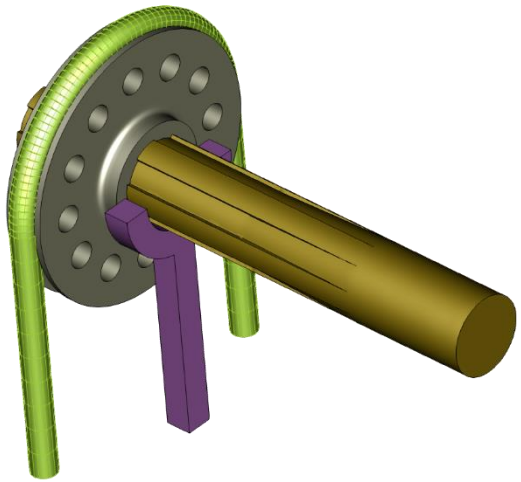
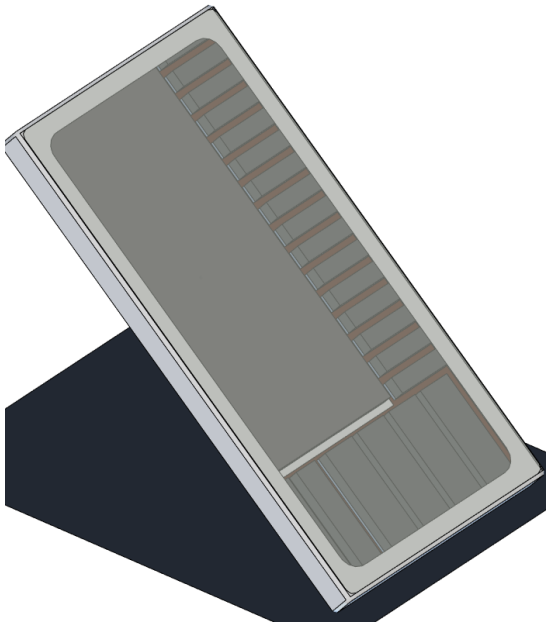
Load history management using *CASE

with dynain.lsda and Mortar contact



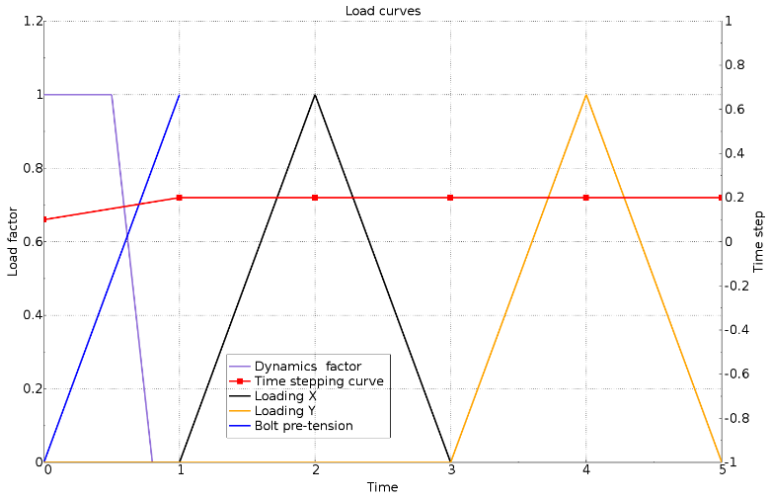


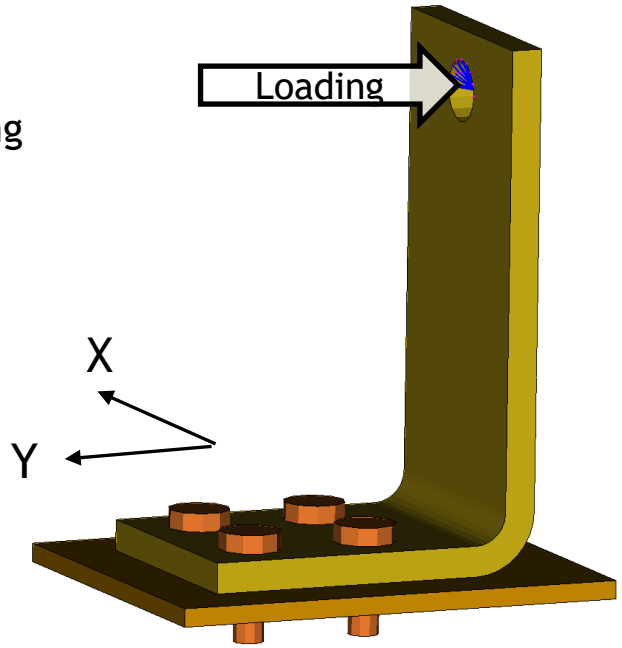
Overview

- Load history management
  - How to treat analyses of a series of operations
- Background
  - `*CASE` and `dynain`
- Examples
  - Different workflow options
  - The Solution Explorer in LS-PrePost
  - Applications
- Summary

