deployment of the airbag, a flag is set and the deployment begins. All load curves relating to the mass flow rate versus time are then shifted by the initiation time. The user subroutine is given below with all the necessary information contained in the comment cards.

```
      subroutine airusr (rbu,rbv,rba,time,dt1,dt2,param,hist,itrnon,
     . rbug,rbvg,rbag,icnv)
c
c*************************************************************
c|  Livermore Software Technology Corporation  (LSTC)         |
c|  ------------------------------------------------------    |
c|  Copyright 1987-2008 Livermore Software Tech.  Corp         |
c|  All rights reserved                                       |
c*************************************************************
c
c     user subroutine to initiate the inflation of the airbag
c
c     variables
c
c     displacements are defined at time n+1 in local system
c     velocites are defined at time n+1/2 in local system
c     accelerations are defined at time n in local system
c
c         rbu(1-3) total displacements in the local xyz directions
c         rbu(3-6) total rotations about the local xyz axes
c         rbv(1-3) velocities in the local xyz directions
c         rbv(3-6) rotational velocities about the local xyz axes
c         rba(1-3) accelerations in the local xyz directions
c         rba(3-6) rotational accelerations about the local xyz axes
c         time is the current time
c         dt1 is time step size at n-1/2
c         dt2 is time step size at n+1/2
c         param is user defined input parameters
c         hist is user defined history variables
c         itrnon is a flag to turn on the airbag inflation
c         rbug,rbvg,rbag, are similar to rbu,rbv,rba but are defined
c                         globally.
c         icnv is the airbag ID
c
c     the user subroutine sets the variable itrnon to:
c
c           itrnon=0 bag is not inflated
c           itrnon=1 bag inflation begins and this subroutine is
c                    not called again
c
      include 'iounits.inc'
      dimension rbu(6),rbv(6),rba(6),param(25),hist(25),
     . rbug(6),rbvg(6),rbag(6)
c
      itrnon=0
      ra=sqrt(rba(1)**2+rba(2)**2+rba(3)**2)
      if (ra.gt.param(1)) then
        itrnon=1
```

APPENDIX D

```
        write(iotty,100) time
        write(iohsp,100) time
        write(iomsg,100) time
      endif
 100  format (' Airbag activated at time ',1pe10.3)
c
      return
      end
```

APPENDIX E: User Defined Solution Control

You can control the I/O, monitor the energies and other solution norms of interest, and shut down the problem whenever you please with user subroutine `uctrl1` in dyn21.F. The arguments are defined in the listing provided below. This subroutine is called each time step and does not need any control card to operate.

```
      subroutine uctrl1 (numnp,ndof,time,dt1,dt2,prtc,pltc,frci,prto,
     . plto,frco,vt,vr,at,ar,ut,ur,xmst,xmsr,irbody,rbdyn,usrhv,
     . messag,totalm,cycle,idrint,mtype,lrb,nrba,rbcor,x,rbv,nrbn,
     . nrb,xrb,yrb,zrb,axrb,ayrb,azrb,dtx,nmmat,rba,fvalnew,fvalold,
     . fvalmid,fvalnxt)
c
c*************************************************************
c|  Livermore Software Technology Corporation  (LSTC)         |
c|  ------------------------------------------------------    |
c|  Copyright 1987-2008 Livermore Software Tech.  Corp        |
c|  All rights reserved                                       |
c*************************************************************
c
c     user subroutine for solution control.  It is called at the
c     beginning of time step n+1.
c
c
c     note:  ls-dyna uses an internal numbering system to
c            accommodate arbitrary node numbering.   to access
c            information for user node n, address array location m,
c            m = lqf8(n,1).  to obtain user node number, n,
c            corresponding to array address m, set n = lqfinv(m,1)
c
c     arguments:
c          numnp = number of nodal points
c          ndof = number of degrees of freedom per node
c          time = current solution time at n+1
c          dt1  = time step size between time n-1 and n
c          dt2  = time step size between time n and n+1
c          prtc = output interval for taurus time history data
c          pltc = output interval for taurus state data
c          frci = output interval for taurus interface force data
c          prto = output time for time history file
c          plto = output time for state data
c          frco = output time for force data
c          vt(3,numnp) = nodal translational velocity vector
c          vr(3,numnp) = nodal rotational velocity vector.  this
c                        array is defined if and only if ndof = 6
c          at(3,numnp) = nodal translational acceleration vector
c          ar(3,numnp) = nodal rotational acceleration vector.  this
c                        array is defined if and only if ndof = 6
c          ut(3,numnp) = nodal translational displacement vector
c          ur(3,numnp) = nodal rotational displacement vector.  this
c                        array is defined if and only if ndof = 6
c          xmst(numnp) = reciprocal of nodal translational masses
c          xmsr(numnp) = reciprocal of nodal rotational masses.  this
c                        array is defined if and only if ndof = 6
```

```
c          fvalold     = array for storing load curve values at time n
c          fvalnew     = array for setting load curve values at time n+1
c                         only load curves with 0 input points may be user
c                         defined.  When the load curve is user set, the
c                         value at time n must be stored in array fvalold.
c          fvalmid     = array for predicting load curve values at time n+3/2
c          fvalnxt     = array for predicting load curve values at time n+2
c                         for some applications it is necessary to predict the
c                        load curve values at time n+2, a time that is not known,
c                         this for instance for boundary prescribed motion.  In
c                         this case the load curve values at time n+3/2 need to
c                         be predicted in fvalmid, and fvalold should be set to
c                         fvalnew and fvalnew should be set to fvalnxt.  See
c                         coding below.
c
c          irbody      = 0 if no rigid bodies
c          rbdyn(numnp)=flag for rigid body nodal points
c                         if deformable node then set to 1.0
c                         if rigid body node then set to 0.0
c                         defined if and only if rigid bodies are present
c                         i.e., irbody.ne.0 if no rigid bodies are
c                         present
c          usrhv(lenhv)=user defined history variables that are stored
c                         in the restart file.  lenhv = 100+7*nummat where
c                         nummat is the # of materials in the problem.
c                         array usrhv is updated only in this subroutine.
c          messag      = flag for dyna3d which may be set to:
c                        'sw1.' ls-dyna3d terminates with restart file
c                        'sw3.' ls-dyna3d writes a restart file
c                        'sw4.' ls-dyna3d writes a plot state
c           totalm     = total mass in problem
c           cycle      = cycle number
c           idrint     = flag for dynamic relaxation phase
c                         .ne.0: dynamic relaxation in progress
c                         .eq.0: solution phase
c          mtype(*)    = material type for each part in the model
c          lrb(*)      = lead rigid body for each rigid body
c                        If part n is rigid and is a lead or has not
c                        been merged then lrb(n)==n.
c          nrba(*)     = starting index in nrb(*) of the nodes for each
c                        rigid body
c          rbcor(3,*)  = rigid body cg coordinates
c          x(3,*)      = Node coordinate array
c          rbv(6,*)    = rigid body cg velocity
c          nrbn(*)     = # nodes in each rigid body
c          nrb(*)      = List of all rigid body nodes
c          xrb(*)      = RB scratch array as long as longest nrbn() value
c          yrb(*)      = RB scratch array as long as longest nrbn() value
c          zrb(*)      = RB scratch array as long as longest nrbn() value
c          axrb(*)     = RB scratch array as long as longest nrbn() value
c          ayrb(*)     = RB scratch array as long as longest nrbn() value
c          azrb(*)     = RB scratch array as long as longest nrbn() value
c          dtx         = (dt1+dt2)*0.5 except at time 0 when it is = dt2
c          nmmat       = number of parts in the model
c          rba(6,*)    = rigid body cg acceleration
c          fvalnew     = array for setting load curve values at time n+1
c                        only load curves with 0 input points may be user
c                        defined.  When the load curve is user set, the
c                        value at time n must be stored in array fvalold.
c          fvalold     = array for storing load curve values at time n
```

```
c           fvalmid     = array for predicting load curve values at time n+3/2
c           fvalnxt     = array for predicting load curve values at time n+2
c                         for some applications it is necessary to predict the
c                        load curve values at time n+2, a time that is not known,
c                         this for instance for boundary prescribed motion.  In
c                         this case the load curve values at time n+3/2 need to
c                         be predicted in fvalmid, and fvalold should be set to
c                         fvalnew and fvalnew should be set to fvalnxt.  See
c                         coding below.
c
      include 'ptimes.inc'
c
c     prtims(1-37)=output intervals for ascii files
c
c      ascii files:
c           ( 1)-cross section forces
c           ( 2)-rigid wall forces
c           ( 3)-nodal data
c           ( 4)-element data
c           ( 5)-global data
c           ( 6)-discrete elements
c           ( 7)-material energies
c           ( 8)-noda interface forces
c           ( 9)-resultant interface forces
c           (10)-smug animator
c           (11)-spc reaction forces
c           (12)-nodal constraint resultant forces
c           (13)-airbag statistics
c           (14)-avs database
c           (15)-nodal force groups
c           (16)-output intervals for nodal boundary conditions
c           (17)-(32) unused at this time
c           (37)-auto tiebreak damage output
c
c     prtlst(32)=output times for ascii files above.  when solution time
c                exceeds the output time a print state is dumped.
c
      common/rbkeng/enrbdy,rbdyx,rbdyy,rbdyz
c
c      total rigid body energies and momentums:
c           enrbdy = rigid body kinetic energy
c           rbdyx = rigid body x-momentum
c           rbdyy = rigid body y-momentum
c           rbdyz = rigid body z-momentum
c
      common/swmke/swxmom,swymom,swzmom,swkeng
c
c      total stonewall energies and momentums:
c           swxmom = stonewall x-momentum
c           swymom = stonewall y-momentum
c           swzmom = stonewall z-momentum
c           swkeng = stonewall kinetic energy
c
      common/deengs/deeng
c
c      deeng = total discrete element energy
c
      common/bk28/summss,xke,xpe,tt,xte0,erodeke,erodeie,selie,selke,
     . erodehg
c
```

```
c      xpe  = total internal energy in the finite elements
c
       common/sprengs/spreng
c
c      spreng = total spr energy
c
       character*(*) messag
       integer cycle
       real*8 x
       dimension vt(3,*),vr(3,*),at(3,*),ar(3,*),
      . xmst(*),xmsr(*),rbdyn(*),usrhv(*),mtype(*),lrb(*),nrba(*),
      . rbcor(3,1),x(*),rbv(6,*),nrbn(*),nrb(*),xrb(*),yrb(*),
      . zrb(*),axrb(*),ayrb(*),azrb(*),rba(6,*),fvalnew(*),
      . fvalold(*),fvalmid(*),fvalnxt(*)
       real*8 ut(3,*),ur(3,*)
c
c      sample momentum and kinetic energy calculations
c
c      remove all comments in column 1 below to activate
       i = 1
       if (i.eq.1) return
c      return
cc
cc
cc     initialize kinetic energy, xke, and x,y,z momentums.
cc
c      xke = 2.*swkeng+2.*enrbdy
c      xm = swxmom+rbdyx
c      ym = swymom+rbdyy
c      zm = swzmom+rbdyz
cc
c      numnp2 = numnp
c      if (ndof.eq.6) then
c        numnp2 = numnp+numnp
c      endif
c      write(iotty,*)ndof
cc
cc
cc     no rigid bodies present
cc
c      if (irbody.eq.0) then
cc       note in blank comment vr follows vt.  this fact is used below.
c        do 10 n = 1,numnp2
c        xmsn = 1./xmst(n)
c        vn1 = vt(1,n)
c        vn2 = vt(2,n)
c        vn3 = vt(3,n)
c        xm = xm+xmsn*vn1
c        ym = ym+xmsn*vn2
c        zm = zm+xmsn*vn3
c        xke = xke+xmsn*(vn1*vn1+vn2*vn2+vn3*vn3)
c   10   continue
cc
cc
cc     rigid bodies present
cc
c      else
cc       nodal accerations for rigid bodies
cc
c        do 12 n = 1,nmmat
```

```
c        if (mtype(n).ne.20.or.lrb(n).ne.n) go to 12
c        lrbn = nrba(n)
c        call stvlut(rbcor(1,n),x,vt,at,ar,vr,rbv(1,n),dt2,
c    .   nrbn(n),nrb(lrbn),xrb,yrb,zrb,axrb,ayrb,azrb,dtx)
c
c        rigid body nodal accelerations
c
c        if (ndof.eq.6) then
c          call rbnacc(nrbn(n),nrb(lrbn),rba(4,n),ar)
c        endif
c
c  12    continue
cc
c        do 20 n = 1,numnp
c        xmsn = 1./xmst(n)
c        vn1 = rbdyn(n)*vt(1,n)
c        vn2 = rbdyn(n)*vt(2,n)
c        vn3 = rbdyn(n)*vt(3,n)
c        xm = xm+xmsn*vn1
c        ym = ym+xmsn*vn2
c        zm = zm+xmsn*vn3
c        xke = xke+xmsn*(vn1*vn1+vn2*vn2+vn3*vn3)
c   20   continue
c        if (ndof.eq.6) then
c          do 30 n = 1,numnp
c          xmsn = 1./xmsr(n)
c          vn1 = rbdyn(n)*vr(1,n)
c          vn2 = rbdyn(n)*vr(2,n)
c          vn3 = rbdyn(n)*vr(3,n)
c          xm = xm+xmsn*vn1
c          ym = ym+xmsn*vn2
c          zm = zm+xmsn*vn3
c          xke = xke+xmsn*(vn1*vn1+vn2*vn2+vn3*vn3)
c   30     continue
c        endif
c      endif
cc
cc     total kinetic energy
c      xke=.5*xke
cc     total internal energy
c      xie = xpe+deeng+spreng
cc     total energy
c      xte = xke+xpe+deeng+spreng
cc     total x-rigid body velocity
c      xrbv = xm/totalm
cc     total y-rigid body velocity
c      yrbv = ym/totalm
cc     total z-rigid body velocity
c      zrbv = zm/totalm
       return
       end
```

APPENDIX F: User Defined Interface Control

This subroutine may be provided by the user to turn the interfaces on and off. This option is activated by the *USER_INTERFACE_CONTROL keyword. The arguments are defined in the listing provided below.

```
      subroutine uctrl2(nsi,nty,time,cycle,nsb,nsbn,nsa,nsan,
     1 thsb,thsa,vt,xi,ut,iskip,idrint,numnp,dt2,ninput,ua,
     2 irectsb,nrtsb,irectsa,nrtsa)
c
c*************************************************************
c|  Livermore Software Technology Corporation  (LSTC)         |
c|  ---------------------------------------------------------  |
c|  Copyright 1987-2008 Livermore Software Tech.  Corp        |
c|  All rights reserved                                       |
c*************************************************************
c
c     user subroutine for interface control
c
c     note:  ls-dyna uses an internal numbering system to
c            accommodate arbitrary node numbering.  To access
c            information for user node n, address array location m,
c            m = lqf8(n,1).  To obtain user node number, n,
c            corresponding to array address m, set n = lqfinv(m,1)
c
c     arguments:
c          nsi          = number of sliding interface
c          nty          = interface type.
c                           .eq.4:single surface
c                           .ne.4:surface to surface
c          time         = current solution time
c          cycle        = cycle number
c          nsb(nsbn)    = list of surfb nodes numbers in internal
c                         numbering scheme
c          nsbn         = number of surfb nodes
c          nsa(nsan)    = list of surfa nodes numbers in internal
c                         numbering scheme
c          nsan         = number of surfa nodes
c          thsb(nsbn)   = surfb node thickness
c          thsa(nsan)   = surfa node thickness
c          vt(3,numnp)  = nodal translational velocity vector
c          xi(3,numnp)  = initial coordinates at time = 0
c          ut(3,numnp)  = nodal translational displacement vector
c          idrint       = flag for dynamic relaxation phase
c                         .ne.0: dynamic relaxation in progress
c                         .eq.0: solution phase
c          numnp        = number of nodal points
c          dt2          = time step size at n+1/2
c          ninput       = number of variables input into ua
c          ua(*)        = users' array, first ninput locations
c                         defined by user.  the length of this
c                         array is defined on control card 10.
c                         this array is unique to interface nsi.
c          irectsb(4,*) = list of surfb segments in internal
```

```
c                          numbering scheme
c          nrtsb         = number of surfb segments
c          irectsa(4,*) = list of surfa segments in internal
c                          numbering scheme
c          nrtsa         = number of surfa segments
c
c     set flag for active contact
c       iskip = 0 active
c       iskip = 1 inactive
c
c*********************************************************************
c
      integer cycle
      real*8 ut
      real*8 xi
      dimension nsb(*),nsa(*),thsb(*),thsa(*),vt(3,*),xi(3,*),
     .          ut(3,*),ua(*),irectsb(4,*),irectsa(4,*)
c
c     The following sample code is provided to illustrate how
c     this subroutine might be used.  Here we check to see if the
c     surfaces in the surface to surface contact are separated.  If
c     so, iskip = 1, and the contact treatment is skipped.
c
c     if (nty.eq.4) return
c     dt2hlf = dt2/2.
c     xminsa = 1.e+16
c     xmaxsa = -xminsa
c     yminsa = 1.e+16
c     ymaxsa = -yminsa
c     zminsa = 1.e+16
c     zmaxsa = -zminsa
c     xminsb = 1.e+16
c     xmaxsb = -xminsb
c     yminsb = 1.e+16
c     ymaxsb = -yminsb
c     zminsb = 1.e+16
c     zmaxsb = -zminsb
c     thksa = 0.0
c     thksb = 0.0
c     do 10 i = 1,nsan
c     dsp1 = ut(1,nsa(i))+dt2hlf*vt(1,nsa(i))
c     dsp2 = ut(2,nsa(i))+dt2hlf*vt(2,nsa(i))
c     dsp3 = ut(3,nsa(i))+dt2hlf*vt(3,nsa(i))
c     x1 = xi(1,nsa(i))+dsp1
c     x2 = xi(2,nsa(i))+dsp2
c     x3 = xi(3,nsa(i))+dsp3
c     thksa = max(thsa(i),thksa)
c     xminsa = min(xminsa,x1)
c     xmaxsa = max(xmaxsa,x1)
c     yminsa = min(yminsa,x2)
c     ymaxsa = max(ymaxsa,x2)
c     zminsa = min(zminsa,x3)
c     zmaxsa = max(zmaxsa,x3)
c  10 continue
c     do 20 i = 1,nsbn
c     dsp1 = ut(1,nsb(i))+dt2hlf*vt(1,nsb(i))
c     dsp2 = ut(2,nsb(i))+dt2hlf*vt(2,nsb(i))
c     dsp3 = ut(3,nsb(i))+dt2hlf*vt(3,nsb(i))
c     x1 = xi(1,nsb(i))+dsp1
c     x2 = xi(2,nsb(i))+dsp2
```

```
c     x3 = xi(3,nsb(i))+dsp3
c     thksb = max(thsb(i),thksb)
c     xminsb = min(xminsb,x1)
c     xmaxsb = max(xmaxsb,x1)
c     yminsb = min(yminsb,x2)
c     ymaxsb = max(ymaxsb,x2)
c     zminsb = min(zminsb,x3)
c     zmaxsb = max(zmaxsb,x3)
c  20 continue
c
c     If thksa or thksb equal zero, set them to some reasonable value.
c
c     if (thksa.eq.0.0) then
c       e1=(xi(1,irectsa(1,1))-xi(1,irectsa(3,1)))**2
c    .    +(xi(2,irectsa(1,1))-xi(2,irectsa(3,1)))**2
c    .    +(xi(3,irectsa(1,1))-xi(3,irectsa(3,1)))**2
c       e2=(xi(1,irectsa(2,1))-xi(1,irectsa(4,1)))**2
c    .    +(xi(2,irectsa(2,1))-xi(2,irectsa(4,1)))**2
c    .    +(xi(3,irectsa(2,1))-xi(3,irectsa(4,1)))**2
c       thksa=.3*sqrt(max(e1,e2))
c     endif
c     if (thksb.eq.0.0) then
c       e1=(xi(1,irectsb(1,1))-xi(1,irectsb(3,1)))**2
c    .    +(xi(2,irectsb(1,1))-xi(2,irectsb(3,1)))**2
c    .    +(xi(3,irectsb(1,1))-xi(3,irectsb(3,1)))**2
c       e2=(xi(1,irectsb(2,1))-xi(1,irectsb(4,1)))**2
c    .    +(xi(2,irectsb(2,1))-xi(2,irectsb(4,1)))**2
c    .    +(xi(3,irectsb(2,1))-xi(3,irectsb(4,1)))**2
c       thksb=.3*sqrt(max(e1,e2))
c     endif
c
c     if (xmaxsa+thksa.lt.xminsb-thksb) go to 40
c     if (ymaxsa+thksa.lt.yminsb-thksb) go to 40
c     if (zmaxsa+thksa.lt.zminsb-thksb) go to 40
c     if (xmaxsb+thksb.lt.xminsa-thksa) go to 40
c     if (ymaxsb+thksb.lt.yminsa-thksa) go to 40
c     if (zmaxsb+thksb.lt.zminsa-thksa) go to 40
c     iskip = 0
c
c     return
c  40 iskip = 1
c
      return
      end
```

APPENDIX G: User-Defined Interface Friction and Conductivity

We provide an easy-to-use user contact interface, where you can define the frictional coefficients (static and dynamic) as well as contact heat transfer conductance as functions of contact pressure, relative sliding velocity, separation and temperature. To be able to use this feature, an object version of the code is required, and you must write your own Fortran (or C) code to define the contact parameters of interest.

The object version comes with various text files in which user subroutines can be included. The subroutines of interest are

```
subroutine usrfrc(fstt,fdyn,...)
```

for defining the frictional coefficients fstt (static) and fdyn (dynamic) and

```
subroutine usrhcon(h,...)
```

for defining the heat transfer contact conductance, *h*. Subroutines `usrfrc` and `usrhcon` are dyn21cnt.F and dyn21.F. Due to the significant change in characteristics between the regular node-to-surface contact and segment-to-segment Mortar contact, the subroutine

```
subroutine mortar_usrfrc(…)
```

in dyn21cnt.F is intended for usage with Mortar contacts. This routine is supposed to return the coefficient of friction for the current contact state and is in this sense very similar to the other routine. For an explanation of all arguments, we refer to the source code.

We emphasize at this point that the user friction interface for non-Mortar differs between SMP and MPP for reasons that have to do with how the contacts are implemented in general. In SMP, you are required not only to define the frictional coefficients but also to assemble and store contact forces and history, whereas in MPP, only the frictional coefficients have to be defined. For Mortar contact, the same routine is called for both SMP and MPP, and only one frictional coefficient needs to be defined.

For the friction interface (SMP and MPP), you may associate history variables with each contact node (or segment for Mortar contact). Unfortunately, the user friction interface is currently not supported by all available contacts, but it should cover the most interesting ones: *CONTACT_(AUTOMATIC/FORMING_)NODES_TO_SURFACE, *CONTACT_(AUTOMATIC/FORMING_)SURFACE_TO_SURFACE, *CONTACT_(AUTOMATIC/FORMING_)ONE_WAY_SURFACE_TO_SURFACE, and all Mortar contacts. Note that non-Mortar segment-to-segment contacts (SOFT = 2 on Optional Card A of *CONTACT_...) are *not* supported. The friction interface is not supported for 2D contacts, except in the case of Mortar contact. Upon request by customers, additional contact types can be supported.

APPENDIX G

One of the arguments to the user contact routines is the curve array `crv`, also available in the user material interface. Note that when using this array, the curve identity must be converted to an internal number or the subroutine `crvval` may be utilized. For more information, see Appendix A on user materials.

For definition of user contact parameters, you must define *USER_INTERFACE_FRICTION for friction and *USER_INTERFACE_CONDUCTIVITY for conductance. The card formats for these two keywords are identical.

There is an alternate route to defining the conductivity parameters for a user defined thermal contact. On the *CONTACT_..._THERMAL_FRICTION optional card the parameter FORMULA may be set to a negative number. This will automatically create a user defined conductivity interface and invoke reading of –FORMULA contact parameters immediately following the card including the FORMULA parameter. Note that FORMULA is related to NOC and NOCI in the *USER_INTERFACE_CONDUCTIVITY keyword as

$$-\text{FORMULA} = \text{NOC} = \text{NOCI}.$$

Note that the pressure is automatically computed for each user conductivity interface, meaning the keyword *LOAD_SURFACE_STRESS is not necessary.

A sample friction subroutine for the non-Mortar case is provided below for SMP. For Mortar contact we refer to the source code for an example.

```
      subroutine usrfrc(nosl,time,ncycle,dt2,intr,areat,xt,yt,zt,
     . lsv,ix1,ix2,ix3,ix4,arear,xx1,xx2,xx3,stfn,stf,fni,
     . dx,dy,dz,fdt2,ninput,ua,side,iisv5,niisv5,n1,n2,n3,fric1,
     . fric2,fric3,fric4,bignum,fdat,iseg,fxis,fyis,fzis,ss,tt,
     . ilbsv,stfk,frc,numnp,npc,pld,lcfst,lcfdt,temp,temp_bot,
     . temp_top,isurface)
c
c*****************************************************************
c|  LIVERMORE SOFTWARE TECHNOLOGY CORPORATION  (LSTC)             |
c|  ------------------------------------------------------------  |
c|  COPYRIGHT © 1987-2007 JOHN O. HALLQUIST, LSTC                 |
c|  ALL RIGHTS RESERVED                                           |
c*****************************************************************
c
c     user subroutine for interface friction control
c
c     note:  LS-DYNA uses an internal numbering system to
c            accomodate arbitrary node numbering.  To access
c            information for user node n, address array location m,
c            m=lqf(n,1).  To obtain user node number, n,
c            corresponding to array address m, set n=lqfinv(m,1)
c
c     arguments:
c
c          nosl       =number of sliding interface
c          time       =current solution time
c          ncycle     =ncycle number
c          dt2        =time step size at n+1/2
c          intr       =tracked node array where the nodes are stored
c                      in ls-dyna internal numbering.  User numbers
c                      are given by function: lqfinv(insv(ii),1)
```

```
c                      for tracked node ii.
c          areat(ii)  =tracked node area (interface types 5&10 only) for
c                      tracked node ii
c          xt(ii)     =x-coordinate tracked node ii (projected)
c          yt(ii)     =y-coordinate tracked node ii (projected)
c          zt(ii)     =z-coordinate tracked node ii (projected)
c          lsv(ii)    =reference segment number for tracked node ii
c          ix1(ii), ix2(ii), ix3(ii), ix4(ii)
c                     =reference segment nodes in ls-dyna internal
c                      numbering for tracked node ii
c          arear(ii)  =reference segment area for tracked node ii.
c          xx1(ii,4)  =x-coordinates reference surface (projected) for
c                      tracked node ii
c          xx2(ii,4)  =y-coordinates reference surface (projected) for
c                      tracked node ii
c          xx3(ii,4)  =z-coordinates reference surface (projected) for
c                      tracked node ii
c          stfn       =tracked node penalty stiffness
c          stf        =reference segment penalty stiffness
c          fni        =normal force
c          dx,dy,dz   =relative x,y,z-displacement between tracked node and
c                      reference surface.  Multipling by fdt2 defines the
c                      relative velocity.
c          n1,n2,n3   =x,y, and z components of reference segment's normal
c                      vector
c
c*********************************************************************
c       frictional coefficients defined for the contact interface
c
c          fric1      =static friction coefficient
c          fric2      =dynamic friction coefficient
c          fric3      =decay constant
c          fric4      =viscous friction coefficient (setting fric4=0
c                      turns this option off)
c
c*********************************************************************
c
c          bignum     =0.0 for one way surface to surface and
c                      for surface to surface, and 1.e+10 for nodes
c                      to surface contact
c          ninput     =number of variables input into ua
c          ua(*)      =users' array, first ninput locations
c                      defined by user.  The length of this
c                      array is defined on control card 10.
c                      This array is unique to interface nosl.
c
c          side       ='surfb' for first pass.  the surfb
c                        surface is the surface designated in the
c                        input
c                     ='surfa' for second pass after surfa and
c                      surfb surfaces have been switched as the
c                      tracked and reference surface for the type 3
c                      symmetric interface treatment.
c
c          iisv5      =an array giving the pointers to the active nodes
c                      in the arrays.
c
c          niisv5     =number of active nodes
c
c          fdat       =contact history data array
c          iseg       =contact reference segment from previous step.
c          fxis       =tracked node force component in global x dir.
c                      to be updated to include friction
c          fyis       =tracked node force component in global y dir.
c                      to be updated to include friction
c          fzis       =tracked node force component in global z dir.
```

```
c                       to be updated to include friction
c          ss(ii)      =s contact point (-1 to 1) in parametric coordinates
c                       for tracked node ii.
c          tt(ii)      =t contact point (-1 to 1) in parametric coordinates
c                       for tracked node ii.
c          ilbsv(ii)   =pointer for node ii into global arrays.
c          stfk(ii)    =penalty stiffness for tracked node ii which was used
c                       to compute normal interface force.
c          frc(1,lsv(ii))
c                     =Coulomb friction scale factor for segment lsv(ii)
c          frc(2,lsv(ii))
c                     =viscous friction scale factor for segment lsv(ii)
c
c***********************************************************************
c       parameters for a coupled thermal-mechanical contact
c
c          numnp      = number of nodal points in the model
c          npc        = load curve pointer
c          pld        = load curve (x,y) data
c          lcfst(nosl)= load curve number for static coefficient of
c                       friction versus temperture for contact
c                       surface nosl
c          lcfdt(nosl)= load curve number for dynamic coefficient of
c                       friction versus temperture for contact
c                       surface nosl
c          temp(j)    = temperature for node point j
c          temp_bot(j)= temparature for thick thermal shell bottom
c                       surface
c          temp_top(j)= temparature for thick thermal shell top
c                       surface
c          numsh12    = number of thick thermal shells
c          itopaz(1)  = 999 ==> thermal-mechanical analysis
c          isurface   = thick thermal shell surface pointer
c
c***********************************************************************
```

APPENDIX H: User-Defined Thermal Material Model

The addition of a thermal user material routine is fairly straightforward. The keyword *MAT_THERMAL_USER_DEFINED, described in Volume II, controls the thermal user material.

The thermal user material can be used alone or in conjunction with any given mechanical material model in a coupled thermal-mechanical solution. A heat source can be included and the specific heat updated so that it is possible to model, for instance, phase transformations including melt energy.

If, for the same part (shell or solid elements), both a thermal and mechanical user material model are defined, the two user material models have (optionally) read access to each other's history variables. If the integration points of the thermal and mechanical elements are not coincident, interpolation or extrapolation is used when reading history variables. Linear interpolation or extrapolation using history data from the two closest integration points is used in all cases except when reading history variables for thick thermal shells (THSHEL = 1 on *CONTROL_SHELL). For the thermal shell, the shape functions of the element are used for the interpolation or extrapolation.

The thermal user materials are thermal material types 11 through 15. File dyn21tumat.F contains these thermal user material subroutines as subroutines `thumat11`, …, and `thumat15`. Subroutine `thusrmat` calls these thermal user material subroutines. File dyn21tumat.F also includes the source code of subroutine `thusrmat`. The comments of subroutines `thusrmat`, `thumat12`, and `umat46` contain additional useful information. Subroutine `umat46` resides in source file dyn21umats.F.

Thermal history variables

Setting NVH on *MAT_THERMAL_USER_DEFINED greater than 0 enables using thermal history variables. Thermal history variables are output to the tprint file; see *DATABASE_TPRINT.

Interchange of history variables with mechanical user material

In a coupled thermo-mechanical solution, a corresponding thermal element exists for each mechanical shell, thick shell, or solid element. A pair consisting of a mechanical and a corresponding thermal element both have integration points and possibly history variables. The mechanical and thermal elements do not necessarily have the same number of integration points. By setting IHVE to 1 on *MAT_THERMAL_USER_DEFINED, a

thermal user material model can read, but not write, the history variables from a mechanical user material model and vice versa.

If the locations of the points where the history variables are located differ between the mechanical and thermal elements, interpolation or extrapolation is used to calculate the history value. More information is available in the comments to the subroutines `thusrmat` and `thumat12`.

Limitations:

The thermal user material implementation has a few limitations. LS-DYNA will, in most cases, give an appropriate warning or error message when such a limit is violated. The limitations include:

1. Option IHVE = 1 is only supported for a limited range of mechanical elements:
   a) Solid elements: ELFORM = 1, 2, 10, 13.
   b) Shell elements: ELFORM = 2, 3, 4, 14, 15, 16. Note that user-defined integration rules are not supported.
2. Thermal history variables limitations:
   a) Thermal history variables are not output to d3plot.
3. The thermal solver includes not only the plastically dissipated energy as a heat source but also wrongly the elastic energy. The latter, however, is, in most cases, not of practical importance.

Example source code:

Example source code for thermal user material models is available in `thumat11` and `thumat12` as well as in `umat46`. Note that there is space for up to 64 material parameters in `r_matp` (the material parameter array), but only 32 can be read from the *MAT_THERMAL_USER_DEFINED card. The material parameters in `r_matp(i)` with $i = 41 - 64$, which are initially set to 0.0, may be used to store additional data.

Subroutines `curveval` and `curveval_v` evaluate load curves. Note that when using `curveval` and `curveval_v`, the load curves are first re-interpolated to 100 equidistant points. See the Load curves and tables subsection of Appendix A for more information on these subroutines.

Here, we provide example keyword input for a thermal user material. This input enables orthotropic conduation. The sample input has SI units.

```
*KEYWORD
   ⋮
```

```
*MAT_THERMAL_USER_DEFINED
$      MID        RO        MT       LMC       NVH      AOPT    IORTHO      IVHE
        21    7800.0      12.0       6.0       1.0       0.0       1.0       0.0
$       XP        YP        ZP        A1        A2        A3
       0.0       0.0       0.0       0.0       0.0       0.0
$       D1        D2        D3
       0.0       0.0       0.0
$  Materials parameter card.  C1 - C3 are the heat conductions in the 11, 22, and
$  33 directions of the material coordinate system.  HC is the heat capacity.
$  HSRC is a load curve giving heat source output as a function of time.  HCFAC
$  is the ID of a load curve giving a scale factor for the heat capacity as a
$  function of time.
$       C1        C2        C3        HC      HSRC     HCFAC
      25.0      25.0      20.0     470.0      11.0      12.0
   ⋮
*END
```

The source code for the user material is:

```
      subroutine thumat12(c1,c2,c3,cvl,dcvdtl,hsrcl,dhsrcdtl,
     1     hsv,hsvm,nmecon,r_matp,crv,
     2     nel,nep,iep,eltype,dt,atime,ihsrcl)
      character*(*) eltype
      dimension hsv(*),hsvm(*),r_matp(*),crv(101,2,*)
      include 'iounits.inc'
c
c     Thermal user-material number 12.
c
c     See comments at the beginning of subroutine thusrmat
c     for instructions.
c
c     Example: isotropic/orthotropic material with k1=P1 and
c     cvl=P2 for solid and shell elements including optional
c     change of heat capacity and a heat source, both functions
c     of time input as load curves.
c
c     Print out some info on start-up, use material parameter 64
c     as a flag.
      if(nint(r_matp(64)).eq.0) then
         r_matp(64)=1.
         write( *,1200)  (r_matp(8+i),i=1,6)
         write(iohsp,1200) (r_matp(8+i),i=1,6)
         write(59,1200) (r_matp(8+i),i=1,6)
      endif
c
c     Calculate response
      c1=r_matp(8+1)
      c2=r_matp(8+2)
      c3=r_matp(8+3)
      cvl=r_matp(8+4)
      dcvdtl=0.0
      eid=nint(r_matp(8+6))
      if(nint(eid).gt.0) then
         call curveval(eid,atime,cvlfac,tmp1)
         cvl=cvl*cvlfac
         dcvdtl=0.0
      endif
c
c     If flux or time step calculation then we are done.
      if(eltype.eq.'soliddt'.or.eltype.eq.'flux'.or.
     .     eltype.eq.'shelldt') return
      eid=nint(r_matp(8+5))
      if(nint(eid).gt.0) then
         ihsrcl=1
         call curveval(eid,atime,hsrcl,tmp1)
         dhsrcdtl=0.0
```

```
      endif
c
c     Update history variables
      hsv(1)=cvl
      hsv(2)=atime
      hsv(3)=hsv(3)+1.0
c
c     Done
      return
 1200 format(/'This is thermal user defined material #12. '/
     1        /' Material parameter c1-c3       : ',3E10.3
     2        /' Material parameter hc          : ',E10.3
     3        /' Heat source load curve         : ',F10.0
     4        /' hc scale factor load curve     : ',F10.0
     5        /' Thermal History variable 1     : cv'
     6        /' Thermal History variable 2-3   : Dummy'/)
      return
      end
```

APPENDIX I: Occupant Simulation Including the Coupling to the CAL3D and MADYMO programs

INTRODUCTION

LS-DYNA is coupled to occupant simulation codes to generate solutions in automotive crashworthiness that include occupants interacting with the automotive structure. In such applications LS-DYNA provides the simulation of the structural and deformable aspects of the model and the OSP (Occupant Simulation Program) simulates the motion of the occupant. There is some overlap between the two programs which provides flexibility in the modeling approach. For example, both the OSP and LS-DYNA have the capability of modeling seat belts and other deformable restraints. The advantage of using the OSP is related to the considerable databases and expertise that have been developed in the past for simulating dummy behavior using these programs.

The development of the interface provided us a number of possible approaches. The approach selected is consistent with the our philosophy of providing the most flexible and useful interface possible. This is important because the field of non-linear mechanics is evolving rapidly and techniques which are used today are frequently rendered obsolete by improved methodologies and lower cost computing which allows more rigorous techniques to be used. This does make the learning somewhat more difficult as there is not any single procedure for performing a coupling.

One characteristic of LS-DYNA is the large number of capabilities, particularly those associated with rigid bodies. This creates both an opportunity and a difficulty: LSDYNA3D has many ways approximating different aspects of problems, but they are frequently not obvious to users without considerable experience. Therefore, in this Appendix we emphasize modeling methods rather than simply listing capabilities.

THE LS-DYNA/OCCUPANT SIMULATION PROGRAM LINK

Coupling between the OSP and LS-DYNA is performed by combining the programs into a single executable. In the case of CAL3D, LS-DYNA calls CAL3D as a subroutine, but in the case of MADYMO, LS-DYNA is called as a subroutine. The two programs are then integrated in parallel with the results being passed between the two until a user defined termination time is reached.

The OSP and LS-DYNA have different approaches to the time integration schemes. The OSP time integrators are based on accurate implicit integrators which are valid for large time steps which are on the order of a millisecond for the particular applications of

interest here. An iterative solution is used to insure that the problem remains in equilibrium. The implicit integrators are extremely good for smoothly varying loads, however, sharp nonlinear pulses can introduce considerable error. An automatic time step size control which decreases the time step size quickly restores the accuracy for such events. The LS-DYNA time integrator is based on an explicit central difference scheme. Stability requires that the time step size be less than the highest frequency in the system. For a coarse airbag mesh, this number is on the order of 100 microseconds while an actual car crash simulation is on the order of 1 microsecond. The smallest LS-DYNA models have at least 1,000 elements. Experience indicates that the cost of a single LS-DYNA time step for a small model is at least as great as the cost of a time step in the OSP. Therefore, in the coupling, the LS-DYNA time step is used to control the entire simulation including the OSP part. This approach has negligible cost penalties and avoids questions of stability and accuracy that would result by using a subcycling scheme between the two programs.

LS-DYNA has a highly developed rigid body capability which is used in different parts of automobile crash simulation. In particular, components such as the engine are routinely modeled with rigid bodies. These rigid bodies have been modified so that they form the basis of the coupling procedure in LS-DYNA to the OSP.

Please contact the technical support team (https://support.ansys.com) for instructions to download and run LS-DYNA executables coupled with Madymo.

AIRBAG MODELING

Modeling of airbags is accomplished by use of shell or membrane elements in conjunction with a control volume (see *AIRBAG_*OPTION*) and possibly a single surface contact algorithm to eliminate interpenetrations during the inflation phase (see *CONTACT_*OPTION*). The contact types showing an "a" in front are most suited for airbag analysis. Current recommended material types for the airbags are:

1. *MAT_ELASTIC (Type 1, Elastic)
2. *MAT_COMPOSITE_DAMAGE (Type 22, layered orthotropic elastic for composites)
3. *MAT_FABRIC (Type 34, fabric model for folded airbags)

Model 34 is a "fabric" model which can be used for flat bags. As a user option this model may or may not support compression.

The elements which can be used are as follows:

1. Belytschko-Tsay quadrilateral with 1 point quadrature. This element behaves rather well for folded and unfolded cases with only a small tendency to

hourglass. The element tends to be a little stiff. Stiffness form hourglass control is recommended.

2. Belytschko-Tsay membrane. This model is softer than the normal Belytschko-Tsay element and can hourglass quite badly. Stiffness form hourglass is recommended. As a better option, the fully integrated Belytschko-Tsay membrane element can be chosen.

3. C0 Triangular element. The C0 triangle is very good for flat bag inflation and has no tendency to hourglass.

4. The best choice is a specially developed airbag membrane element with quadrilateral shape. This is an automatic choice when the fabric material is used.