Example: Loading of a L-bracket

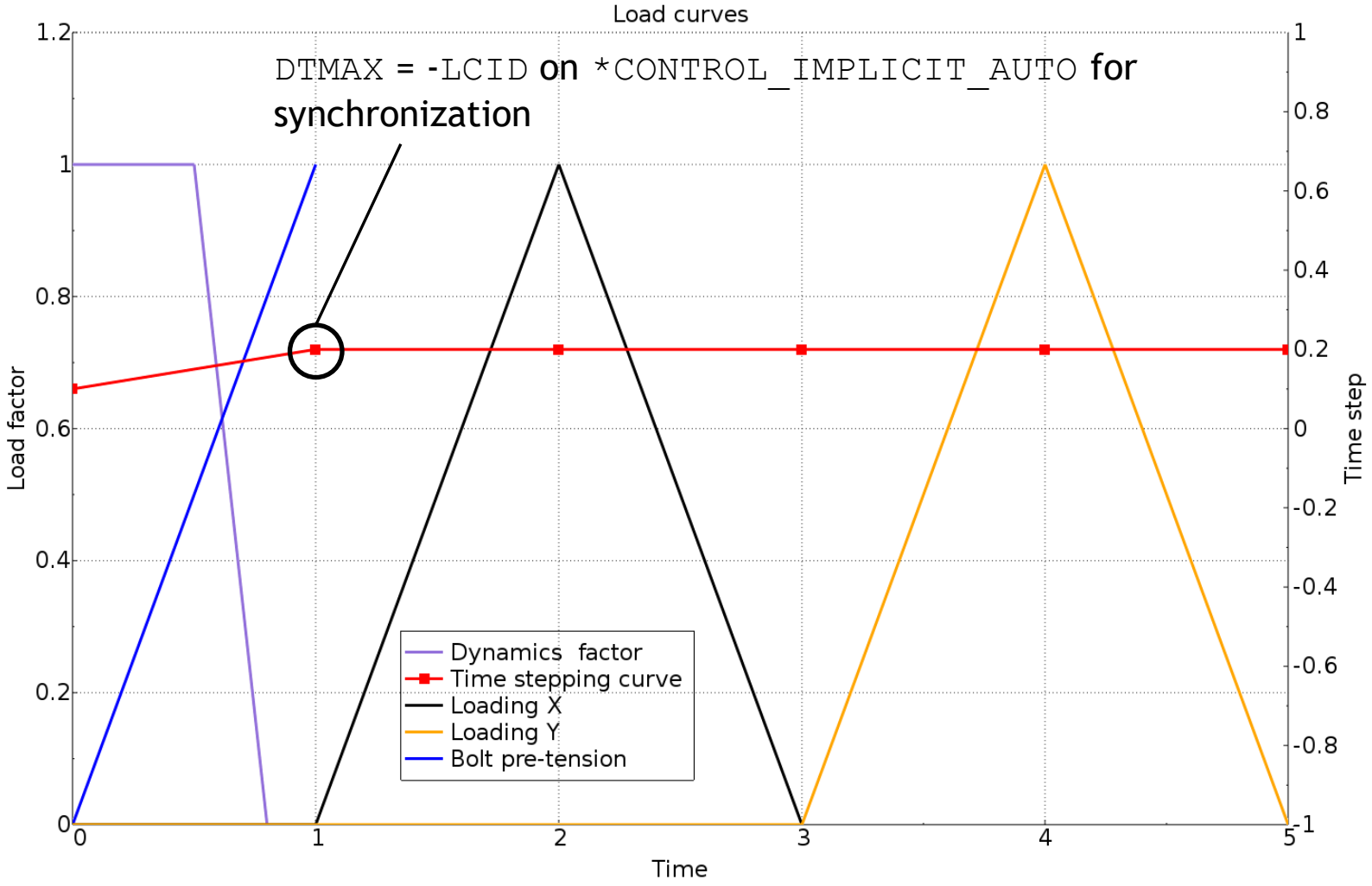
- Load history:
  - A structure is subjected to 1 kN loading/unloading in the X-direction, followed by
    1 kN loading/unloading in the Y-direction.
  - But first, bolt pre-tensioning is applied
- Traditional "time-domain" set-up:
  - This can be seen as five "load steps"
  - Set `endtim` = 5 on `*CONTROL_TERMINATION`.
  - Define load curves for pre-tensioning, X-loading and Y-loading
  - Set DTMAX = -LCID on `*CONTROL_IMPLICIT_AUTO`.





The concept of time in implicit analyses

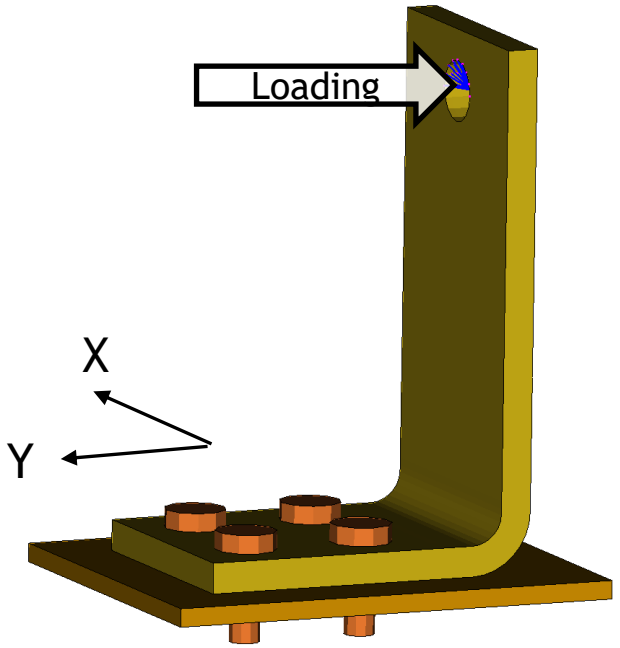
- The load curves(*) required to set-up this load case in one image