As an airbag inflates, a considerable amount of energy is transferred to the surrounding air. This energy transfer decreases the kinetic energy of the bag as it inflates. In the control volume logic, this is simulated either by using either a mass weighted damping option or a back pressure on the bag based on a stagnation pressure. In both cases, the energy that is absorbed is a function of the fabric velocity relative to a rigid body velocity for the bag. For the mass weighted case, the damping force on a node is proportional to the mass times the damping factor times the velocity vector. This is quite effective in maintaining a stable system, but has little physical justification. The latter approach using the stagnation pressure method estimates the pressure needed to accelerate the surrounding air to the speed of the fabric. The formula for this is:

$$P = \text{Area} \times \alpha \times \left[\left(\vec{V}_i - \vec{V}_{cg}\right) \cdot \hat{n}\right]^2$$

This formula accomplishes a similar function and has a physical justification. Values of the damping factor, α, are limited to the range of 0 to 1, but a value of 0.1 or less is more likely to be a good value.

COMMON ERRORS

1. Improper airbag inflation or no inflation.

The most common problem is inconsistency in the units used for the input constants. An inflation load curve must also be specified. The normals for the airbag segments must all be consistent and facing outwards. If a negative volume results, this can sometimes be quickly cured by using the "flip" flag on the control volume definition to force inward facing normals to face outwards.

2. Excessive airbag distortions.

Check the material constants. Triangular elements should have less distortion problems than quadrilaterals. Overlapped elements at time zero can cause locking to occur in the contact leading to excessive distortions. The considerable

energy input to the bag will create numerical noise and some damping is recommended to avoid problems.

3. The dummy passes through the airbag.

A most likely problem is that the contacts are improperly defined. Another possibility is that the models were developed in an incompatible unit system. The extra check for penetration flag if set to 1 on the contact control cards variable PENCHK in the *CONTACT_... definitions may sometimes cause nodes to be prematurely released due to the softness of the penalties. In this case the flag should be turned off.

4. The OSP fails to converge.

This may occur when excessively large forces are passed to the OSP. First, check that unit systems are consistent and then look for improperly defined contacts in the LS-DYNA input.

5. Time step approaches zero.

This is almost always in the airbag. If elastic or orthotropic (*MAT_ELASTIC or *MAT_COMPOSITE material 1 or 22) is being used, then switch to fabric material *MAT_FABRIC which is less time step size sensitive and use the fully integrated membrane element. Increasing the damping in the control volume usually helps considerably. Also, check for “cuts” in the airbag where nodes are not merged. These can allow elements to deform freely and cut the time step to zero.

APPENDIX J: Interactive Graphics Commands

NOTE: The former Appendix J has been removed due to obsolescence.

APPENDIX K: Interactive Material Model Driver

INTRODUCTION

The interactive material model driver in LS-DYNA allows calculation of the material constitutive response to a specified strain path. Since the constitutive model subroutines in LS-DYNA are directly called by this driver, the behavior of the constitutive model is precisely that which can be expected in actual applications. In the current implementation the constitutive subroutines for both shell elements and solid elements can be examined.

INPUT DEFINITION

The material model driver is invoked when no *NODE or *ELEMENT commands are present in a standard LS-DYNA input file. The number of material model definitions should be set to one, the number of load curves should be nine, and the termination time to the desired length of the driver run. The complete state dump interval as given in *DATABASE_BINARY_D3PLOT serves as the time step to be used in the material model driver run. Plotting information is saved in core for the interactive plotting phase.

The input deck typically consists only of *KEYWORD, *DATABASE_BINARY_D3PLOT, *CONTROL_TERMINATION, one each of *PART/*MAT/*SECTION, and nine load curves (*DEFINE_CURVE) describing the strain path. These nine curves define the time history of the displacement gradient components shown in Table 61-1.

The velocity gradient matrix, L_{ij}, is approximated by taking the time derivative of the components in Table 61-1. If these components are considered to form a tensor S_{ij} , then

$$L_{ij}(t) = \frac{S_{ij}(t) - S_{ij}(t_{k-1})}{(t - t_k)}$$

and the strain rate tensor is defined as

$$d_{ij} = \frac{L_{ij} + L_{ij}^t}{2}$$

and the spin tensor as

$$\omega_{ij} = \frac{L_{ij} - L_{ij}^t}{2}$$

Load Curve Number	Component Definition
1	$\frac{\partial u}{\partial x}$
2	$\frac{\partial v}{\partial y}$
3	$\frac{\partial w}{\partial z}$
4	$\frac{\partial u}{\partial y}$
5	$\frac{\partial v}{\partial x}$
6	$\frac{\partial u}{\partial z}$
7	$\frac{\partial w}{\partial x}$
8	$\frac{\partial v}{\partial z}$
9	$\frac{\partial w}{\partial y}$

Table 61-1. Load Curve Definitions versus Time

INTERACTIVE DRIVER COMMANDS

After reading the input file and completing the calculations, LS-DYNA gives a command prompt to the terminal. A summary of the available interactive commands is given below. An on-line help package is available by typing HELP. Only available in Unix and Linux.

ACCL	Scale all abscissa data by f. Default is f = 1.
ASET amin omax	Set min and max values on abscissa to amin and amax, respectively. If amin = amax = 0, scaling is automatic.
CHGL n	Change label for component n. LS-DYNA prompts for new label.

CONTINUE	Re-analyze material model.
CROSS c_1 c_2	Plot component c_1 versus c_2.
ECOMP	Display component numbers on the graphics display: EQ.1: x-stress, EQ.2: y-stress, EQ.3: z-stress, EQ.4: xy-stress, EQ.5: yz-stress, EQ.6: zx-stress, EQ.7: effective plastic strain, EQ.8: pressure, EQ.9: von Mises (effective) stress, EQ.10: 1st principal deviatoric stress, EQ.11: 2nd principal deviatoric stress, EQ.12: 3rd principal deviatoric stress, EQ.13: maximum shear stress, EQ.14: 1st principal stress, EQ.15: 2nd principal stress, EQ.16: 3rd principal stress, EQ.17: ln (v/v0), EQ.18: relative volume, EQ.19: v0/v - 1.0, EQ.20: 1st history variable, EQ.21: 2nd history variable. Adding 100 or 400 to component numbers 1-16 yields strains and strain rates, respectively.
FILE name	Change pampers filename to name for printing.
GRID	Graphics displays will be overlaid by a grid of orthogonal lines.
NOGRID	Graphics displays will not be overlaid by a grid of orthogonal lines.
OSCL	Scale all ordinate data by f. Default is f = 1.
OSET omin omax	Set min and max values on ordinate to omin and omax, respectively. If omin = omax = 0, scaling is automatic.

PRINT	Print plotted time history data into file "pampers." Only data plotted after this command is printed. File name can be changed with the "file" command.
QUIT, END, T	Exit the material model driver program.
RDLC m n r_1 z_1 ... r_n z_n	Redefine load curve m using n coordinate pairs (r_1,z_1) (r_2,z_2),...(r_n,z_n).
TIME c	Plot component c versus time.
TV n	Use terminal output device type n. LS-DYNA provides a list of available devices.

Presently, the material model drive is implemented for solid and shell element material models. The driver does not yet support material models for beam elements.

Use the keyword *CONTROL_MPP_MATERIAL_MODEL_DRIVER and run the input deck only on one processor if a distributed memory executable (MPP) is used.

Beginning with Release R11, it is possible to use the material model driver in batch mode. That means if a file called "mmd.bat" exists in the working directory, then the commands contained therein get executed sequentially without opening an interactive command prompt. Supported commands are print, cross, time, and quit. Example for contents of "mmd.bat" to get effective stress versus plastic strain in a file called "sigeps" and effective stress versus time in "sigtim" is given here (read format '(a8,2x,2e10.0)'):

```
print
sigeps
cross               9          7
print
sigtim
time                9
quit
```

APPENDIX L: VDA Database

VDA surfaces describe the surface of geometric entities and are useful for the simulation of sheet forming problems. The German automobile and automotive supplier industry (VDA) has defined the VDA guidelines [VDA 1987] for a proper surface definition used for the exchange of surface data information. In LS-DYNA, this format can be read and used directly. Some files have to be provided for proper linkage to the motion of the correlation parts/materials in LS-DYNA.

Linking is performed via names. To these names surfaces are attached, which in turn can be linked together from many files externally to LS-DYNA. Thus, arbitrary surfaces can be provided by a preprocessor and then can be written to various files. The so-called VDA file given on the LS-DYNA execution line via V = **vda** contains references to all other files. It also contains several other parameters affecting the treatment in the contact subroutines; see below.

The procedure is as follows. If VDA surfaces are to be used, the file specified by **vda** must have the following form. The file is free formatted with blanks as delimiters. Note that the characters "}" and "{" must be separated from the other input by spaces or new lines. The **vda** file may contain any number of input file specifications of the form:

file **afile bfile** {
    alias definitions
}
alias definitions
followed by optional runtime parameters and a final end statement.

The file, **afile,** is optional, and if given must be the name of an ASCII input file formatted in accordance with the VDA Surface Interface Definitions as defined by the German automobile and automotive supply industry. **bfile** is required, and is the name of a binary VDA file. In a first run **afile** is given and **bfile** is created. In any further run, if the definitions have not changed, **afile** can be dropped and only **bfile** is needed. The purpose of **bfile** is that it allows for much faster initialization if the same VDA surfaces are to be used in a future LS-DYNA run.

If **afile** is given, **bfile** will always be created or overwritten. The alias definitions are used for linking to LS-DYNA and between the various surface definitions in the files defined by **afile** and **bfile.**

The alias definitions are of the form

alias **name** { el1 el2 ... eln }

where **name** is any string of up to 12 characters, and el1,...,eln are the names of VDA elements as specified in **afile**. The list of elements can be empty, in which case all the SURF and FACE VDA elements in **afile** will be used. Care should be taken to ensure that the alias **name** is unique, not only among the other aliases, but among the VDA element

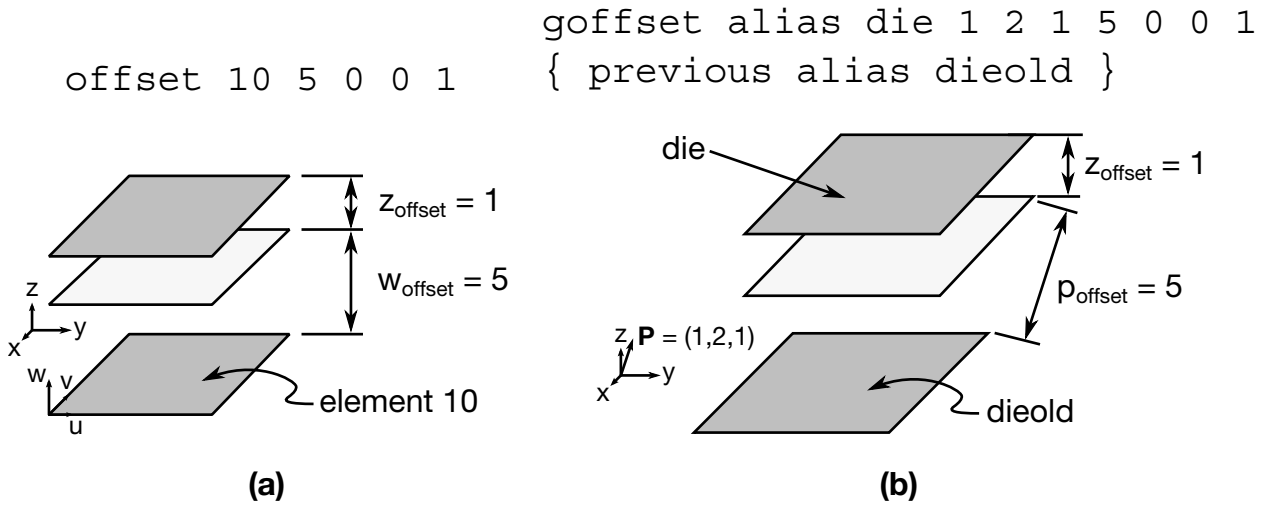


Figure 62-1. (a) a schematic illustration of offset version 1, and (b) is a schematic illustration of offset version 2.

names in **afile**. This collection of VDA elements can later be indicated by the alias **name**. In particular, **name** may appear in later alias definitions.

Often it is required that a punch or die be created by a simple offset. This can be achieved in the **vda** files in two ways, either on VDA elements directly, or on parts defined by aliases. This feature offers great capability in generating and using surface data information.

Offset Version 1

As an option, the keyword **offset** may appear in the alias list which allows a new surface to be created as a normal offset (plus translation) of a VDA element in the file. The keyword **offset** my be applied to VDA elements only, not aliases. The usage of **offset** follows the form

offset elem **normal x y z**

where **normal** is the amount to offset the surface along the normal direction, and **x**,**y**,**z** are the translations to be applied. The default normal direction is given by the cross product of the local u and v directions on the VDA surface, taken in that order. **normal** can be negative.

Offset Version 2

Frequently, it is convenient to create a new alias **name** by offsetting and translating an existing **name**. The keyword **goffset** provides this function:

goffset **alias name xc yc zc normal x y z** { previous **alias name** }

where **normal**, **x**, **y**, and **z** are defined as in the offset keyword. A reference point $\mathbf{x_c}$, $\mathbf{y_c}$, and $\mathbf{z_c}$ defines a point in space which determines the normal direction to the VDA surface, which is a vector from the origin to $P(x_c,y_c,z_c)$. See example below.

Finally, several parameters affecting the VDA surface iteration routines can be reset in the file **vda**. These parameters, and their default values in square brackets [ ], are:

gap [5.0] The maximum allowable surface gap to be filled in during the iterations. Points following the surface will effectively extend the edges of surfaces if necessary to keep them from falling through cracks in the surface smaller than this. This number should be set as small as possible while still allowing correct results. In particular, if your VDA surfaces are well formed (having no gaps), this parameter can be set to 0.0. The default value is 5.0.

t**rack** [2.0] A point must be within this distance of contact to be continually tracked. When a point not being tracked comes close to a surface, a global search is performed to find the near surface point. While a point is being tracked, iterations are performed every cycle. These iterations are much faster, but if the point is far away it is faster to occasionally do the global search. The default value is 2.0.

t**rack2** [5.0] Every VDA surface is surrounded by a bounding box. When a global search needs to be performed but the distance from a point to this box is > t**rack2**, the actual global search is not performed. This will require another global search to be performed sooner than if the actual distance to the surface were known, but also allows many global searches to be skipped. The default value is 5.0.

ntrack [4] The number of VDA surfaces for which each point maintains actual distance information. A global lower bound on distance is maintained for all remaining surfaces. Whenever the point moves far enough to violate this global lower bound, all VDA surfaces must have the global search performed for them. Hence, this parameter should be set to the maximum number of surfaces that any point can be expected to be near at one time (the largest number of surfaces that come together at one point). Setting **ntrack** higher will require more memory but result in faster execution. If **ntrack** is too low, performance may be unacceptably slow. The default value is 4.0.

toroid [.01] Any surface with opposing edges which are within distance [t] of each other is assumed to be cylindrical. Contacts occurring on one edge can pass to the adjacent edge. The default value is 0.01.

converge [.01] When surface iterations are performed to locate the near point, iteration is continued until convergence is detected to within this distance (all VDA coordinates are in mm). The default value is 0.01.

iterate [8] Maximum number of surface iterations allowed. Since points being tracked are checked every cycle, if convergence fails it will be tried again next cycle, so setting this parameter high does not necessarily help much. On the other hand, a point converging to a crease in the VDA surface (a crease between patches with discontinuous derivative, for example) may bounce back and forth between patches up to this many times, without actually moving. Hence, this value should not be too large. The default value is 8.

el_size [t mx mn]

Controls the generation of elements where:

t = surface tolerance for mesh generation,
mx = maximum element size to generate,
mn = minimum element size to generate.

The default values are [0.25 100. 1.0]

aspect [s1 s2] Controls the generation of elements where:

s1 = maximum difference in aspect ratio between elements generated in neighboring VDA patches,
s2 = maximum aspect ratio for any generated element.

The default values are [1.5 4.0]

cp_space [10] Determines the spacing around the boundaries of parts at which the size of elements is controlled. In the interior of the part, the element size is a weighted function of these control points as well as additional control points in the interior of the region. If there are too few control points around the boundary, elements generated along or near straight boundaries, but between control points, may be too small. The default value is 10.

meshonly The existence of this keyword causes LS-DYNA to generate a file containing the mesh for the VDA surfaces and then terminate.

onepatch The existence of this keyword causes LS-DYNA to generate a single element on each VDA patch.

somepatch [n] Like onepatch, but generates an element for 1 out of every [n] patches.

Example for file V = **vda**. It contains the following data:

```
file vda1 vda1.bin {
        alias die {
                sur0001
                sur0003
                offset fce0006 1.5 0 0 120
        }
        alias holder1 { sur008 }
}
file vda2 vda2.bin {
        alias holder2 { sur003 }
}
alias holder { holder1 holder2 }
ntrack 6
gap 0.5

end
```

Explanation:

vda1	This file contains the surfaces/face elements sur0001,sur0003, fce0006, and sur0008.
alias die face	Combines the surface/face elements sur0001, sur0003, and the offsetted fce0006 to a global surface.
alias holder1	Defines the surface/face element sur0008 as holder1.
vda2	This file contains the surface/face element sur0003.
alias holder2	Defines the surface/face element sur0003 as holder2.
alias holder	Combines the surfaces holder1 and holder2 into a combined surface holder.
ntrack 6	For each point the actual distances to 6 VDA surfaces are maintained.
gap 0.5	Surface gaps of 0.5mm or less are filled.
end	Closes reading of this file.

APPENDIX M: Commands for Two-Dimensional Rezoning

NOTE: The former Appendix M has been removed due to obsolescence.

APPENDIX N:
Rigid-Body Dummies

The two varieties of rigid body dummies available in LS-DYNA are described in this appendix. These are generated internally by including the appropriate *COMPONENT keyword. A description of the GEBOD dummies begins on this page and the HYBRID III family on page N.7.

GEBOD Dummies

Rigid body dummies can be generated and simulated within LS-DYNA using the keyword *COMPONENT_GEBOD. Physical properties of these dummies draw upon the GEBOD database [Cheng et al. 1994] which represents an extensive measurement program conducted by Wright-Patterson AFB and other agencies. The differential equations governing motion of the dummy are integrated within LS-DYNA separate from the finite element model. Interaction between the dummy and finite element structure is achieved using contact interfaces (see *CONTACT_GEBOD).

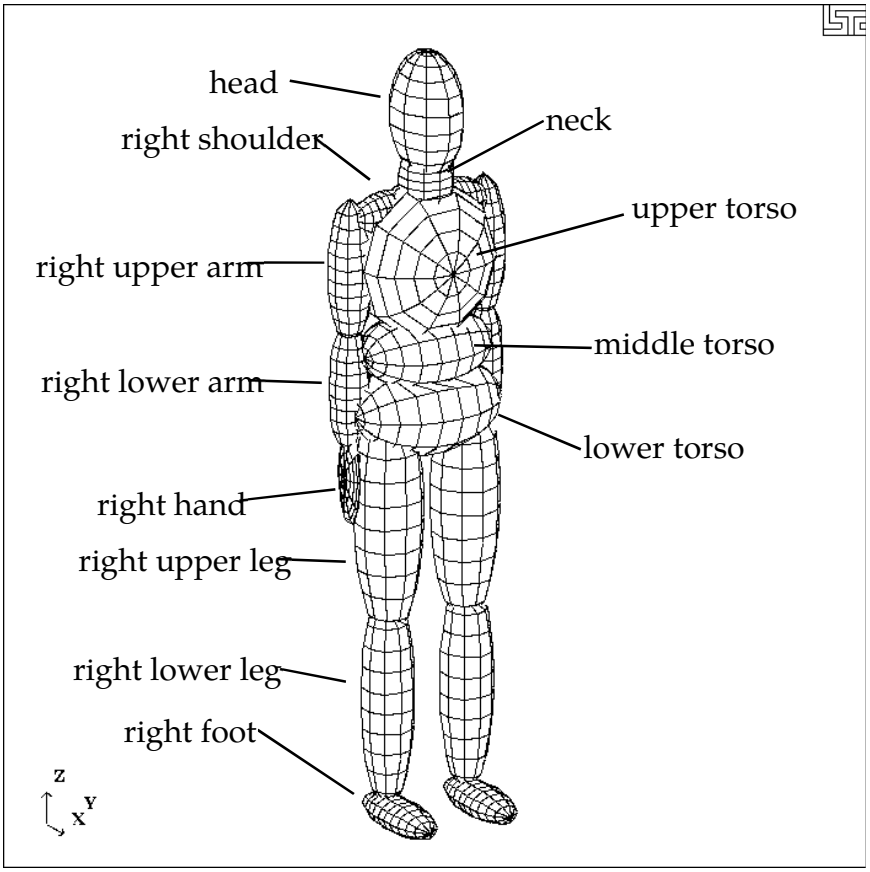


Figure 64-1. 50th percentile male dummy in the nominal position.

APPENDIX N

The dynamical system representing a dummy is comprised of fifteen rigid bodies (segments) and include: lower torso, middle torso, upper torso, neck, head, upper arms, forearms/hands, upper legs, lower legs, and feet. Ellipsoids are used for visualization and contact purposes. Shown in Figure 64-1 is a 50th percentile male dummy generated using the keyword command *COMPONENT_GEBOD_MALE. Note that the ellipsoids representing the shoulders are considered to be part of the upper torso segment and the hands are rigidly attached to the forearms.

Each of the rigid segments which make up the dummy is connected to its neighbor with a joint which permits various relative motions of the segments. Listed in the Table 64-2 are the joints and their applicable degrees of freedom.

Joint Name	Degree(s) of Freedom		
	1st	2nd	3rd
pelvis	lateral flexion (x)	forward flexion (y)	torsion (z)
waist	lateral flexion (x)	forward flexion (y)	torsion (z)
lower neck	lateral flexion (x)	forward flexion (y)	torsion (z)
upper neck	lateral flexion (x)	forward flexion (y)	torsion (z)
shoulders	abduction-adduction (x)	internal-external rotation (z)	flexion-extension (y)
elbows	flexion-extension (y)	n/a	n/a
hips	abduction-adduction (x)	medial-lateral rotation (z)	flexion-extension (y)
knees	flexion-extension (y)	n/a	n/a
ankles	inversion-eversion (x)	dorsi-plantar flexion (y)	medial-lateral rotation (z)

Table 64-2. Joints and associated degrees of freedom. Local axes are in parentheses.

Orientation of a segment is effected by performing successive right-handed rotations of that segment relative to its parent segment - each rotation corresponds to a joint degree of freedom. These rotations are performed about the local segment axes and the sequence is given in Table 64-2. For example, the left upper leg is connected to the lower torso by the left hip joint; the limb is first abducted relative to lower torso, it then undergoes lateral rotation, followed by extension. The remainder of the lower extremity (lower leg and foot) moves with the upper leg during this orientation process.

By default all joints are assigned stiffnesses, viscous characteristics, and stop angles which should give reasonable results in a crash simulation. One or all default values of a joint may be altered by applying the *COMPONENT_GEBOD_JOINT_*OPTION* command to the joint of interest. The default shape of the resistive torque load curve used by all joints is shown in Figure 64-6. A scale factor is applied to this curve to obtain the proper stiffness relationship. Listed in Table 64-3 are the default values of joint characteristics for dummies of all types and sizes. These values are given in the English system of units;

the appropriate units are used if a different system is specified in card 1 of *COMPONENT_GEBOD_*OPTION*.

joint degrees of freedom	load curve scale factor (in·lbf)	damping coef. (in·lbf·s/rad)	low stop angle (degrees)	high stop angle (degrees)	neutral angle (degrees)
pelvis - 1	65000	5.77	-20	20	0
pelvis - 2	65000	5.77	-20	20	0
pelvis - 3	65000	5.77	-5	5	0
waist - 1	65000	5.77	-20	20	0
waist - 2	65000	5.77	-20	20	0
waist - 3	65000	5.77	-35	35	0
lower neck - 1	10000	5.77	-25	25	0
lower neck - 2	10000	5.77	-25	25	0
lower neck - 3	10000	5.77	-35	35	0
upper neck - 1	10000	5.77	-25	25	0
upper neck - 2	10000	5.77	-25	25	0
upper neck - 3	10000	5.77	-35	35	0
l. shoulder - 1	100	5.77	-30	175	0
r. shoulder - 1	100	5.77	-175	30	0
shoulder - 2	100	5.77	-65	65	0
shoulder - 3	100	5.77	-175	60	0
elbow - 1	100	5.77	1	-140	0
l. hip - 1	10000	5.77	-25	70	0
r. hip - 1	10000	5.77	-70	25	0
hip - 2	10000	5.77	-70	70	0
hip - 3	10000	5.77	-140	40	0
knee - 1	100	5.77	-1	120	0
l. ankle - 1	100	5.77	-30	20	0
l. ankle - 1	100	5.77	-20	30	0
ankle - 2	100	5.77	-20	45	0
ankle - 3	100	5.77	-30	30	0

Table 64-3. Default joint characteristics for all dummies.

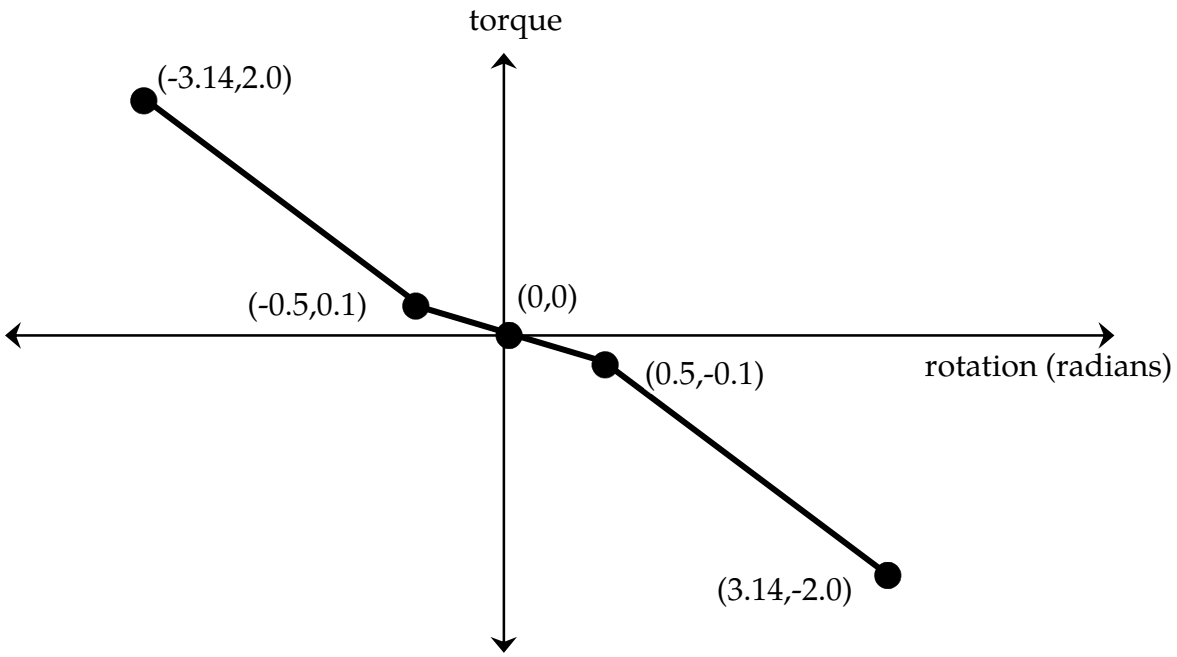


Figure 64-4 Characteristic torque curve shape used by all joints.

The dummy depicted in Figure 64-1 appears in what is referred to as its "nominal" position. In this position the dummy is standing upright facing in the positive x direction and the toe-to-head direction points in positive z. Additionally, the dummy's hands are at the sides with palms facing inward and the centroid of the lower torso is positioned at the origin of the global coordinate system. Each of the dummy's segments has a local coordinate system attached to it and in the nominal position all of the local axes are aligned with the global axes.

When performing a simulation involving a *COMPONENT_GEBOD dummy, a positioning file named "gebod.*did*" must reside in the directory with the LS-DYNA input file; here the extension *did* is the dummy ID number, see card 1 of *COMPONENT_GEBOD_*OPTION*. The contents of a typical positioning file is shown in Table 64-5; it consists of 40 lines formatted as (59a1,e30.0). All of the angular measures are input as degrees, while the lower torso global positions depend on the choice of units in card 1 of *COMPONENT_GEBOD_*OPTION*. Setting all of the values in this file to zero yields the so-called "nominal" position.

Table 64-5. Typical contents of a dummy positioning file.

lower torso	centroid global x position	0.0
lower torso	centroid global y position	0.0
lower torso	centroid global z position	0.0
total body	global x rotation	0.0
total body	global y rotation	-20.0
total body	global z rotation	180.0

pelvis	lateral flexion	+ = tilt right	0.0
pelvis	forward flexion	+ = lean fwd	0.0
pelvis	torsion	+ = twist left	0.0
waist	lateral flexion	+ = tilt right	0.0
waist	forward flexion	+ = lean fwd	0.0
waist	torsion	+ = twist left	0.0
lower neck	lateral flexion	+ = tilt right	0.0
lower neck	forward flexion	+ = nod fwd	0.0
lower neck	torsion	+ = twist left	0.0
upper neck	lateral lexion	+ = tilt right	0.0
upper neck	forward flexion	+ = nod fwd	0.0
upper neck	torsion	+ = twist left	0.0
left shoulder	abduction-adduction	+ = abduction	30.0
left shoulder	internal-external rotation	+ = external	-10.0
left shoulder	flexion-extension	- = fwd raise	-40.0
right shoulder	abduction-adduction	- = abduction	-30.0
right shoulder	internal-external rotation	- = external	10.0
right shoulder	flexion-extension	- = fwd raise	-40.0
left elbow	flexion-extension	+ = extension	-60.0
right elbow	flexion-extension	+ = extension	-60.0
left hip	abduction-adduction	+ = abduction	0.0
left hip	medial-lateral rotation	+ = lateral	0.0
left hip	flexion-extension	+ = extension	-80.0
right hip	abduction-adduction	- = abduction	0.0
right hip	medial-lateral rotation	- = lateral	0.0
right hip	flexion-extension	+ = extension	-80.0
left knee	flexion-extension	+ = flexion	50.0
right knee	flexion-extension	+ = flexion	50.0
left ankle	inversion-eversion	+ = eversion	0.0
left ankle	dorsi-plantar flexion	+ = plantar	0.0
left ankle	medial-lateral rotation	+ = lateral	0.0
right ankle	inversion-eversion	- = eversion	0.0
right ankle	dorsi-plantar flexion	+ = plantar	0.0
right ankle	medial-lateral rotation	- = lateral	0.0

In Figure 64-6 the 50th percentile male dummy is shown in a seated position and some of its joints are labeled. The file listed in Table 64-5 was used to put the dummy into the position shown. Note that the dummy was first brought into general orientation by setting nonzero values for two of the lower torso local rotations. This is accomplished by performing right-handed rotations successively about local axes fixed in the lower torso, the sequence of which follows: the first about local x, next about local y, and the last about local z. The dummy in Figure 64-6 was made to pitch backward by setting "total

body global y rotation" equal to -20. Setting the "total body global z rotation" equal to 180 caused the dummy to rotate about the global z-axis and face in the -x direction.

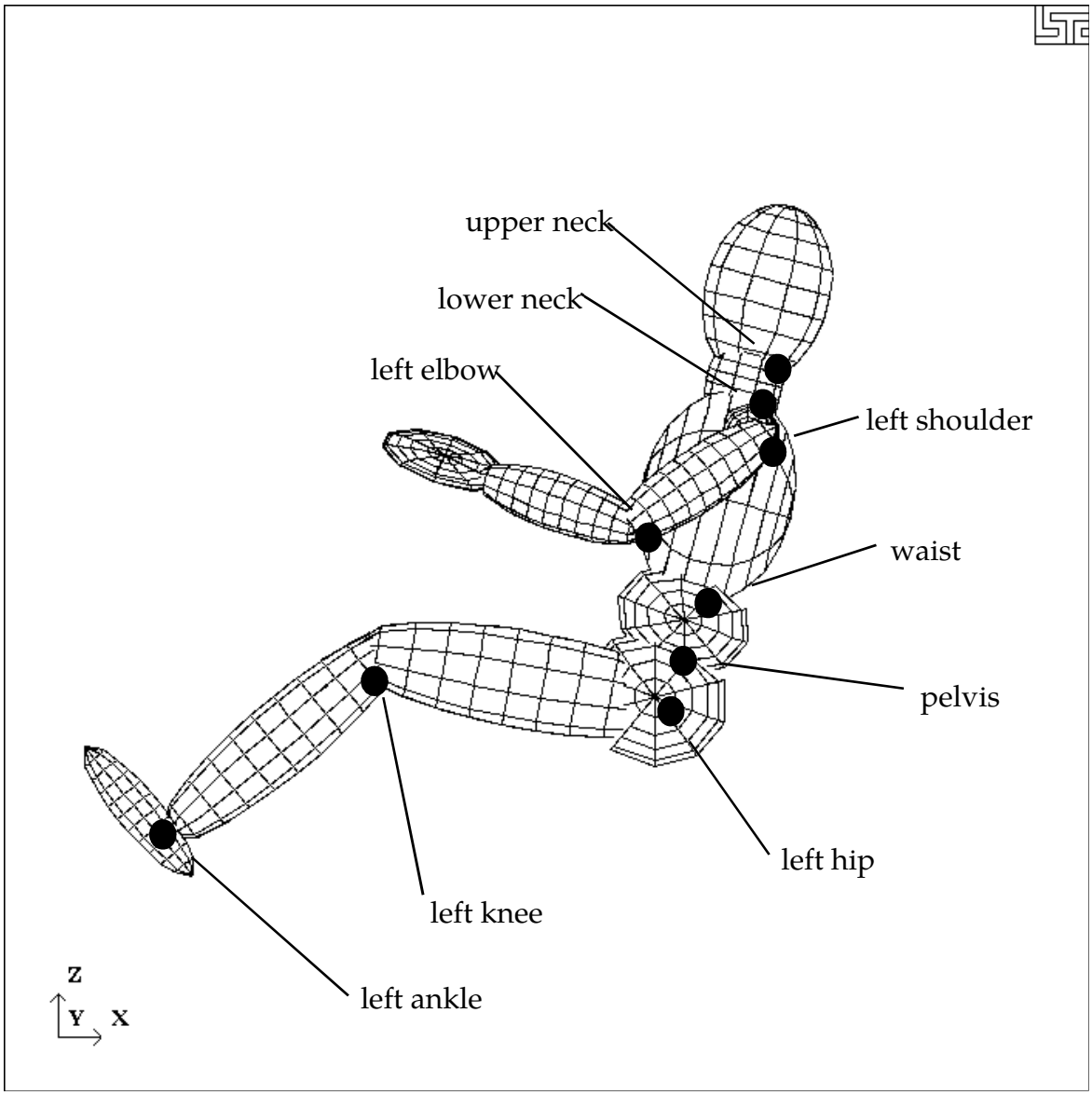


Figure 64-6. Dummy seated using the file listed in Table 64-5.

HYBRID III Dummies

A listing of applicable joint degrees of freedom of the Hybrid III dummy is given below.

Joint Name	Degree(s) of Freedom		
	1st	2nd	3rd
lumbar	flexion (y)	torsion (z)	
lower neck	flexion (y)	torsion (z)	
upper neck	flexion (y)	torsion (z)	
shoulders	flexion-extension (y)	abduction-adduction (x)	n/a
elbows	flexion-extension (y)	n/a	n/a
wrists	flexion-extension (x)	n/a	n/a
hips	abduction-adduction (x)	medial-lateral rotation (z)	flexion-extension (y)
knees	flexion-extension (y)	n/a	n/a
ankles	inversion-eversion (x)	medial-lateral rotation (z)	dorsi-plantar flexion (y)
sternum	translation (x)	rotation (y)	rotation (z)
knee sliders	translation (z)		

Table 64-7. Joints and associated degrees of freedom. Local axes are in parentheses.

Joint springs of the *COMPONENT_HYBRIDIII dummies are formulated in the following manner.

$$
\begin{aligned}
T &= a_{\mathrm{lo}}(q - q_{\mathrm{lo}}) + b_{\mathrm{lo}}(q - q_{\mathrm{lo}})^3, && q \le q_{\mathrm{lo}} \\
T &= a_{\mathrm{hi}}(q - q_{\mathrm{hi}}) + b_{\mathrm{hi}}(q - q_{\mathrm{hi}})^3, && q \ge q_{\mathrm{hi}} \\
T &= 0, && q_{\mathrm{lo}} < q < q_{\mathrm{hi}}
\end{aligned}
$$

Where,

T is the joint torque
q is the joint generalized coordinate
a_{lo} and b_{lo} are the linear and cubic coefficients, respectively, for the low regime
a_{hi} and b_{hi} are the linear and cubic coefficients, respectively, for the high regime
q_{lo} and q_{hi} are the activation values for the low and high regimes, respectively

APPENDIX O: LS-DYNA MPP User Guide

This is a short user's guide for the MPP version of LS-DYNA. For a general guide to the capabilities of LS-DYNA and a detailed description of the input, consult the LS-DYNA User's Manual. If you have questions about this guide, find errors or omissions in it, please email https://support.ansys.com.

Supported Features

Most LS-DYNA features are available for MPP. Those that are not supported are being systematically added. Unless otherwise noted here, all options of LS-DYNA are supported by MPP/LS-DYNA.

The following are *unsupported* features:

*BOUNDARY_THERMAL_WELD

*BOUNDARY_USA_SURFACE

*CONTACT_1D

*DATABASE_AVSFLT

*DATABASE_MOVIE

*DATABASE_MPGS

*LOAD_SUPERPLASTIC_OPTION

*TERMINATION_NODE

Contact Interfaces

MPP/LS-DYNA uses a completely redesigned, highly parallel contact algorithm. The contact options currently *unsupported* include:

*CONTACT_FORCE_TRANSDUCER_CONSTRAINT

Because these options are all supported via the new, parallel contact algorithms, slight differences in results may be observed as compared to the serial and SMP versions of LS-DYNA. Work has been done to minimize these differences, but they may still be evident in some models.

APPENDIX O

For each of the supported types of contact, the keyword option MPP can be used. Adding this keyword option triggers the reading of a new control card immediately following the keyword (unless an ID card is included). See the section on *CONTACT for details of the parameters and their meanings.

Output Files and Post-Processing

For performance reasons, MPP/LS-DYNA combines many of the ASCII output files normally created by LS-DYNA into a binary format. MPP/LS-DYNA writes this output only in binary format, irrespective of the setting of the variable BINARY in *DATABASE_*OPTION*. There is a post-processing program l2a, which reads the binary database of files and produces as output the corresponding ASCII files. MPP/LS-DYNA creates the new binary files in the directory specified as the global directory in the pfile (see section pfile). The files (up to one per processor) are named binout*nnnn*, where *nnnn* is replaced by the four-digit processor number. To convert these files to ASCII, simply feed them to the l2a program like this:

l2a binout*

LS-PREPOST can read the binout files directly, so conversion is not required. We provide it for backward compatibility.

The *supported* ASCII files are:

*DATABASE_ABSTAT

*DATABASE_BNDOUT

*DATABASE_CURVOUT

*DATABASE_DEFGEO

*DATABASE_DEFORC

*DATABASE_DISBOUT

*DATABASE_ELOUT

*DATABASE_FSI

*DATABASE_FSI_SENSOR

*DATABASE_GCEOUT

*DATABASE_GLSTAT

*DATABASE_JNTFORC

*DATABASE_MATSUM

*DATABASE_NCFORC

*DATABASE_NODFOR

*DATABASE_NODOUT

*DATABASE_PBSTAT

*DATABASE_PRTUBE

*DATABASE_RBDOUT

*DATABASE_RCFORC

*DATABASE_RWFORC

*DATABASE_SBTOUT

*DATABASE_SECFORC

*DATABASE_SLEOUT

*DATABASE_SPCFORC

*DATABASE_SPHMASSFLOW

*DATABASE_SPHOUT

*DATABASE_SPHVICINITY

*DATABASE_SWFORC

*DATABASE_TRACER_ALE

*DATABASE_TPRINT

Some of the normal LS-DYNA files will have corresponding collections of files produced by MPP/LS-DYNA, with one per processor. These include the d3dump files (new names = d3dump.*nnnn*), the messag files (now mes*nnnn*) and others. Most of these will be found in the local directory specified in the pfile.

The format of the d3plot file has not been changed. It will be created in the global directory, and can be directly handled with your current graphics post-processor.

APPENDIX O

Parallel Specific Options

There are a few new command line options that are specific to the MPP version of LS-DYNA.