Example: Loading of a L-bracket

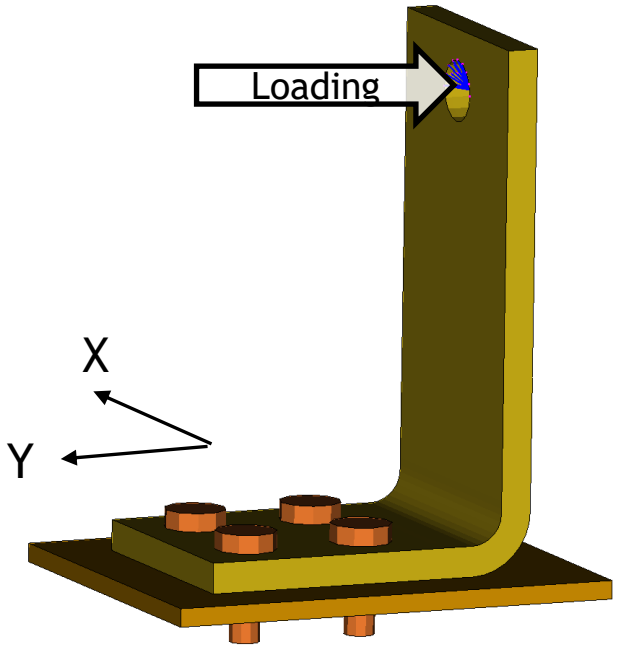
- Load history:
  - A structure is subjected to 1 kN loading/unloading in the X-direction, followed by
    1 kN loading/unloading in the Y-direction.
  - But first, bolt pre-tensioning is applied
  - This can be seen as three “load stages”

1. Bolt pre-tensioning
2. X-loading and un-loading
3. Y-loading and un-loading



Example: Loading of a L-bracket

- Load history:
  - A structure is subjected to 1 kN loading/unloading in the X-direction, followed by
    1 kN loading/unloading in the Y-direction.
  - But first, bolt pre-tensioning is applied
  - This can be seen as three “load stages” or `*CASE`s
- Case 1: Bolt pre-tensioning
  - Use initial implicit dynamics to overcome rigid-body modes
- Case 2: X-loading and un-loading
  - Start from pre-loaded configuration: case1.dynain.lsda
- Case 3: Y-loading and un-loading
  - Start from previous case: case2.dynain.lsda
- Run the cases in a sequence using LS-DYNA’s case driver



Background

- `*CASE` in LS-DYNA
  - Functionality for setting up many analyses in one keyword file
  - Each analysis is called a case
  - Cases are run sequentially
  - Each case “has its own time”, t = 0 .. ENDTIM
  - Separate families of results: case1.d3plot, case1.d3hsp, case2.d3plot ...

- `dynain` files
  - A keyword file containing the final state of an analysis
  - Deformed geometry (`*NODE`, `*ELEMENT_SHELL_THICKNESS`, ...)
  - Elements(*)
  - Residual stress state (`*INITIAL_STRESS_`...) and history variables

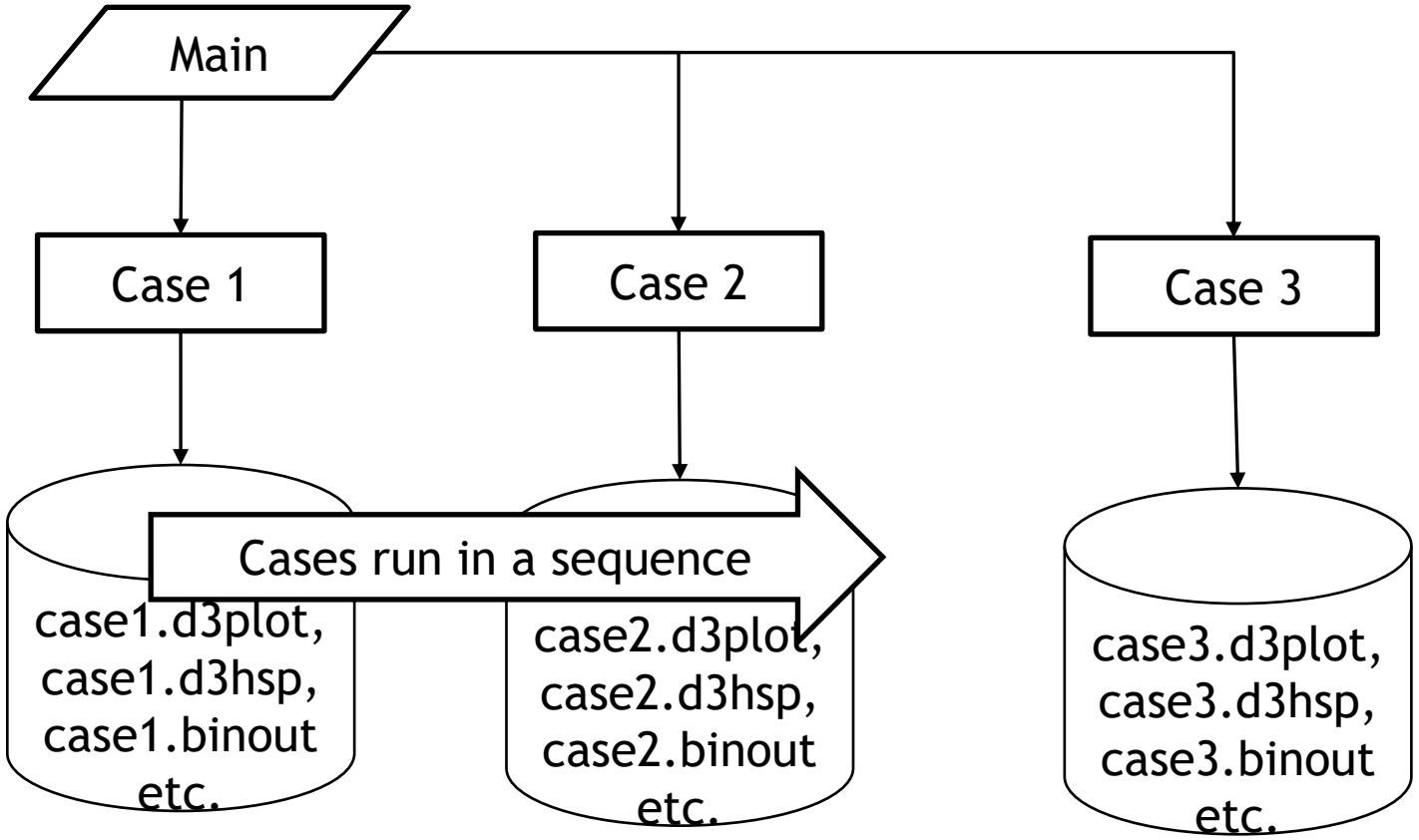
- Combining `*CASE` and `dynain` files: a new approach for multi-step (implicit) analyses in LS-DYNA