In the serial and SMP versions of LS-DYNA, the amount of memory required to run the problem can be specified on the command line by adding "memory=*XXX*", where *XXX* is the number of words of memory to be allocated. For the MPP code, this will result in each processor allocating *XXX* words of memory. If pre-decomposition has not been performed, one processor must perform the decomposition of the problem. This can require substantially more memory than will be required once execution has started. For this reason, there is a second memory command line option, "memory2=*YYY*." If used together with adding "memory=*XXX*," the decomposing processor will allocate *XXX* words of memory, and all other processors will allocate *YYY* words of memory.

For example, in order to run a 250,000 element crash problem on 4 processors, you might need memory=80m and memory2=20m. To run the same problem on 16 processors, you still need memory=80m, but can set memory2=6m. The value for *memory2* drops nearly linearly with the number of processors used to run the program, which works well for shared-memory systems.

See the "Memory" subsection of the "Linear Equation Solver" section in Appendix P, "Implicit Solver," for additional remarks pertaining to memory for implicit analyses.

The full deck restart capability is supported by the MPP version of LS-DYNA, but in a manner slightly different than the SMP code. Each time a restart dump file is written, a separate restart file is also written with the base name d3full. For example, when the third restart file d3dump03 is written (one for each processor, d3dump03.0000, d3dump03.0001, etc), there is also a single file written named d3full03. This d3full file (or alternately the runfull file, which is written at the interval specified in *DATABASE_BINARY_RUNRSF) is required in order to do a full deck restart in MPP instead of a d3dump or runrsf file. In order to perform a full deck restart with the MPP code, you first must prepare a full deck restart input file as for the serial/SMP version. Then, instead of giving the command line option r = d3dump03 for example, you would use the special option n = d3full03. The presence of this command line option tells the MPP code that this is a restart, not a new problem, and that the file d3full03 contains the geometry and stress data from the previous run.

PFILE

A pfile contains MPP-specific parameters that affect the execution of the program. The command line option p = pfile causes LS-DYNA to use this file. The file is split into sections, with several options in each section. Currently, the sections directory, decomposition, contact, and general are available. First, here is a sample pfile:

```
directory {
```

```
        global rundir
        local /tmp/rundir
    }
    contact {
        non_blocking
    }
```

The file is case-insensitive and has a free format input. The sections and options currently supported are discussed below.

directory:

Holds directory-specific options.

global *path*

Path to a directory where program output should be written. If **transfer_files** is not given, this directory needs to be accessible to all processors – otherwise it is only accessed by processor 0. This directory will be created if necessary.
Default = current working directory

global_message_files

If this option appears, the message files are written in the global directory rather than the local directory
Default = disabled (message files go in the local directory)

local *path*

Path to a processor-specific local directory for scratch files. This directory will be created if necessary. If a pfile is not provided, the local directory can be defined through the environment variable LSTC_DIR_LOCAL. This directory should be on a local disk on each processor for performance reasons.
Default = global path

rmlocal

If this option, which is not available for the Windows OS, is given and **transfer_files** is active, the program attempts to clean up the **local** directories on each processor. In particular, it deletes files that are successfully transferred back to the **global** directory and removes the **local** directory if it was created. It will not delete any files if there is a failure during file copying, nor will it delete directories it did not create.
Default = disabled

repository *path*

Path to a safe directory accessible from processor 0. This directory will be created if necessary. This is intended to be used as a safekeeping/backup of files during execution and should only be used if **transfer_files** is also given. If this directory is specified, then the following actions occur:

- At program start up, any required files (d3dump, binout, etc) that cannot be located in the **global** directory are looked for in the **repository** for copying to the **local** processor directories.
- Important output files (d3dump, runrsf, d3plot, binout and others) are synchronized to the repository regularly. That is, every time one of these files is updated on the node local or the global directories, a synchronized copy is updated in the repository.

The intention is that the repository be on a redundant disk, such as NAS, to allow restarting the problem if a hardware failure should occur on the machine running the problem. It must be noted that some performance penalty must be paid for the extra communication and I/O. Effort has been made to minimize this overhead, but this option is not recommended for general use.
Default = unspecified

transfer_files
If this option is given, then processor 0 will write all output and restart files to the **global** directory (see "global" below), and scratch files to the **local** directory. All other processors will write all data to their **local** directory. At normal termination, all restart and output files will be copied from the processor specific **local** directories to the **global** directory. Also, if this is a restart from a dump file, the dump files will be distributed to the processors from the **global** directory. With this option enabled, there is no need for the processors to have shared access to a single disk for output – all files will be transferred as needed to and from the **global** directory.
Default = disabled

decomposition:

Holds decomposition-specific options.

automatic
If this option is given, an attempt is made to automatically determine a reasonable decomposition, primarily based on the initial velocity of nodes in the model. Use of the **show** option is recommended to verify a reasonable decomposition.

aledist
Distribute ALE elements to all processors.

ctcost *id1,id2,…,sf*
Apply a scale factor to the partition weight of all elements defined in the surfa and surfb sides in the list of contacts. A prefix of "a," or "b," can be added before "id1" to apply different scale factors to the surfa or surfb sides, respectively. Repeat this option as desired.

dcmem *n*

It may be in some cases that the memory requirements during the first phase of decomposition are too high. If that is found to be the case (if you get out of memory errors during decomposition phase 1), then this may provide a work around. Specifying a value *n* here will cause some routines to process the model in blocks of *n* items, when normal processing would read the whole set (of nodes, elements, whatever) all at once. This will reduce memory requirements at the cost of greater communication overhead. Most users will not need this option. Values in the range of 10,000 to 50,000 would be reasonable.

dunrefIc

Time dependent load curves are usually applied to the following boundary or loading conditions on each node. By default, those curves are copied to all MPP subdomain without checking for the presence of that node in the domain or not. The curves are evaluated every cycle and may consume substantial CPU time. This command will remove curves which are not referenced in the MPP subdomain for the following keywords.

*BOUNDARY_PRESCRIBED_MOTION_NODE

*LOAD_NODE

*LOAD_SHELL_ELEMENT

*LOAD_THERMAL_VARIABLE_NODE

file_*option filename*

This command is for creating a pre-decomposition file that can be used in subsequent runs to skip performing the decomposition or for reading a pre-decomposition created in a previous run. *filename* is the name of the file that holds the decomposition information or will hold the decomposition information. *option* can be unset or set to *READ* or *WRITE*. If *option* is unset, this file will be created in the current working directory if it does not exist. If the file already exists in the working direction, the file will be used for the decomposition. When *READ* is used, the pre-decomposition file must be in the working directory. The job will terminate if the file is not present. When *WRITE* is used, the program creates the file. The job will terminate if the pre-decomposition file already exists in the working directory.
Default = None

method *name*

Currently, there are two decomposition methods supported, namely *rcb* and *greedy*. Method *rcb* is Recursive Coordinate Bisection. Method *greedy* is a simple neighborhood expansion algorithm. The impact on overall runtime is problem dependent, but *rcb* generally gives the best performance.
Default = rcb

newcost

If this option appears in the decomposition section, a newer version of the element cost data is used during the decomposition with the goal of improving the load balance.

numproc *n*

The problem will be decomposed for n domains. If $n > 1$ and you are running on 1 processor, or if the number of processors you are running on does not evenly divide n, then execution terminates immediately after decomposition. Otherwise, the pre-decomposition file is written, and execution continues. For a decomposition only run, both **numproc** and **file** should be specified.
Default = the number of processors you are running on

outdecomp *itype*

If this option appears in the decomposition section, the decomposition is written to the output file decomp_parts.lsprepost for *itype* = 1, decomp_parts.ses for *itype* = 2, and decomp_parts.lsprepost_s###### where ###### is a d3plot state number for *itype* = 3. The d3plot file is not changed.

For versions prior to R18, the problem runs normally after decomposition. For R18 and later versions, the problem runs normally unless it is run on one processor. If it is run on a single processor, the job terminates at the end of the decomposition phase, and the code outputs a file with the final decomposition. Note that *itype* must be set to 1 or 3 for the single processor case. To specify the number of domains onto which the problem is decomposed, use **numproc**. This single processor case replaces the functionality of **show**.

region rx ry rz sx sy sz c2r s2r 3vec mat

See the section below on Special Decompositions for details about these decomposition options.

rcblog *filename*

This option is ignored unless the decomposition method is *rcb*. A record is written to the indicated file recording the steps taken during decomposition. This is an ASCII file giving each decomposition **region** (see the section on Special Decompositions) and the location of each subdivision for that **region**. Except for the addition of this decomposition information, the file is otherwise equivalent to the current pfile. Thus it can be used directly as the pfile for a subsequent problem, which will result in a decomposition as similar as possible between the two runs. For example, suppose a simulation is run twice, but the second time with a slightly different mesh. Because of the different meshes the problems will be distributed differently between the processors, resulting in slightly different answers due to roundoff errors. If an rcblog is used, then the resulting decompositions would be as similar as possible.

show

If this option appears in the decomposition section, the d3plot file is modified so that the final decomposition can be viewed with a post-processor. Displaying material 1 in a post-processor shows the portion of the problem assigned to processor 0, and so on. The problem does not actually run, but the code terminates once the initial d3plot state is written. This file can be created using a single processor with a decomposition over *n* domains by also using **numproc**. Warning: As of R18, this option is deprecated. Instead, use **outdecomp** to write the visualization file with **numproc** to specify the number of domains , and run the job on a single processor.

timing_start *n*

Begin timing of element calculations on cycle *n*. The appearance of this option will trigger the generation of a file named DECOMP_TIMINGS.OUT during normal termination. This file will contain information about the actual time spent doing element calculations, broken down by part.

timing_end *n*

End timing of element calculations on cycle *n*. A reasonable value is probably **timing_start** + 50 or 100.

timing_file *filename*

The file filename is assumed to be the output file DECOMP_TIMINGS.OUT from a previous run of this or a similar model. The computational cost of each part that appears in this file is then used during the decomposition instead of the built-in internal value for that part. Matching is based strictly on the user part ID. The first two lines of the file are skipped and only the first two entries on each of the remaining lines are relevant (the part ID and the cost per element).

transform_keyword

Including this option will cause global decomposition transformations to apply to all the decomposition regions created by the decomposition keywords _PARTS_-DISTRIBUTE, _PARTSET_DISTRIBUTE, _ARRANGE_PARTS, and _CONTACT_-DISTRIBUTE. If this option is not given, these regions are decomposed without any coordinate transformation applied.

vspeed

If this option is specified, a brief measurement is taken of the performance of each processor by timing a short floating point calculation. The resulting information is used during the decomposition to distribute the problem according to the relative speed of the processors. This might be of some use if the cluster has machines of significantly different speed.

APPENDIX O

contact:

This section has been largely replaced by the MPP keyword option on the normal contact card. The remaining useful options are:

alebkt *n*
Sets the bucket sort frequency for FSI (fluid-structure interaction) to once every *n* cycles.
default = 50

groupable_exclude_ag
Excludes *CONTACT_AUTOMATIC_GENERAL from groupable contacts, overriding any other settings.

non_blocking
The non-blocking algorithm applies to groupable contacts. With this option, data for contact is packed/unpacked before the element loop. With this option, the contact cost is considered along with the element cost for better load balancing.

general:

Holds general options.

lstc_reduce
If this option appears, our own reduce routine is used to get consistent summation of floating point data among processors. See also, *CONTROL_MPP_IO_LSTC_REDUCE which has the same effect.

nobeamout
Generally, whenever a beam, shell, or solid element fails, an element failure report is written to the d3hsp and message files. This can generate a lot of output. If this option appears, the element failure report is suppressed.

nodump
If this option appears, all restart dump file writing will be suppressed: d3dump, runrsf, d3full and runfull files will not be written.

nod3dump
If this option appears, writing of d3dump and runrsf files will be suppressed.

nofull
If this option appears, writing of d3full and runfull (full deck restart) files will be suppressed.

notiedio
If this option appears, the tied_nodes temp files are stored in memory instead of being output to disk. This option helps avoid problems on distributed file systems.

runrsfonly
If this option appears, writing of d3dump files will not occur – runrsf files will be written instead. Any time a d3dump *or* runrsf file would normally be written, a runrsf file will be written.

swapbytes
If this option appears, the d3plot and interface component analysis files are written in swapped byte ordering.

Special Decompositions:

These options appear in the "decomposition" section of the pfile and are only valid if the decomposition method is *rcb*. The *rcb* decomposition method works by recursively dividing the model in half, each time slicing the current piece of the model perpendicularly to one of the three coordinate axes. It chooses the axis along which the current piece of the model is longest. The result is that it tends to generate cube shaped domains aligned along the coordinate axes. This is inherent to the algorithm but is often not the behavior desired.

This situation is addressed by providing a set of coordinate transformation functions which are applied to the model before it is decomposed. The resulting deformed geometry is then passed to the decomposition algorithm, and the resulting domains are mapped back to the undeformed model. As a simple example, suppose you wanted rectangular domains aligned along a line in the *xy*-plane, 30 degrees from the *x*-axis, and twice as long along this line as in the other two dimensions. If you applied these transformations:

```
sx 0.5
rz -30
```

then you would achieve the desired effect.

Furthermore, it may be desirable for different portions of the model to be decomposed differently. It is now possible to specify different regions of the model to be decomposed with different transformations. The general form for a special decomposition would look like this:

```
decomposition {
        region { <region specifiers> <transformation> <grouping> }
        region { <region specifiers> <transformation> <grouping> }
        <transformation>
}
```

APPENDIX O

Where the region specifiers are logical combinations of **box**, **sphere**, **cylinder**, **parts**, and **silist**.
The transformation is a series of **sx**, **sy**, **sz**, **rx**, **ry**, **rz**, **c2r**, **s2r**, **3vec**, and **mat**. The grouping is either **lumped**, **together**, **nproc**, or empty. The portion of the model falling in the first region will be decomposed according to the given transformation. Any remaining part of the model in the second region will then be treated, and finally anything left over will be decomposed according to the final transformation. Any number of regions may be given, including 0. Any number of transformations may be specified. They are applied to the region in the order given.

The region specifiers are:

box xmin xmax ymin ymax zmin zmax
A box with the given extents

sphere xc yc zc r
The sphere centered at (x_c, y_c, z_c) with radius r. If r is negative, it is treated as infinite.

cylinder xc yc zc ax ay az r d
A cylinder with center at (x_c, y_c, z_c) with radius r, extending out in the direction of (a_x, a_y, a_z) for a distance of d. If d is 0, the cylinder is infinite in both directions.

parts n1 n2 n3 n4….
All parts whose user ID matches one of the given values are included in the region. Any number of values may be given.

partsets n1 n2 n3 n4….
All partsets whose user ID matches one of the given values will have all of their parts included in the region. Any number of values may be given.

silist n1 n2 n3 n4….
All elements involved in a contact interface whose user ID matches one of the given values are included in the region.

The transformations available are:

local
This is only useful in conjunction with the *CONTROL_MPP_PFILE keyword, when used inside a file included with*INCLUDE_TRANSFORM. At this point in the region processing, a transformation is inserted to invert the transformation used by *INCLUDE_TRANSFORM. The effect is that if this option appears before any other transformation options, all those options (and the subsequent decomposition) are performed in the coordinate system of the included file itself, not the global coordinate system.

sx t
Scale the current x coordinates by t.

sy t
Scale the current y coordinates by t.

sz t
Scale the current z coordinates by t.

rx t
Rotate around the current x axis by t degrees.

ry t
Rotate around the current y axis by t degrees.

rz t
Rotate around the current z axis by t degrees.

txyz x y z
Translate by (x, y, z).

mat m11 m12 m13 m21 m22 m23 m31 m32 m33
Transform the coordinates by matrix multiplication:

$$\text{Transformed Coordinates} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

3vec v11 v12 v13 v21 v22 v23 v31 v32 v33
Transform the coordinates by the inverse of the transpose matrix:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} v_{11} & v_{21} & v_{31} \\ v_{12} & v_{22} & v_{32} \\ v_{13} & v_{23} & v_{33} \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$
$$= \mathbf{V}^{\mathrm{T}} \times \text{transformed coordinates}$$

This appears complicated, but in practice is very intuitive: instead of decomposing into cubes aligned along the coordinate axes, *rcb* will decompose into parallelepipeds whose edges are aligned with the three vectors (v_{11}, v_{12}, v_{13}), (v_{21}, v_{22}, v_{23}), and (v_{31}, v_{32}, v_{33}). Furthermore, the relative lengths of the edges of the decomposition domains will correspond to the relative lengths of these vectors.

C2R x0 y0 z0 vx1 vy1 vz1 vx2 vy2 vz2
The part is converted into a cylindrical coordinate system with origin at (x_0, y_0, z_0), cylinder axis (v_{x1}, v_{y1}, v_{z1}) and $\theta = 0$ along the vector (v_{x2}, v_{y2}, v_{z2}). You can think of this as tearing the model along the (v_{x2}, v_{y2}, v_{z2}) vector and unwrapping it around the (v_{x1}, v_{y1}, v_{z1}) axis. The effect is to create decomposition domains that are "cubes" in cylindrical coordinates: they are portions of cylindrical shells. The actual transformation is:

$$\text{new}(x, y, z) = \text{cylindrical coordinates}(r, \theta, z)$$

Knowing the order of the coordinates is important if combining transformations, as in the example below.

S2R x0 y0 z0 vx1 vy1 vz1 vx2 vy2 vz2

Just like the above, but for spherical coordinates. The (v_{x1}, v_{y1}, v_{z1}) vector is the $\phi = 0$ axis.

$$\text{new}(x, y, z) = \text{spherical coordinates}(\rho, \theta, \phi)$$

The grouping qualifier is:

lumped

Group all elements in the region on a single processor. If this qualifier is not given, the elements in the region are distributed across all processors.

together

Group all elements in the region as one "super" element and join decomposition together with other remaining parts. This qualifier works like "lumped" to keep all elements in the region on a single processor. It also assigns them on the same processor with their neighboring elements with load balanced. In general, this qualifier is recommended over "lumped."

nproc *n frstp*

This region will be distributed to n processors which usually is a subset of **numproc** and starting from rank **frstp**. If **frstp** is not given, the code will select the n least costly processors from **numproc** automatically.

Examples:

rz 45

will generate domains rotated -45° around the z-axis.

C2R 0 0 0 0 0 1 1 0 0

will generate cylindrical shells of domains. They will have their axis along the vector (0,0,1), and will start at the vector (1,0,0). Note that the part will be cut at (1,0,0), so no domains will cross this boundary. If there is a natural boundary or opening in your part, the $\theta = 0$ vector should point through this opening. Note also that if the part is, say, a cylinder 100 units tall and 50 units in radius, after the C2R transformation the part will fit inside the box: $x = [0,50]$, $y = [0, 2\pi)$, $z = [0,100]$. In particular, the new y coordinates (θ) will be very small compared to the other coordinate directions. It is therefore likely that every decomposition domain will extend through the complete transformed y-direction. This means that each domain will be a shell completely around the original cylinder. If you want to split the domains along radial lines, try this pair of transformations:

C2R 0 0 0 0 0 1 1 0 0
SY 5000

This will do the above C2R, but then scale y by 5000. This will result in the part appearing to be about 30,000 long in the y-direction, long enough that every decomposition domain will divide the part in this (transformed) y-direction. The result will be decomposition domains that are radial "wedges" in the original part.

General combinations of transformations can be specified, and they are applied in order:

SX 5 SY .2 RZ 30

will scale x, then y, and then rotate.

A more general decomposition might look like:

```
decomposition { rx 45 sz 10
    region { parts 1 2 3 4 5 and sphere 0 0 0 200 lumped }
    region { box 0 100 –1.e+8 1.e+8 0 500 or sphere 100 0 200 200 rx 20 }
}
```

This would take elements that have user ID 1, 2, 3, 4, or 5 for their part *and* that lie in the sphere of radius 200 centered at (0,0,0), and place them all on one processor.

Then, any remaining elements that lie in the given box *or* the sphere of radius 200 centered at (100,0,200) would be rotated 20° in the x-direction and then decomposed across all processors. Finally, anything remaining would be rotated 45° in the x-direction, scaled 10 in the z-direction, and distributed to all processors. In general, region qualifiers can be combined using the logical operations *and*, *or*, and *not*. Grouping using parentheses is also supported.

Execution of MPP/LS-DYNA

MPP/LS-DYNA runs under a parallel environment provided by the hardware vendor. The execution of the program therefore varies from machine to machine. On some platforms, command line parameters can be passed directly on the command line. For others, the use of the names file is required. The names file is supported on all systems.

The serial/SMP code supports the use of the SIGINT signal (usually Ctrl-C) to interrupt the execution and prompt for user input, generally referred to as "sense switches." The MPP code also supports this capability. However, on many systems a shell script or front end program (generally "mpirun") is required to start MPI applications. Pressing Ctrl-C on some systems will kill this process, and thus kill the running MPP-DYNA executable. As a workaround, there are two ways to issue this sense-switch:

1. Create a file called D3KIL with the desired sense switch in it. LS-DYNA checks for this file every 100 cycles and executes this sense-switch.

2. When the MPP code begins execution, it creates a file bg_switch in the current working directory. This file contains the following single line:

   ssh -x <machine name> kill -INT <PID>

   where < machine name > is the hostname of the machine on which the root MPP-DYNA process is running, and <PID> is its process ID. Thus, simply executing this file will send the appropriate signal and prompt the user to enter the desired sense switch. Alternatively, a file with name switch can be created where the desired sense switch is included in the file. When bg_switch is executed, the desired 'sense switch' is applied.

Here is a simple table to show how to run the program on various platforms. Of course, scripts are often written to mask these differences.

Platform	Execution Command
DEC Alpha	dmpirun –np n mpp-dyna
Fujitsu	jobexec –vp n –mem m mpp-dyna
Hitachi	mpirun –np n mpp-dyna
HP	mpp-dyna –np n
IBM	#!/bin./ksh export MP_PROC=n export MP_LABELIO=no export MP_EUILIB=us export MPI_EUIDEVICE=css0 poe mpp-dyna
NEC	mpirun –np n mpp-dyna
SGI	mpirun –np n mpp-dyna
Sun	tmrun –np n mpp-dyna

In the above, **n** is the number of processors, **mpp-dyna** is the name of the MPP/LS-DYNA executable, and **m** is the MB of real memory.

APPENDIX P: Implicit Solver

Introduction

The terms implicit and explicit refer to time integration algorithms. In the explicit approach, internal and external forces are summed at each node point, and a nodal acceleration is computed by dividing by nodal mass. The solution is advanced by integrating this acceleration in time. The maximum time step size is limited by the Courant condition, producing an algorithm which typically requires many relatively inexpensive time steps.

While explicit analysis is well suited to dynamic simulations such as impact and crash, it can become prohibitively expensive to conduct long duration or static analyses. Static problems, such as sheet metal springback after forming, are one application area for implicit methods. In the implicit method, a global stiffness matrix is computed, inverted, and applied to the nodal out-of-balance force to obtain a displacement increment. The advantage of this approach is that time step size may be selected by the user. The disadvantage is the large numerical effort required to form, store, and factorize the stiffness matrix. Implicit simulations therefore typically involve a relatively small number of expensive time steps.

In a dynamic implicit simulation these steps are termed *time steps*, and in a static simulation they are *load steps*. Multiple steps are used to divide the nonlinear behavior into manageable pieces, to obtain results at intermediate stages during the simulation, or perhaps to resolve a particular frequency of motion in dynamic simulations. In each step, an equilibrium geometry is sought which balances dynamic, internal and external forces in the model. The *nonlinear equation solver* performs an iterative search using one of several Newton based methods. *Convergence* of this iterative process is obtained when norms of displacement and/or energy fall below user-prescribed tolerances. Within each implicit iteration there is a *line search* performed for enhancing robustness of the procedure.

The implicit analysis capability was first released in Version 950. Initially designed for metal forming springback simulation, this new capability allowed static stress analysis. Version 970 added many additional implicit features, including new element formulations for linear and modal analysis. A milestone in implicit was the advent of Version 971 in which implicit analysis was carried over to MPP and thus allowed much larger problems to be solved, and from there it has evolved extensively to presently contend well among competitive softwares. Still, it can be a gruesome task to set up an implicit input problem that runs all the way to completion with acceptable results, especially when contacts are involved. The purpose of this text is to explain keywords of interest and suggest values of important parameters in order to give users tools for developing own solution strategies to these kinds of problems.

APPENDIX P

A prerequisite for running implicit is to use a double precision version of LS-DYNA, and a machine with a significant amount of memory. A theoretical overview of implicit can be found in the LS-DYNA Theory Manual.

Nonlinear Implicit Analysis

Activating Implicit Analysis

The keyword *CONTROL_IMPLICIT_GENERAL is used to activate the implicit method, and in principle it is sufficient to set

IMFLAG = 1
DT0 = some reasonable initial time step.

The initial time step should be chosen small enough to resolve the frequency spectrum of interest and/or provide decent convergence properties, but large enough to benefit from an implicit analysis. With no other implicit options, this converts an explicit input deck to a static implicit input deck with a constant time step throughout the simulation and the problem will terminate upon convergence failure. Time stepping strategies to prevent this are discussed below.

Singularities and Eigenvalue Analysis

The first concern in implicit analysis is to prevent singularities in the stiffness matrix, otherwise the chance of succeeding is close to none. The major cause of a singular stiffness matrix is the presence of rigid body modes in a static problem which can be revealed in an eigenvalue analysis. This is done by using *CONTROL_IMPLICIT_EIGENVALUE and setting

NEIG = number of modes to extract.

First, run an eigenvalue analysis and extract (if practically possible) enough modes to see all the rigid body modes just to get an impression of the properties of the model. The frequencies are in the output text file eigout and the mode shapes can be animated in the binary output file d3eigv using LS-PrePost. Some rigid body modes may come as a surprise and should be constrained. A typical example of this is a beam that is free to rotate around its own axis. Other rigid body modes are a consequence of the nature of the problem and cannot be constrained without destroying the connection to reality. An example of this could be components that are to be constrained with contacts but are initially free to move. There are several strategies to deal with these latter rigid body modes; some are discussed in the context of contacts but here dynamics is used.

Dynamics and Intermittent Eigenvalue Analysis

Dynamics alleviates singular rigid body modes and is activated on *CONTROL_IMPLICIT_DYNAMICS by setting

IMASS = 1 or a negative number.

If the purpose is to solve a dynamic problem in the first place, then just use IMASS = 1 and don't bother about the other parameters; otherwise some other strategy is necessary. The idea is start with a dynamic analysis until the rigid modes have been constrained by contacts at which point dynamics is turned off. The simplest way of doing this is to set

TDYBIR = 0
TDYDTH = time when contacts have been established and rigid body modes are constrained
TDYBUR = time after TDYDTH for fading out dynamics between TDYDTH and TDYBUR.

If the dynamic results are of no interest but just a way to proceed to the static solution, then it is recommended to use numerical damping to prevent unnecessary oscillations, that is, set

GAMMA = 0.6
BETA = 0.38.

This should be enough to start up most problems of interest. A restriction with this is that when dynamics has been turned off it cannot be turned on again. If this is necessary for some reason, then a negative number of IMASS should be used to control dynamics through a load curve.

If it is hard to deduce how to choose the time when to turn off dynamics, the use of intermittent eigenvalues may be of great help. By setting

NEIG = a negative number

on *CONTROL_IMPLICIT_EIGENVALUE the user may extract eigenvalues at given time points during a nonlinear simulation and deduce from that if all rigid body modes have been eliminated.

The Geometric Stiffness Contribution

Stiffness singularities may also occur during a simulation due to a complex global phenomenon involving the stress state and geometry. For instance, slender components with compressive stresses could give rise to zero eigenvalues, commonly known as buckling. The mathematical explanation for these kinds of singularities is that the material and geometric stiffness contributions cancel out, and for this reason the geometric contribution to the stiffness matrix in LS-DYNA is optional. It is activated by putting

IGS = 1

on *CONTROL_IMPLICIT_GENERAL. Most often singularities due to this contribution have a negative effect on convergence although it sometimes helps. It is recommended to leave this as default and turn it on if other strategies fail. A controlled way of getting past singular points of this type is to use so called arc length methods for which the geometric stiffness should be turned on. This way of dealing with the problem is discussed in the Theory Manual.

BFGS or Full Newton

The nonlinear solver parameters are set on *CONTROL_IMPLICIT_SOLUTION where the solver type is specified as

NSOLVR = nonlinear or linear implicit analysis option.

By default, a nonlinear BFGS solution strategy is used when the stiffness matrix is reformed every 11th iteration and a maximum of 15 reformations is allowed (for a linear solution set NSOLVR = 1). These parameters are set by

ILIMIT = iterations between reforming stiffness matrix
MAXREF = maximum number of stiffness reformations

on *CONTROL_IMPLICIT_SOLUTION. Reforming the stiffness matrix is computationally expensive but stabilizes the solution procedure, and the best strategy in this context is very much problem dependent. The recommendation is to start with the default strategy and if necessary, change them. For hard problems decrease ILIMIT while for problems that converge well increase ILIMIT. For really bad problems switch to full Newton (the safe bet) by for instance setting

ILIMIT = 1
MAXREF = 30.

Keep the maximum number of reformations reasonably low or otherwise it may take an unnecessary amount of time just to reach convergence failure.

Convergence

Tolerances

Convergence is based on changes in displacements, energies and optionally residual forces, and the tolerance levels are set by

DCTOL = Relative displacement tolerance
ECTOL = Relative energy tolerance
RCTOL = Relative residual tolerance
ABSTOL = Absolute tolerance
DMTOL = Maximum displacement tolerance
EMTOL = Maximum energy tolerance
RMTOL = Maximum residual tolerance

on *CONTROL_IMPLICIT_SOLUTION. The tolerances on the left are based on Euclidian norms of the involved vectors, whereas the ones on the right are based on maximum norms. The maximum norm tolerances are optional and should be activated if increased accuracy is desired at the price of more implicit iterations. To keep things simple, the discussion that follows pertains only to the Euclidian norm tolerances.

By default, the progress of the equilibrium search is shown to the screen. The box below shows a typical iteration sequence, where the norms of displacement and energy are displayed. When these norms are reduced below user prescribed tolerances (default 0.001 and 0.01, respectively), the iteration process is said to have *converged*, and the solution proceeds to the next time step. The message files, messag in SMP and mesxxxx in MPP, typically contain a whole lot more information that will be dealt with further in output diagnostics and convergence issues.

```
BEGIN static time step     3
============================================================
               time =  1.50000E-01
  current step size =  5.00000E-02
Iteration:   1     *|du|/|u| =  3.4483847E-01     *Ei/E0 =  1.0000000E+00
Iteration:   2     *|du|/|u| =  1.7706435E-01     *Ei/E0 =  2.9395439E-01
Iteration:   3     *|du|/|u| =  1.6631174E-03     *Ei/E0 =  3.7030904E-02
Iteration:   4     *|du|/|u| =  9.7088516E-05     *Ei/E0 =  9.6749731E-08
```

Premature Convergence

The last of these parameters (ABSTOL) overrides the other three (DCTOL, ECTOL, RCTOL) in the sense that if the residual force is small enough, convergence is detected regardless of if the other three criteria are fulfilled or not. It has been seen that this sometimes give rise to so called *premature convergence*, converged states are not *really* converged in the sense that the residual norm is small enough. This gives bad results and it is usually recommended to tighten this tolerance to 10^{-20} to prevent this. On the other hand, if the problem is very hard this may prevent convergence to the extent that going back to the default is more or less necessary. It is difficult to give a general recommendation that holds for every problem.

As for the other three parameters, the one that usually comes into practice is the displacement criterion DCTOL. The energy criterion ECTOL is often easy to fulfill and the residual criterion is disabled by default and there may be no reason to activate this (that is unless a completely different strategy is used, or if convergence issues arise). Using the default values on all three is usually good enough to give acceptable results in decent time. For problems with poor convergence, the question of tightening or relaxing these parameters (that is the displacement convergence criterion) is up for debate. It is tempting to believe that relaxing the constraints will give better convergence which may or may not be true. Sloppy convergence criteria may once again result in premature convergence, and this will have a negative effect on the subsequent steps. The general recommendation would have to be to leave these unchanged, and be aware that relaxing them may not result in getting further in the implicit simulation.

Using Residual Tolerance

A novel strategy that seems to make sense and works well for some problems is to only use tolerances on residual forces RCTOL. This relies on the degree of nonlinearity of the simulation model being moderate, but it could be well worth trying. The first thing to do would be to set DCTOL and ECTOL to large numbers to disable them while setting RCTOL to a relatively small number (start with 0.01 and see how that works out). This must be complemented with an absolute tolerance on the residual forces to get past the initial stages where the residual forces are small enough to not fulfill the relative tolerance. To do this, set ABSTOL to a negative number, which is a different absolute criterion compared to setting it to a positive number. Now, this number is very problem dependent as this says that convergence is attained as soon as the Euclidian norm of the residual force vector is smaller than the absolute value of ABSTOL. It goes without saying that this requires running the problem for a few steps to monitor the residual norm in the message files as described below; this should give an indication on how to determine the ABSTOL value.

Convergence Norms

The degrees of freedom encountered in an LS-DYNA simulation are either *translational* or *rotational,* where the latter comes from the presence of beams and shells. Historically, convergence check is on norms of the translational degrees of freedom only, which is unsettling because rotational residual forces (moments) are equally important in an implicit simulation. It is therefore recommended to use

NLNORM = choice of convergence norms

on *CONTROL_IMPLICIT_SOLUTION to change this default (which is = 2). While NLNORM = 3 will incorporate residual degrees of freedom separately and is a more satisfying approach, the recommended option is to use NLNORM = 4 or a negative number. NLNORM = 4 will treat the entire force and displacement vector, respectively, as one complete mathematical vector and the convergence norms and scalar products used for checks are simply the Euclidian norms of these vectors. To make the displacement and residual norms unit consistent, a scale factor l, corresponding to a characteristic element size in the model, is introduced and accounted for when computing the norms. This means for instance that displacements are summed to l length unit times angular increments, and moments are summed to l length unit times forces. For NLNORM = 4 the scale factor is echoed to the output files and screen. If the internally calculated value is inappropriate for some reason, it may be convenient to use NLNORM = -(length scale) to supersede this scale factor.

Time Stepping Strategies

By default, LS-DYNA will terminate when a step fails to converge. This is unfortunate since it may just be that the step is too large to achieve convergence and taking smaller steps would solve this problem. Instead of starting all over with a smaller step size, set

IAUTO = 1 to activate automatic time stepping

on *CONTROL_IMPLICIT_AUTO. When the problem fails to converge LS-DYNA will with this option go back to the previously converged state and retry with a smaller step size. If the problem converges well, the step size is increased for subsequent steps. This process of backing up and retrying difficult steps lends much persistence to the analysis and is often the only procedure for solving highly nonlinear problems short of adjusting the step size manually. We recommend always turning automatic time stepping on.

Another parameter worth mentioning in this context is

DTMAX = Max time step allowed, or a negative number

on the same card. This is the maximum step size possible and should be chosen to not lose necessary information in the results, such as resolving frequencies, contacts and non-linear material response to a satisfactory degree. A negative value of this parameter will give the maximum step size as a function of the simulation time and allows for hitting key points that are of interest. The user may for instance look for the peak stress for a certain external load and hitting the point in time when this happens is crucial. It also allows for taking smaller steps during critical stages in the simulation without wasting resources away from these stages.

Line Search

A good line search strategy is crucial in solving nonlinear implicit problems; without it only simple problems would be solvable. There are a few line search options available and these are activated by

LSMTD = Line search method

on *CONTROL_IMPLICIT_SOLUTION. The default method is based on minimizing a potential energy along the search direction and at the same time keeping track of the residual force magnitude. It also detects when a BFGS step results in negative initial energy which causes the stiffness matrix to be reformed for robustness (this can only happen when the stiffness matrix has negative eigenvalues). Line search type 2 is based on residual forces and is more robust than the others; the drawback is that it typically results in too small steps and is not practically useful. Line search type 3 is very similar to type 4, but it does not track the residual force or avoid negative volumes to the same extent. In sum, there is no practical reason for choosing methods 1 through 3. Worth mentioning is however line search type 5 which combines the energy and residual method in a stricter sense. That is, it minimizes the potential energy just like type 4, but it only allows the residual norm to double at each implicit iteration. This has shown to be robust but slow in convergence and is to be used for problems that have difficulties converging with the default line search strategy. This has had a huge impact on the treatment of rubber models for instance. An interesting combination is to use line search method 5 together with the strategy of converging based on residual forces only.

Finally, the line search tolerance

LSTOL = Line search tolerance

on *CONTROL_IMPLICIT_SOLUTION is fine as it is; don't change it.

Output

As previously mentioned, the output to the message file is more extensive than that to the screen/stdout. To novice or average implicit users this information may be more than can be digested at first and this section may be skipped, but with experience the output can become really useful when running into convergence issues. For this discussion it is important to distinguish between different *types* of iterations, and we use the term *time/load step* to mean the actual advance in simulation time, *implicit iteration* to indicate points in the iterative solution procedure where a new search direction is obtained (corresponding to either Newton or BFGS) and *line search iteration* to indicate the search for an optimal step size along a given search direction. We begin by showing a typical iteration output for when NLPRINT = 3 following by a numbered list explaining the content.

```
[1]BEGIN implicit dynamics step      2 t = 2.5849E-01
 ============================================================
                time =  2.58489E-01
   current step size =  1.58489E-01
.
.
.

[2]Newton step computed
 Initial translational energy =  1.4031740E-01
[3]Initial total energy         =  1.4031740E-01
 Initial residual norm        =  7.5755334E-01
[4]Translational nodes norm     =  7.1314670E-01
 Translational nodes max      =  1.9159278E-01 at node         440788
 Translational rigid body norm =  3.5324028E-03
 Translational rigid body max =  2.3897586E-03 at body        4000024
 Rotational rigid body norm   =  2.5553155E-01
 Rotational rigid body max    =  1.7834357E-01 at body        4000024

[5]Evaluated residual for full step
 Current translational energy =  9.0141343E-02
[6]Current total energy         =  9.0141343E-02
 Current residual norm        =  2.2038666E+00
 Translational nodes norm     =  2.1950775E+00
 Translational nodes max      =  1.0295509E+00 at node         440984
 Translational rigid body norm =  3.4677960E-02
 Translational rigid body max =  2.6701145E-02 at body        4000024
 Rotational rigid body norm   =  1.9354585E-01
 Rotational rigid body max    =  1.2669458E-01 at body        4000024

[7]Evaluated residual for step size  6.6666667E-01
 Current translational energy =  1.1123844E-01
 Current total energy         =  1.1123844E-01
 Current residual norm        =  1.5007375E+00
 Translational nodes norm     =  1.4922142E+00
 Translational nodes max      =  7.1187918E-01 at node         440984
```

```
 Translational rigid body norm =  1.6121536E-02
 Translational rigid body max =  1.2659274E-02 at body        4000024
 Rotational rigid body norm   =  1.5890244E-01
 Rotational rigid body max    =  1.0852930E-01 at body        4000024

 Evaluated residual for step size  3.3333333E-01
 Current translational energy =  1.2543971E-01
 Current total energy         =  1.2543971E-01
 Current residual norm        =  8.3754668E-01
 Translational nodes norm     =  8.1775945E-01
 Translational nodes max      =  3.0290699E-01 at node         440984
 Translational rigid body norm =  5.4770134E-03
 Translational rigid body max =  4.5377150E-03 at body        4000024
 Rotational rigid body norm   =  1.8089755E-01
 Rotational rigid body max    =  1.2913973E-01 at body        4000024

 Line search continues
 Max relative step            =  0.3499001E+00
[8]Lower bound                =  3.3333333E-01
 Upper bound                  =  1.0000000E+00

 Evaluated residual for step size  7.7777778E-01
 Current translational energy =  1.0487486E-01
 Current total energy         =  1.0487486E-01
 Current residual norm        =  1.7421066E+00
 Translational nodes norm     =  1.7342352E+00
 Translational nodes max      =  8.2830775E-01 at node         440984
 Translational rigid body norm =  2.1464316E-02
 Translational rigid body max =  1.6693623E-02 at body        4000024
 Rotational rigid body norm   =  1.6402030E-01
 Rotational rigid body max    =  1.0799892E-01 at body        4000024

 Evaluated residual for step size  5.5555556E-01
 Current translational energy =  1.1674955E-01
 Current total energy         =  1.1674955E-01
 Current residual norm        =  1.2575584E+00
 Translational nodes norm     =  1.2472759E+00
 Translational nodes max      =  5.8455890E-01 at node         440984
 Translational rigid body norm =  1.1655353E-02
 Translational rigid body max =  9.2884622E-03 at body        4000024
 Rotational rigid body norm   =  1.6006264E-01
 Rotational rigid body max    =  1.1222751E-01 at body        4000024

[9]Line search converged in    2 iterations
[10]Max relative step            =  7.1106671E-01

[11]Iteration:    5    displacement            energy              residual
 ----------           not conv.             not conv.            converged
 norm ratio           1.013E+00             1.000E+00              n/a
 current norm         8.446E-01             1.403E-01            1.258E+00
 initial norm         8.338E-01             1.403E-01            7.575E+00
-------------    trans       rot      trans       rot        trans       rot
max node norm   8.533E-02  0.000E+00  2.052E-02  0.000E+00  5.846E-01  0.000E+00
   at node ID    4000674    9007526     440984    9007526     440984    9007526
-------------
       RB max   3.449E-02  6.613E-04 -5.507E-04  7.681E-05  9.288E-03  1.122E-
01
     at RB ID   4000024    4000024    4000024    4000024    4000024    4000024
```

Referring to the superscripts at the beginning of the lines in the excerpt above, and taking them in order of appearance, we have at

1. Beginning a *dynamic* implicit time/load step, the goal is to get to a given time point (in this case 2.5849E-01 using a step size of 1.58489E-01). Implicit time/load steps are either dynamic, *static* or *semidnmc*, where the last one indicates a transition phase between dynamics and statics.