Background

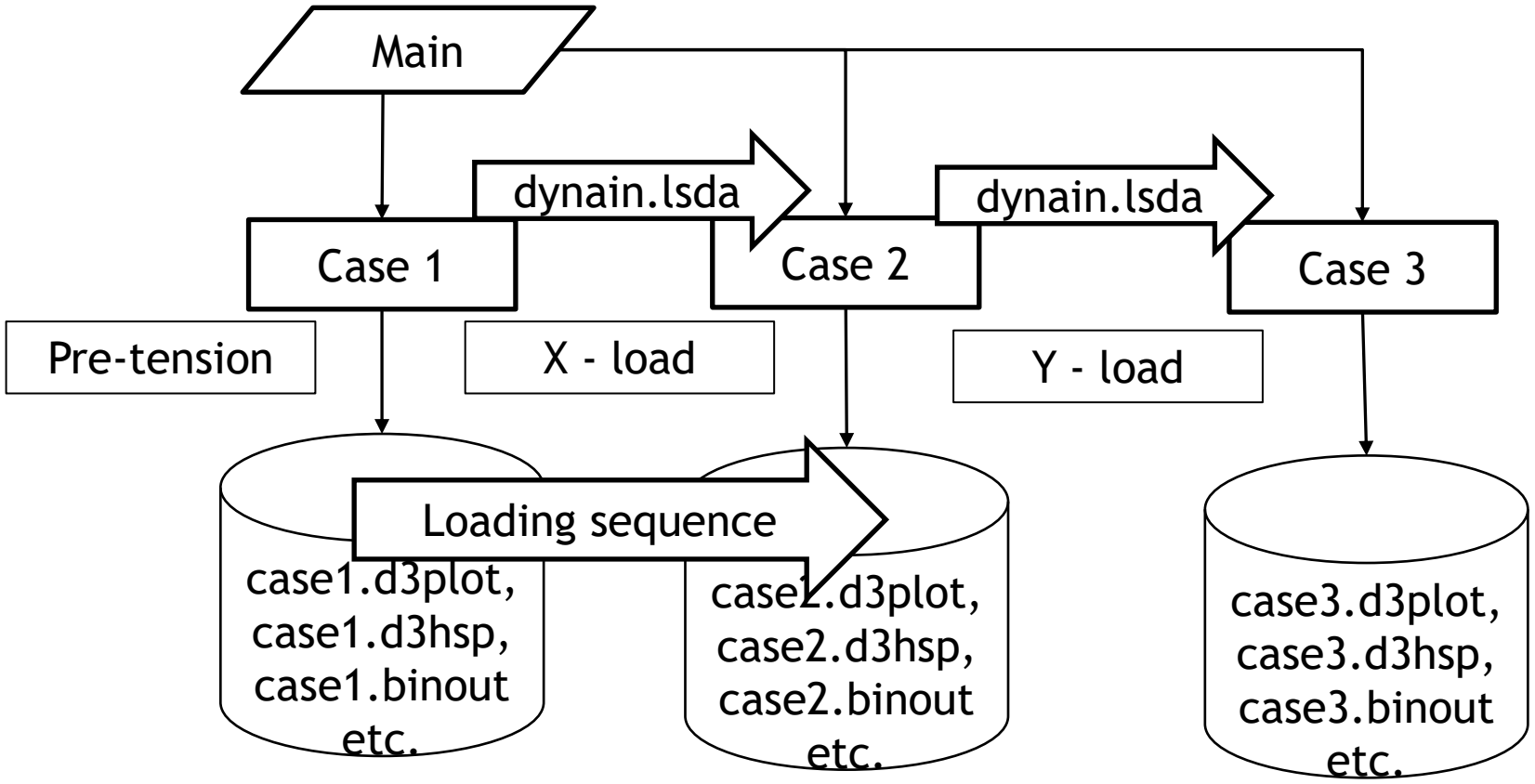
- *CASE in LS-DYNA
  - Functionality for setting up many analyses in one keyword file
  - Each analysis is a called a case
  - Cases are run sequentially
- dynain files
  - A keyword file containing the final state of an analysis
  - Deformed geometry (*NODE, *ELEMENT_SHELL_THICKNESS, ...)
  - Elements(*)
  - Residual stress state (*INITIAL_STRESS_...) and history variables