2. A *Newton* implicit iteration is performed, meaning that a full reformation of the stiffness matrix has just been made. Other implicit iterations can be either a *linear* if at the first iteration of a time/load step, or *BFGS* if the implicit iteration corresponds to a quasi-Newton step.

3. Initial norms are displayed at the beginning of each implicit iteration; these numbers are used as reference for the line search algorithm when checking line search convergence.

4. Detailed initial norms are displayed, separating out translational/rotational as well as nodal/rigid body Euclidian/max norms. The node number and rigid body attaining the global maximum norms are displayed and can be used to spot critical points in the model. In this particular case node 440984 and part 4000024 would be the likely candidates for trouble.

5. A full step along the search direction is taken, as a candidate for the next implicit iteration.

6. Norms for the full step is displayed. The line search algorithm compares these with the ones from 3 to deduce if the line search convergence criterion is fulfilled.

7. Line search is in this case needed, and information for further iterates along the search direction is displayed in analogy to previous line search iterates. If no line search had been needed, the string *Line search is skipped* would have been displayed.

8. The way line search is performed, each line search iteration narrows in on the optimal step and the bounds within which the optimal step is sought is continuously displayed. The size of this interval becomes smaller and smaller with increasing number of line search iterations, but convergence should preferably be attained before it tends to zero.

9. Line search converged here in 2 iterations; as a rule of thumb the number of iterations for line search should be about 10 or less. For critical steps one might accept 20 iterations but that should be among the exceptions. If the string *Line search did not converge* is displayed, that is an indication of something being terribly wrong.

10. After each implicit iteration, *Max relative step* indicates how large steps have been taken during line search so far. It also indicates how well prescribed motion is approximated where a value of 1 is perfect. For values below 1 and in the presence of non-zero prescribed motion, convergence will not be accepted even though all norms are within prescribed tolerances. This will instead issue the warning *Convergence prevented due to unfulfilled boundary conditions* and iterations continue.

11. Diagnostics for iteration 5 is displayed; similar numbers as shown during line search but in another format. Furthermore, the initial norms here refer to the initial norms from the beginning of the implicit time/load step and not from the beginning of the implicit iteration. This table can be used to deduce how close the implicit time/load step is at converging. The single important number for comfortably assessing the accuracy of an implicit iterate is the residual norm, currently displayed as 1.258E+00 (bold-faced in the table), since this should be small for good results.

At equilibrium, the output ends with a summary on how convergence was achieved. It may look like

```
Convergence detected as a combination of
      1.Ratio of euclidian displacement
         Value =   2.3417E-03 vs Tolerance =  2.5000E-3
      2.Ratio of euclidian energy
         Value =   3.2269E-08 vs Tolerance =  1.0000E-2
```

which tells us that the ratio of displacement was below DCTOL = 2.5E-3 and the ratio of energy was below ECTOL = 1E-2. If other criteria are satisfied, these will be listed as well.

So, with NLPRINT = 3 information from every force evaluation is given, including type of step taken, the line search step size, the potential energy value used in line search and the magnitude of the residual force. A point is made here, already touched upon and will be repeated, and it is the following: the nonlinear implicit solver is solving for zero residual force, so basically the number observed for the magnitude of the residual force should be small upon convergence (bold-faced in the table). If it is not, then the convergence is premature, and the results may not be correct and subsequent steps are in danger. What "small" means in this context is hard to say since this depends on units as well as loads and geometry of the problem, but a good sign is that the residual force does not grow by more than the external loads in the problem do. Another thing to observe is how the line search is doing; if the line search starts needing many iterations and very large variations in the residual force along the search direction is observed, then it is likely that something in the model is causing this behavior. A safe bet is that contact states are changed along the line search direction causing discontinuities in the residual force and indicates that some remodeling has to be done.

APPENDIX P

We will come back to this section when discussing convergence problems and strategies to prevent them.

Linear Equation Solver

General

Within each equilibrium iteration, a linear system of equations of the form $\mathbf{K}\Delta\mathbf{u} = \mathbf{R}$ must be solved. To do this, the stiffness matrix **K** is factorized and applied to the out-of-balance load or residual **R**, yielding a displacement increment $\Delta\mathbf{u}$. Storing and solving this linear system represents a large portion of the memory and CPU costs of an implicit analysis. LS-DYNA uses a multi-frontal sparse direct solver. For a problem with N number of nodes, the number of operations and memory storage is asymptotically proportional to $N^{3/2}$ and $NlogN$, respectively. This should give a ballpark indication on the growth when increasing the size of a given model. It must be stressed, however, that it is *a priori* difficult to predict the actual value of these numbers as they are highly dependent on the problem solved, in particular on the nodal connectivity through elements and contacts.

We here focus on the memory, and the purpose is twofold. First it is a documentation of the memory diagnostics that is written to the output message files (messag in SMP and mesxxxx in MPP) of LS-DYNA for the linear solvers. The level of information is regulated by the

LPRINT = Linear solver diagnostics level

input parameter on *CONTROL_IMPLICIT_SOLVER, and this text will cover LPRINT = 2 as the information for higher values is of no particular interest to others than developers. Second it should help understanding the interrelationship between the size of a model, and the memory required to obtain a solution. We emphasize however that different classes of problems most likely require different guidelines (shell and solid structures typically result in different matrix topologies for instance) and to this end the user is left at earning this experience on his own.

Memory

The memory is specified and reported in terms of *words*, where 1 word is equivalent to 8 bytes in double precision arithmetic. To simplify things, a word can in this context be seen as the equivalent of a real number or an integer. For implicit calculations the executable is only provided in double precision.

When starting a simulation, the user typically specifies the *static* memory, for instance in SMP

```
ls-dyna i = in.k memory = 200m
```

meaning that LS-DYNA tries to allocate 200 Mwords for storing statically allocated data. In MPP, the physical memory is distributed among processes and this option applies to each individual process. In MPP there is also an option to add a second memory flag, for instance

```
mpp-dyna i = in.k memory = 1000m memory2 = 200m
```

and this means that 200 Mwords are allocated for each process while 1000 Mwords is allocated on the first process to handle everything up to the point when the model data has been distributed among all the involved processes. The memory size may also be specified on *KEYWORD.

Beginning with LS-DYNA version R11.0.0, more and more of the statically allocated memory is being moved to dynamically allocated memory. For explicit simulation, the will allow real time adaptivity and dynamic load balancing. Implicit has migrated all of the linear algebra storage (matrices, factorization) to dynamically allocated storage. This vastly simplifies the specification of memory and memory2 for implicit jobs.

If the model is initially explicit, the following message will be printed:

```
Memory required to process keyword: 38147436
```

For this case memory = 39m should be used on the command line.

For SMP, implicit will require an additional $(19 + 2 \times \text{ILIMIT}) \times (6 \times \text{NUMNP})$ words of memory for a transient nonlinear simulation where ILIMIT is specified on *CONTROL_-IMPLICIT_SOLUTION (default of 11; for modal or linear analysis ILIMIT = 0) and NUM-NP is the number of nodes in the model. To approximate the minimum value of memory needed to run the implicit analysis using SMP, one could add this additional value to the "Memory required to process keyword". In practice, one doesn't need to go to the trouble of calculating the required memory since the code will tell you what is needed as explained below.

For MPP, the nodes will be distributed across p processors, so a rough estimate of the addition to memory will reduced by $1/p$.

For an actual implicit execution look for

ALERT

which will tell you exactly the setting to use.

When a certain feature requires a slot in the memory, this is reported in the output text files, for instance as

```
contracting memory to    2306940 d 696365    implicit friction
expanding   memory to    2306955 d 696365    joints
```

The first number in each line above is the current value of statically allocated memory. The second number (following the "d") is the current value of dynamically allocated memory.

In the example above, the static memory pointer after having allocated for implicit friction is at 2306940; allocating the joints reserves 15 words and the static memory pointer is incremented accordingly. So the memory used up to this point is roughly 2.3 Mwords for this particular process. At the end of the initialization a memory report is written and can look like

```
S t o r a g e   a l l o c a t i o n
Memory required to begin solution      :     929 k
Linear Alg dynamically allocated memory:     196 k
Additional dynamically allocated memory:    2285 k
                                  Total:    3409 k
```

The total amount of memory used up to this point is roughly 3.5 Mwords, which is partitioned in the static memory specified and *dynamic* memory. Whenever a memory violation occurs, for instance when trying to allocate more memory than physically available or if more memory needs to be specified, LS-DYNA terminates with appropriate error information. The dynamic memory for the linear solver is allocated thereafter, which is the topic for the next section.

Linear Solver Memory Consumption

For LPRINT = 2, some memory information is given in the output files, and this is here presented in order of execution, thus as they appear in the files. We repeat that this is the information obtained in the message files and is thus referring to the memory consumption for this particular process if MPP is considered. For the sake of completeness we also cover the diagnostics related to the CPU time for the involved stages in the linear solution.

The last statically allocated memory for implicit is reserved and reported as

```
expanding   memory to    125081 d 2652145 matrix assembly allocation
```

After this allocation all remaining memory is dynamically allocated. All storage for the linear algebra modules is dynamically allocated. Typically this will be a 3 phase process of symbolical factorization, numeric factorization, and numeric solve.

Before the symbolic factorization takes place, current information on the system matrix and workspace reserved for implicit is reported

```
local number of rows      =             68490
local size of matrix      =           2338945
```

The first number refers to the number of independent degrees of freedom in the model, meaning the number of rows in the system matrix, and the second to the number of non-zero entries before factorization takes place, that is, after assembly.

The symbolic factorization is now performed and information on the CPU time for doing this together with the estimated size of the factorized matrix is reported.

```
CPU: symbolic factor      =          0.200
WCT: symbolic factor      =          0.203
storage currently in use  =        1038915
storage needed            =       12913838
factor speedup            =     1.9197E+00
solve  speedup            =     1.9378E+00
est.  factor nonzeroes    =     5.1432E+07
est.  factor operations   =     1.2099E+11
est.  total factor nz.    =     9.9666E+07
est.  max.  factor op.    =     1.2099E+11
est.  total factor op.    =     2.3293E+11
est.  max.  factor nz.    =     5.1432E+07
```

The first two rows reports the CPU and wall clock time for doing the symbolic factorization in terms of seconds. The next two are storage requirements for the symbolic factorization, of which the latter is the memory reserved for this. After this we have estimations of the speedup in the factorization and subsequent solve of the linear system of equations; this is a report from an MPP run on 2 processors. The last lines refer to the estimated requirement for storing and factorizing the system matrix and distinguish between the local, total and maximum number of each of these two entries. This allows for determining the memory scaling in the case of an MPP run.

Now LS-DYNA is in the position of reserving space for the factorization of the system matrix, and a report on the storage needed for this is given.

```
symbolic storage        1 =           1038915
in-core numeric  storg  1 =          62762269
out-of-core num.  storg 1 =          13168676
```

Here basically the same information is repeated, except for a slight increment in order to account for potential penalty when doing the actual factorization. The first is again the memory reserved for the symbolic factorization and the other two are the memory required to perform an in-core and out-of-core factorization, respectively. If the memory available is not sufficient for doing an in-core factorization, LS-DYNA warns about this and attempts to do an out-of-core factorization, which means that the hard disk is used to store information during factorization.

At this point a redistribution of the system matrix is performed and a short report is given on this

```
symbolic storage        2 =           1038915
in-core numeric  storg  2 =          68238926
out-of-core num.  storg 2 =          11553502
```

Now the actual factorization takes place and a report from this is given where the information of interest is

```
CPU: factorization        =            59.025
WCT: factorization        =            50.151
act.  factor nonzeroes    =       5.1236E+07
act.  factor operations   =       1.2092E+11
act.  max.  factor nz.    =       5.1236E+07
act.  max.  factor op.    =       1.2092E+11
act.  total factor nz.    =       9.9666E+07
act.  total factor op.    =       2.3293E+11
```

which basically is the same information as from the symbolic factorization except that now it is the actual instead of the estimated values that is reported.

Finally, the forward and backward substitutions are allocated for and performed. The information given is

```
symbolic storage        3 =          1038915
in-core numeric storg   3 =         68299122
out-of-core num.  storg 3 =         16476693
CPU: numeric solve        =            0.191
WCT: numeric solve        =            0.194
```

More information is available with the command line options DIAG = 1 (memory) and IMPCON = 1 (time stamps).

Elements and Materials

Implicit Accuracy

Implicit and explicit analysis differ in many respects. An important one is that the deformation during a single step is much larger in implicit than a typical one in explicit. In the context of elements and materials, the demand for stronger objectivity and higher accuracy in implicit is obvious. The notion of implicit executing the same algorithms as a corresponding explicit analysis is therefore not adopted in general. This can optionally be further extended, using

IACC = 1 to increase implicit accuracy

on *CONTROL_ACCURACY will make selected elements strongly objective and enhance the accuracy for selected materials. For instance finite rotations will be represented exactly and fully iterative plasticity adopted. In addition, the flag applies to tied contacts as will be elaborated on below. Currently the following elements are supported for this option

Solid elements -2,-1,1,2

Shell elements 4,-16,16,23,24
Beam elements 1,2,9

and materials 24 and 123 use fully iterative plasticity.

Contacts

Contacts are probably among the hardest features to treat in a nonlinear implicit simulation. They are divided into the categories *tied* (bilateral) and *sliding* (unilateral) contact, and the characteristics of the two are quite different. Tied contacts are most often applied as constraints, only sometimes using a penalty formulation, and are fairly easy to deal with as they are only moderately nonlinear. In contrast, sliding contacts are exclusively implemented as penalty contacts and hard to deal with because of the unilateral condition. While the number of contacts, see *CONTACT, in LS-DYNA is abundant, we will here only present the few contacts that make most sense to use in implicit.

Tied Contacts

Tied contacts in implicit analysis should be accompanied with the implicit accuracy option. This restriction essentially reduces the number of relevant contacts to six. These six contacts are

*CONTACT_TIED_NODES_TO_SURFACE
*CONTACT_TIED_NODES_TO_SURFACE_CONSTRAINED_OFFSET
*CONTACT_TIED_NODES_TO_SURFACE_OFFSET
*CONTACT_TIED_SHELL_EDGE_TO_SURFACE
*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET
*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET

The first and fourth of these project the nodes on the tracked side to the reference surface, while the rest are offset contacts for which nodes will remain in place. The fourth, fifth and sixth constrain rotations while the others do not. The third and sixth are penalty-based, while the others are constraint-based. In sum, these six contacts cover most reasonable scenarios.

The implicit accuracy option will in this context make these six tied contact strongly objective, and built-in intelligence will detect whether nodes contain rotations to constrain or not. A nice feature with the latter is that torsion is automatically applied with physical consistency. Thus, it is no problem to use single beams to model spot welds between solid elements. When using the same connection technique between shells, the beam axial rotational degrees of freedom will be constrained to the translational degrees of the

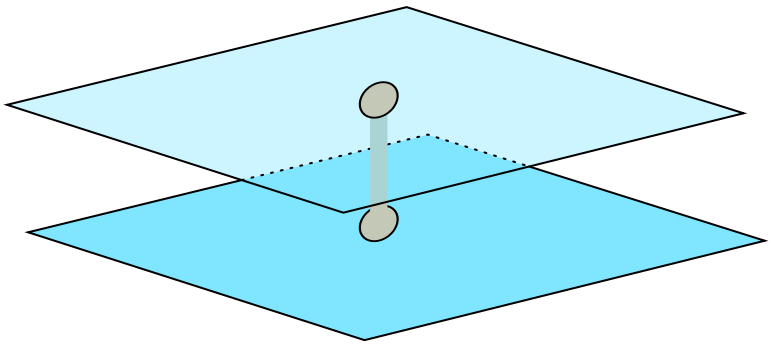


Figure 66-1. Modeling spotwelds by connecting a beam element to two shell elements.

shells, avoiding the relatively weak (and non-physical) drilling degree of freedom. This situation is depicted in Figure 66-1.

The choice of contact depends on the situation, but from a conceptual point of view it should be alright to use *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINED_OFFSET for most cases. The only problem then would be if complicated geometry results in termination from overconstraining, for which a switch to the corresponding penalty version *CONTACT_TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET is motivated. See General Remarks in the *CONTACT section of the LS-DYNA keyword manual for more information.

Sliding Contact

In this section we will discuss Mortar contact as this seems to be the best implicit contact algorithm when considering a combination of speed, accuracy and robustness. Mortar contact features smoothness and continuity that is highly appreciated in implicit analysis but is expensive enough to not be recommended for explicit analysis. Many other contact algorithms are supported for implicit analysis and execute faster than the Mortar contact, but this is often seen when IGAP flag is set to the default. This flag manipulates the stiffness matrix to the extent that accuracy may be deteriorated. If used, the results should be thoroughly checked. For Mortar contact IGAP has a different meaning as will be described IGAP. See General Remarks for *CONTACT_OPTION in the LS-DYNA keyword manual for more information.

Basics

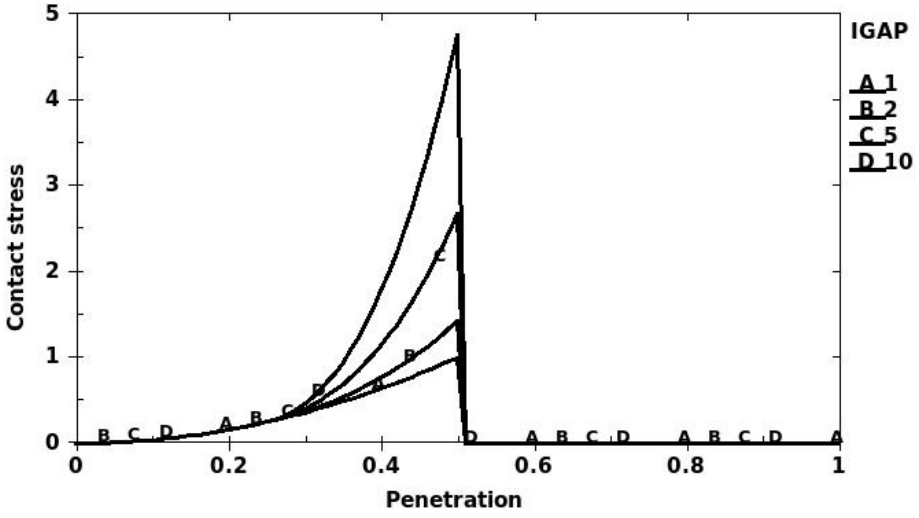
Mortar contact is activated by typically appending the suffix MORTAR to automatic single surface, automatic surface-to-surface or forming surface-to-surface keywords. It can also be run as tied and tiebreak contacts. All Mortar contacts are segment-to-segment and penalty-based. The tied and tiebreak contacts are always offset, meaning the tie occurs on the outer surfaces of shells and not on the mid-surfaces. For automatic contacts, edge contact with flat edges is always active. It possesses features that are of particular interest to implicit and are not available for other contacts. It is supported in both SMP and MPP, but the keyword option MPP does not apply. The SOFT flag does not apply. To summarize it is a contact algorithm especially intended for implicit analysis.

Recommendations

For Mortar forming contact the rigid tools must be meshed so that the normal vectors are directed towards the blank. Contacts from above and below the blank must be separated into two or more interfaces because contact can only occur from one side of the blank for a given contact interface. Rigid shells on the tracked side have no contact thickness. This is not the case for automatic Mortar contacts. For these contacts, there are no restrictions on the mesh and even rigid shells have contact thickness. For all Mortar contacts, part or part set based tracked and reference sides are recommended but not mandatory. If the two sides in the contact interface have different stiffnesses, use the weak part as the tracked side to get the best possible implicit convergence behavior. The single surface Mortar contact automatically takes care of this.

Characteristics

The contact pressure in Mortar contact is a parabolic function of the penetration in combination with a cubic stiffening phase. In short, the contact stiffness depends on the tracked side material and a characteristic length of the tracked side segment. The characteristic length for shells is the shell thickness and for solids is a median of the edges in the tracked side of the contact interface. The maximum penetration allowed is given as 95% of the average characteristic lengths on reference and tracked sides. For solids the definition of the characteristic length may cause the stiffness to become unnecessarily high if some elements are much smaller than most, so stiffness adjustments may be necessary. Furthermore, the characteristic length also determines the maximum penetration as well as the search radius for finding contact pairs; for this reason, the characteristic length can optionally be increased by assigning it on PENMAX on optional card B. In most cases it is expected that default value (= 0.0) for this parameter will suffice.



IGAP

The contact stiffness is parabolic with respect to penetration up to a penetration depth corresponding to half of the maximum penetration. For IGAP = 1 it will remain parabolic

for even larger penetrations, but the user may increase IGAP which means that the contact will stiffen for larger penetrations. In fact it will become cubic according to the picture above. The purpose of increasing IGAP could be to prevent the penetration from becoming larger than the maximum allowed penetration, because if convergence is attained with penetrations larger than this value this contact will be released in subsequent steps and the simulation is likely destroyed. Penetrations of this depth are likely to cause discontinuities along line searches and other discouraging phenomena. The user may of course scale the stiffness by increasing SFSA but this also scales the stiffness for small penetrations and probably has a negative effect on convergence.

Output for Debugging

Just as for implicit in general, information is always a good thing to have when convergence starts deteriorating and considering the release of contact in the previous section, it would be interesting to know if penetrations are large enough for this to be a potential danger. First, initial penetrations are always reported in the message file(s), including the maximum penetration and how initial penetrations are to be handled. The latter depends on the value of the IGNORE flag and this is dealt with below. In addition, by setting

MINFO = 1

on *CONTROL_OUTPUT, LS-DYNA will report the absolute maximum penetration as well as the maximum penetration in percentage after each equilibrium. If the relative maximum penetration reaches above 99% a warning message is printed as this particular contact is close to being released. The output is exemplified in the following.

```
     Contact sliding interface          1
     Number of contact pairs          527

     Maximum penetration is  0.2447797E+00 between
     elements     306774 and     306672

     Maximum relative penetration is  0.2266694E+02 % between
     elements     306742 and     306733

     Contact sliding interface          2
     Number of contact pairs        16209

     Maximum penetration is   0.5027643E+00 between
     elements     219492 and      94935

     Maximum relative penetration is  1.0366694E+02 % between
     elements     219492 and      94935
 *** Warning Penetration is close to maximum before release
```

This percentage value should ideally be kept below some 90% to have some sort of comfort margin. There are three ways to reduce maximum relative penetrations, and these are *(i)* to increase IGAP, *(ii)* to increase PENMAX for solids or *(iii)* to increase SFSA.

Initial Penetrations

As mentioned above, initial penetrations are always reported in the message file(s), including the maximum penetration and how initial penetrations are to be handled. The IGNORE flag governs the latter and the options are listed below.

IGNORE < 0	Same functionality as the corresponding absolute value but contact between segments belonging to the same part is ignored completely.
IGNORE = 0	Initial penetrations will give rise to initial contact stresses, meaning the tracked contact surface is not modified. *This is not available for Mortar but listed for the sake of completeness*.
IGNORE = 1	Initial penetrations will be tracked, that is, the tracked contact surface is translated to the level of the initial penetrations and subsequently follows the reference contact surface on separation until the unmodified level is reached.
IGNORE = 2	Initial penetrations will be ignored, meaning the tracked contact surface is translated to the level of the initial penetrations, optionally with an initial contact stress governed by MPAR1 (*this is the default*).
IGNORE = 3	Initial penetrations will be removed over time, that is, the tracked contact surface is translated to the level of the initial penetrations and pushed back to its unmodified level over a time determined by MPAR1.
IGNORE = 4	Same as IGNORE = 3, but it allows for large penetrations by also setting MPAR2 to at least the maximum initial penetration.

The use of IGNORE depends on the problem. If no initial penetrations are present, there is no need to use this parameter at all. If penetrations are relatively small in relation to the maximum allowed penetration, then IGNORE = 1 or IGNORE = 2 seems to be the appropriate choice. For IGNORE = 2 the user may specify an initial contact stress small enough to not significantly affect the physics, but large enough to eliminate rigid body modes and thus singularities in the stiffness matrix. The intention with this is to constrain loose parts that are initially close but not in contact by pushing out the contact surface using SFSAT and applying the IGNORE = 2 option. It is at least good for debugging problems with many singular rigid body modes. IGNORE = 3 is the Mortar interference counterpart, used for instance if there is a desire to fit a rubber component in a structure. With this option the contact surfaces are restored linearly in time from the beginning of the simulation to the time specified by MPAR1. A drawback with IGNORE = 3 is that initial penetration must be smaller than half the characteristic length of the contact or otherwise they will not be detected in the first place. For this reason, IGNORE = 4 was introduced where initial penetrations may be of arbitrary size, but it requires that the user provides crude information on the level of penetration of the contact interface. This is done in MPAR2 which must be larger than the maximum penetration or otherwise and error termination will occur. IGNORE = 4 only applies to solid elements at the moment.

When eliminating penetrations by simulation for models with many parts, some parts may contain thin members that cause spurious self-contacts. These may be difficult to work around by only adjusting contact parameters, but fortunately there is rarely any loss of generality in ignoring contact within parts completely since those are usually not of interest in such a context. The option IGNORE < 0 was implemented for this purpose and is a way to approach this problem.

Damping
Damping can be activated in dynamic implicit analysis using VDC. A problem with contact damping in implicit is that the time step is usually large enough to not resolve the time in contact to get the desired damping effect. Therefore, it is not recommended to use damping.

Troubleshooting Convergence Problems

Convergence of the nonlinear equilibrium iteration process presents one of the greatest challenges to using the implicit mode of LS-DYNA. At risk of repeating what has already been mentioned, below are some useful troubleshooting approaches.

Eigenvalue Analysis

Many convergence problems in static implicit analysis are caused by unconstrained rigid body modes. These are created when an insufficient number of constraints are applied to the model, or when individual model parts are left disconnected. Eigenvalue analysis is an excellent diagnostic tool to check for these problems, both initially and at critical points in the simulation. The procedure for performing an eigenvalue analysis was discussed above.

D3Iter Plot Database

To diagnose convergence trouble which develops in the middle of a simulation, get a picture of the deformed mesh. Adjust the d3plot output interval to produce an output state after every step leading up to the problematic time. An additional binary plot database named d3iter is available which shows the deformed mesh during each equilibrium iteration. This output is activated by

D3ITCTL = 1 to activate d3iter plot database

on *CONTROL_IMPLICIT_SOLUTION. View this database using LS-PrePost to detect abnormal displacements. The problem may become obvious, especially as deformation is magnified. If not, there is yet another flag to activate to get the residual forces into both this database as well as d3plot for fringing. Setting

RESPLT = 1 to get residual data to binary databases

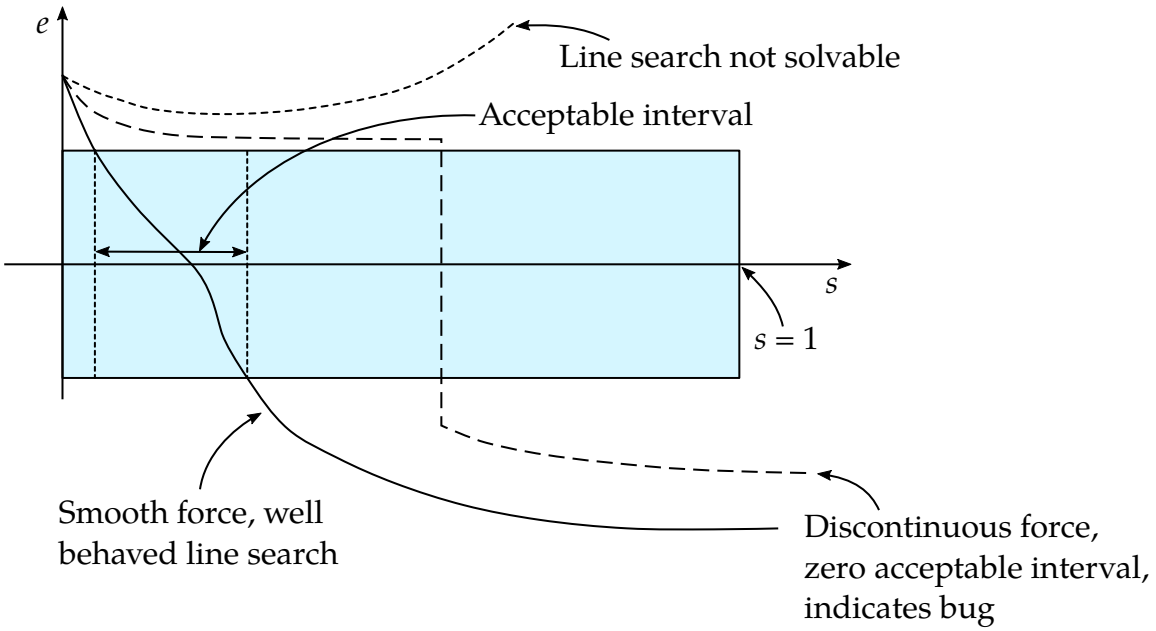
on *DATABASE_EXTENT_BINARY will do just that. With this option the residual forces are output to the d3plot and d3iter databases for fringing under the *NdV* menu. This is a great tool for locating areas in the model where the residual forces are not being reduced to a satisfactory level and take appropriate actions.

Taking Advantage of ASCII Information

If requested through NLPRINT = 3 on *CONTROL_IMPLICIT_SOLUTION and MINFO = 1 on *CONTROL_OUTPUT, a lot can be drawn from the information in the message file(s). This may be considered as a piece of advanced topic but may become useful in due time.

Residual Norm

Starting with the nonlinear output diagnostics, the following basic principle should be held in mind; *if the residual norm is zero, the problem is solved*. So it all comes down to make this number (11) small enough to trust the results. Therefore, we suggest monitoring the residual norm and interpreting it as an indicator of whether the convergence characteristics is good. Hopefully you would see this number decrease with implicit iterations and finally reach a relatively small number at convergence. Although it may (and will) increase on occasion, the trend should be a decreasing one. If this is a problem from the get-go, and you checked the model integrity through eigenvalue analysis and common practice, it may be that a feature is not properly supported in implicit.



Line Search

Another thing to look at is the line search convergence (9), as the relatively loose line search convergence tolerance should render a reasonably small number of line search iterations. A rule of thumb would be less than 10, at most 20. If more is used, or if the line search does not converge, there may be something in the model causing a jump in the residual forces. This is not supposed to happen in implicit and may suggest a bug.

APPENDIX P

Either of the two following scenarios for a feature discontinuity (and many line search iterations) is possible

- The line search step size becomes ridiculously small, and the current residual norm (7) is not approaching the initial one (3).
- The interval in which the optimal step is to be found (8) becomes ridiculously small, and the residual norm on the left and right interval points are not approaching each other. This situation is depicted in the figure above (dashed line).

Many line search iterations can of course be due to high nonlinearities but if the above is observed it should probably be investigated, for instance by trying to identify model features that may be causing the bad behavior. If it is obvious that a discontinuity exist, like if the step size goes down to ~1^{-100} or the interval size becomes zero, and it is not due to a model error, please send a bug report.

Stretching for Convergence
If the problem runs fine for a significant number of load/time steps with subsequent convergence issues, then the information received up to this point may be used intelligently and suggest a different (unorthodox) implicit strategy. To justify the approach, we begin by recalling that the convergence is detected by the following numbers becoming small; $d = \|\Delta \boldsymbol{u}\|$ (displacement norm), $r = \|\boldsymbol{R}\|$ (residual norm) and $e = \left|\boldsymbol{R}^T \Delta \boldsymbol{u}\right|$ (energy norm). But all these numbers are linked through the linear system of equations, $\boldsymbol{K}\Delta \boldsymbol{u} = \boldsymbol{R}$, so if $\boldsymbol{K}$ reasonably "nice" at all times it doesn't matter which numbers we use for detecting convergence. A mathematical way to state this is; if the condition number of $\boldsymbol{K}$ is "good," then all the norms are equivalent throughout the solution process. An intuitive statement would be to say that the problem is only moderately nonlinear, and convergence is usually never a problem. *But*, what happens if the properties of $\boldsymbol{K}$ are not that nice, or if $\boldsymbol{K}$ shifts character every time it is reformed? This is sort of saying that the problem is highly nonlinear, for instance due to frequent change of contact state or onsets/offsets of plasticity. In those cases the equivalence between d and r is lost, and typically an oscillatory behavior of the displacement norm is observed *even though the residual norm is reducing*, which in turn may lead to potential convergence problems just because the displacement criterion cannot be satisfied. Or equally bad, it could lead to premature convergence just by the coincidence that the displacement norm happens to become small. In those cases we essentially want to come up with a strategy where the displacement criterion is taken out of the convergence check and instead detect convergence based on residual only.

So assume we have say 10 converged states, after which the convergence problems begin. Then we can look at what the residual norm is for each of these converged states (11). For instance, we may see the sequence

Step 1, residual norm 2859
Step 2, residual norm 1581

Step 3, residual norm 2119
Step 4, residual norm 2511
Step 5, residual norm 11570
Step 6, residual norm 4904
Step 7, residual norm 3586
Step 8, residual norm 3157
Step 9, residual norm 3315
Step 10, residual norm 2825.

For each of these steps we may validate the results, by post-processing contact force curves, checking force/moment balance etc., in LS-PrePost, and usually there is a correlation between "good results" and small residual norms. Step 5 above is for instance an outlier and may be associated with a prematurely converged step, something that can be confirmed or rejected from looking at the result. The goal with these observations is to establish a reasonable residual norm for which we can safely say that the results are good, and the numbers above indicates that 3000 may be a good candidate. The strategy is then to simply put DCTOL = 10^{-16} to ignore the displacement and set ABSTOL = -3000, which means we can be assured that convergence will not be detected until the residual norm is below 3000. We "know" from having learned about the problem that this will yield good results. It should be mentioned that this may not work if the character (force level) changes significantly later on in the simulation, as 3000 then may not be the number we would trust.

Contacts
Continuing with the contact output, a general rule is that there should be *no* warnings of large penetrations and preferably the maximum relative penetration should be below 90%. These numbers can be monitored after each load/time step. If large penetrations occur, either increase IGAP or contact stiffness, whatever makes most sense, but first make sure that the converged step is *really* converged (no premature convergence) by monitoring the residual forces and checking results as indicated above.

Checklist

So, to summarize

- Activate implicit by setting IMFLAG = 1 on *CONTROL_IMPLICIT_GENERAL
    - Set DT0 to a reasonable initial time step
- Check possible singularities in an eigenvalue analysis by requesting NEIG eigenvalues on *CONTROL_IMPLICIT_EIGENVALUE
    - Find a way to constrain spurious modes

- Initially use dynamic analysis if rigid body modes are present in a static problem by setting IMASS = 1 on *CONTROL_IMPLICIT_DYNAMICS
    - Use TDYBIR, TDYDTH and TDYBUR
    - Use numerical damping by putting GAMMA = 0.6 and BETA = 0.38
- Only activate geometric stiffness contribution as a last resort, except for arc length methods
- Use default BFGS parameters ILIMIT and MAXREF on *CONTROL_IMPLICIT_-SOLUTION
    - Increase or decrease ILIMIT based on convergence characteristics
    - Use full Newton (ILIMIT = 1) for hard problems
    - Keep MAXREF to a reasonably low number
- Use default convergence tolerances DCTOL, ECTOL and RCTOL on *CONTROL_IMPLICIT_SOLUTION
    - ABSTOL may be set to 10^{-20} to prevent premature convergence
    - Relaxing DCTOL may not necessarily result in better convergence
    - Future strategy may be to focus on residual
- Activate automatic time stepping on *CONTROL_IMPLICIT_AUTO
    - Set DTMAX to a number that resolves features of interest
        - Contacts
        - Material nonlinearities
        - Frequencies
- Use default line search method on *CONTROL_IMPLICIT_SOLUTION
    - Switch to LSMTD = 5 if hard problem (typically rubbers)
    - Don’t change LSTOL
- Use at least NLPRINT = 2 on *CONTROL_IMPLICIT_SOLUTION to get convergence diagnostics into log files
    - Use NLPRINT = 3 to thoroughly track the residual norm and monitor line search behavior if debugging model is necessary
- Use *CONTROL_IMPLICIT_SOLVER only in special occasions
- Set D3ITCTL = 1 on *CONTROL_IMPLICIT_SOLUTION to get a database with Newton iterates when debugging a model

    - Complement this with RESPLOT = 1 on *DATABASE_EXTENT_BINARY to get the possibility to fringe the residual force vector

- For forming Mortar contact, make sure

    - Tools are oriented towards the blank
    - Contact on top and bottom of blank are separated among contact interfaces

- Use part (set) definitions of SURFA and SURFB in Mortar contact
- Always use weak part as SURFA (tracked) in a Mortar contact definition to get best possible convergence
- Set PENMAX to a reasonable characteristic length when the SURFA side consists of solid elements
- If penetrations are large, activate penetration diagnostics on *CONTROL_CONTACT
- To avoid release of contact pairs, either

    - Increase stiffness scaling factor SFSA
    - Increase IGAP for progressive stiffness increase
    - Increase PENMAX for solids while at the same time increasing SFSA

- Use IGNORE appropriately to deal with initial penetrations

    - Check initial penetrations in messag file

- Don’t use contact damping

APPENDIX Q: User Defined Weld Failure

The addition of a user weld failure subroutine into LS-DYNA is relatively simple. The UWELDFAIL subroutine is called every time step when OPT = 2 is specified in MAT_-SPOTWELD. As data, the identification number for the spotweld material, six constants specified in the input by thfe locations NRR through MTT, the radius of the cross section of the spotwelds, the current time, and the current values of the resultants for the spotwelds, which are stored in array STRR, are passed to the subroutine. The subroutine loops over the welds from LFT through LLT, and sets the values of the failure flag array FLAG.

```
      SUBROUTINE UWELDFAIL(IDWELD,STRR,FAIL,FIBL,CM,TT,LFT,LLT)
C*************************************************************
C|  LIVERMORE SOFTWARE TECHNOLOGY CORPORATION  (LSTC)          |
C|  ------------------------------------------------------    |
C|  COPYRIGHT 2002 JOHN O. HALLQUIST, LSTC                     |
C|  ALL RIGHTS RESERVED                                        |
C*************************************************************
C
C***  SPOTWELD FAILURE ROUTINE
C
C***  LOCAL COORDINATES: X IS TANGENT TO BEAM, Y & Z ARE NORMAL
C
C***  VARIABLES
C          IDWELD ---- WELD ID NUMBER
C          STRR ------ STRESS RESULTANTS
C                      (1) AXIAL (X DIRECTION) FORCE
C                      (2) Y SHEAR FORCE
C                      (3) Z SHEAR FORCE
C                      (4) MOMENT ABOUT Z
C                      (5) MOMENT ABOUT Y
C                      (6) TORSIONAL RESULTANT
C          FAIL ------ FAILURE FLAG
C                      = 0 NOT FAILED
C                      = 1 FAIL ELEMENT
C          FIBL ------ LOCATION (1,*) GIVES THE SPOTWELD DIAMETER
C          CM -------- 6 CONSTANTS SUPPLIED BY USER
C          TT -------- CURRENT SIMULATION TIME
C          LFT,LLT --- DO-LOOP RANGE FOR STRR
C
      DIMENSION IDWELD(*),STRR(6,*),FAIL(*),CM(*),FIBL(5,*)
C
C
      RETURN
      END
```

APPENDIX R: User Defined Cohesive Model

The addition of a user cohesive material subroutine into LS-DYNA is relatively simple. The UMATiC subroutine is called every time step where i ranges from 41 to 50. Input for the material model follows the *MAT_USER_DEFINED_MATERIAL definition. The user has the option of providing either a scalar or vectorized subroutine. As discussed in the Remarks for the user-defined material, the first two material parameters are reserved to specify how the density is treated and the number of integration points required for the failure of the element.

The cohesive model calculates the tractions on the mid-surface of the element as a function of the differences of the displacements and velocities of the upper (defined by nodes 5-6-7-8) and lower surfaces (defined by nodes 1-2-3-4). The displacements, velocities, and the calculated tractions are in the local coordinate system of the element, where the first two components of the vectors are in the plane of the mid-surface and the third component is normal to the mid-surface.

A stiffness must also be calculated by the user for the explicit time step calculation in LS-DYNA. This stiffness must provide an upper bound on the stiffness in all three directions.