- Combining *CASE and dynain files: a new approach for multi-step (implicit) analyses in LS-DYNA
  - For a background, see
    Borrvall, T., Solution explorer in LS-PrePost- a GUI for Nonlinear Implicit FE, 12th European LS-DYNA conference, Koblenz 2019.
  - Some information provided in the recently (21-04-19) updated "Guideline for implicit analyses using LS-DYNA"
  - Appendix in coming revisions of the keyword manual

The *CASE - concept



The *CASE / dynain.lsda - concept



The *CASE / dynain.lsda - concept

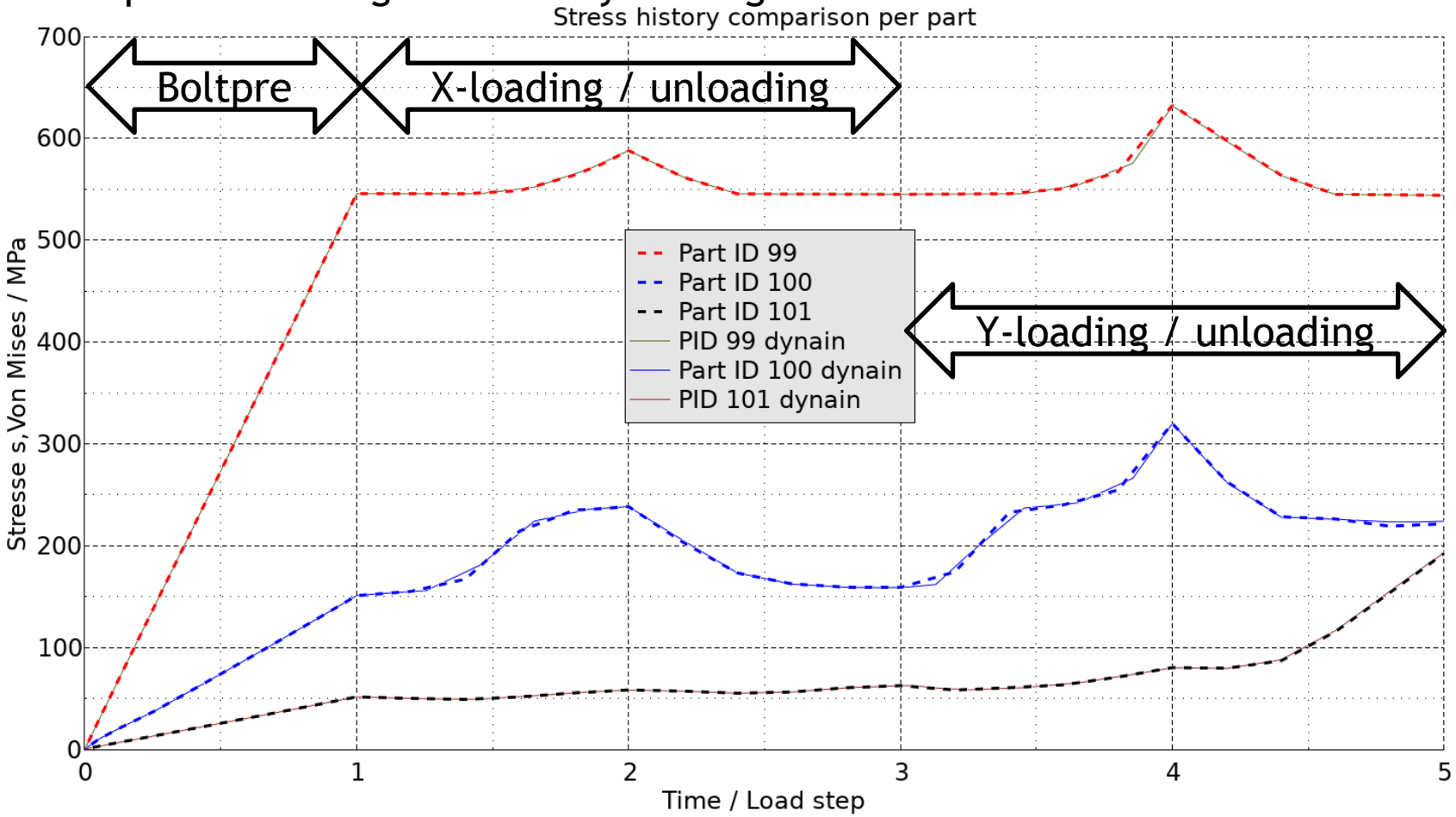
- Analysis of a sequence of load steps is run as a sequence of analyses
  - A more intuitive way of setting up a load history using load steps
  - Clear definition of which features that are active in each step
  - Reduce need for birth/death times of contacts, boundary conditions etc.

- Information is propagated from step to step using a binary dynain file
  - *INTERFACE_SPRINGBACK_LSDYNA
  - *ftype* = 3, *cflag* = 1, ... ➔ **dynain.lsda**
  - Contains deformations, stresses, history variables, contact state etc.
  - **NOTE! Must use Mortar contacts!**
  - Can be opened for 3d visualization in LS-PrePost (from version 4.8)
- *CASE to describe the sequence of steps