The material fails at an integration point when `ifail=.true`. For an element to be deleted from the calculation, the number of integration points specified by the second material parameter must fail. If the second parameter is zero, elements cannot fail regardless of the specification of IFAIL in the user-defined material input. For example, the user may choose to reject an implicit step is the displacement increment is too

For implicit analysis, the subroutine is called with `maketan=.true.` and the user must provide the elastic moduli in the three local directions in the respective diagonal terms of the `dsave` array. The parameter `reject`, if set to `.true.` by the user, will signal to LS-DYNA that the current implicit iteration is unacceptable. For example, the user may choose to reject an implicit step if the traction changes too much from the last time step. In this situation, LS-DYNA will print a warning message `Material model rejected current iterate' and retry the step with a smaller time step. If chosen carefully (by way of experimenting), this may result in a good trade-off between the number of implicit iterations per step and the step size for overall speed.

The following example is a vectorized model with two elastic constants and failure:

```
      subroutine umat41c(idpart,cm,lft,llt,fc,dx,dxdt,aux,ek,
     & ifail,dt1siz,crv,nnpcrv,nhxbwp,cma,maketan,dsave,ctmp,elsiz,
     & reject,ip,nip)
      include 'nlqparm'
c
c***  vector cohesive material user model example
c
```

APPENDIX R

```
c***  variables
c          idpart ---- Part ID
c          cm -------- material constants
c          lft,llt --- start and end of block
c          fc -------- components of the cohesive force
c          dx -------- components of the displacement
c          dxdt ------ components of the velocity
c          aux ------- history storage
c          ek -------- max.  stiffness/area for time step calculation
c          ifail ----- =.false.  not failed
c                      =.true.  failed
c          dt1siz ---- time step size
c          crv ------- curve array
c          nnpcrv ---- # points per curve for crv array
c          nhxbwp ---- internal element id array, lqfinv(nhxbwp(i),2)
c                      gives external element id
c          cma ------- additional memory for material data defined by
c                      LMCA in 2nd card, 6th field of *MAT_USER_DEFINED
c          maketan --- true for implicit
c          dsave ----- material stiffness array, define for implicit
c          ctmp ------ current temperature
c          elsiz ----- characteristic element size (=sqrt(area))
c          reject ---- set to .true.  if this implicit iterate is
c                      to be rejected for some reason (implicit only)
c          ip -------- integration point number
c          nip ------- total number of integration points
c
c***  dx, dxdt, and fc are in the local coordinate system:
c     components 1 and 2 are in the plane of the cohesive surface
c     component 3 is normal to the plane
c
c***  cm storage convention
c     (1) =0 density is per area
c         =1 density is per volume
c     (2) number of integration points for element deletion
c         =0 no deletion
c     (3:48) material model constants
c
      logical ifail,maketan,reject
      dimension cm(*),fc(nlq,*),dx(nlq,*),dxdt(nlq,*),
     &          aux(nlq,*),ek(*),ifail(*),dt1siz(*),crv(101,2,*),
     &          nhxbwp(*),cma(*),dsave(nlq,6,*),ctmp(*),elsiz(*)
      integer nnpcrv(*)
c
c
      et=cm(3)
      en=cm(4)
      eki=max(et,en)
      fcfail=cm(5)
c
      do i=lft,llt
        fc(i,1)=et*dx(i,1)
        fc(i,2)=et*dx(i,2)
        fc(i,3)=en*dx(i,3)
        ek(i)=eki
        ifail(i)=fc(i,3).gt.fcfail
      enddo
c
      if(maketan) then
        do i=lft,llt
          dsave(i,1,1)=et
          dsave(i,2,1)=0.
          dsave(i,3,1)=0.
          dsave(i,1,2)=0.
          dsave(i,2,2)=et
          dsave(i,3,2)=0.
```

```
          dsave(i,1,3)=0.
          dsave(i,2,3)=0.
          dsave(i,3,3)=en
        enddo
      endif
      return
      end
```

The second example implements the Tveergard-Hutchinson cohesive model with failure in both the vectorized (UMAT42C) and scalar (UMAT43C) forms. Note the LFT and LLT are passed to the scalar version, however their value is zero.

```
      subroutine umat42c(idpart,params,lft,llt,fTraction,jump_u,dxdt,
     & aux,ek,ifail,dt1siz,crv,nnpcrv,nhxbwp,cma,maketan,dsave,ctmp,elsiz,
     & reject,ip,nip)
      include 'nlqparm'
c
c***  vector cohesive material user model example
c
c     Tveergard-Hutchinson model based on:
c     tahoe/src/elements/cohesive_surface/cohesive_models/TvergHutch3DT.cpp
c
c     the declaration below is processed by the C preprocessor and
c     is real*4 or real*8 depending on whether LS-DYNA is single or double
c     precision
c
      REAL L,jump_u
      logical ifail,maketan,reject
      dimension params(*),fTraction(nlq,*),jump_u(nlq,*),dxdt(nlq,*),
     &          aux(nlq,*),ek(*),ifail(*),dt1siz(*),crv(101,2,*),
     &          nhxbwp(*),cma(*),dsave(nlq,6,*),ctmp(*),elsiz(*)
      integer nnpcrv(*)
c
      fsigma_max=params(3)
      fd_c_n=params(4)
      fd_c_t=params(5)
      fL_1=params(6)
      fL_2=params(7)
      fpenalty=params(8)
c
      fK=fpenalty*fsigma_max/(fL_1*fd_c_n)
c
      fac=min(fd_c_n/fd_c_t**2,1./fd_c_n)
c
      do i=lft,llt
      u_t1 = jump_u(i,1)
      u_t2 = jump_u(i,2)
      u_n = jump_u(i,3)
c
      r_t1 = u_t1/fd_c_t
      r_t2 = u_t2/fd_c_t
      r_n = u_n/fd_c_n
      L = sqrt(r_t1*r_t1 + r_t2*r_t2 + r_n*r_n)
c
      if (L .lt.  fL_1) then
         sigbyL=fsigma_max/fL_1
      else if (L .lt.  fL_2) then
         sigbyL = fsigma_max/L
```

```
      else if (L .lt.  1.) then
         sigbyL = fsigma_max*(1. - L)/(1. - fL_2)/L
      else
         sigbyL = 0.0
         ifail(i)=.true.
      endif
c
      fTraction(i,1) = sigbyL*r_t1*(fd_c_n/fd_c_t)
      fTraction(i,2) = sigbyL*r_t2*(fd_c_n/fd_c_t)
      fTraction(i,3) = sigbyL*r_n
c
c     penetration
      if (u_n .lt.  0) fTraction(i,3)=fTraction(i,3)+fK*u_n
c
c     approximate stiffness for time step
      if (u_n .lt.  0) then
        ek(i)=fac*sigbyL+fK
      else
        ek(i)=fac*sigbyL
      endif
c
      if (maketan) then
        dsave(i,1,1)=sigbyL/fd_c_t*(fd_c_n/fd_c_t)
        dsave(i,2,1)=0.
        dsave(i,3,1)=0.
        dsave(i,1,2)=0.
        dsave(i,2,2)=sigbyL/fd_c_t*(fd_c_n/fd_c_t)
        dsave(i,3,2)=0.
        dsave(i,1,3)=0.
        dsave(i,2,3)=0.
        dsave(i,3,3)=sigbyL/fd_c_n
        if (u_n.lt.0) dsave(i,3,3)=dsave(i,3,3)+fk
      endif
      enddo
c
      return
      end
c
c
c
      subroutine umat43c(idpart,params,lft,llt,fTraction,jump_u,dxdt,
     & aux,ek,ifail,dt1siz,crv,nnpcrv,nhxbwp,cma,maketan,dsave,ctmp,elsiz,
     & reject,ip,nip)
c
c***  scalar cohesive material user model example
c
c     Tveergard-Hutchinson model based on:
c     tahoe/src/elements/cohesive_surface/cohesive_models/TvergHutch3DT.cpp
c
c     the declaration below is processed by the C preprocessor and
c     is real*4 or real*8 depending on whether LS-DYNA is single or double
c     precision
c
      REAL L,jump_u
      logical ifail,maketan,reject
      dimension params(*),fTraction(nlq,*),jump_u(nlq,*),dxdt(nlq,*),
     &          aux(nlq,*),ek(*),ifail(*),dt1siz(*),crv(101,2,*),
     &          nhxbwp(*),cma(*),dsave(nlq,6,*),ctmp(*),elsiz(*)
      integer nnpcrv(*)
c
      fsigma_max=params(3)
      fd_c_n=params(4)
      fd_c_t=params(5)
      fL_1=params(6)
      fL_2=params(7)
      fpenalty=params(8)
```

```
c
      fK=fpenalty*fsigma_max/(fL_1*fd_c_n)
c
      fac=min(fd_c_n/fd_c_t**2,1./fd_c_n)
c
      u_t1 = jump_u(1)
      u_t2 = jump_u(2)
      u_n = jump_u(3)
c
      r_t1 = u_t1/fd_c_t
      r_t2 = u_t2/fd_c_t
      r_n = u_n/fd_c_n
      L = sqrt(r_t1*r_t1 + r_t2*r_t2 + r_n*r_n)
c
      if (L .lt.  fL_1) then
         sigbyL=fsigma_max/fL_1
      else if (L .lt.  fL_2) then
         sigbyL = fsigma_max/L
      else if (L .lt.  1.) then
         sigbyL = fsigma_max*(1. - L)/(1. - fL_2)/L
      else
         sigbyL = 0.0

      ifail=.true.
      endif
c
      fTraction(1) = sigbyL*r_t1*(fd_c_n/fd_c_t)
      fTraction(2) = sigbyL*r_t2*(fd_c_n/fd_c_t)
      fTraction(3) = sigbyL*r_n
c
c     penetration
      if (u_n .lt.  0) fTraction(3)=fTraction(3)+fK*u_n
c
c     approximate stiffness for time step
      if (u_n .lt.  0) then
        ek=fac*sigbyL+fK
      else
        ek=fac*sigbyL
      endif
c
      if (maketan) then
        dsave(1,1)=sigbyL/fd_c_t*(fd_c_n/fd_c_t)
        dsave(2,1)=0.
        dsave(3,1)=0.
        dsave(1,2)=0.
        dsave(2,2)=sigbyL/fd_c_t*(fd_c_n/fd_c_t)
        dsave(3,2)=0.
        dsave(1,3)=0.
        dsave(2,3)=0.
        dsave(3,3)=sigbyL/fd_c_n
        if (u_n.lt.0) dsave(3,3)=dsave(3,3)+fk
      endif
      return
      end
```

APPENDIX S: User Defined Boundary Flux

A user defined boundary flux interface is provided in LS-DYNA where it is possible to define the thermal heat flux (power per surface area) in or out of a surface segment as an arbitrary function of temperature and history. The user may associate history variables with each individual flux interface and also use load curves.

The user flux interface is invoked using the keyword *BOUNDARY_FLUX_*OPTION.* This is accomplished with the parameter NHISV. When it is defined with a value greater than 0, the user subroutine

```
subroutine usrflux(fl,flp,…)
```

is called to compute the flux (`fl`) defined as heat (energy) per time and per surface area.

Other parameters that are passed to the user flux subroutine include the segment nodal temperatures at the previous (T_0) and current time (T_1), the segment nodal coordinates and the time integration parameter α. Also, the current thermal simulation time t, the time step Δt and average segment temperature (T_α) at time $t+\alpha\Delta t$ is provided together with the curve array for accessing defined load curves in the keyword input file. For computing load curve values, note that load curve IDs need to be transformed to internal numbers or the subroutine crvval should be used, see the appendix on user defined materials for details.

The segment coordinates available in the subroutine are such that the outward normal vector follows the well-known right-hand rule, thus segments corresponding to the lower surface of thick thermal shells are reversed before passed to the subroutine. For shells in general, the segment connectivity should follow the connectivity of the actual shell element to avoid problems.

Optionally, the user may define the derivative of the flux `fl` with respect to the average segment temperature (T_α) at time $t+\alpha\Delta t$, `flp`. This value is used in the nonlinear thermal solver for assembling the correct stiffness matrix and must be set by the user. If possible, it is recommended to use a value that reflects the nonlinearity of the flux model, otherwise the value 0 should be used.

An array of history variables, identical with the input parameters defined in the keyword input file, are passed to the subroutine that can be updated with time or kept constant throughout the simulation. An example of usage would be to integrate the flux with time to keep track of the dissipated energy per surface area in order to simulate the effects of spray cooling in hot-stamping.

```
      subroutine usrflux(fl,flp,x,tnpl,tnl,nodes,
```

```
     .     alpha,atime,atemp,dt,time,fhsv,nfhsv,crv)
C***********************************************************************
C|  LIVERMORE SOFTWARE TECHNOLOGY CORPORATION  (LSTC)                   |
C|  ------------------------------------------------------------------  |
C|  COPYRIGHT © 2007 JOHN O. HALLQUIST, LSTC                            |
C|  ALL RIGHTS RESERVED                                                 |
C***********************************************************************
c
c     User subroutine for boundary thermal flux
c
c     Purpose:  To define thermal flux parameter (heat per surface area and
c      time)
c
c     Variables:
c
c     fl              = flux intensity (output)
c     flp             = flux intensity derivative wrt atemp (output)
c     x(3,nodes)      = global segment coordinates (input)
c     tnpl(nodes)     = temperatures at time time (input)
c     tnl(nodes)      = temperatures at time time-dt (input)
c     nodes           = number of nodes in segment (3,4 or 6) (input)
c     alpha           = time integration parameter (input)
c     atime           = time+(alpha-1)*dt
c     atemp           = average segment temperature at time atime
c     dt              = time step size (input)
c     time            = time at which the new temperature is sought (input)
c     fhsv(nfhsv)     = flux history variables (input/output)
c     nfhsv           = number of flux history variables for this segment
c      (input)
c     crv             = curve array (input)
c
      include 'nlqparm'
      dimension x(3,*),tnpl(*),tnl(*)
      dimension fhsv(*),crv(lq1,2,*)
c
c     Define flux by linear convection
c     that optionally decays (in an ad-hoc way) as power
c     dissipates from surface
c
c     fhsv(1) = convection coefficient
c     fhsv(2) = ambient temperature
c     fhsv(3) = total amount of energy per surface area available
c     fhsv(4) = dissipated energy per surface area at current
c
      hcon=fhsv(1)
      tinf=fhsv(2)
      flin=hcon*(tinf-atemp)
      if (nfhsv.gt.2) then
         q=(1.-fhsv(4)/fhsv(3))/
     .        (1.+.5*dt*flin/fhsv(3))
         flp=-q*hcon
         if (q.gt.1.) then
            q=1.
            flp=-hcon
         elseif (q.lt.0.) then
            q=0.
            flp=0.
         endif
         fl=q*flin
         fhsv(4)=fhsv(4)+dt*.5*fl
         fhsv(4)=min(fhsv(3),fhsv(4))
      else
         fl=flin
         flp=-hcon
      endif
c
```

```
return
end
```

APPENDIX T: Metal Forming Glossary

A TYPICAL DRAW DIE ENGINEERING PROCESS

Clay models of a new vehicle are scanned and the outer shell surfaces are created in a design studio. Body-in-white engineers and designers are responsible to create all the inner parts and various structure and underbody parts. Flanges are created on outer surface parts to be joined with the inner panels. These parts are typically created in the car axis; with global X-axis runs from the front to the back along the car's center line, global Y-axis from the driver side to the passenger side and Z-axis going straight up.

During the clay design and shaping, simultaneous and multidisciplinary engineering may be practiced involving divisions/departments from design, engineering and manufacturing. Material suppliers may also be involved in this stage for consultation if advanced materials are to be applied. Multidisciplinary involvement in this early stage of a vehicle development allows manufacturing engineers to capture any part designs with "no make" conditions that would be costly if they were allowed to proceed to a later stage; while design envelopes can be pushed to their maximum potential within the state-of-art manufacturing capabilities.

During the advance feasibility phase, parts from the design studio are processed quickly to go through an engineering process involving mainly the process engineer and FEA simulation engineer. Addendum and binder are roughly built in order to conduct a reliable draw die simulation. The exact process eventually would be used to build the die may not yet be established, therefore similar processes from knowledge base are used as references. Rarely are secondary dies (all dies except draw die) simulated. The main task here is to provide some quick assessment of the part's manufacturability through fast design/engineering iterations.

During the hard die design and construction phase, stamping manufacturing processes (some in three dimensional) are established, by referring to existing knowledge base, with necessary modifications needed for the current part design, and with the limits of manufacturing equipment such as press type, shut height, maximum tonnage and automation, etc. The stamping process includes the number of dies (to make a complete part of the final shape), part tipping, draw height, trimming (direct or aerial), flanging, springback compensation requirements, etc. Not all areas of the part may be formed to its final shape in one draw die. Some involve redraw die if one area of the part is especially deep compared to the rest of the part, which may otherwise cause uncontrollable wrinkles, splits, or exceed the draw height limit without the redrawing. Some areas of the part (which may have "undercut" or "die locked" conditions) may be unfolded to the addendum, then trimmed and flanged in a flanging die later. A typical example of such can be found along the hood line of a fender outer. There may be multiple trimming and flanging dies, since not all areas of the drawn panel may be feasible for trimming or flanging all in one trim or flanging die.

*APPENDIX T

Referring to Figure 70-1, a typical draw die development process flow for an outer part is illustrated. The part is tipped from the global car axis to a 'draw die' axis, which takes into account of balancing the internal draw angles over the entire part, and minimizing the overall draw height, etc. Hem flanges are unfolded off the part breakline (Figure 70-1) in one of the following ways:

1)Tangent extension of the part surface,
2)Horizontal surface,
3)Vertical down-standing surface,
4)Any angled surface between horizontal and vertical surface,
5)To the addendum surfaces to be designed; in this case, the unfolding will happen after the addendum design is complete.

Flange unfold must take into account the trimming condition later in the trim dies. Direct trim represents the trim steel going down in a draw direction (vertical); while aerial (cam) trim steel covers all other directions, driven by a 'cam driver' driven further by the vertical downward motion of the trim die. There are specific trim angle requirements for direct and aerial trim operations. Next, the part boundary is smoothed, filling up any gaps, holes and sharp features. The boundary will likely be modified later during the addendum build. Binder is created, based on the unfolded part boundary and overall part curvature. A developable binder surface, which consists of planes, cylindrical surfaces or a combination of both, is preferred; however, in some cases a doubly curved binder surface (undevelopable surface) must be designed for the purpose of reducing the draw depth at critical locations for material utilization, alleviating thinning, or both. Theoretical punch opening line (P.O. line, Detail #2 in Figure 70-1) can be offset from the smoothed part boundary in the plane normal to the draw axis, projected to the binder surface in the draw direction; some smoothing of the P.O. lines may be required. Generally, the finished P.O. lines should be consisted of mostly straight lines and radii, with generous transition among corners. P.O. lines do not follow tight corners, avoiding formation of wrinkles during the draw. In addition, design of P.O. lines must take into consideration the material utilization issue, especially in the blank sizing critical locations. Once P.O. lines design is complete, binder surfaces inside of the P.O. are trimmed and the remaining binder surface design is sent for binder closing simulation. Given a blank initial shape, simulation of the binder closing action can determine the quality of the binder design. Typically, for exterior outer panels, no buckles or wrinkles are allowed during the closing; for inner panels, some wrinkles are acceptable, or even desirable, depending on the part shape. Lower punch/post support may be introduced to reduce the draw height, alleviate thinning, or remove the initial blank drape into the binder cavity. Based on the binder closing simulation results, unacceptable binder design is sent back for rework, and the satisfactory binder design proceeds into the next step of addendum build, which is used to fill the space between the part boundary and binder surface. It is noted that the P.O. lines may need to be adjusted during the addendum design. The trimming condition may also be affected and modified during the addendum design. Next, draw simulation ensues, assessing the overall formablity of the draw design. If quality targets are not achieved in the simulation, the process may go

all the way back to the tipping for redesign/reprocess; otherwise successful simulation will direct the draw surface design to the next stage of draw die structure design.

TYPES OF DRAW DIES

There are many different types of draw dies used to punch the part to their intended shape, as shown in Figure 70-2. They basically can be divided into either single action or double action dies. Specifically (names may vary among different companies),

1)Air draw – Single action. It is a 3-piece die system with 1 piece upper (cavity) and 2-piece (binder and punch/post) lowers. The upper cavity (driven by press ram) moves down in one action to close with the lower binder, and then closes with the lower punch to draw the part to the home position. The lower binder is either sitting on an air cushion through pins that go through the press bed, or directly on nitrogen cylinders arranged uniformly between the bottom of the binder structure and the press bed; the punch is fixed onto the press bed. This is the most popular draw type, mainly because of its speed and efficiency. Its limitation includes a maximum draw height of 10".

2)Toggle draw – Double action. It is a 3-piece die system with 2-piece upper (binder and punch) and 1 piece (cavity) lower. Upper binder, driven by the outer ram of the press, moves down to clamp the blank with the lower cavity; then the upper punch, driven down by the inner ram, closes with the lower cavity to complete the draw. Since this adds another action, it is slower than the air draw; however, this type of draw die is well suited to control the wrinkles created during the forming of difficult part, such as liftgate and door inner. Furthermore, this type of draw has a relatively large draw height, which is limited only by the press.

3)Air draw with pressure pad – Single action. This is very similar to 1), except an additional pressure pad, driven by nitrogen cylinders mounted on the upper die structure, closes first with the lower punch, then the entire upper comes down to finish the draw. This is similar to 1) in efficiency.

4)Stretch Draw (four piece) – Double action. It is a 4-piece die system with 2-piece uppers (upper binder and punch) and 2-piece lowers (lower binder and cavity). Upper binder moves down to close with lower binder, moving together for a certain distance (up to 2"), then upper punch comes down to completely close with lower cavity. Finally the binders move down together to their home position. This process is not used as often as 1), 2) and 3), however, it is very capable in forming difficult inner parts, especially those prone for wrinkles, such as liftgate inner, door inner and floor pan. Since there is a 'pre-stretch' action with the binders clamping the blank and moving down together, strain path change in the part during the forming is expected. This is the slowest draw type.

5)Crash Form Die – Single action. It is a 2-piece die system with no upper or binders. Upper and lower tool takes the same shape, with upper moving down to close with the lower tool. This is obviously a very simple die, which can handle simple

parts, with not too much draw depth variation (near constant draw depth all around).

TYPES OF FLANGING DIES

There are three types of flanging dies, as shown in Figure 70-3. All three types have a fixed lower (trim) post upon which the drawn (or partially trimmed part) is sitting, and a pressure pad (or multiple pads) which holds the part (which is loaded onto the post in a vertical direction) in place against the post. In a direct flanging process, the flanging steel (Figure 70-3) moves vertically down to 'wipe' or 'bend' the part to its flanged position. In an aerial flanging process, the flanging steel moves to form the flanges in an angle rather than the vertical direction. The steel is held by the cam slide, driven by a cam driver which in turn is driven by the trim die's downward movement. Since the finished flange forms a 'die lock' condition, meaning the flanged part will not be able to be lifted (retract) out of the trim die into the next die (or station), the filler cam (Figure) is moved horizontally out of the way so the part can be lifted up and out. Once the part is removed, the filler cam moves back into its home position ready for the next drawn panel to be loaded. In the rotary cam flanging process, the filler cam is called a 'rotor', which rotates out of the way for the flanged part to be lifted up and out. Some parts of the rotary cam flanging design are a patented process. In comparison to the conventional cam flanging, rotary flanging has the advantage of being very compact.

TYPES OF HEMMING DIES

There are two types of hemming dies, as shown in Figure 70-4. Both processes have a fixed hemming bed, on which the flanged outer part is loaded; the inner panel is then loaded onto the outer panel; proper clamps are applied to hold tight the panels together and in place. In press (or table top) hemming, a pre-hemming tool is moved to form the flange into a halfway position, followed by the final hemming tool pressing down on the flange against the inner and outer panels to its final position. Many different shapes of hem tips can be achieved, as shown in the figure. In roller hemming, a pre-roller moves in a three dimensional curvilinear path following the hem tip, to form the flange partially. This is followed by a final roller, moving in another three dimensional path, to finish the hem shape. In the hemming of complex or high-end parts, many passes (rollers) are needed to achieve high quality hem surfaces. Similarly, with the design of different roller shapes, many shapes of hem tips can be achieved.

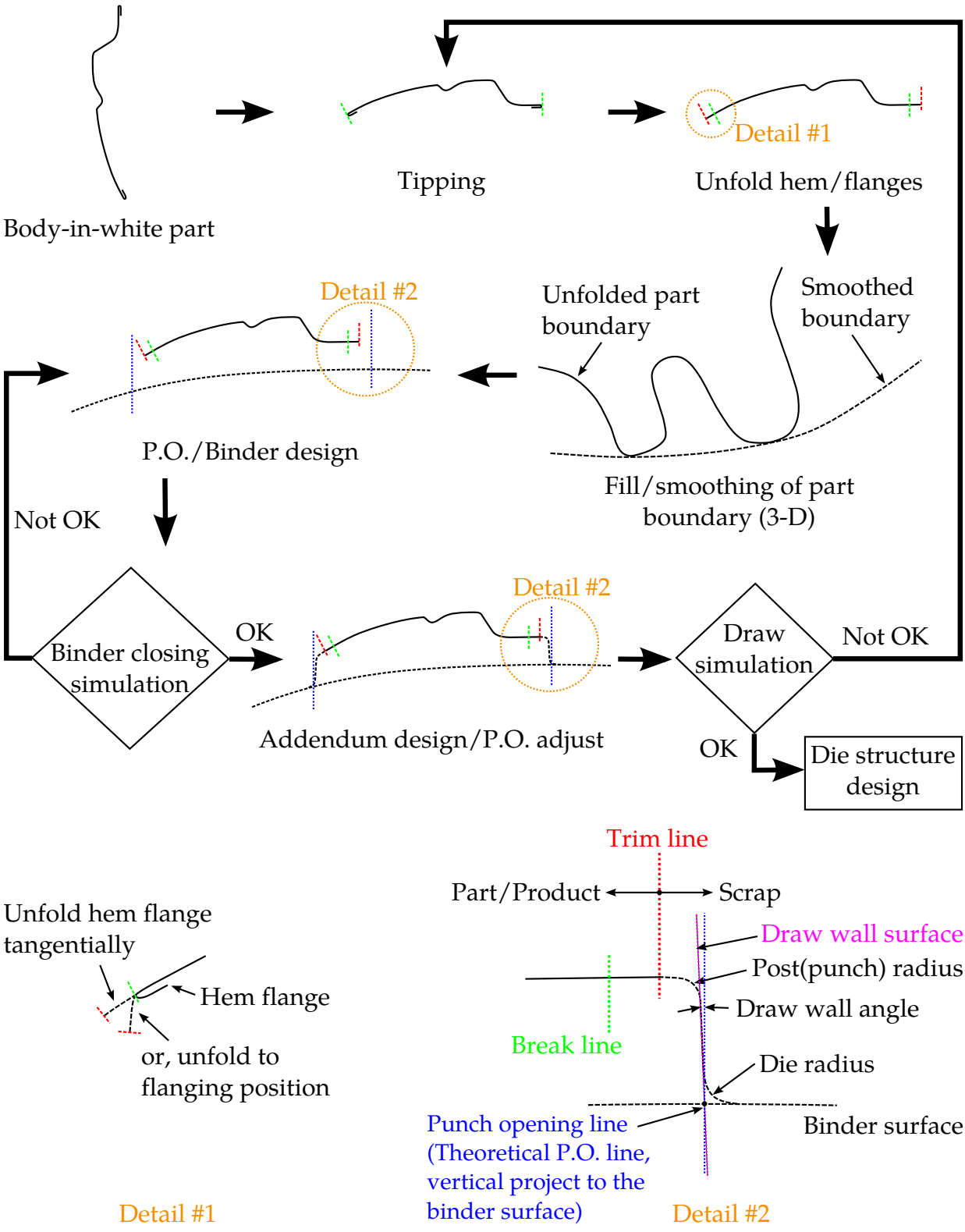
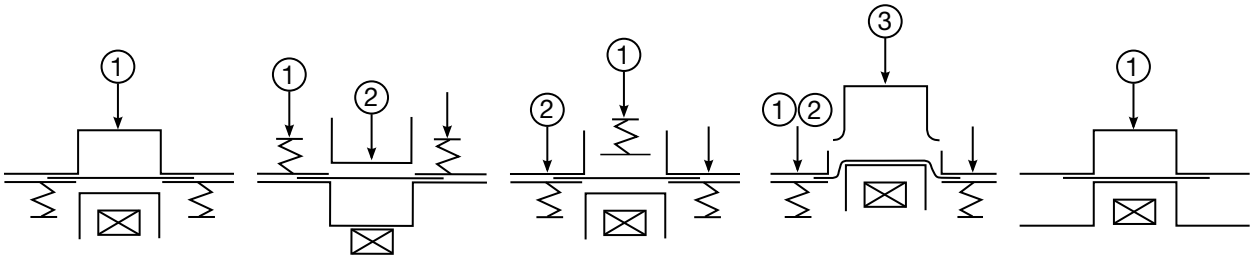
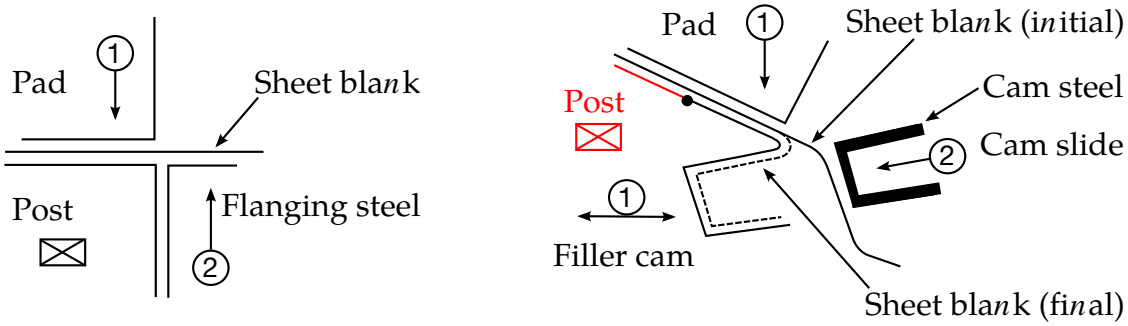


Figure 70-1. A typical draw die engineering process



+ : single action
++ : double action

Figure 70-2. Types of draw dies



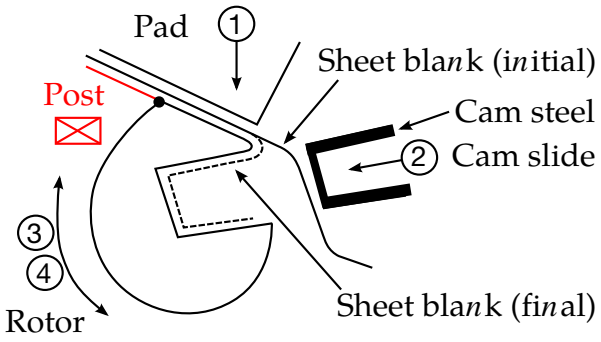


Figure 70-3. Types of flanging dies

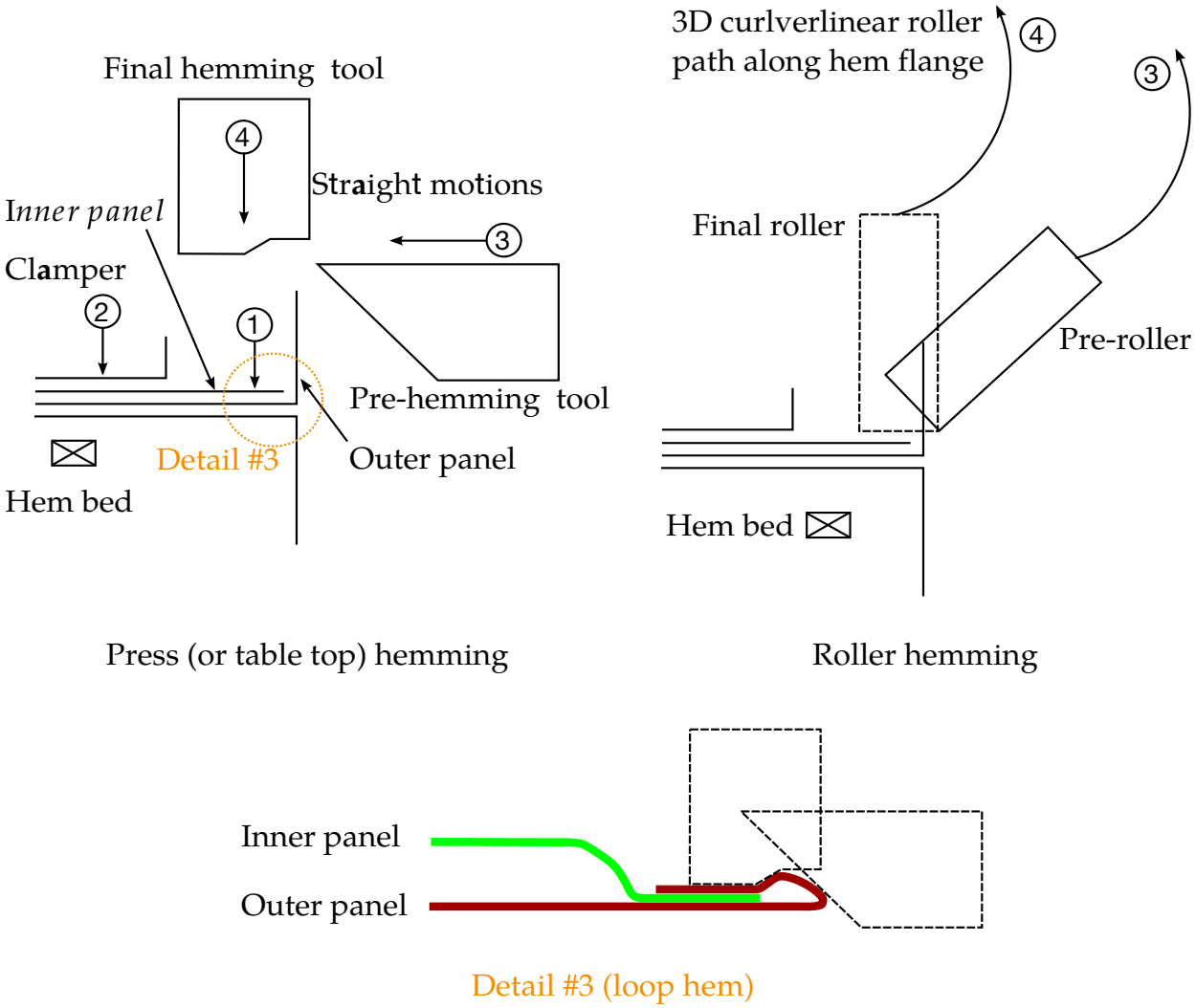


Figure 70-4. Types of hemming dies

APPENDIX U: Reserved Variable Names

This appendix lists variable names that are reserved internally by LS-DYNA.

LS-DYNA Functions:

ACCM
ACCX
ACCY
ACCZ
AKISPL
ARYVAL
AX
AY
AZ
BEAM
BISTOP
BUSH
CHEBY
CUBSPL
CURVE
CVCV
CX
CY
CZ
DIF
DIF1
DM
DMRB
DX
DXRB
DY
DYRB
DZ
DZRB
ELHIST
FIELD
FM
FORCOS
FORSIN
FX
FY
FZ
GFORCE
HAVSIN
IF
IMPACT
JOINT
JPRIM
MOTION
NFORCE
PHI
PINVAL
PITCH
POLY
POUVAL
PSI
PTCV
RCFORC
ROLL
RX
RY
RZ
SENSOR
SFORCE
SHF
SPDP
STEP
STEPL
TEMP
THETA
TM
TX
TY
TZ
VARVAL
VFORCE
VM
VR
VTORQ
VX
VY
VZ
WDTM
WDTX
WDTY
WDTZ
WM
WX
WY
WZ
YAW

Types:

char
cstring
double
float
generic
int
MatrixNxM (N, M = 1, 2, 3, …)
pointer
VectorN (N = 1, 2, 3, …)

*APPENDIX U

Constants:

DTOR	NULL	RTOD
MODE	PI	

System:

abort()	inv()	rx()
argc	isalnum()	ry()
argv[]	isalpha()	rz()
atof()	isascii()	SEEK_CUR
atoi()	iscntrl()	SEEK_END
become_daemon()	isdefined()	SEEK_SET
ceil()	isdigit()	setdebug()
CharArray()	isgraph()	setdefinedstring()
clearerr()	islower()	sizeof()
cross()	isprint()	sleep()
DataCopy()	ispunct()	sprint()
DataCopyTo()	isspace()	sprintf()
die()	isupper()	stderr
dovariablesubstitution()	isxdigit()	stdin
EOR	length()	stdout
exit()	logoutput()	strcasecmp()
fclose()	malloc()	strcat()
fdprint()	memcpy()	strcmp()
feof()	memmove()	strcpy()
ferror()	memset()	strdup()
fgetc()	New()	strdupf()
fgets()	NewArray()	string
FILE	notify()	strlen()
floor()	out()	strncasecmp()
fopen()	print()	strncat()
fprintf()	printf()	strncmp()
fputs()	PrintGlobals()	strncpy()
fread()	printsp()	strrchr()
free()	putenv()	strtok()
fseek()	ReadFileField()	tr()
fwrite()	remove()	unlink()
getdefinedstring()	rewind()	whatis()
getenv()	rot()	

Error Handling:

clearerrno()	E2BIG	EACCES

EADDRINUSE
EADDRNOTAVAIL
EAFNOSUPPORT
EAGAIN
EALREADY
EBADF
EBADMSG
EBUSY
ECANCELED
ECHILD
ECONNABORTED
ECONNRESET
EDEADLK
EDESTADDRREQ
EDOM
EDQUOT
EEXIST
EFAULT
EFBIG
EHOSTNOTFOUND
EHOSTUNREACH
EIDRM
EILSEQ
EINPROGRESS
EINTR
EINVAL
EIO
EISCONN
EISDIR
ELOOP
EMFILE
EMLINK
EMSGSIZE
EMULTIHOP
ENAMETOOLONG
ENETDOWN
ENETRESET
ENETUNREACH
ENFILE
ENOADDRESS
ENOBUFS
ENODATA
ENODEV
ENOENT
ENOEXEC
ENOLCK
ENOLINK
ENOMEM
ENOMSG
ENOPROTOOPT
ENOSPC
ENOSR
ENOSTR
ENOSYS
ENOTCONN
ENOTDIR
ENOTEMPTY
ENOTSOCK
ENOTSUP
ENOTTY
ENXIO
EOK
EOPNOTSUPP
EOVERFLOW
EPERM
EPIPE
EPROTO
EPROTONOSUPPORT
EPROTOTYPE
ERANGE
EROFS
ESPIPE
ESTALE
ETIME
ETXTBSY
EWOULDBLOCK
EXDEV
geterror()
geterrstr()
perror()
seterror()
strerror()
sys_err()

Regular Expressions:

chomp()
compilepattern()
gsubst()
research()
search()
subst()

Binary File Library:

access()
bufferopen()
close()
F_OK
fileclear()
filestrerror()
ftruncate()
linediscipline()
lseek()
MAX_FD
MSG_PEEK
O_APPEND
O_CREAT
O_EXCL
O_RDONLY
O_RDWR
O_TRUNC
O_WRONLY
open()
R_OK
read()
readline()
readlinetest()
stat
truncate()
W_OK
write()

*APPENDIX U

X_OK

Processes/Signals:

chdir()
chmod()
ctime()
execute()
fgetpid()
getcwd()
getpid()
kill()
mkdir()
PROC_BYLINE
PROC_MONITOR
PROC_NOCONSOLE
PROC_NOWAIT
PROC_PIPEIN
PROC_PIPEOUT
PROC_REDIR
PROC_REDIRIN
PROC_TERMINAL
PROC_TERMINALIN
PROC_TERMINALOUT
rename()
setfiletimes()
SIG_DFL
SIG_IGN
SIGABRT
SIGALRM
SIGBUS
SIGCHLD
SIGCONT
SIGFPE
SIGHUP
SIGILL
SIGINT
SIGIO
SIGKILL
signal()
SIGPIPE
SIGPWR
SIGQUIT
SIGSEGV
SIGSTOP
SIGSYS
SIGTRAP
SIGTSTP
SIGTTIN
SIGTTOU
SIGURG
SIGUSR1
SIGUSR2
SIGWINCH
system()
time()
touch()
wait()
which()

Sockets (probably not enabled):

accept()
closemonitor()
drain()
fd_set
fdset()
fdunset()
fdzero()
gethostbyname()
gethostname()
hostent
ipsocket()
isset()
localaddr()
localport()
monitor()
readtest()
remoteaddr()
remoteport()
select()
socket_blocking()
socket_reuseaddr()
tcpconnect()
tcpserver()
tcpsocket()
timeval

XDR Routines:

xdr_buf
xdr_bytes
xdr_data
xdr_data1
xdr_float
xdr_int
xdr_readargs
xdr_status
xdr_writeargs
xdrfree()
xdrgetbytes()
xdrgetdata()
xdrgetfloat()
xdrgetint()
xdrgetstat()
xdrgetstatus()
xdrlinkdata()
xdrlinkgetbytes()
xdrlinkgetdata()
xdrlinkgetfloat()
xdrlinkgetint()
xdrlinkgetstat()
xdrlinkgetstatus()
xdrlinkputbytes()

xdrlinkputfloat()
xdrlinkputint()
xdrlinkrecv()
xdrlinksend()
xdrputbytes()
xdrputdata()
xdrputfloat()
xdrputint()
xdrsave()
xdrsend()

Remote Functions:

clientloop()
currentlink;
ftp()
ftpchdir()
ftpdebug()
ftpfilesize()
ftpget()
ftplist()
ftpmkdir()
ftpmode()
ftpmodtime()
ftpput()
ftppwd()
ftpquit()
ftpreply()
ftprmdir
ftpsetcounter()
ftptimeout()
ftpunlink()
LINK_COMMAND
LINK_STDERR
LINK_STDOUT
linkaccept()
linkbegin()
linkclose()
linkcommand()
linkcommandf()
linkend()
linkinfo()
linkinput()
linkopen()
linkoutput()
linkoutputhandler()
linkread()
linkstderr()
linkstdout()
linkwrite()
outputtolink()
raccess()
rcopyfile()
remote_exec
rexecute()
rfstat()
ropen()
rremove()
rrename()
rstat()
rsystem()
runlink()
serverloop()
telnet()
telnetdebug()
telnetkill()
telnettimeout()

Appendix V: How to Read Card Summaries

LST has written the LS-DYNA manual an encyclopedic reference for LS-DYNA. The first goal of the manual is to fully document all features and their corresponding input structures. Secondly, we have structured the manual so that any chunk of keyword input can be quickly interpreted. As of R11 we have introduced a new standard format aimed at making the manual even more useful for *quickly* interpreting keyword input. The definitions below, which are provided for clarity, are followed by a lengthy discussion of this new format.

Definitions: Decks, Cards, and Keywords

1. **Decks.** The text file containing the LS-DYNA input is called *the input deck.* The term *deck* is historical in origin, originally referring to actual punch-cards.

2. **Cards.** Each line of the input file is called a *card*, as in a punch-*card.* Taken together the *cards* comprise the *deck*.

3. **Keywords.** A typical input deck consists of two kinds of cards: (1) keyword cards and (2) data cards. Keyword cards are easily recognized by the "*"-character in the first column followed directly by a (case insensitive) *keyword*. These cards form the skeleton of the deck. LS-DYNA treats each keyword card as a kind of command (or function or subroutine) for which the arguments are passed with data cards.

4. **Keyword Options.** Each keyword card consists of a base keyword combined with option words. Option words are appended to the base keyword using underscores. For example, to invoke the *node* keyword with the *merge* option one would include a card (a line) consisting of the text "*NODE_MERGE" where the "*"-character is in the first column. We have, for clarity, adopted a convention of capitalizing (the case insensitive) keywords.

Introduction to Card Summaries

The keyword manual (the enormous PDF you're looking at) is organized by keyword. As mentioned in item 3 above, keywords are like subroutines (or functions, if you program in C). Like subroutines, keywords take input. Unlike subroutines, keyword input is not divided into a fixed number of strongly-typed fields. For instance, for a given keyword the number of fields and their data types can, and often does, depend on what is contained in the input itself. Keyword input is, then, best described by flow chart.

APPENDIX V

After processing a keyword card all cards until the next keyword card are treated as the keyword's input. Collectively, these cards are called the *data cards*. Most of the LS-DYNA manual is devoted to describing how each keyword interprets the contents of its data cards (the flowchart mentioned in the previous paragraph). *Beginning in 2018 (R11), new and updated entries to the manual will contain a section called "card summary" that expresses the structure of the keyword's data in a concise and standardized format*. We hope that these summaries will be simple enough that you can understand them without reading this document.

Motivation for Introducing the Card Summary Section.

1. **"Optional" Cards/Blank Cards.** Traditionally, some data cards have been called *optional* and others have been called *required*. This can be confusing. Some examples of frequently asked questions are:

   Can the second optional card be omitted? The answer is: not if you intend to include the third optional card.

   Can I add blank lines to make my deck more human-readable? The answer is "no" because LS-DYNA interprets blank lines as data. For instance, each data card after a *NODE is expected to contain a node definition. A blank card is interpreted as an invalid node definition.

2. **Branching (and Switch Constructs).** Often the format of the next data cards depends on what was in the preceding cards or on the options applied to the keyword. Sometimes the branching can be complicated. We have adopted a compact convention to express branching in the *card summaries*.

3. **Repeated Cards (DO/FOR Loops Constructs).** There has been confusion over the meaning of the *card number* as presented in traditional manual entries. In some cases, a data card's card number indicates its position relative to the keyword; in other cases, the card number serves as a kind of format identifier. Iterated cards, however, make the line-number interpretation impossible without saying something like *Card n*, where *n* is an integer, and we almost never do that. Card summaries clear up this confusion.

4. **Default Values.** When a field is blank or set to 0, then LS-DYNA applies a default value. The meaning of "0" for objects like node sets can be confusing and, in the past, hasn't been explicitly spelled out. As part of the format revamp we are adopting new conventions for ambiguous cases.

"Optional" Cards (How LS-DYNA Parses Blank Lines).

When LS-DYNA reads an input deck some data cards contain fields having no default value. These data cards are said to be *required*; LS-DYNA will error terminate when required cards are missing. In contrast, data cards consisting entirely of fields that have default values can, generally, be left blank. Such cards are called *optional.*

The details of the algorithm for processing optional cards are subtle, but the application of optional cards is nonetheless mostly intuitive. An *optional card* is only optional in the sense that if *all the cards* that come after it are also optional, then the card can be omitted, but *only* if *all* the subsequent cards *are also omitted*. For example, if a keyword has three optional data cards and some of the values on the third optional card need to be changed from their default values, then the two previous optional cards must be included, at the very least, as blank lines. Note that there are a very small number of irregular cases where an omitted card is treated differently from a blank card; such special cases are documented when they appear.

Example:

The example below illustrates some of the subtleties involved in deciding when to omit an optional card. In this example, two rigid wall cylinders are defined with a single instance of the *RIGIDWALL_GEOMETRIC_CYLINDER_ID_DISPLAY_MOTION keyword by including two sets of data cards, one for each cylinder. The DISPLAY option activates an optional data card. Since two cylinders are being defined, the DISPLAY data card must be included, at least as a blank line, for the first cylinder; otherwise, LS-DYNA will read the first line for the second cylinder as new defaults for the DISPLAY option data card causing an error termination due to format mismatch. The last card included in the input below for the second cylinder is *not* the DISPLAY data card. For this case, it does not have to be input since it is optional and no further data cards follow. A blank line may be included for the second DISPLAY data card, but is not necessary, and excluding it makes no difference.

```
      ⋮
*RIGIDWALL_GEOMETRIC_CYLINDER_ID_DISPLAY_MOTION
$ first cylinder
$       ID
       100
$     nsid    nsidex     boxid     birth     death
                 0         0     0.000     0.000
$       xt        yt        zt        xh        yh        zh      fric
-20.000000     0.000-50.000000-21.000000     0.000-50.000000     0.000
$   radcyl    lencyl
 20.000000 30.000000
$     lcid       opt        vx        vy        vz
        99         0     0.000  1.000000     0.000
$
$ Blank line for card data associated with DISPLAY. This blank line is necessary since
$ 2 card sets are being used even though this card is optional.  If the blank line was
$ not included, then LS-DYNA would read 101 as the DISPLAY data card line which would
$ lead to an error termination
$

$ second cylinder
```

APPENDIX V

```
101
                    0          0     0.000     0.000
-20.000000     0.000 10.000000-21.000000     0.000 10.000000     0.000
 20.000000 30.000000
         2                0.000  1.000000     0.000
$ No blank line is needed here since this is the last card set and the DISPLAY data
$ card is optional.
*control_contact
      ⋮
```

The Structure of Manual Entries.

The following table compares traditional entries to summary table formatted entries:

Old Format (no summary table)		Summary Table Format	
1.	The entry begins with a "purpose" statement	1.	No change.
2.	The purpose statement is followed by an explanation of options, if there are any. Sometimes the purpose statement and option sections are reversed.	2.	No change.
3.		3.	Summary table
4.	Several small tables of between 2 and 4 rows are included, roughly, one for each type of data card. Each table contains a short note about how many times its repeated.	4 & 5.	Each card table is followed directly by a short variable list. Generally, each branching is represented by a *different* card table.
5.	These tables are followed by a single long "variable list table," which defines the meaning of all the fields on all of the preceding tables. Branching is mostly explained in the variable list.		
6.	Lastly, manual entries often contain a "remarks" section.	6.	No change.

As indicated in the above table, the differences between the two formats are confined to items 3, 4, and 5. The traditional format, when applied to keywords with large data sets suffers a few drawbacks which the summary table format addresses:

1. For keywords having many data cards (> 5) the variable lists may be several pages removed from the original data card. In the summary table format a data card's fields are explained right after the data card.

2. The entire set of data card tables, itself, may span several pages. Ideally, it should be on one page so that a user can look at a single page when trying to interpret his input deck. The card summary table is compact (and more likely to fit on one page) and contains enough information to match each field in an input deck to the name of the field in the manual.

3. In cases where the meaning of a data card's field depends on previous data cards, the traditional format puts the branching explanations entirely in the variable list. Interpreting an input deck then requires lots of page flipping. Moreover, since groups of fields often branch together, the same branching verbiage is often repeated in field after field requiring you to read each-and-every repetition in case one field works differently from the others, which sometimes happens.

   In the card summary format branching is explained in the summary table and sometimes repeated on the card representations. Since the fields are explained after the card, the branching is *not* generally dealt with in the variable lists.

Summary Table Format

The summary table itself consists of two rows per data card (see Example 1 directly below). The first row consists of pseudo-code containing an identifier for the card type, the logic used to determine when a data card is included, and the logic used to determine how many times it is repeated. The second row represents the data card's fields following the convention from the traditional manual entry.

The summary tables will, for the most part, adhere to the conventions described below.

1. Cards are labeled as "Card [identifier]". In the simplest case, the identifier is an integer. See Example 1.

2. In the case of a group of mutually exclusive cards (or sets of cards), the variants are expressed by adding a letter onto the identifier. Letters increment alphabetically. See Example 2.

3. Cards that are grouped together (in contrast to ones that are mutually exclusive) are indicated by adding a "*.n*" onto the base identifier, where *n* is the card number with the first card in the group being numbered "1". This construct is analogous to increasing indention in python or nesting with curly braces in C. See Example 3.

4. Cards with italicized names are optional. See Example 2.

APPENDIX V

Example 1.

The data cards for the *RIGIDWALL_PLANAR keyword have neither exclusive branches or conditional sets. The cards are, therefore, numbered linearly. To the extent there is branching, it is only to include extra data cards as options are appended to the keyword.

ID Card. First data card of the card set if the ID keyword option is used in the keyword name. Otherwise Card 1 is the first data card.

ID							

Card 1. This card is required.

NSID	NSIDEX	BOXID	OFFSET	BIRTH	DEATH	RWKSF	

Card 2. This card is required.

XT	YT	ZT	XH	YH	ZH	FRIC	WVEL

Card 3. This card is included only if the ORTHO option is used.

SFRICA	SFRICB	DFRICA	DFRICB	DECAYA	DECAYB		

Card 4. This card is included only if the ORTHO option is used.

NODE1	NODE2	D1	D2	D3			

Card 5. This card is included only if the FRICTION option is used.

XHEV	YHEV	ZHEV	LENL	LENM			

Card 6. This card is included only if the MOTION option is used.

MASS	V0						

Card 7. This card is included only if the FORCES option is used.

SOFT	SSID	N1	N2	N3	N4		

For the *RIGIDWALL_PLANAR_MOTION_FORCES_ID keyword the summary table is interpreted as

ID	ID							
1	NSID	NSIDEX	BOXID	OFFSET	BIRTH	DEATH	RWKSF	
2	XT	YT	ZT	XH	YH	ZH	FRIC	WVEL
6	MASS	V0						
7	SOFT	SSID	N1	N2	N3	N4		

For the *RIGIDWALL_PLANAR_ORTHO_FRICTION keyword the summary table is interpreted as

1	NSID	NSIDEX	BOXID	OFFSET	BIRTH	DEATH	RWKSF	
2	XT	YT	ZT	XH	YH	ZH	FRIC	WVEL
3	SFRICA	SFRICB	DFRICA	DFRICB	DECAYA	DECAYB		
4	NODE1	NODE2	D1	D2	D3			
5	XHEV	YHEV	ZHEV	LENL	LENM			

Example 2.

Card 1. This card is required.

NEIG	CENTER	LFLAG	LFTEND	RFLAG	RHTEND	EIGMTH	SHFSCL

Card 2. Card 2 and beyond are optional. If Card 3a or 3b are included, then Card 2 must be included; Card 2 can be a blank line in this case, so default values are used.

ISOLID	IBEAM	ISHELL	ITSHELL	MSTRES	EVDUMP	MSTRSCL	

Card 3a. Card 3a is read only when EIGMTH = 101. It is optional.

IPARM1	IPARM2	IPARM3	IPARM4	RPARM1			

Card 3b. Card 3b is read only when EIGMTH = 102. It is optional.

IPARM1	IPARM2			RPARM1	RPARM2		

APPENDIX V

Example 3.

ID Card. First data card of the card set if the ID keyword option is used in the keyword name. Otherwise Card 1 is the first data card.

RWID	HEADING

Card 1. This card is required.

NSID	NSIDEX	BOXID	BIRTH	DEATH			

Card 2. This card is required.

XT	YT	ZT	XH	YH	ZH	FRIC	

Card 3a. This card is included only for the FLAT shape.

XHEV	YHEV	ZHEV	LENL	LENM			

Card 3b. This card is included only for the PRISM shape.

XHEV	YHEV	ZHEV	LENL	LENM	LENP		

Card 3c. This card is included only for the CYLINDER shape.

RADCYL	LENCYL	NSEGS					

Card 3c.1. NSEGS instances of this card are included only for the CYLINDER shape.

VL	HEIGHT						

Card 3d. This card is included only for the SPHERE shape.

RADSPH							

Card 4. This card is required if the MOTION keyword option is used.

LCID	OPT	VX	VY	VZ			

Card 5. This card is read only if the DISPLAY keyword option is used. It is optional and may be omitted unless more than one card set is being read in; if more than one card set is being used, then at least a blank line must be included for all but the last card set. If not input, the defaults will be used.

PID	RO	E	PR	↓			

Default Values for Data Cards.

The keyword reader always, with a vanishingly small number of exceptions, fills in blank fields with the value 0. LS-DYNA, then replaces the value of 0 with *the default value*. The default value row of a keyword card table adheres to the following conventions:

1. Often, the row of default values is missing from the keyword card tables. In this case see the variable list following the table (or after the tables for traditional format) for information about default values.

2. A "↓" character indicates that the default value is explained in the variable list. This, likely, indicates that the explanation involves more content than the table can accommodate.

3. If the field is required, then the default value is written as *none*.

4. The symbol {∅} indicates an empty set.

5. "Rem *n*" where *n* is an integer indicates that the field is explained in a remark.

6. If the default value is listed as being 0, then one of two things is being described: (1) the default value is *actually* 0, or (2) the table is indicating the tautological, namely, that the keyword reader fills blank fields with zero, which always happens for every field. In the second case the meaning of zero is almost always explained within the variable list. Going forward, LST will try to avoid this second case.

Appendix W: Acoustic Volume Element and Spectral Element Solvers

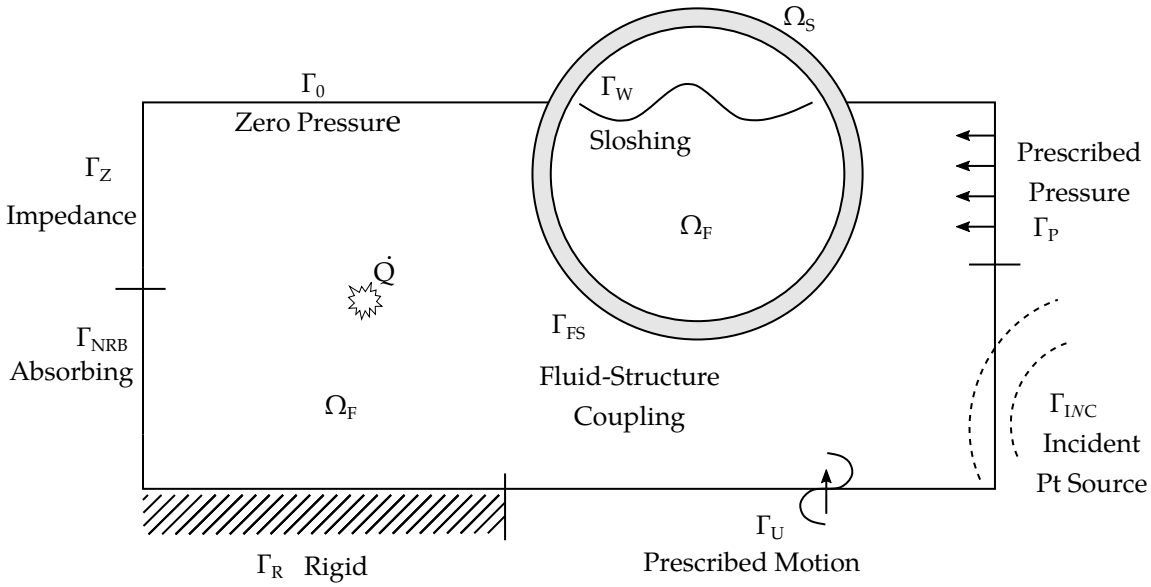


Figure 74-1. Sketch of the structural-acoustic system

Explicit Transient Acoustic Iso-parametric Element Solutions

The longstanding and default solution path for transient acoustic solutions in LS-DYNA uses low-order iso-parametric finite elements. It is capable of treating shock-induced cavitation and uses the displacement potential as the fundamental nodal unknown. This is the approach used if neither *CONTROL_IMPLICIT_SSD_DIRECT nor *CONTROL_ACOUSTIC_SPECTRAL is used.

Boundary condition treatment is summarized in Table 74-1. This solution path uses an inverted boundary condition strategy that may be confusing to analysts familiar with potential formulations. Kinematic boundary conditions are interpreted as if the formulation were a Lagrangian, displacement-based model like *MAT_ELASTIC. Thus, fluid faces that are constrained with nodal displacement constraints are rigid, sliding faces. Fluid faces that are left free are pressure release or zero-pressure faces. And fluid faces that are merged with structural nodes are coupled faces. Structural coupling when there is fluid on both sides of a shell element requires *BOUNDARY_ACOUSTIC_COUPLING or *BOUNDARY_ACOUSTIC_COUPLING_MISMATCH. Non-reflecting and absorbing boundaries use either *BOUNDARY_NON_REFLECTING or the perfectly matched layer of *MAT_PML_ACOUSTIC.

APPENDIX W

Table 74-1. Keywords for explicit transient acoustic iso-parametric finite element solutions

Model Parameters	Keywords
Material Properties, Ω_F	*MAT_ACOUSTIC
Structural Coupling, Γ_{FS}	Merge fluid and structural nodes on one side *BOUNDARY_ACOUSTIC_COUPLING *BOUNDARY_ACOUSTIC_COUPLING_MISMATCH
Prescribed Boundary Motion, Γ_U	No support
Prescribed Boundary Pressure, Γ_P	*LOAD_SEGMENT_SET
Rigid Boundary, Γ_R	*BOUNDARY_SPC
Impedance Boundary, Γ_Z	*BOUNDARY_ACOUSTIC_IMPEDANCE
Absorbing Boundary, Γ_{NRB}	*BOUNDARY_NON_REFLECTING *MAT_PML_ACOUSTIC
Zero Pressure Boundary, Γ_{NRB}	Leave these nodes unconstrained
Small Amplitude Wave Boundary, Γ_W	No support
Internal Point Source, $\dot{Q}$	No support

Explicit Transient Acoustic Spectral Element Solutions

The spectral acoustic elements in LS-DYNA invoked with *CONTROL_ACOUSTIC_SPECTRAL use higher-order interpolants based on Legendre polynomials. The elements converge exponentially and support integration orders from 2 to 15. They are especially recommended for ultrasonic wave propagation. These elements have internal degrees-of-freedom (invisible to users). These degrees-of-freedom are generated automatically based on the integration order chosen. You may change the integration order and accuracy of the solution without remeshing.

For the explicit solution to the transient, fluid-structure interaction problem with acoustic spectral elements, the following nonsymmetric system is solved at every time step:

$$\begin{bmatrix} M_{ss} & 0 \\ -\rho c^2 T_{fs}^T & M_{ff} \end{bmatrix} \begin{bmatrix} \ddot{u} \\ \ddot{p} \end{bmatrix} + \begin{bmatrix} W_{ss} & 0 \\ 0 & W_{ff} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{p} \end{bmatrix} + \begin{bmatrix} K_{ss} & T_{fs} \\ 0 & K_{ff} \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} r_s \\ r_f \end{bmatrix}$$

Here M_{ss}, W_{ss}, and K_{ss} are the structural mass, damping, and stiffness. M_{ff}, W_{ff}, and K_{ff} are the equivalent matrices for the fluid. T_{fs} is the fluid-structure coupling matrix. Vector u is the structural displacements, and p is the pressure. The time step used is the smallest stable time step in either of the acoustic or structural domains. The applicable keywords are summarized in Table 74-2.

Table 74-2. Keywords for explicit transient acoustic spectral element solutions

Model Parameters	Keywords
Material Properties, Ω_F	*MAT_ACOUSTIC *MAT_ACOUSTIC_DAMP
Structural Coupling, Γ_{FS}	*BOUNDARY_ACOUSTIC_COUPLING_-SPECTRAL
Weak Structural Coupling, Γ_{FS}	*INTERFACE_ACOUSTIC *BOUNDARY_ACOUSTIC_INTERFACE
Prescribed Boundary Motion, Γ_U	*BOUNDARY_ACOUSTIC_PRESCRIBED_-MOTION
Prescribed Boundary Pressure, Γ_P	*BOUNDARY_ACOUSTIC_PRESSURE_SPEC-TRAL
Rigid Boundary, Γ_R	This is a natural condition
Impedance Boundary, Γ_Z	*BOUNDARY_ACOUSTIC_IMPEDANCE
Absorbing Boundary, Γ_{NRB}	*BOUNDARY_ACOUSTIC_NON_REFLECT-ING
Zero Pressure Boundary, Γ_{NRB}	*BOUNDARY_ACOUSTIC_FREE_SURFACE
Small Amplitude Wave Boundary, Γ_W	*BOUNDARY_ACOUSTIC_FREE_SURFACE
Internal Point Source, $\dot{Q}$	*LOAD_ACOUSTIC_SOURCE

The perfectly matched layer of *MAT_ACOUSTIC_PML is not currently supported in spectral acoustic element solutions.

For post-processing, pressure-time histories are written with *DATABASE_ACE-OUT and *DATABASE_HISTORY_ACOUSTIC. Plot states for the user's mesh are written to the D3PLOT file. The integration point used for those plot states is derived from the closest spectral pressure degree-of-freedom.

Implicit Direct Steady State Acoustic FE Solutions

For the direct solution to the forced vibration problem with acoustic fluid coupling, a complex symmetric system of equations is solved. This starts with

$$\begin{bmatrix} M_{ss} & 0 \\ 0 & M_{ff} \end{bmatrix} \begin{bmatrix} \ddot{u} \\ \ddot{q} \end{bmatrix} + \begin{bmatrix} W_{ss} & T_{fs} \\ T_{fs}^T & W_{ff} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{q} \end{bmatrix} + \begin{bmatrix} K_{ss} & 0 \\ 0 & K_{ff} \end{bmatrix} \begin{bmatrix} u \\ q \end{bmatrix} = \begin{bmatrix} r_s \\ r_f \end{bmatrix}$$

Here M_{ss}, W_{ss}, and K_{ss} are the structural mass, damping and stiffness. M_{ff}, W_{ff}, and K_{ff} are the equivalent matrices for the fluid. T_{fs} is the fluid-structure coupling matrix. Vector u contains the displacements of structural nodes. Vector q contains the integral of the pressure, that is, the fluid nodal pressure equals $\dot{q}$.

APPENDIX W

Assuming $u = \overline{U_s}e^{i\omega t}$, $q = \overline{\Phi_f}e^{i\omega t}$, and $r_{f/s} = \overline{F_{f/s}}e^{i\omega t}$, the assembled system of equations becomes complex

$$\begin{bmatrix} -\omega^2 M_{ss} + i\omega W_{ss} + \overline{K_{ss}} & i\omega T_{fs} \\ i\omega T_{fs}^T & -\omega^2\overline{M_{ff}} + i\omega\overline{W_{ff}} + \overline{K_{ff}} \end{bmatrix} \begin{bmatrix} \overline{U_s} \\ \overline{\Phi_f} \end{bmatrix} = \begin{bmatrix} \overline{F_s} \\ F_f \end{bmatrix}$$

Submatrices K_{ss}, M_{ff}, W_{ff}, and K_{ff} may also be complex and may be frequency dependent. This is the system that is solved for every forcing frequency with *CONTROL_IMPLICIT_SSD_DIRECT.

Table 74-3. Keywords for implicit, direct steady-state acoustic finite element solutions

Model Parameters	Keywords
Material Properties, Ω_F	*MAT_ACOUSTIC *MAT_ACOUSTIC_DAMP *MAT_ACOUSTIC_COMPLEX *MAT_ACOUSTIC_POROUS_DB
Structural Coupling, Γ_{FS}	*BOUNDARY_ACOUSTIC_COUPLING_MISMATCH
Weak Structural Coupling, Γ_{FS}	*INTERFACE_ACOUSTIC *BOUNDARY_ACOUSTIC_INTERFACE
Prescribed Boundary Motion, Γ_U	*BOUNDARY_ACOUSTIC_PRESCRIBED_MOTION
Rigid Boundary, Γ_R	This is a natural condition
Impedance Boundary, Γ_Z	*BOUNDARY_ACOUSTIC_IMPEDANCE *BOUNDARY_ACOUSTIC_COMPLEX *BOUNDARY_ACOUSTIC_MECHANICAL
Absorbing Boundary, Γ_{NRB}	*BOUNDARY_ACOUSTIC_NON_REFLECTING
Zero Pressure Boundary, Γ_{NRB}	*BOUNDARY_ACOUSTIC_FREE_SURFACE
Small Amplitude Wave Boundary, Γ_W	*BOUNDARY_ACOUSTIC_FREE_SURFACE
Internal Point Source, $\dot{Q}$	*LOAD_ACOUSTIC_SOURCE
Incident Wave Point Source, Γ_{INC}	*LOAD_ACOUSTIC_SOURCE

For post-processing, real and imaginary nodal response is written with *DATABASE_ACEOUT and *DATABASE_HISTORY_ACOUSTIC.

Appendix X: Multistage Analysis

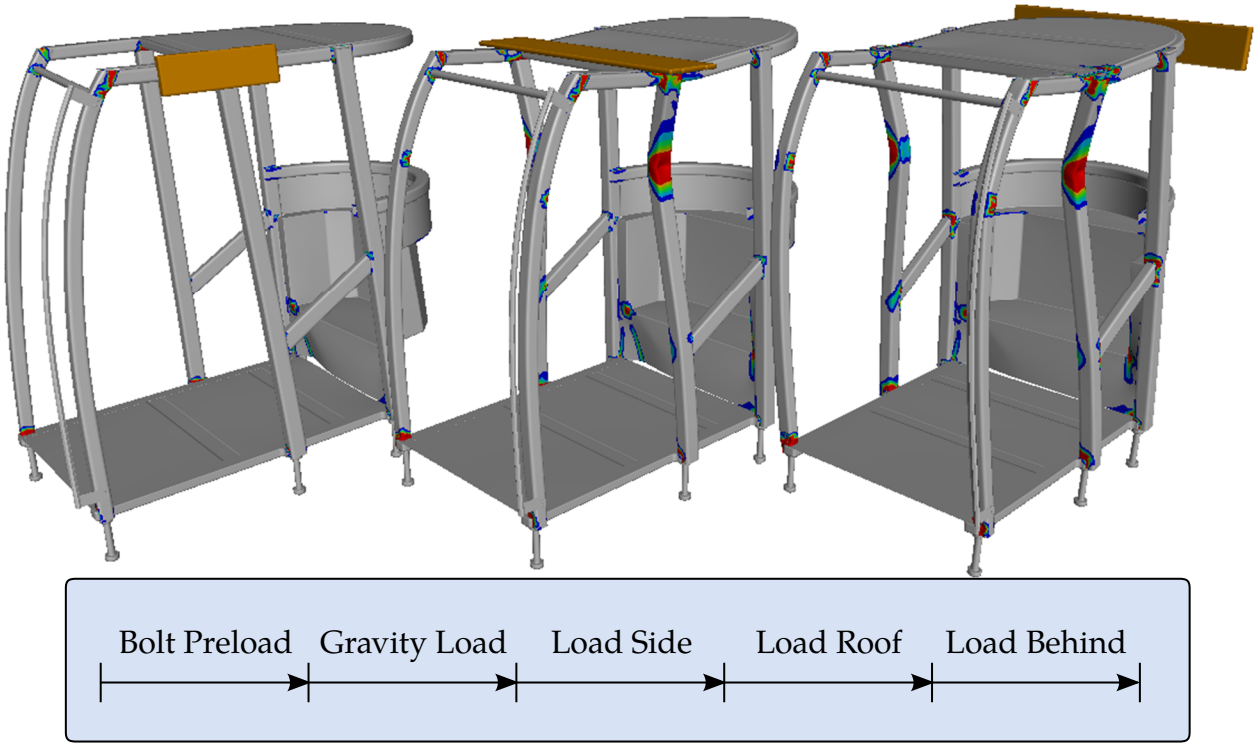


Figure 75-1. Example of Roll-Over Protection Systems simulation with Bolt Preload and Gravity Load

WHY USE STAGES

A typical LS-DYNA simulation is a boundary value problem with initial conditions. It is solved from an initial time 0 (zero) to an appropriate end time T. The initial configuration is usually assumed to be *stress free*, which basically means that all internal variables in all active features are trivially initialized. If the application is sufficiently complex, setting up the input can be cumbersome. The resulting simulation can also be difficult to tackle. In those situations, we can divide the entire problem into several *stages.* A stage is associated with an isolated and well-defined phase of the real physical process. In this sense, a stage naturally defines a *standard* analysis in the following ways:

- No birth or death time in any feature[2]
- Few and simple boundary conditions
- Only implicit or only explicit with no switching between

[2] The only exception is turning on and off dynamics in implicit, which is regarded as a stabilization technique for solving implicit static analyses. See Appendix P for details.

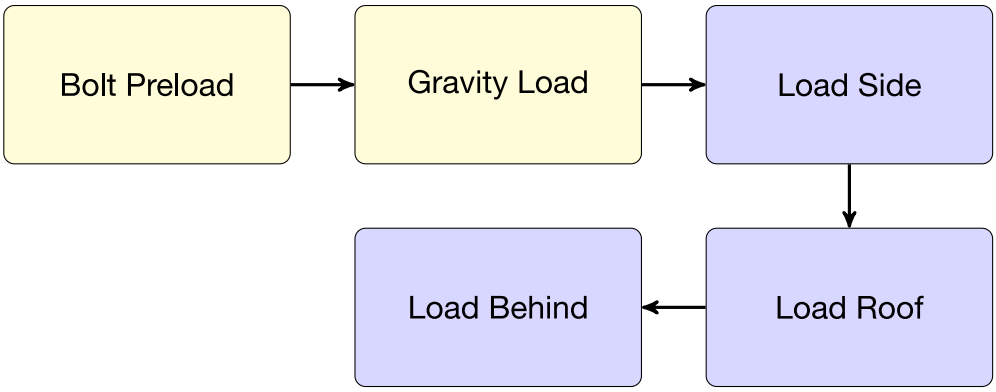


Figure 75-2. Flowchart of how to separate the ROPS test simulation into 5 stages

- Using well defined loadings with simple curves

Each stage is itself a boundary value problem with initial conditions. The only caveat is that the initial configuration may now be *prestressed* because of what occurred in the previous stage. In other words, internal variables must be propagated from the previous stage in order to make sense of future results. Apart from this, each stage can be executed and investigated separately instead of having to complete the entire simulation in a single run.

Example of Splitting into Stages:

To make things less abstract, consider the simple example of a hypothetical cab design subjected to a *ROPS (Roll-Over Protective Systems)* test, ISO requirement 3471, as depicted in Figure 75-1. The legal requirements state conditions for sequential quasi-static loads from the *side*, the *roof* and *behind*, which are graphically illustrated. To make things a little more interesting, we add *bolt preload* and *gravity load* before applying the external loads. Assume for the sake of simplicity that the entire process takes 10 seconds. As indicated, completing the entire simulation in one single run would require everything to be set up accordingly, with the potential risk of having to restart from the beginning in case of execution failure.

The alternative is to identify the 5 stages. Assuming each stage takes 2 seconds, you then set up a separate input for all of them. You run these input files sequentially in the order they apply. Figure 75-2 illustrates this method for setting up the simulation. For the moment we don't worry about the problem of inheriting the internal variables but instead think of what advantages this approach might bring:

- If the simulation fails, restart can be made from the last successful stage instead of rerunning the entire simulation. Thus, turnaround time decreases.
- Some stages may require only a subset of the entire model. Thus, this method reduces simulation time.

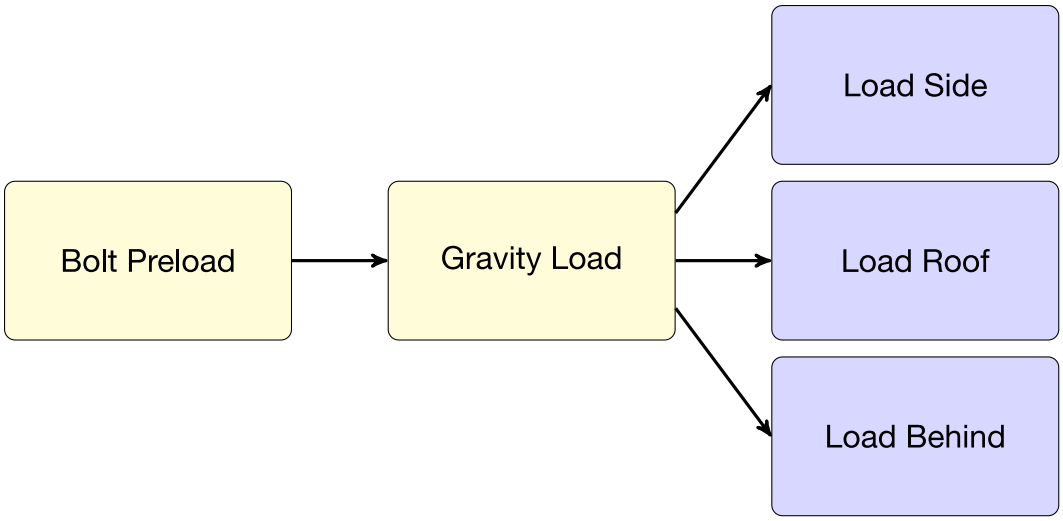


Figure 75-3. Flowchart of a multiple case staged analysis

- Some parts can be rigid during stages, saving simulation time.
- Data from a stage can be reused for many different types of simulations, such as different load cases and eigenvalue analyses.
- A prestressed component can be duplicated in subsequent stages.

So far, we only considered the situation where the stages are executed sequentially, and each case depends on the results of the previous. It is, however, easily seen that a *multiple case* analysis with preloads fits within this framework. Assume, for instance, that you want the response from each of the three external loads *independently*, but with the effect from both bolt preload and gravity load. This situation is illustrated in Figure 75-3. The only difference is that internal variables resulting from the gravity load is imported to each of the three different load cases. The three load cases can be executed in parallel if you have sufficient computer resources.

Simulating a particular stage is on the one hand rather straightforward, but we are left to explain how internal variables are exported from one stage and imported to the next. Although the principles are simple and much can be figured out from logical reasoning, certain situations and features call for a more thorough explanation. With this appendix, we intend to provide the basics and then meticulously go through each feature and situation that requires a more in-depth treatment. We will also explain how all stages can be simulated seamlessly without manual interaction, which is handy if submitting overnight runs.

THE “DYNAIN” APPROACH

The “dynain” approach is a well-known concept among experienced LS-DYNA users, especially in the forming simulation community. The common usage is to output

geometry and stresses of a blank from a forming simulation to a file called dynain, and then include this file in a subsequent simulation, such as springback. This is the simplest form of a multistage analysis, that is, porting stress between two simulations. Here we describe this approach for general situations.

Formats:

The first thing to observe is that the format of a dynain file can be one out of three: *ascii, binary* or *lsda* format.

1. The *ascii* format is popular because you can open it in a text editor and, thereby, inspect and edit the file. It is not, however, capable of representing the full *system state* of a particular simulation result. The *system state* is not only stresses and strains, but everything that is needed to seamlessly continue the process without losing any information. In particular, it requires the state from the contacts, like friction history and information from tied contacts.

2. The *binary* format is not used extensively and is therefore of no interest here.

3. The format we want to use for multistage analysis is the *lsda* format, since it handles most of the internal variables for obtaining an acceptable result. From now on we will refer to the file as the dynain.lsda file.

LSDA format:

The dynain.lsda file is in a sense "just" a regular keyword file. It is bound to the rules and regulations for that particular reader.

It is generated by LS-DYNA from a previous run, but the content is whatever you may find in any other keyword file, such as nodes, elements, initial stresses, etc. It can be included from any other keyword file. Given this prerequisite, and that you are familiar with the keyword format, much of what is about to be presented will make sense from your own experience.

It may contain keywords that are not available in the ascii format. These keywords are for representing the system state as accurately as possible. In particular, it contains stabilization forces and contact history. These are yet to be documented, but the lsda format does not currently have documentation, so this is not a showstopper.

Finally, it is a high-level *binary* format. The content is organized like a file system. To process it, you need an lsda *reader*. Both LS-PrePost and LS-DYNA have built-in lsda readers, but at this moment LS-PrePost is not able to *interpret* all the data correctly. We have an external reader called *ioq*, that can be used to parse through the hierarchical structure of the file with standard linux commands such as *cd, ls* and *cat*. This reader cannot

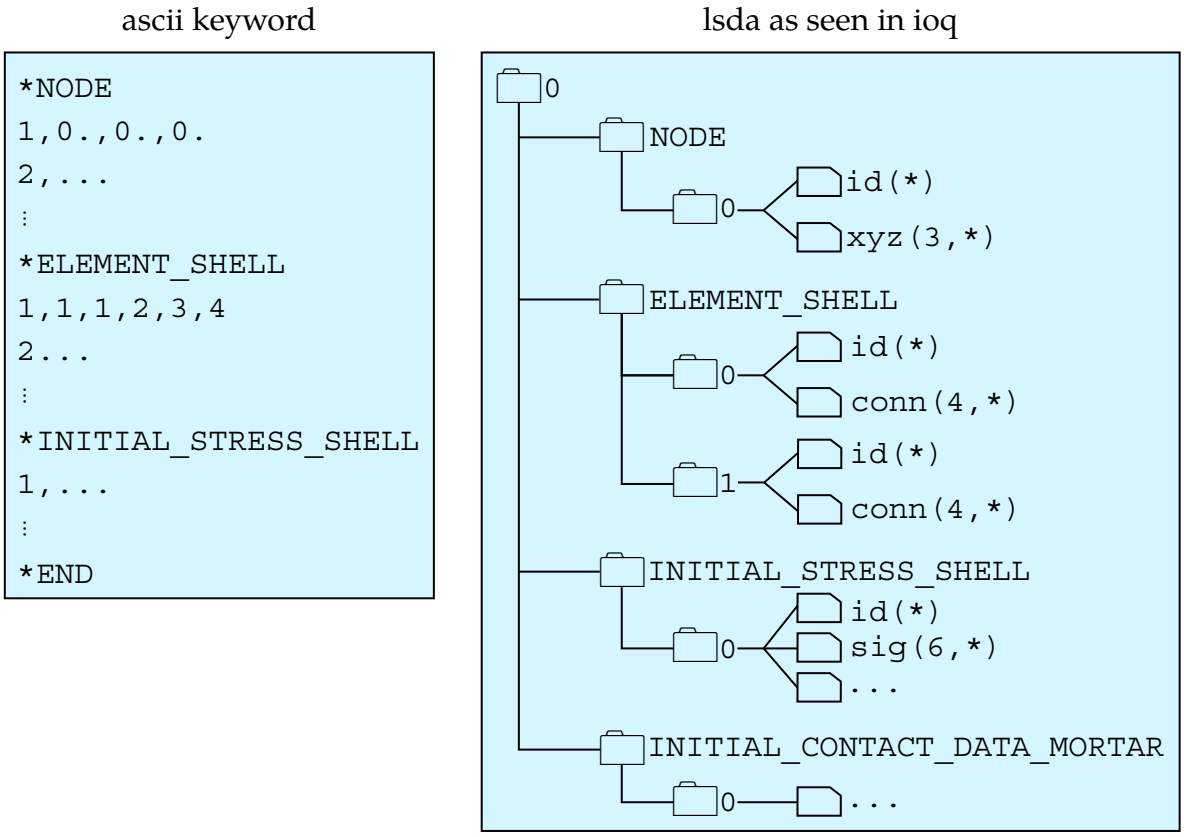


Figure 75-4. Comparison of the structure of *ascii* and *lsda* formats for the dynain file

edit the file. Figure 75-4 schematically illustrates the contents of a dynain file for the ascii and lsda formats, respectively. Note that the enumerations "0" and "1" for some directories are for organizational purposes. We also highlight that the lsda file may contain keywords not supported by the ascii format (here exemplified by INITIAL_CONTACT_-DATA_MORTAR).