- Easy to re-use pre-tensioned configuration is obtained after each step

Loading of L-bracket: Results comparison

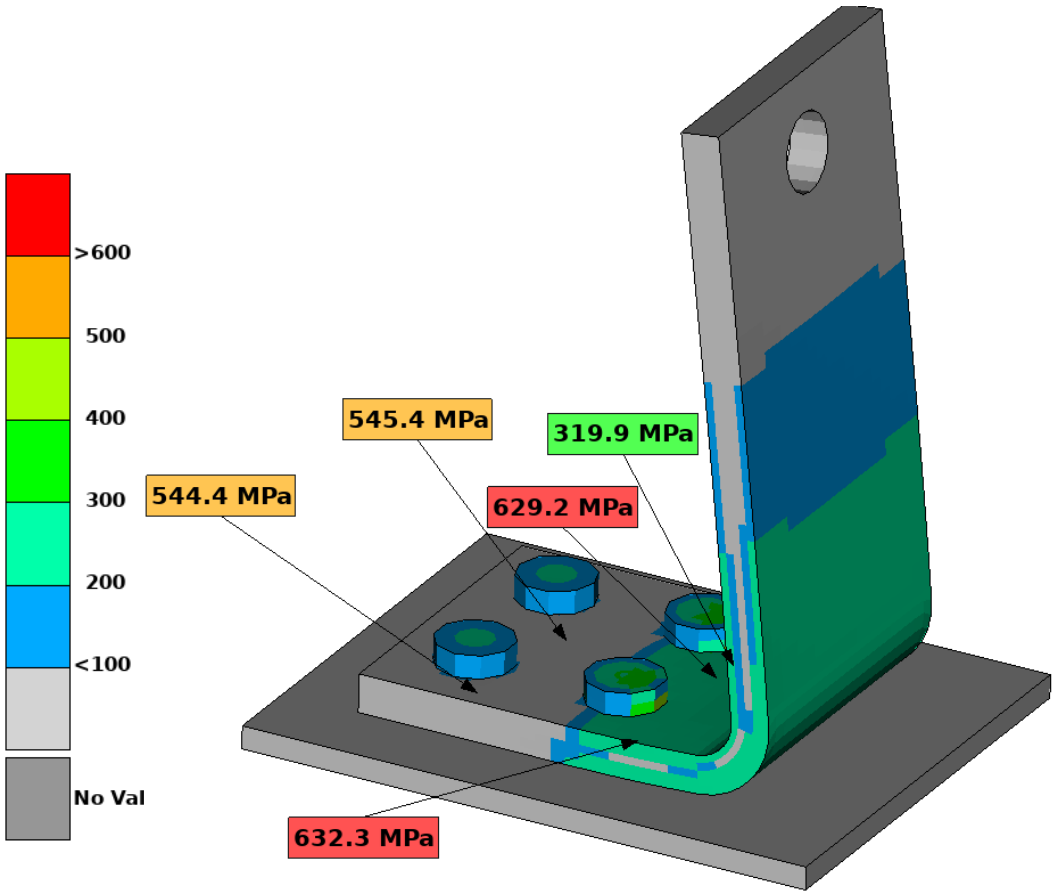
- Bolt pre-tensioning followed by loading



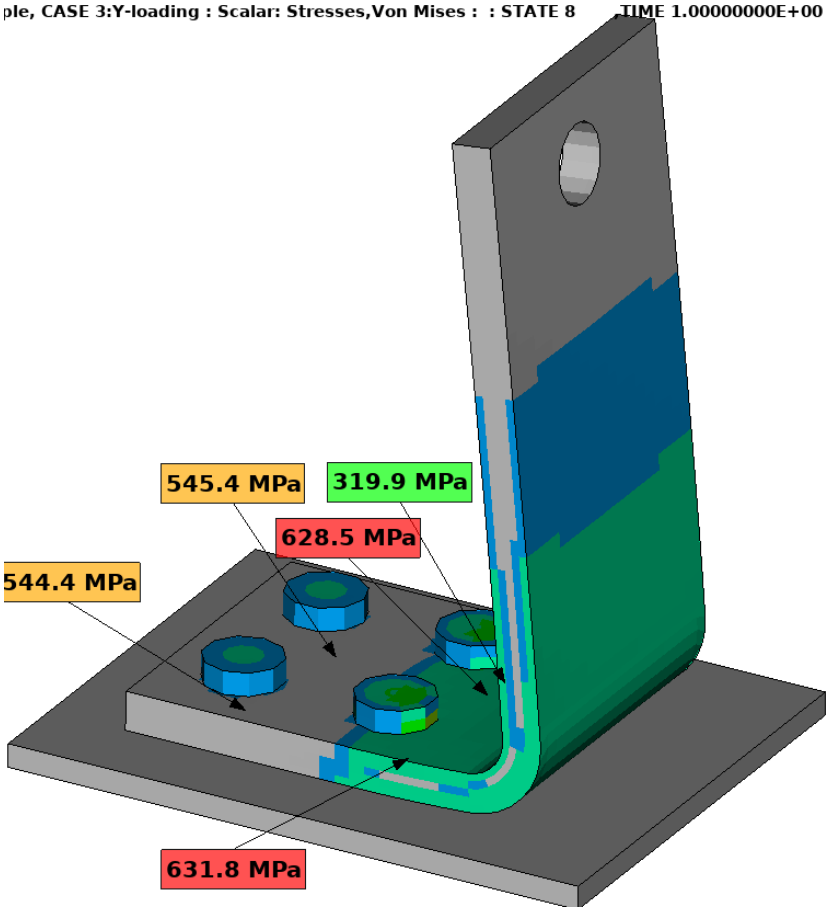
Stress history comparison: dynain.lsda w. solid lines, all-in-one with dashed

Loading of L-bracket: Results comparison

- Stress at peak Y-loading



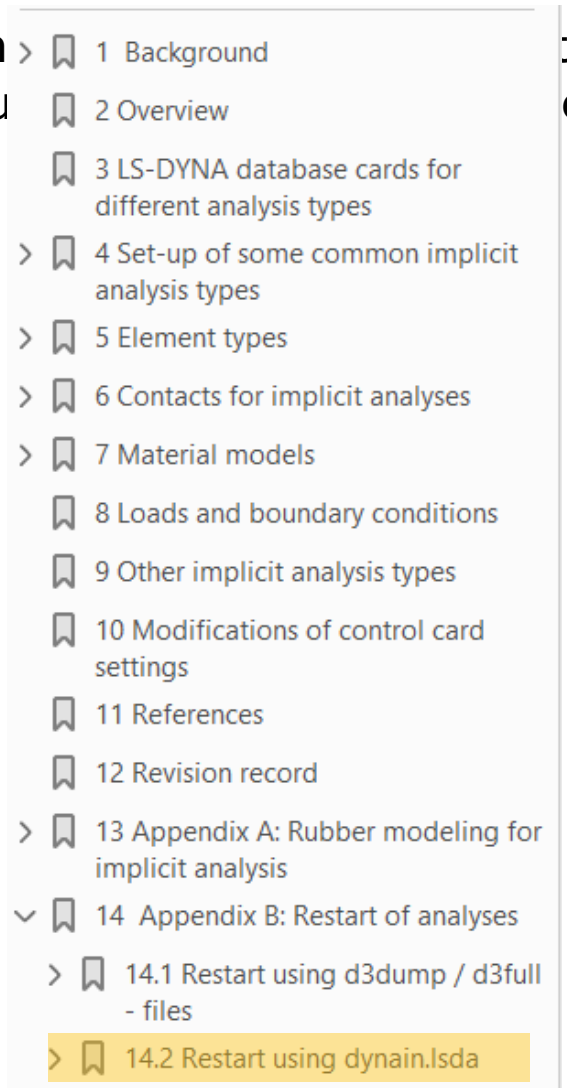
Peak stress time history

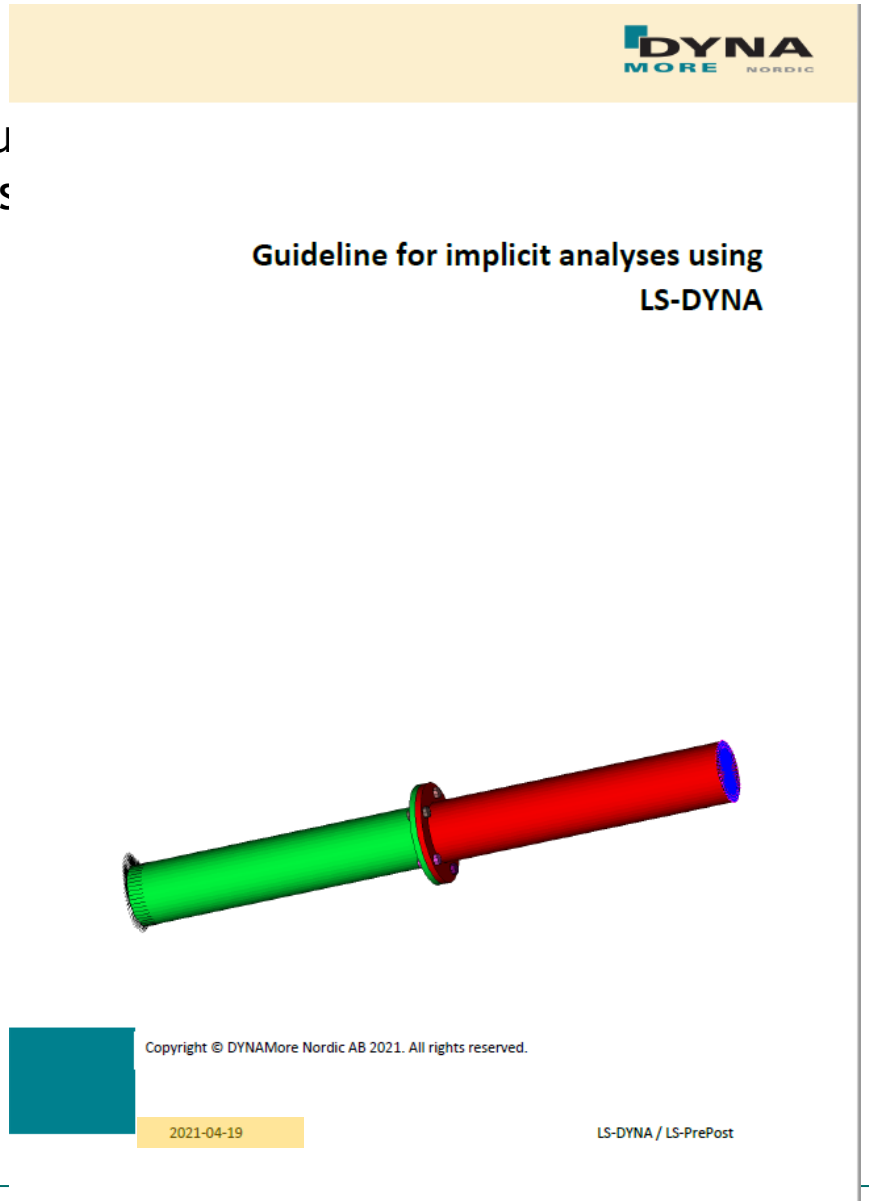


Peak stresses using dynain.lsda

Loading of L-bracket

- Th… …he u…
  Gu… …es us…





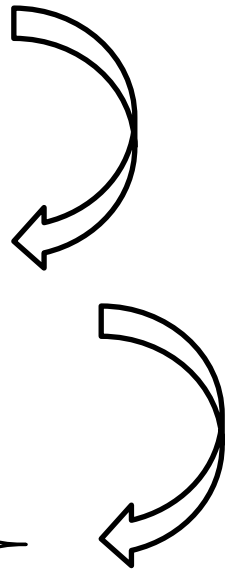
The *CASE / dynain.lsda - concept

- By use of the *CASE / dynain.lsda - concept, an analysis consisting of a sequence of separate loadings or load steps are divided into a sequence of LS-DYNA analyses.
  - Available from LS-DYNA version R12.0.0 and R11.2
  - Under development: More features added continuously
- Information is propagated via dynain.lsda - files from one load step to the next
  - Binary file containing deformed geometry, initial stresses, history variables, contact state from **Mortar contacts**, etc.
  - A bit similar to a restart file
  - Generated by the keyword *INTERFACE_SPRINGBACK_LSDYNA
  - Full flexibility to modify control cards settings, add parts etc. between cases
- The cases are run automatically in a sequence, using LS-DYNA's case driver