Discussion

Juggling many stages soon becomes overwhelming. We recommend giving each stage a *job ID*, which will put a prefix on the dynain.lsda (and all other output) files so that you can distinguish one stage from another. One may either use jobid=A on the command line when executing the stage, or use

```
*KEYWORD_ID
A
```

in the main input file to make sure the dynain.lsda file will be named A.dynain.lsda to indicate that this is data from stage *A*. Using *CASE statements is another way of organizing the stages. *CASE will be described later.

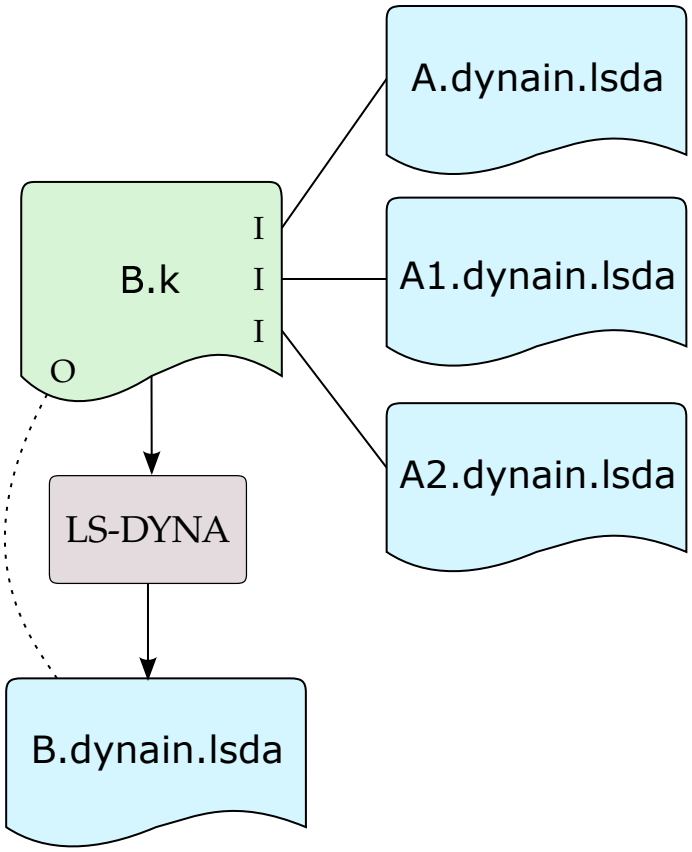


Figure 75-5. Basic schematic of including dynain.lsda files from previous stages in stage *B*. The I in the B.k box stands for an instantiation of *INCLUDE. In this case, A.dynain.lsda, A1.dynain.lsda, and A2.dynain.lsda are output by previous stages and are included in stage *B*. The O in the figure stands for output through *INTERFACE_SPRINGBACK_LSDYNA. With this keyword in the input deck for stage *B*, LS-DYNA exports B.dynain.lsda.

Remember that dynain.lsda *represents* the keyword format in a different way than its ascii counterpart, but that it is nevertheless processed by the keyword reader and should be seen as a natural inclusion to a standard keyword input file. A strong argument in favor of this approach, when compared to other multistage functionalities, is *repeatability*. By restricting all inputs to be keyword and be processed by the same reader facilitates maintenance of the software and thus reduces the risk of i/o problems and related bugs. Stages are advantageous since they constitute a *simple* keyword input. With stages, we can avoid complexities associated with activating/deactivating, switching, using complicated curves and other forms of out-of-the-box solutions that are not part of main stream simulations.

This being said, the dynain approach is by no means trivially understood and failsafe, especially without an appropriate graphical user interface (GUI) to aid the user. But with the right information and tools at hand, it is a powerful framework to deal with complicated processes. We now intend to provide this information and present the tools for dealing with multistage analyses. We refer to Figure 75-5 to enhance understanding. In

particular, we will meticulously go through the details for exporting data from one stage and importing this data to the next.

EXPORTING DATA

Referring to Figure 75-5, assume that we are about to simulate stage *B*, represented by the main keyword file B.k together with include files generated by previous *A* stage(s). For the sake of simplicity, we assume that only dynain.lsda files are included, and B.k is the only ascii file. The keyword required for exporting the results from stage *B* for use in a subsequent stage is *INTERFACE_SPRINGBACK_LSDYNA. It should be part of the input in B.k. A more detailed description of this keyword can be found elsewhere in the keyword manual, but here we provide a recommended setup and discuss the various parameters.

Outputting Data with *INTERFACE_SPRINGBACK_LSDYNA

In the two tables below, we list the needed input in the data cards for *INTERFACE_-SPRINGBACK_LSDYNA with recommended below. In this section we will give motivation for the recommended values of each of the variables.

Card 1	1	2	3	4	5	6	7	8
Variable	PSID	NHSV	FTYPE					
Value	999	999	3					

Card 2	1	2	3	4	5	6	7	8
Variable	OPTC				NDFLAG	CFLAG	HFLAG	
Value	OPTCARD				1	1	1	

PSID = 999
We recommend reserving 999 (or some other favorite number) as the ID for a part set (see *SET_PART) containing parts to be carried over from stage *B* to stage *C*. The resulting B.dynain.lsda file will have the nodes with coordinates and elements with stresses/forces for *at least* those particular parts at the end of the simulation. More nodes will be output (see discussion of NDFLAG below) but not more elements.

Note that none of the *PART, *SECTION, and *MAT keywords are output to the B.dynain.lsda file. They must be inserted in the main file C.k for stage *C* or the simulation will error terminate. Using the same section and material for the same part throughout the entire process is convenient and often sufficient *but not mandatory*. For instance, when going from a quasi-static implicit stage to a dynamic explicit stage, you may want to switch from a fully integrated element formulation to a formulation with hourglass control. In this case, the stresses will be mapped onto the new integration point configuration, resulting in loss of accuracy. In rare situations you might want to alter the material as well. This situation is discussed in conjunction with NHSV below.

We *strongly* recommend associating part, node and element numbers with the actual physical components and their material locations throughout the process. By this, we mean that if a particular node number or element number is used for a certain material point or region, then the same number should *not* be reused for some *other* material point or region. For instance, say that we have a forming simulation where the tools have node numbers between 1 and 100000, and the blank between 100001 and 200000. Then you decide to only output the blank to the dynain.lsda file because the tools are no longer of importance. In the next stage you want to mount the blank onto a vehicle as part of the construction process, where some *new* parts are introduced to act as the rest of the car. It is tempting to reuse node numbers 1 through 100000 since these numbers are "available" again. You should use new numbers and *not* reuse the numbers because you will eventually run into trouble with duplicated nodes. The same recommendation applies to element and part numbers.

Selecting parts for PSID is difficult since it presupposes knowledge of what will be performed next. Any part that is *not* included in PSID can in general not be reintroduced without resetting stresses and history. Parts that *are* in PSID *will* be included in the next stage no matter what. But what if the situation demands that some parts are of interest in one particular branch of a continuation, and some other parts of interest in another? You might, for instance, want to do several sub-model analyses on disjoint components given the stresses from a global gravity load. You might also forget to include a certain part that you need. You can always output all parts with:

```
*SET_PART_LIST_GENERATE
999
1,99999999
```

and then *exclude* some parts when importing the data in the subsequent analysis. This strategy might lead to a large dynain.lsda file, but it is convenient. We will discuss it further when dealing with importing data, specifically in the section Excluding Parts in Subsequent Stages g.

NHSV = 999

You should set NHSV to a "large" number to include all history variables for the elements contained in the part set PSID above. 999 should be enough assuming no material has no

more than 999 history variables. We assume that the material models used for a particular element in stages *B* and *C* are not necessarily the same but *compatible.*

Compatible material models are those for which the mapping of stresses and history variables make physical sense. It is difficult to know which ones *are* compatible without consulting expertise. Generally, we only recommend switching from rigid to some deformable material model. This switch is useful when the deformation of some parts can be neglected for some stages. These parts can be defined as rigid until the stage of interest begins. All the stresses and histories for these parts will be trivially initialized. Note that we do not recommend switching a deformable part to a rigid part. A part that is deformable for some duration and then switched to rigid will be *reset* in terms of stress and history variables.

FTYPE = 3
Setting FTYPE = 3 causes the output dynain file to be in lsda format, which is what we *always* want.

NDFLAG = 1
As mentioned earlier, by default, only the nodes and elements belonging to parts in PSID are output to the dynain.lsda file. This behavior is not always sufficient for adequately continuing a process. For instance, some nodes could be part of nodal rigid bodies or interpolation constraints without being associated to *any* part. The updated locations and orientations of these nodes also need to be carried over between stages. Setting NDFLAG = 1 will make sure that *all* nodes are output to the dynain.lsda file. As long as you follow the practice of not reusing a used node, there should be no reason to not *always* set this. Nodes that for some reason are not hooked up to parts or connectors in remaining stages will simply be constrained or massless and should not affect the outcome of the simulation.

CFLAG = 1
This flag is for output of contact history and tied contact pairs. It should *always* be set to not lose any such information. When this flag is active, *all supported* contacts will be output to the dynain.lsda file, regardless of which ones will be of interest in later stages. The philosophy behind this decision is the same as for elements and nodes discussed above. When importing the data, we simply discard everything that is not hooked up to contact entities in later simulations.

Note that the *CONTACT card itself will *not* be output to the dynain.lsda file, only the segments and their associated data. Therefore, you must copy and paste these lines from one main input file to the next in order for simulations to work properly. The pasting can be done with modifications in ways that will be apparent when we discuss importing data, specifically in the section Contact.

APPENDIX X

Because of the abundance of contact formulations in LS-DYNA, the flag applies only to *some* regular and *some* tied contacts. We will discuss these two classes of formulations separately, starting with regular (non-tied) contacts.

The flag is supported for *all* Mortar contacts, but none of the other contact algorithms. The multiple stage approach is in a sense limited to applications where Mortar contact is the preferred choice or is good enough. In situations, where for instance an implicit gravity load is to be followed by a crash simulation, you may need to switch from mortar to a (faster) contact at the expense of losing contact stress history.

The output is organized in chunks of the form *{contact segment ID, data for this segment}* or *{contact node ID, data for this node}*, depending on the type of contact. Since Mortar contact is a segment-to-segment contact, we only have the former situation at the moment. We will only discuss this situation in the following.

The *contact segment ID* is uniquely defined as the pairing *{contact ID, segment node IDs}*, where *contact ID* is simply the identity of the contact given through the option _ID on the *CONTACT card. To properly map the data, you must specify a contact ID *and* use the same identity for the same contact in subsequent stages. The *segment node IDs* are the node numbers as given by you in the input file, reordered so that the smallest number comes first. The mapping of data occurs when a matching *contact segment ID* is found in the dynain.lsda file and among the actual contacts in the input file.

The *data for this segment* is currently specific to the contact of choice. The format is of little interest to the user. It simply contains enough information for properly migrating the state of the contact between stages, and the mapping is performed with no sophistication. It is, for instance, not possible to map data between different *types* of contacts, which essentially means that the same contact type must be used throughout the entire process. There are *some* allowed operations for changing the behavior of the contact between stages. The section concerning contact in importing data will discuss these allowed operations.

Our implementation for tied contacts follows similar conventions. For instance, the *CONTACT card must be copied between main files for the simulations to work properly. Before delving into further details, we need to elaborate on the extent of support for various tied contact formulations. Explicit and implicit tied contacts are in general different formulations which adds some complications. Explicit tied contacts are *incrementally* objective while implicit are *strongly* objective. Because of this, the internal variables used for keeping track of the tied stress logic is different, and it is nontrivial to map between the two. We added IACC = 2 on *CONTROL_ACCURACY to unite the tied contact formulations. IACC = 2 allows for switching between explicit and implicit with no loss of information. It ensures that the implicit contacts are executed even for explicit analysis. We, therefore, recommend always setting IACC to 2 for multiple stage analysis. If IACC = 1, internal variables are not properly ported. The actual *pairs* will be ported

correctly, but the forces/stresses that have been built up during a stage will not be correct in subsequent stages. As an alternative, you could always set IACC to 0 for which the explicit contacts will be executed even for implicit analysis. IACC set to 0 should also be used if all stages are explicit.

The output to the dynain.lsda file for a tied contact is chunks in the form *{contact ID, node ID, segment node IDs, data}*. As before, *contact ID* is from the option _ID on *CONTACT. *{node ID, segment node IDs}* are the tied contact pairs in user node numbers. The *data* field constitutes the internal variables needed for computing the tied contact forces. As soon as a contact using a specific *contact ID* occurs in the dynain.lsda file, the tied contact pairs for that particular contact are generated solely from the information given therein, and *no* new tied contact pairs are generated even if the logic for tying is fulfilled. All mortar tied contacts are also supported, for which the chunks are instead in the form *{contact ID, segment node IDs, segment node IDs, data}* since the tying occurs between segments and not nodes and segments. Except for this, the same treatment applies.

HFLAG = 1
By default, internal variables associated with *integration points*, such as stresses and material related history, are ported between stages. Stabilization forces, such as drilling and hourglass forces, may depend on their own internal variables. These variables may also need to be transferred for a complete description of the system state. We recommend setting HFLAG to 1 to deal with these internal variables. We will proceed to discuss the two mentioned types of stabilization.

In implicit analysis, *drilling* stabilization is on by default, otherwise shell elements would suffer from incompleteness. Usually, everything is taken care of automatically. If all stages are implicit, we can end the discussion here. If the implicit analysis is followed by an explicit analysis, the drilling stabilization will be turned *off* in the latter since explicit drilling resistance is by default handled through the rotational inertia of the shell element nodes. This transition results in loss of equilibrium. In practice this may not mean much, but, if you are picky, you can deal with this by using DRCPSID on *CONTROL_SHELL to invoke drilling stabilization even in explicit analyses. Doing so for all shell parts leads to force continuity between the implicit and explicit stages.

For elements with reduced integration (shells or solids), stiffness based *hourglass* stabilization may be invoked to complete the element. These hourglass models update the internal forces in a similar fashion to how the integration point stresses are incremented. These forces need to be output to the dynain.lsda file. To maintain equilibrium, all stages need to keep the same element formulation and hourglass model. Otherwise, these forces will be lost in the transition. Again, this may be of little practical importance, but nevertheless a good thing to know.

APPENDIX X

Boundary Conditions

Before discussing the importing of data from the previous stage, we need to emphasize that the two lines

```
*INTERFACE_SPRINGBACK_EXCLUDE
BOUNDARY_SPC_NODE
```

must be present in all stage files. Otherwise, the boundary conditions are inherited from the previous stage. With these lines, we are free to use whatever boundary conditions we like for each stage.

IMPORTING DATA

Let's assume stage *B* has been completed. Then, to import stage *B*'s data for stage *C*, add the lines

```
*INCLUDE
B.dynain.lsda
```

to the main file C.k. Assuming no conflict (that is, all element and node numbers are unique), you can include several files, each resulting from separate previous runs

```
*INCLUDE
B.dynain.lsda
B1.dynain.lsda
B2.dynain.lsda
…
```

You can also include the same file several times, but the data must be transformed differently for each include so that again there are no conflicts

```
*INCLUDE
B.dynain.lsda
*INCLUDE_TRANSFORM
B.dynain.lsda
*INCLUDE_TRANSFORM
…
```

Adding these lines is in principle how importing data works with the catch that the include files need to be complemented with accompanying *PART, *SECTION, *MAT, *CONTACT, etc. cards in the main file C.k to render a sensible continuation of the process. In this section we lay out the details concerning which keywords need to be reinserted into C.k and which ones appear in the include file B.dynain.lsda from the previous stage.

Excluding Parts in Subsequent Stages:

First, let's continue the discussion about PSID = 999 from the section on Outputting Data with *INTERFACE_SPRINGBACK_LSDYNA. Recall that only the parts specified in this particular part set in B.k are output to the B.dynain.lsda file. This requirement could be a problem if the parts to be used in subsequent stages are not known *a priori*, or if you want to perform several sub-modelling analyses using data from the same previous run.

We previously indicated that you may output *all* parts and then use only those of interest in subsequent stages. This strategy will circumvent this conundrum. For it to work you will need to include *CONTROL_LSDA in the main file C.k. For details we refer to the corresponding keyword description elsewhere in the manual, but its application is best explained through a simple example.

For the sake of simplicity, assume that stage *B* consists merely of three parts, enumerated from 1 to 3. By include the following in B.k,

```
*SET_PART_LIST
999
1,2,3
*INTERFACE_SPRINGBACK_LSDYNA
999,…
*PART
Part 1
1,1,1
*PART
Part 2
2,2,2
*PART
Part 3
3,3,3
```

LS-DYNA will output all three parts to B.dynain.lsda, including their respective nodal positions and integration point data.

We can now in one particular continuation, say stage *C_1*, insert

```
*CONTROL_LSDA
$ Number of parts to exclude
1
$ Part to exclude
3
*PART
Part 1
1,1,1
*PART
Part 2
2,2,2
```

into C_1.k to actually use only parts 1 and 2. And in another, say stage *C_2*, insert

```
*CONTROL_LSDA
$ Number of parts to exclude
2
$ Parts to exclude
1,2
*PART
Part 3
3,3,3
```

into C_2.k to use only part 3.

This approach is illustrated graphically in Figure 75-6. Note that the *PART card for the active parts need to be reinserted in each continuation.

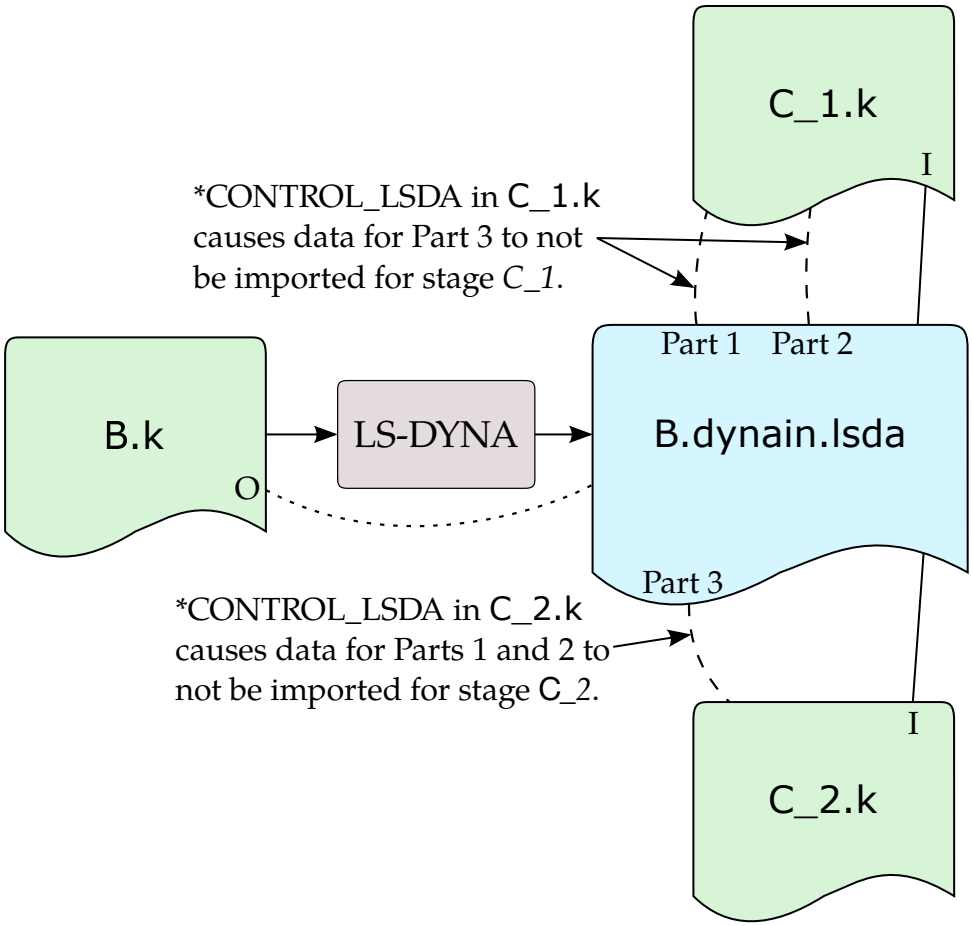


Figure 75-6. Illustration of how parts can be excluded in subsequent analyses. In this example. Stage *B* contains the full model. For stage *C_1* Part 3 is excluded from the analysis due to *CONTROL_LSDA while for stage *C_2* Parts 1 and 2 are excluded.

Changing Element and Material Types:

Just as the *PART card is needed in a continuation, the *SECTION and *MAT cards are needed as well. In almost every situation you would just copy them from the previous stage, assuming things can proceed without sacrificing execution speed or accuracy. In rare cases, you may want to change element formulation or maybe even material model.

The element formulation is preferably changed when switching from an explicit stage to an implicit stage, or vice versa. Fully integrated elements (such as type -16) enhance accuracy in implicit without significant speed penalty while reduced integration elements make explicit computationally efficient while maintaining sufficient accuracy. For example, stage *B* may have

```
*CONTROL_IMPLICIT_GENERAL
1,1.0
*SECTION_SHELL
```

```
1,-16
1.,1.,1.,1.
```

in *B.k*, and stage *C* continues the process with

```
*CONTROL_IMPLICIT_GENERAL
0
*SECTION_SHELL
1,2
1.,1.,1.,1.
```

in *C.k*. The stresses from elements of type -16 in stage *B* will be averaged in the plane for use in the type 2 shells of stage *C*, leading to loss of accuracy. Had it been the other way around, the single point stress in the Belytschko-Tsay shell would be scattered to the four in-plane integration points of the fully integrated shell, impacting the accuracy of the transition. We recommend maintaining section properties between stages as much as possible.

We do not recommend switching materials either in general. As mentioned in the discussion of NHSV = 999 under Outputting Data with *INTERFACE_SPRINGBACK_LS-DYNA, you *may* use a rigid material for a subset of parts during the first few stages, and then switch them to a deformable material when loaded accordingly. Switching back to a rigid material later will rigidize the parts in the deformed state, causing you to lose the internal variables. You must decide if losing this information for subsequent stages is acceptable.

Inertia Properties of Rigid Bodies:

Concerning the *PART card, the particular keyword *PART_INERTIA needs special attention. Again, recall that LS-DYNA will interpret the keyword as written which is a problem if the inertia properties are provided in global coordinates. Consider a rigid body that initially has a center of mass located at

$$\mathbf{x}_c = \begin{pmatrix} 5 \\ 0 \\ 0 \end{pmatrix}$$

and an inertia tensor along the global axes

$$\mathbf{I}_c = \begin{pmatrix} 3 & & \\ & 2 & \\ & & 1 \end{pmatrix}.$$

The part input for this body would be

```
*PART_INERTIA
Global inertia
1
…
$ XC YC ZC TM
5,0,0,1
$ IXX IXY IXZ IYY IYZ IZZ
3,0,0,2,0,0,1
$ No velocities
…
```

Now, if the body translates a distance 1 in the global x direction and rotates 90 degrees about this axis during the first stage, then the rigid body properties at the end of the simulation are

$$\mathbf{x}_c = \begin{pmatrix} 6 \\ 0 \\ 0 \end{pmatrix}$$

and

$$\mathbf{I}_c = \begin{pmatrix} 3 & & \\ & 1 & \\ & & 2 \end{pmatrix}.$$

Consequently, if the *PART_INERTIA card is copied between stages as we have recommended, the inertia properties will always take the values from the beginning of the entire process, resulting in apparent errors. The strategy for circumventing this is to define inertia properties with respect to local positions and a local coordinate system that follows the motion of the body. To do this, put a node at the center of mass and let NODEID refer to this node. Also, set IRCS = 1, and let CID refer to a coordinate system defined by nodes attached to the rigid body in question. The nodes needed for NODEID and CID can be defined by *CONSTRAINED_EXTRA_NODES. The keywords for the example above would be

```
*PART_INERTIA
Local inertia
1
…
$ XC YC ZC TM IRCS NODEID
,,,1,1,999
$ IXX IXY IXZ IYY IYZ IZZ
3,0,0,2,0,0,1
$ No velocities

$ XL YL ZL XLIP YLIP ZLIP CID
0,0,0,0,0,0,99
*NODE
999,5,0,0
9999,6,0,0
99999,5,1,0
*CONSTRAINED_EXTRA_NODES_NODE
1,999
1,9999
1,99999
*DEFINE_COORDINATE_NODES
99,999,9999,99999
```

The above is for the first stage. In all remaining stages the nodes will be contained in corresponding dynain.lsda files, while the other cards are copied between stages.

The same method must be used for *CONSTRAINED_NODAL_RIGID_BODY_INERTIA which has the same keyword data (except the coordinate system in input in field CID2) and *ELEMENT_INERTIA which can similarly be defined using a coordinate system based on nodal coordinates. In sum, care should be taken when simulating components with the INERTIA option.

Deformation Gradient and Stress Update:

All materials have their own set of history variables that are used for computing stresses. We loosely call a material *hyperelastic* if the deformation gradient or stretch tensor is an essential part of the history content and needed for computing the stress.

The deformation gradient can be computed in two different ways, governed by FMATRX on *CONTROL_SOLID. For FMATRX = 1, it is numerically incremented based on the strain and spin rate, while for FMATRX = 2 ,it is computed directly as a relation between the current and reference configurations. In the FMATRX = 1 case, nothing needs to be known about the reference configuration. It would be just like a *hypoelastic* material situation where everything is incrementally updated. For FMATRX = 2, the reference configuration needs to be the same throughout the entire process. Before version R16, this setting affected how to transition stresses between stages. Starting with R16, LS-DYNA handles the treatment of the deformation gradient automatically without requiring user intervention.

The discussion above pertains to solid elements. For shells and beams, the deformation gradient is, in general, numerically incremented, except for fabric materials (*MAT_FABRIC). For fabric materials, the discussion is slightly modified. For instance, no corresponding FMATRX flag exists. The deformation gradient is computed based on the reference geometry from the first stage. Also here, LS-DYNA handles the treatment of reference configurations between stages automatically in versions R16 and later.

Contact:

As discussed in the CFLAG = 1 section, contact histories are ported for a selection of contacts when CFLAG is 1 on *INTERFACE_SPRINGBACK_LSDYNA. The *CONTACT card itself, for the contact with a given ID, must be explicitly included in the keyword file in the continuing simulation. Recall that the dynain.lsda file contains the contact history for all supported contact types from the previous simulation. To read this information, the *CONTACT card ID must match the ID for the history data in the file must match. As discussed earlier, involved contact segments must also match in the sense of having the same user nodal IDs. Let's discuss some hypothetical situations beginning with the penalty contacts.

In a common scenario, the contact definition in subsequent runs is identical to the first run, using the same contact ID, same number of contact segments, and same corresponding user nodal IDs. We *are*, however, allowed to deviate from this restriction. For instance, if a part is added to the subsequent simulation, this part may be included in the contact. This additional part will render additional segments for which there will be a clean sheet of history variables. These segments may come into new contact situations while the other segments will have their history inherited. We can also consider removing a part from the model and consequently from the contact. Any remaining segments

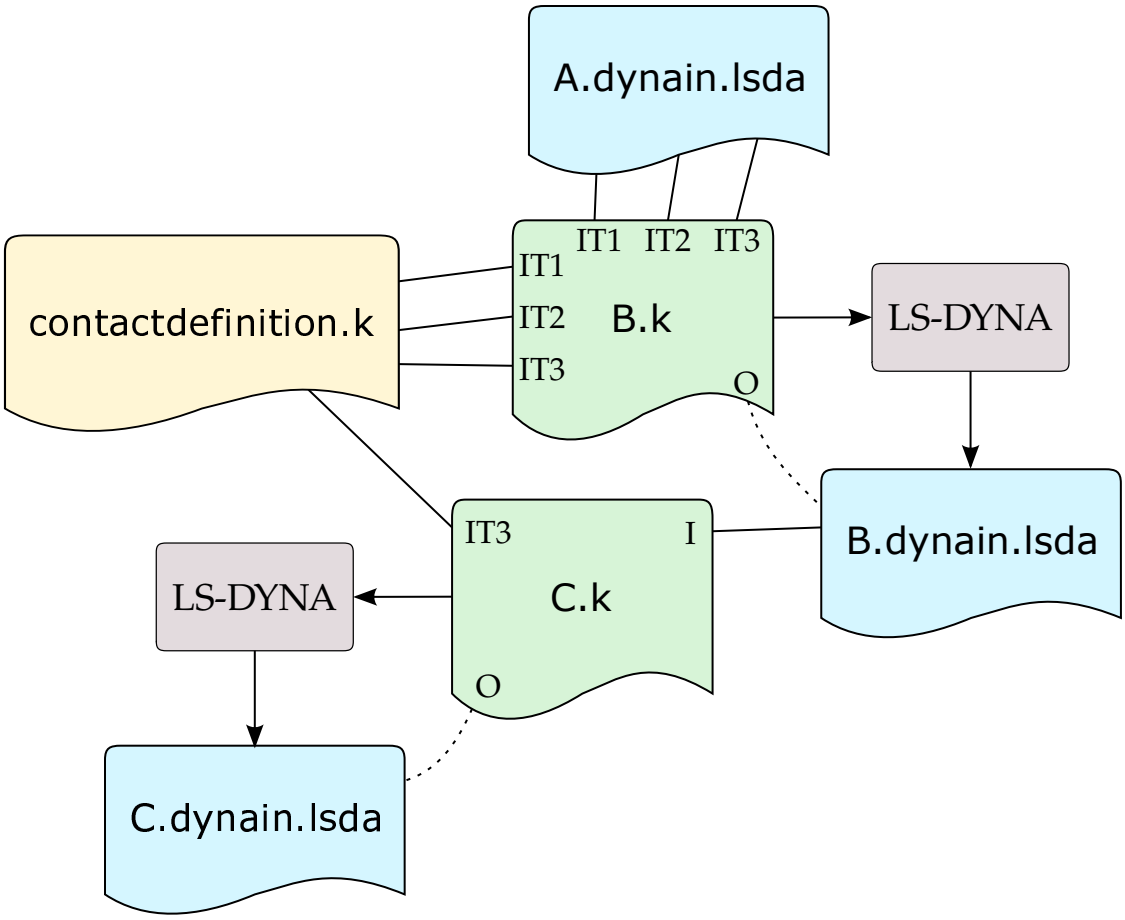


Figure 75-7. Example of including a component several times with transformations. In this example the component from stage *A* is included in Stage *B* three times. Because the same file is being included several times, you must use the keyword *INCLUDE_TRANSFORM. IT*i* indicates a *INCLUDE_TRANSFORM with TRANID *i*. The contact definition must also be included for each component with the same corresponding TRANIDs. For subsequent stage *C*, if we only want the third component, then we include the contact definition again with TRANID = 3. B.dynain.lsda does not need to be transformed since it already contains the transformed component.

that *were* in contact with the deleted part will be given no contact force and will be reset in terms of history variables. Finally, you can change the contact stiffness scale factor SFSA if needed.

For tied contacts, there aren't many exceptions to the rule of keeping the contact as is. Departing from the same contacts tends to result in situations for which the behavior is ambiguous. The mortar tied contacts offer some interesting switch options that may be worth discussing. For instance, you can execute a stage using the keyword *CONTACT_..._MORTAR_TIED_WELD. This contact essentially starts off as a regular, non-tied contact but will tie segments based on weld conditions as a result of the simulation. During the next stage, the keyword may be changed to *CONTACT_..._MORTAR_TIED,

or *CONTACT_..._TIEBREAK_MORTAR, assuming the welding part of the process is now done. The segments that were tied during the first stage are inherited to the next and the behavior will for the remaining process be just as for a regular tied or tiebreak contact.

Sometimes you need to include the same dynain.lsda file several times with different transformations using *INCLUDE_TRANSFORM. For instance, a small, preloaded component may be spatially distributed in a larger assembly. This feature is supported for the state of contact, including ID and nodal offsets, but it requires that *all* segments in contact are transformed as a unified rigid body. Note also that you need a *CONTACT card for each included file where the ID is transformed as the ID in the dynain.lsda file. If many such transformations are required, you can put the contact definition in a separate file and include that the same way as the dynain.lsda file. Figure 75-7 illustrates the case where a component is simulated in stage *A* and distributed in three instances for stage *B*. Note the treatment of the contact definition.

The keyword syntax in B.k would be

```
*INCLUDE_TRANSFORM
A.dynain.lsda
…
$TRANID
1
*INCLUDE_TRANSFORM
A.dynain.lsda
…
$TRANID
2
*INCLUDE_TRANSFORM
A.dynain.lsda
…
$TRANID
3
*INCLUDE_TRANSFORM
contactdefinition.k
…
$TRANID
1
*INCLUDE_TRANSFORM
contactdefinition.k
…
$TRANID
2
*INCLUDE_TRANSFORM
contactdefinition.k
…
$TRANID
3
```

The contact definition file is the same as when simulating stage *A*, just copied between stages. In stage *C*, assume we just want to keep the third component, meaning that we only need to keep the contact definition for that particular contact. The syntax in C.k would then be

```
*INCLUDE
B.dynain.lsda
*INCLUDE_TRANSFORM
```

```
contactdefinition.k
…
$TRANID
3
```

Note that the included dynain.lsda file needs no transformation since it already has the transformed quantities from the first transformation between stages *A* and *B*. This hypothetical example should provide insight into the interrelation between the resulting dynain.lsda files and main keyword ascii files. It should all make sense when recalling the basics of the keyword format.

Stabilization Forces:

Porting stabilization forces is done more or less automatically. the only concern is the compatibility between implicit and explicit. Following the discussion earlier (HFLAG = 1), we recommend invoking drilling restraints with DRCPSID on *CONTROL_SHELL, even for explicit analysis. Furthermore, using hourglass type 6 for solid elements and type 4 for shell elements is also a good practice. It would lead to continuity in the stabilization forces between stages. Departing from these recommendations may not be a terrible thing to do, but you should be aware of the possible consequences.

CASES

So far we have assumed that you execute runs sequentially and manually. Sometimes simulating all stages in a single run is convenient. For instance, suppose you need to submit an overnight run where you want all stages to be simulated, and you want to take advantage of multistage capability for reasons to be given. The keyword *CASE allows you to do setup a multistage simulation in a single run. Here we exemplify this usage by reconsidering the example at the beginning of this Appendix, the ROPS test. We emphasize that this problem may be set up in many equivalent ways. We will present here merely a suggested organization of files that should serve as a reasonable template. To shorten this section, we will skip the "gravity load" and the "load from behind" for this discussion.

The main process file would contain the definition of all cases (= stages), the card for writing the dynain.lsda file and perhaps other data that is common for all stages. It would look something like

```
*KEYWORD
$
$ Meta data for cases
$
*CASE
1,BoltPreload
*CASE
3,LoadSide
*CASE
4,LoadRoof
$
$ First case is defined all in ascii
```

```
$
*CASE_BEGIN_1
*INCLUDE
BoltPreload.k
*CASE_END_1
$
$ Remaining cases define bulk in ascii,
$ and includes the state from previous dynain.lsda file
$
*CASE_BEGIN_3
*INCLUDE
LoadSide.k
BoltPreload.dynain.lsda
*CASE_END_3
*CASE_BEGIN_4
*INCLUDE
LoadRoof.k
LoadSide.dynain.lsda
*CASE_END_4
$
$ Output "everything" to dynain.lsda
$
*INTERFACE_SPRINGBACK_LSDYNA
…
*INTERFACE_SPRINGBACK_EXCLUDE
BOUNDARY_SPC_NODE
$
$ Mounting of the cab, present in all stages
$
*BOUNDARY
…
*END
```

We then need to define each of the cases. We will start with the bolt preload. The bolt preload is the "easy" load case since it does not involve any inclusions of data from previous stages. Without further ado, BoltPreload.k would look something like

```
*KEYWORD
*TITLE
Bolt preload
$
$ Part data for the cab
$
*PART
…
*PART_INERTIA
…
*SECTION
…
*MAT
…
$
$ Nodes and elements for the cab
$
*NODE
…
*ELEMENT_...
…
$
$ Potential other cards required for the cab
$
*ELEMENT_MASS
…
*ELEMENT_INERTIA
…
*CONTACT_...
```

```
…
*CONSTRAINED_...
…
$
$ The specific load case follows
$
*INITIAL_STRESS_SECTION
…
*DATABASE_CROSS_SECTION
…
*END
```

We will not write out the details. You should imagine the general content of the cab model.

We then set up the load from the side assuming the bolt preload is already finished and the corresponding BoltPreload.dynain.lsda contains everything we now know it contains. The keyword file LoadSide.k needs to contain the complementary information as well as the impactor from the side and associated data. The input file would look like

```
*KEYWORD
*TITLE
Side load
$
$ Part data for the side impactor
$
*PART

1000,…
*SECTION
…
*MAT
…
$
$ Motion of the side impactor
$
*BOUNDARY_PRESCRIBED_MOTION_RIGID
1000,…
$
$ Geometry for the side impactor
$
*NODE
…
*ELEMENT
…
$
$ Contact between the side impactor and cab
$
*CONTACT_..._ID
1000,…
…
$
$ Data for the cab is exactly as in bolt preload,
$ except for nodes and elements which are in the
$ BoltPreload.dynain.lsda file
$
…
*END
```

We will highlight a few things. First, the nodes and elements for the cab are all in the dynain.lsda file, so they do not need to be copied over. Second, keywords like *ELEMENT_MASS/INERTIA and *CONSTRAINED_NODAL_RIGID_-

BODY/INTERPOLATION are to be seen as integrated components of the cab that are *not* written to the dynain.lsda file. These cards refer to quantities that are either updated between stages (nodes) or invariant to any deformation (masses). Therefore, these need to be copied.

Starting the third stage, we tend to get even more fancy because now we not only introduce the roof impactor but need to take out the side impactor. The LoadRoof.k would look like

```
*KEYWORD
*TITLE
Roof load
$
$ Ignore impactor from side in dynain.lsda file,
$ also taking out all of its part and contact data
$
*CONTROL_LSDA
1
1000
$
$ Part data for the roof impactor
$
*PART

2000,…
*SECTION
…
*MAT
…
$
$ Motion of the roof impactor
$
*BOUNDARY_PRESCRIBED_MOTION_RIGID
2000,…
$
$ Geometry for the roof impactor
$
*NODE
…
*ELEMENT
…
$
$ Contact between the roof impactor and cab
$
*CONTACT_..._ID
2000,…
…
$
$ Data for the cab is exactly as in side load,
$ except for nodes and elements which are now in the
$ LoadSide.dynain.lsda file
$
…
*END
```

In principle, this example defines one simulation run in three steps. You may wonder why this is better than defining all in one simulation. Assume you start this simulation overnight, and in the morning you can post-process the results for each of the three cases. What if the first two stages were successful, and the third stage failed? Well, then you can

simply make the modifications necessary in LoadRoof.k that you think takes care of whatever mistakes you made in the first place, and then redefine the main process file as

```
*KEYWORD
$
$ Meta data for cases, first two already successful
$
*CASE
4,LoadRoof
$
$ and we can start directly on the roof load
$
*CASE_BEGIN_4
*INCLUDE
LoadRoof.k
LoadSide.dynain.lsda
*CASE_END_4
*INTERFACE_SPRINGBACK_LSDYNA
…
*INTERFACE_SPRINGBACK_EXCLUDE
BOUNDARY_SPC_NODE
$
$ Mounting of the cab, present in all stages
$
*BOUNDARY
…
*END
```

You can submit this run without having to rerun the first two stages.

Multistage processes also allow you to cleanly organize a process compared to defining everything in a single input with birth/death times, deformable to rigid cards, implicit/explicit switches, complicated curves, etc. Admittedly, either way you do it manipulating text files in text editors is not the way to go. This appendix should be seen as an introductory guide for setting up and troubleshooting this kind of problem. An ultimate goal would be to do the setup in a high-level GUI where much of what has been presented in here as "recommendations" or "requirements" are dealt with automatically.

APPENDIX Y: LS-DYNA HYBRID User Guide

This is a short user's guide for the hybrid version of LS-DYNA.

INTRODUCTION

Since the LS-DYNA code was first introduced in the early nineties, continuous improvements have been made to the code to meet the needs of engineers who are using the code for increasingly complex applications. As this complexity has increased from purely structural problems to multi-physics problems and modeling details have raised the number of elements and solution variables, the performance of the code on multi-processor computers and computer clusters has become even more important. Calculation times have also risen with the use of more complex material models, multiple contacts, coupling between different modeling approaches, smaller element sizes, and more expensive element formulations. Contemporaneous to the increasing complexity of features and models, the increases in processor performance and the design of sophisticated clusters and networks have opened new possibilities for the solution of complex analysis models in a more acceptable timeframe. In order for software to take advantage of these increasingly common computational environments with multiple cores and compute clusters, parallel programming methods that enable the software to calculate portions of the model in parallel on several processors are required.

LS-DYNA offers two parallel programming models. SMP (Symmetric Multi-Processing), which originated from the serial code and uses OpenMP® directives to split the threads thereby enabling them to be run on multiple cores; and MPP (Massively Parallel Processing), which uses a message passing protocol to exchange information between the cores on a board or over the network. The SMP parallel programming model runs on computers with multiple identical cores with the cores and memory connected via a bus that is shared between all cores. The MPP programming model first performs a decomposition of the problem (domain decomposition) and then distributes the sub-domains to different cores using MPI (Message Passing Interface) protocol for communications between the subdomains during calculation. These sub-domains use their own exclusive part of memory and exchange data that is needed on other cores via MPI protocol. The MPP model is, therefore, not restricted to hardware where all cores have access to the same memory. This is important when using compute clusters, where an arbitrary number of cores are gathered and connected via a network. For the MPI protocol, messages can be passed between cores on one motherboard or over the network, which provides the opportunity to build clusters of single compute nodes connected via a network.

The LS-DYNA MPP version is scalable over a wide range of core counts and can reduce calculation wall clock times dramatically. While the MPI protocol connects all cores,

increasing the number of cores rapidly increases the network load, which has a negative impact on the performance and scalability. The LS-DYNA HYBRID version takes advantage of a combined SMP and MPP programming model to address this issue.

LS-DYNA HYBRID VERSION

In general, LS-DYNA HYBRID combines SMP and MPP parallel programming models to reduce wall clock time when using an increasing number of cores. Additionally, the LS-DYNA HYBRID version can be used (in a specific setup) to obtain consistent outputs for different core counts. "Output" is meant here to be the various result quantities obtained from binary or ASCII format result files, e.g., the value of specific node displacements for a node in a specific coordinate direction at a specific time.
In the following section, performance, and scalability as well as output consistency are discussed for the LS-DYNA HYBRID version. This includes the discussion of basic characteristics of the LS-DYNA SMP and LS-DYNA MPP versions since the LS-DYNA HYBRID version is a combination of these two programming models.

PERFORMANCE AND SCALABILITY

One of the main measures of performance is the elapsed wall clock time $T_{elapsed}$, which can be split up as follows:

for LS-DYNA SMP:	$T_{elapsed} = T_{computation}$	$+ T_{IO} + T_{overhead}$
for LS-DYNA MPP:	$T_{elapsed} = T_{computation} + T_{communication}$	$+ T_{IO}$
for LS-DYNA HYBRID:	$T_{elapsed} = T_{computation} + T_{communication}$	$+ T_{IO} + T_{overhead}$

where $T_{computation}$ is the actual time the cores do operations on the problem and T_{IO} is the time needed to perform the Input / Output operations to the hard disk. $T_{overhead}$ is related to the OpenMP® thread overhead in the SMP programming model. This thread overhead is related to the time needed to manage and create threads in the SMP programming model that also includes managing the memory for these threads and delays due to mismatches in computational time between threads. $T_{communication}$ is the time which is required by the MPI protocol to exchange data between different cores.

If the number of cores is increased for an MPP application, the performance and scalability may not also increase as would be expected. In MPP applications, communication is done between all cores used for the calculation. If the core count increases, the time needed for the communication $T_{communication}$ increases. If the amount of data exchanged via MPI protocol hits the bandwidth of the network connections, communication speed decreases rapidly, and when the time used for communication becomes dominant, the analysis fails to scale.

To keep the core count constant, but reduce the MPI communication, the LS-DYNA HYBRID version combines SMP and MPP parallel programming models. In the LS-DYNA HYBRID programming model a portion of the MPP ranks are replaced by an SMP process (the SMP process contains all SMP threads). The SMP calculations can use several cores, while the MPI protocol communicates only between the SMP processes, which reduce the amount of communication when compared to a pure MPP run.

OUTPUT CONSISTENCY

The drawback of having different parallel programming models is that some of the code subroutines for LS-DYNA keywords such as the *CONTACT, *ALE, *AIRBAG, *BOUNDARY, *CONSTRAINED keywords are different, while others such as the *ELEMENT and *MAT keywords are, for the most part, the same code. This discrepancy leads to different computed output for LS-DYNA SMP, LS-DYNA MPP and LS-DYNA HYBRID versions compared to each other.

Additional differences can be seen for all three parallel programming methods when using different core counts. The order of summation of result vectors depends on the number of cores used to compute them. Due to round off errors, the result of this summation is order dependent, which shows up as different output for different core counts. Methods to avoid or suppress these differences in output are discussed for the three parallel programming models in the next paragraphs.

When running LS-DYNA SMP versions, there is a consistency flag available that enforces the order of the summation of certain result vectors, thereby maintaining output numerical consistency when using different core counts. There is, however, a time penalty of about 10-15% of the wall clock time for this LS-DYNA SMP consistency option.

When running LS-DYNA MPP versions, there are numerical variations due to round off errors during the summation of certain result vectors. Using double precision, a finer mesh, or avoiding instabilities in the model, may help reduce these numerical variations. Unfortunately, a consistency flag option for MPP would increase wall clock time. Certain MPI protocols sum up the result vectors on the cores that belong to one compute node first and then sum up all results from the compute nodes connected via the network. If the compute nodes have different core counts, this could end up with different results even for constant core counts. To avoid this effect, the “lstc_reduce” option in pfile should be set.

The LS-DYNA HYBRID version inherits the merits and limits from both programming models, SMP and MPP that it was derived from. Using LS-DYNA HYBRID without the LS-DYNA SMP consistency flag would show differences in output when using different core counts for the SMP threads, as would different MPP processors counts show differences in the computed output. However, for the LS-DYNA HYBRID version, using the merits of the consistency flag for SMP threads and keeping the MPP processors constant,

output consistency is maintained for varying SMP thread counts. This enables a consistent output if the number of MPP processors is kept constant and the SMP thread count varies.

SUPPORTED FEATURES

For a list of supported features please refer to "Appendix O MPP User Guide" of this manual.

CONTACT INTERFACES

LS-DYNA HYBRID uses the parallel contact algorithm of LS-DYNA MPP. For supported and unsupported contact options please refer to "Appendix O MPP User Guide" of this manual.

OUTPUT FILES AND POST-PROCESSING

For performance reasons, many of the ASCII output files normally created by LS-DYNA have been combined into a binary format. For further information on the available output files formats please refer to "Appendix O MPP User Guide" of this manual.

PARALLEL SPECIFIC OPTIONS

For memory options and restart capabilities please refer to "Appendix O MPP User Guide" of this manual

PFILE

LS-DYNA HYBRID uses the same pfile definitions as LS-DYNA MPP. For details on these definitions please refer to "Appendix O MPP User Guide" of this manual.

EXECUTION OF LS-DYNA HYBRID

LS-DYNA HYBRID, as well as LS-DYNA MPP, uses a message passing interface to communicate between several MPP ranks. The message passing interface is not part of the LS-DYNA implementation. A third party MPI product software is needed to execute LS-DYNA HYBRID when more than one MPP rank is used. There are several MPI products available and LSTC provides LS-DYNA HYBRID executables for all major implementations of the MPI standard.

APPENDIX Y

For the execution syntax of the MPI product, please refer to the user documentation of this product. In the following paragraphs <MPI-EXE> is used to identify the run command for the MPI product and "-np" is used as a flag to the MPI product executable to specify the number of MPP ranks.

The generic example for an execution syntax for LS-DYNA HYBRID is:

<MPI-EXE> -np < number of MPP ranks > <LS-DYNA HYBRID name > ncpu = - < number of OpenMP threads > <general LS-DYNA command line options >

The number of MPP ranks are specified in the same way as for LS-DYNA MPP versions. In addition to this specification, the OpenMP threads per MPP rank must specified as well. This is done in the same way as for the LS-DYNA SMP version, via the command line option "ncpu=". It is strongly recommended to use a negative number here to activate the SMP consistency flag.

APPENDIX Z: Using the System Coupling Software with LS-DYNA

This discussion has been adapted from the presentation given by David Aspenberg at the 2025 Ansys EMEA Transportation Summit and LS-DYNA Conference in Munich, Germany (Aspengberg [2025]) and the accompanying paper (Aspenberg et al [2025]).

ABOUT THE SYSTEM COUPLING SOFTWARE

The System Coupling software manages coupling among two or more supported Ansys products to obtain multiphysics solutions. It can also couple Ansys products with external data sources. This physics solver connection software handles the data exchange and coordinates the execution of each solver. It also features convergence checking, solution stabilization, and HPC deployment. See the System Coupling User's Guide for a more detailed discussion of its features.

The System Coupling software supports four different execution methods:

- Interactively through the System Coupling standalone GUI,
- Batch execution using a command line,
- Through integration in Workbench, and
- Using PySystemCoupling.

Currently, coupling with the LS-DYNA solver is supported through the standalone GUI and through batch execution. We will limit our discussion to execution via the standalone GUI (see the Coupled Simulation Setup and Post-Processing section and the Examples section).

COUPLING WITH LS-DYNA

Supported Couplings

LS-DYNA can be coupled with Ansys Fluent®. For this coupling, LS-DYNA gives Fluent displacements, and Fluent gives LS-DYNA forces.

Note that both System Coupling and LS-DYNA support coupling through the FMI standard. However, performing a coupled analysis with FMUs yields lower-fidelity results at the system level, where the force and displacement fields are not fully resolved. This guide does not cover using FMUs.

Versions

System Coupling with LS-DYNA is currently available as a beta feature. It is available for both LS-DYNA MPP.

LS-DYNA version	System Coupling version
R16.2 (beta)	2026R1 or later when using LS-DYNA explicit 2025R2 or later when using LS-DYNA implicit
R17.0 (beta)	2026R1 or later when using LS-DYNA explicit 2025R2 or later when using LS-DYNA implicit
R18.0	TBD

We recommend using the most recent version of System Coupling possible. System Coupling can be downloaded with the Ansys installation package.

System Coupling is only supported for MPP. You must use Intel MPI with it. The following MPI versions or newer must be used:

System Coupling version	MPI Version for Windows	MPI Version for Linux
2025R2	Intel MPI 2021.8	Intel MPI 2021.14.1
2026R1	Intel MPI 2021.8	Intel MPI 2021.14.2

Fluid-Structure Interaction (FSI) using Fluent and LS-DYNA

In FSI, the fluid flow may exert pressure and/or thermal loads on a structure. Here, we will discuss only structural loads from the fluid, as thermal loads have not yet been implemented. When the pressure load is large enough to cause significant deformation of the structure, such that the fluid flow itself is altered, the pressure load and deformation must be exchanged between the solvers to obtain a correct solution. Hence, capturing the physics requires a coupled approach. Conversely, a sequential approach, in which the flow problem is solved first separately with the pressure loads applied in a subsequent structural simulation, is sufficient for small structural deformations.

For the Fluent/LS-DYNA coupling, Fluent handles the fluid flow and LS-DYNA the structural deformations. Both one-way and two-way coupling types are available. When System Coupling reads the LS-DYNA input file during setup using the System Coupling GUI, it automatically determines whether the structural solution in LS-DYNA is using the explicit or implicit solver, and this will determine the iterative approach used by System Coupling to resolve the coupling between the two solvers. In the following, it is assumed that Fluent is run with an implicit solution scheme.

APPENDIX Z

When using LS-DYNA's implicit solver, a two-way iterative coupling approach, as illustrated in Figure 77-1, is automatically suggested. For each coupling time step, several coupling iterations are performed in which forces coming from the pressure load are sent from Fluent to LS-DYNA, and a displacement is sent from LS-DYNA to Fluent. Within each coupling loop, the different solvers independently seek convergence in the solver loop, in which absolute convergence is not an absolute requirement until the last coupling iteration. If different time step sizes are required on each solver side, we advise picking the smallest of these as the common time step. For this approach, the System Coupling framework also has convergence stabilization and acceleration methods, such as relaxation factors in the data transfer, available.

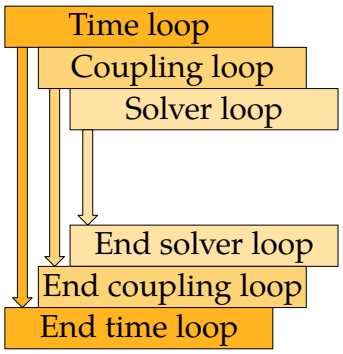


Figure 77-1. Illustration of the two-way iterative solving scheme

In contrast, LS-DYNA's explicit solver traverses through its (smaller) explicit time steps once at the start of each coupling time step and does not participate in coupling iterations. This coupling could be considered a one-way workflow, in that it does not involve coupling iterations. LS-DYNA uses the forces from the last Fluent step. Structural displacements are still sent to the fluid solver; it is merely the coupling loop that is omitted. This approach is, therefore, inherently less numerically stable than the fully implicit approach. Furthermore, no stabilization schemes are currently available to improve stability for this type of coupling. The only possible remedy for instabilities is to decrease the implicit time step on the fluid side.

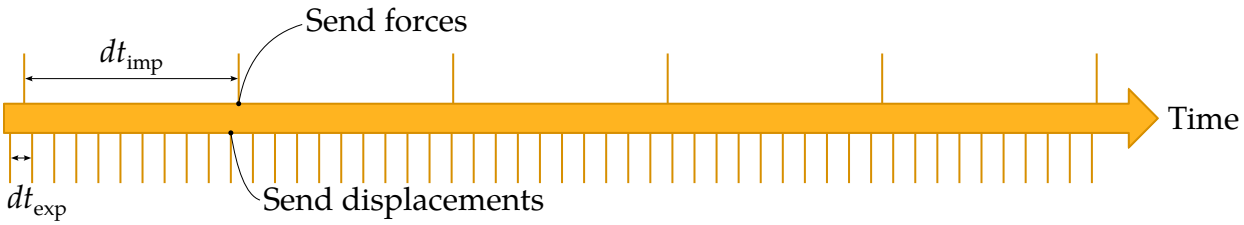


Figure 77-2. Illustration of one-way coupling

COUPLED SIMULATION SETUP AND POST-PROCESSING

This section covers the additional steps required to go from a standalone simulation with each solver to a coupled simulation.

LS-DYNA Input

For a coupled simulation, the LS-DYNA input deck needs three additional keywords: *SET_SEGMENT_N8, *COSIM_SYC_CONTROL, and *COSIM_SYC_INTERFACE. *SET_SEGMENT_N8 defines which regions in the model are to be coupled. For linear elements, the segments are described by three or four nodes, whereas second-order elements require six or eight nodes. This specification differs from how segment sets are normally created in LS-DYNA, which only have corner nodes. However, both the generation and visualization of *SET_SEGMENT_N8 sets can be made in LS-PrePost 4.13 and later. *COSIM_SYC_CONTROL defines the coupling and the general co-simulation settings. Lastly, *COSIM_SYC_INTERFACE specifies the variables to be exchanged across the different coupling regions.

Fluent Input

On the Fluent side, you must add Dynamic Meshing to allow the fluid mesh to deform as the structure deforms. To couple the interface to LS-DYNA and allow System Coupling to modify it, select the interface in Fluent and assign the System Coupling Dynamic Mesh Zone type to it (see Figure 77-3). System Coupling, then, controls the deformation of these interfaces with information coming from the deformations calculated in LS-DYNA.

In the Figure 77-3 (cf. the reed valve example presented below), the deformation of the coupling interfaces also affects the mesh on the adjacent symmetry planes. Therefore, these symmetry planes must also have a Dynamic Mesh definition, but with the Deforming type.

For Fluent, you need a System Coupling Participant setup file (.scp file), which can be generated from a Fluent .cas file. This .scp file is XML-formatted and provides System Coupling with the participant-specific information needed to connect the participants to a coupling run. It can be generated from a .cas file from Fluent. Note that this step is not required on the LS-DYNA setup side. The .scp file contains the regions where the data is transferred.

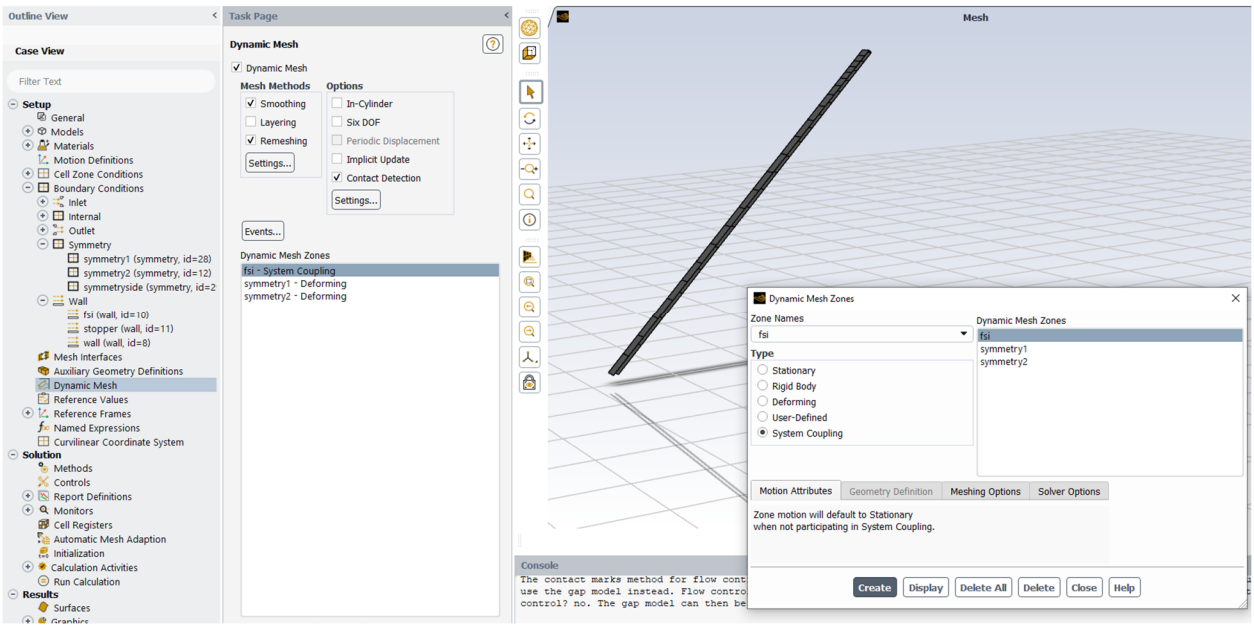


Figure 77-3. Specifying the coupled interfaces in Fluent

Setting up the Solution in the System Coupling GUI

To set up the coupled simulation run in the System Coupling GUI, the current workflow follows these steps:

1. Add the two participants by pointing at the keyword file for LS-DYNA and the .scp file for Fluent.

2. Add a coupling interface and select, for instance, Fluent as side one and LS-DYNA as side two, along with their respective defined coupling regions.

3. Add FSI data transfer.

4. Select the LS-DYNA executable to be used.

5. Fill in End Time, Time Step Size, and Minimum and Maximum Iterations to fit your needs.

The rest is automatically set, but with many options for modifications if needed. Currently, it is also necessary to activate the hidden beta features in the System Coupling GUI.

To see if the regions of the two sides match, you can:

- Check the mapping summary given by System Coupling, or

- Activate the "Live Visualization" to see the interfaces in the System Coupling GUI.

Post-Processing Interface Results

Post-processing of the coupled simulation can be performed in a standard manner on each side of the coupled run. In other words, to post-process the structural results, use, for instance, LS-PrePost, and to post-process the fluid results, use the Fluent GUI.

For the transferred forces and displacements, the System Coupling GUI provides a direct link to Ansys EnSight™, a general-purpose post-processing software. As an example, Figure 77-4 shows a vector plot of the transferred nodal forces applied to the LS-DYNA structure.

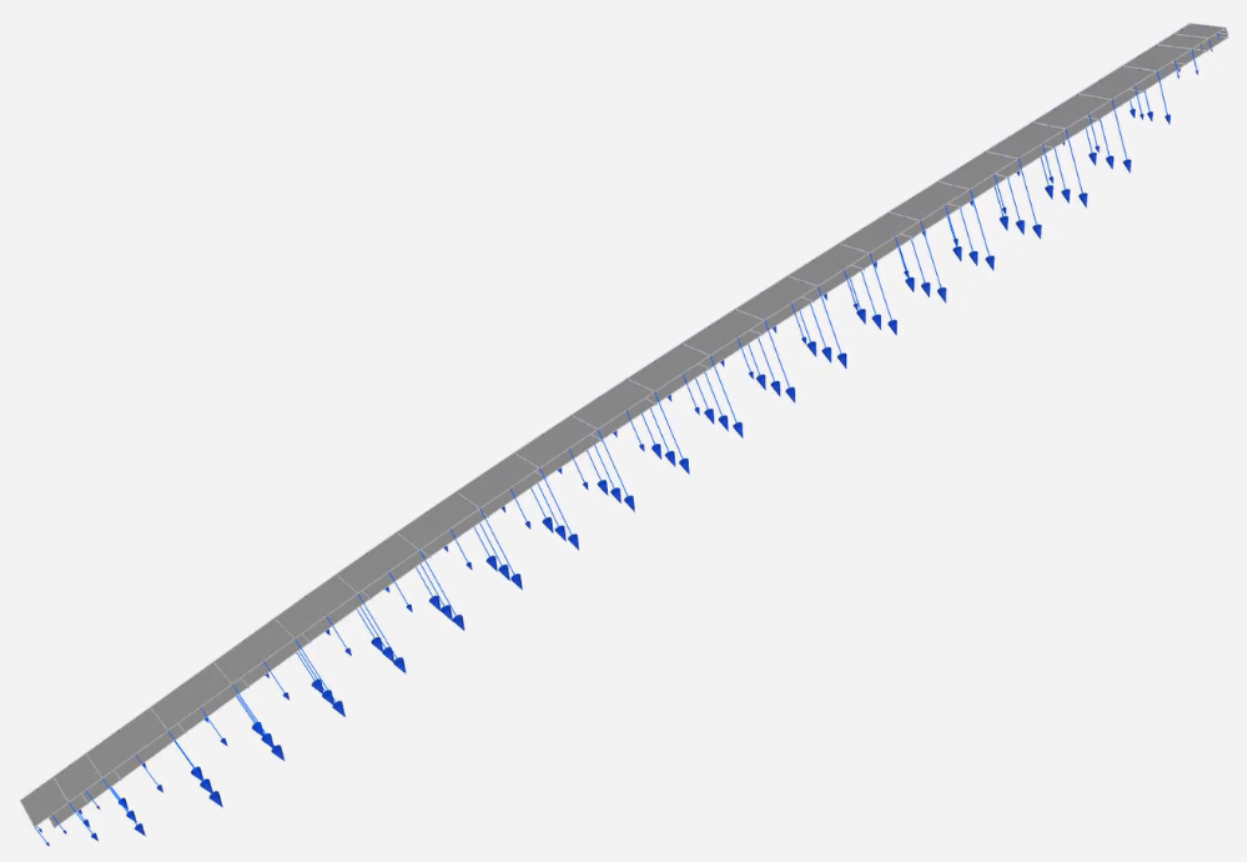


Figure 77-4. A vector plot of the transferred forces from Fluent to LS-DYNA

RESTARTS

Restarts are supported for System Coupling analyses. However, the LS-DYNA input deck cannot be modified, meaning only a simple restart. The System Coupling software controls the restart. See Stops and Restarts of System Coupling Analyses in the System Coupling User's Guide for details.

APPENDIX Z

EXAMPLES

In this section, we will present several examples. The input files for the examples are available in a zip file at https://lsdyna.ansys.com/wp-content/uploads/2026/08/system_coupling_examples.zip.

Reed Valve Example

To demonstrate the coupling, we have a simple reed valve example. The fluid domain modeled in Fluent has a pressure inlet boundary condition applied at one end and a zero-gauge-pressure outlet at the other. The deformable reed is modelled in LS-DYNA with an elastic material. Our example set includes versions for both implicit and explicit LS-DYNA.

Here, we will discuss setting up the coupling using a provided Python script imported into the System Coupling GUI and solely using the GUI. Note that this discussion and the accompanying screenshots are from the 2025R2 version of the System Coupling GUI.

Implicit LS-DYNA version with Python Script

The Python script run.py, located in the reed_valve/implicit_mpi folder, sets up the coupling and runs the simulation. You only need to change the path to the LS-DYNA executable. After you modify this path, you can run the Python script in the System Coupling GUI.

The System Coupling GUI first has you set the working directory. Select the folder that includes the folder containing the LS-DYNA input deck and the folder containing the Fluent .scp, .dat, and .cas files. Once the System Coupling GUI loads, select File → Run Script. The GUI will then have you select the Python script.

Alternatively, this Python script can be run in batch mode with `[path to System Coupling binary] -R run.py`.

Implicit LS-DYNA version with the System Coupling GUI

When you run the System Coupling program, it first prompts you to select the working folder. This working folder should include the folder containing the LS-DYNA input deck and the folder containing the Fluent .scp, .dat, and .cas files.

After the System Coupling GUI loads, select Setup and then Activate Hidden Features in the drop-down menu. This places Hidden Feature Control in the tree under Setup. Click Hidden Feature Control and toggle the boolean next to Beta Features from False to True.

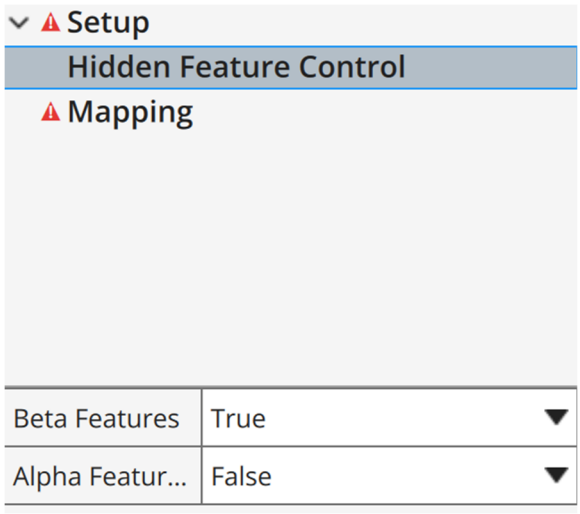


Figure 77-5. Enable beta features

The next step is to add the participants to the coupling. We will add Fluent first. Click Setup → Add Participant. In the pop-up box, make sure the Input File box is checked. Then click the button to specify the location of the .scp and click Add.

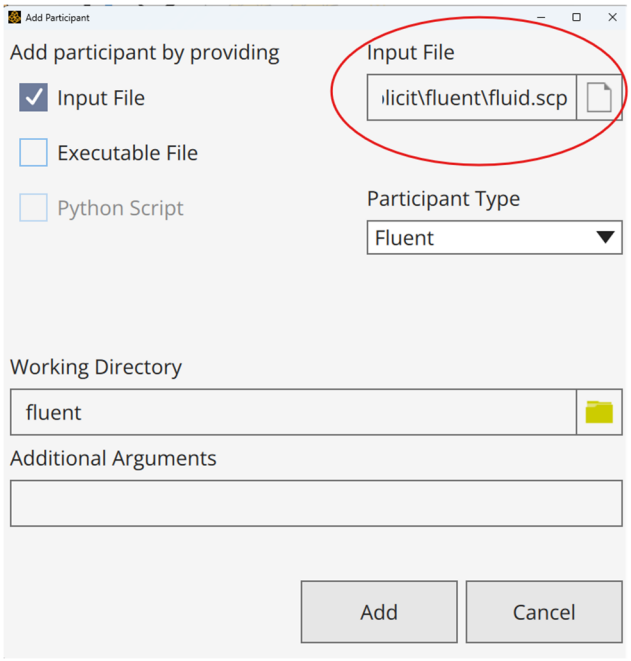


Figure 77-6. Add Fluent as a participant

This will populate the tree with Fluent as a Coupling Participant. Only one item needs to be changed. Click on Fluent Input under Execution Control, and in the Option drop-down menu, change InitialCase to InitialCaseAndDateFile. Then add the fluent.dat.h5 file as the Data File.

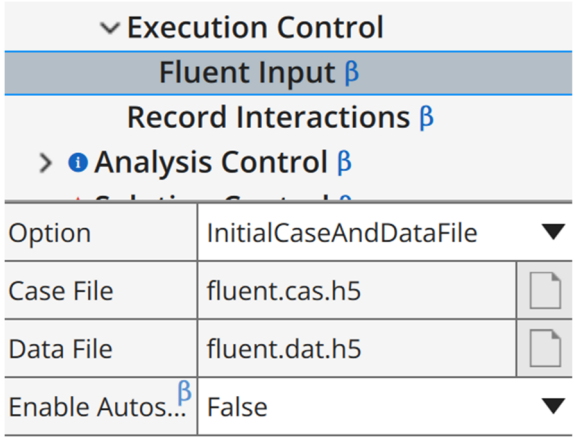


Figure 77-7. Add the .dat.h5 file for Fluent

Next, add LS-DYNA as a coupling partner. To do this, again select Setup → Add Participant. Then, check both the input file and executable file check boxes and browse for each of them, respectively. After hitting add, go to Execution Control for LS-DYNA and change Option from UserDefined to ProgramControlled (see Figure 77-9). Otherwise, System Coupling will fail to connect to LS-DYNA during the solve.

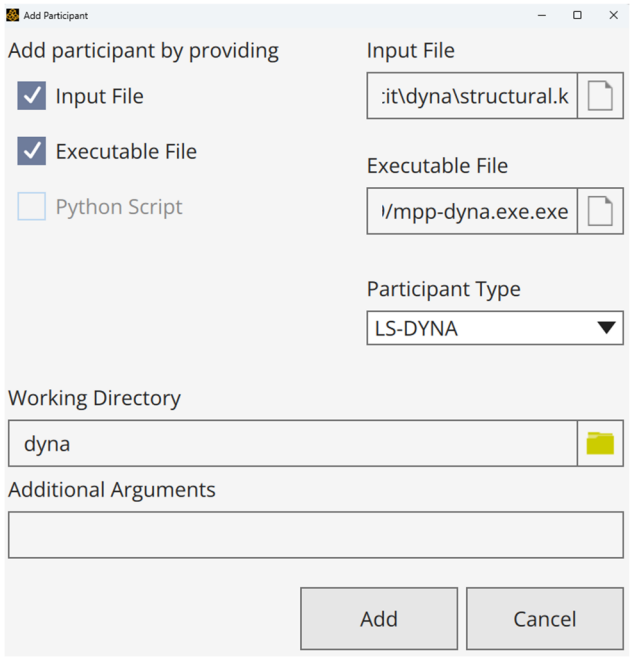


Figure 77-8. Adding LS-DYNA as a coupling participant

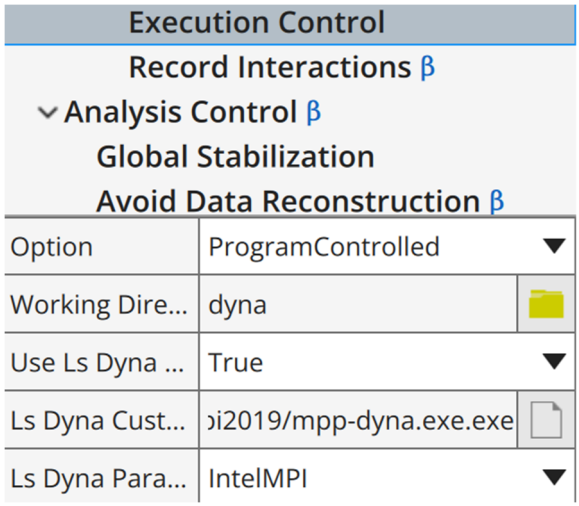


Figure 77-9. Screenshot of Execution Control for LS-DYNA. Option must be set to ProgramControlled with IntelMPI set as the LS-DYNA Parallel Option.

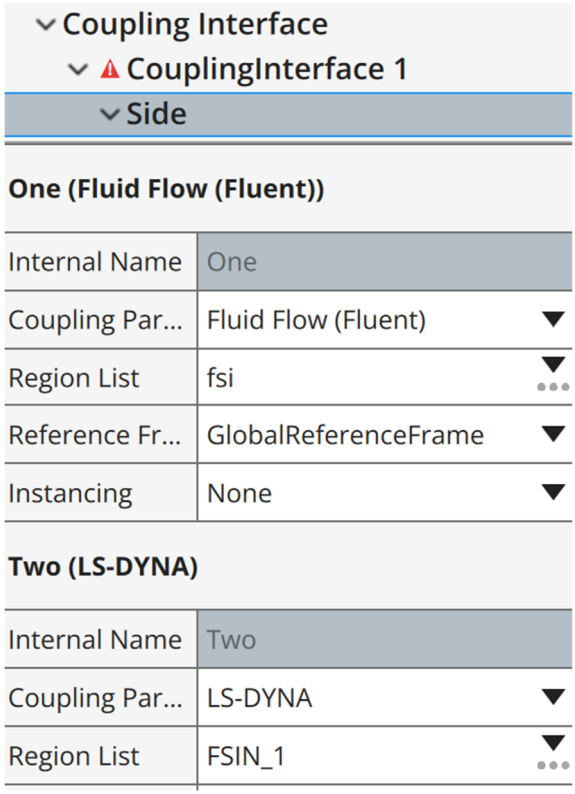


Figure 77-10. Example of how to set the sides in the coupling interface

The next step is to add a coupling interface for sharing data between the two solvers (see Figure 77-10). To do this, select Setup → Add Coupling Interface. Coupling Interface will now appear in the tree. Select Side in the specific coupling interface under the tree. Pick Fluent as the first coupling participant and LS-DYNA as the second. For Fluent, select fsi from the Region List, and for LS-DYNA select FSIN_1.

After this, right-click the name of the coupling interface as shown in Figure 77-11 and select Add Data Transfer. Next, rename the data transfer to force, just to keep track of the transfers more easily, and select LS-DYNA as the target side (see Figure 77-12). We now need to add a second data transfer for Fluent to receive the displacements from LS-DYNA. To do this, you can repeat the step of right-clicking the name of the coupling interface and selecting Add Data Transfer as before, or you can right-click Data Transfer under the coupling interface in the tree and select Add. As before, rename this transfer, but in this case, to displacement, and keep Fluent as the target side.

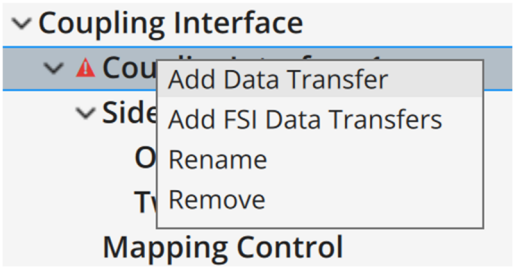


Figure 77-11. Add the data transfer for the coupling interface

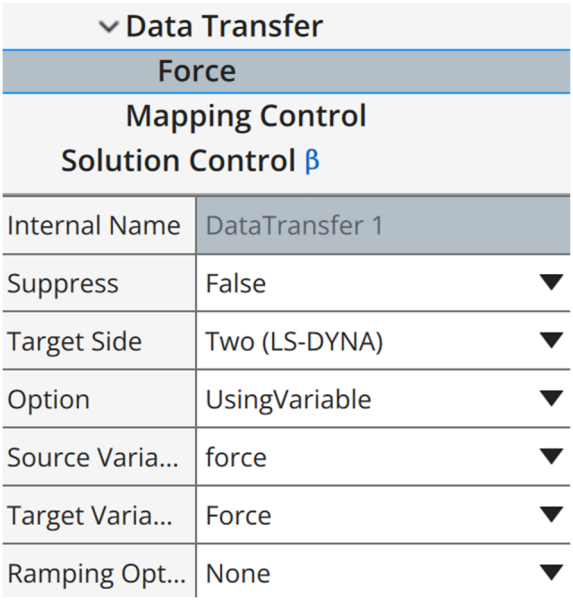


Figure 77-12. Setting up the data transfer for force with LS-DYNA as the target side

The final step in finishing the setup is to select Results under Output Control in the tree and change Option to EveryStep.

Explicit LS-DYNA Version with Python Script

The steps for using a Python script in the coupling for the explicit version are the same as the implicit version. Note that the scripts differ between the two versions in that the end time and time step size must be set in this case.

Explicit LS-DYNA version with the System Coupling GUI

The System Coupling GUI automatically detects that the LS-DYNA input deck is for an explicit run. Thus, all the steps in the Implicit LS-DYNA Version with the System Coupling GUI remain the same. However, in this case, the Time Step Size must be set in Solution Control.

Turek-Hron FSI Benchmark

We also have the Turek-Huron benchmark available as an example [Turek and Hron 2006]. It was used to validate the coupling results. This problem is inherently unstable; the flexible structure deforms significantly under fluid pressure, resulting in self-induced oscillations. This problem can pose numerical stability challenges, particularly when using a more elastic material or when the fluid and solid densities are of similar magnitudes. We have straightforwardly adopted the parameters from the setup named FSI2 in the paper [Turek and Hron 2006]. See Figure 77-13 and Table 77-1.

Table 77-1. Model parameters appleid in the Turek-Hron benchmark example

Parameter	Value according to FSI2 setup
Solid	
Density ρ^s [10^3 kg/m^3]	10
Poisson's ratio, ν^s	0.4
Lamés second constant (shear modulus), μ^s [10^6 kg/(ms^2)]	0.5
Fluid	
Density ρ^f [10^3 kg/m^3]	1
Kinematic viscosity, ν^f [10^{-3} m^2/s]	1
Mean inflow velocity, $\overline{U}$ [m/s]	1

We applied the two-way iterative coupling approach (implicit LS-DYNA) in this example. We used an implicit time step of 0.01 s. To aid convergence, we applied a relaxation factor of 0.5 on the force transfer, without which the solution became unstable. No other stabilizations were required. Note that we also attempted to solve this example by using the explicit solver of LS-DYNA with an unchanged time step on the Fluent side, but this solution approach was found to be unstable.

In this case, we provide the settings.h5 file to set up the coupling. You have to change the LS-DYNA executable in Execution Control part of the tree under LS-DYNA. Note that this example only works for LS-DYNA versions R16.2 and later and R17 and later.

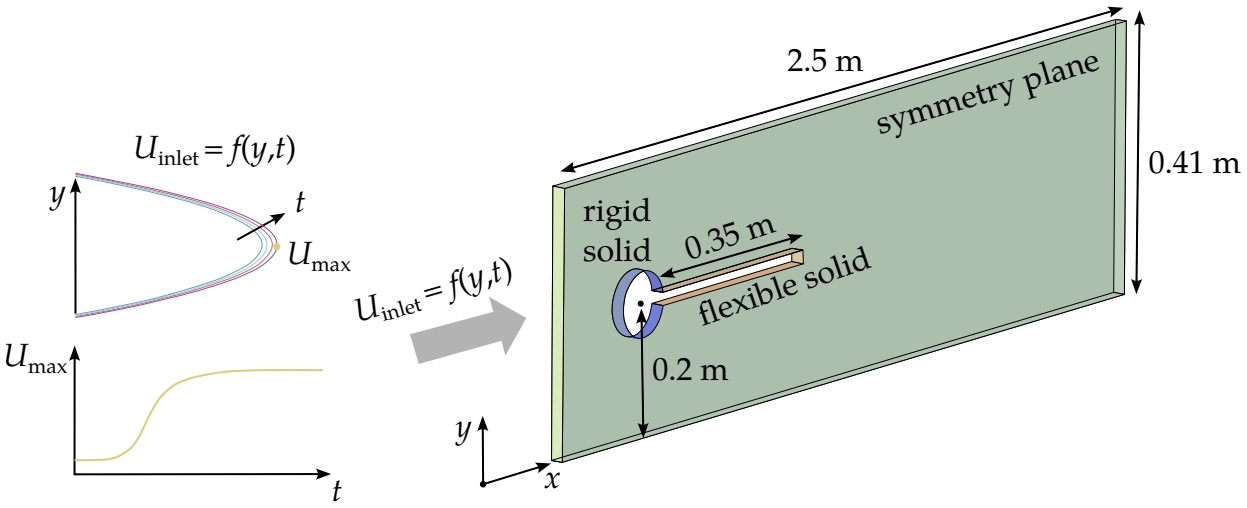


Figure 77-13. Geometry and flow profile for the Turek-Hron FSI benchmark example

LIMITATIONS

A current limitation is that System Coupling is not allowed to control the time step in LS-DYNA. In two-way coupling, you must manually make sure that the time steps are equal on both sides. A fix for this has already been implemented in the development version of LS-DYNA but is yet to be tested. When tested and validated, this limitation will be lifted in future LS-DYNA releases.