  - Requires case to be present on the run command line, for example
    mppdyna 10 i=run.key memory=100m case
  - Possible to run a selected set of cases
  - "Families" of output files: case1.d3hsp, case1.d3plot ..., case2.d3hsp, case2.d3plot ...
- Problem time re-starts at t=0 for each case

The *CASE - concept

- In keyword format (pseudo – code)

```
*KEYWORD
*CASE_BEGIN_1
*INCLUDE
run_pretens.key
*CASE_END_1
*CASE_BEGIN_2
*INCLUDE
case1.dynain.lsda
*INCLUDE
run_xload.key
*CASE_END_2
*CASE_BEGIN_3
*INCLUDE
case2.dynain.lsda
*INCLUDE
run_yload.key
*CASE_END_3
*END
```

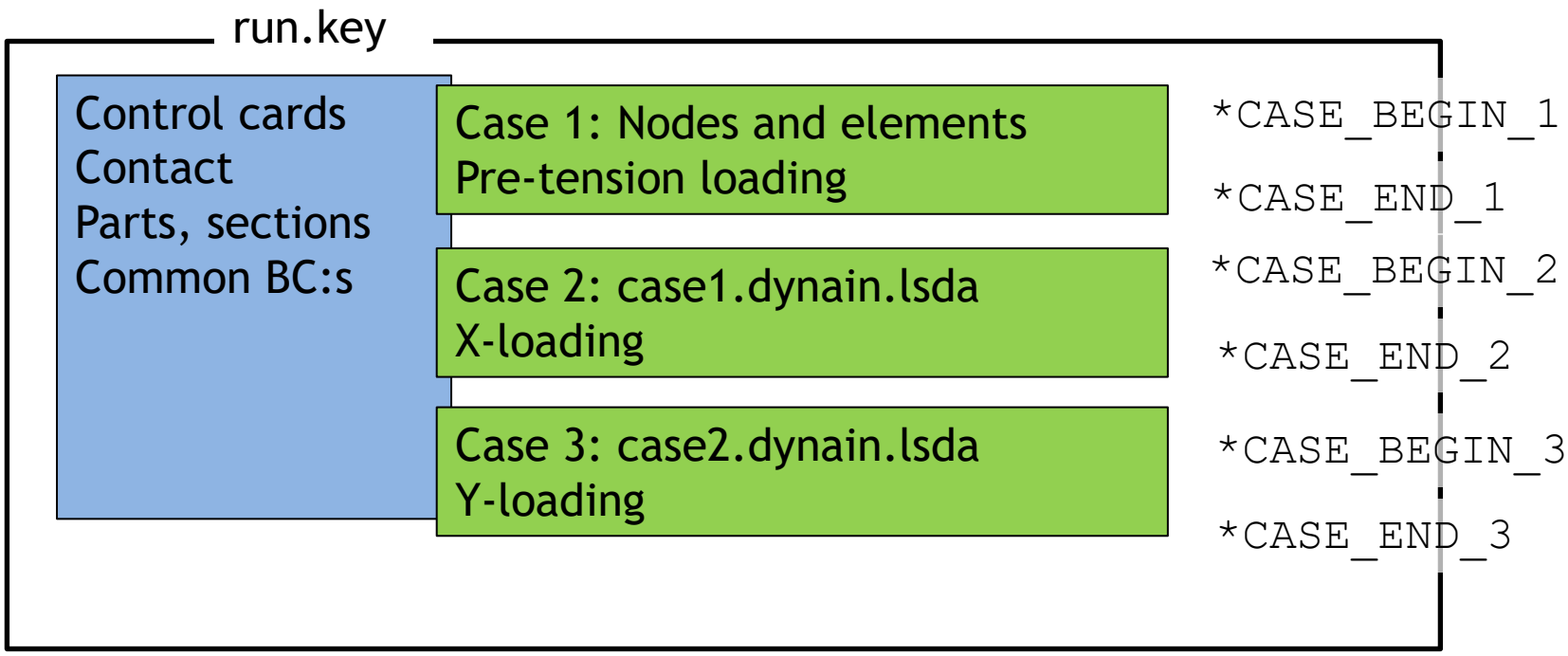


case1.dynain.lsda

case2.dynain.lsda

The *CASE - concept

- Keywordfile partitioning



The *CASE - concept

- In keyword format (pseudo – code)

```
*KEYWORD
*CONTROL_...
Control cards common to all cases
*CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR_ID
Common contacts
etc. other common features
...
*CASE_BEGIN_1
*INCLUDE
geo001.key
*CONTROL_TERMINATION
...
Other specific controls
*LOAD_...
*BOUNDARY_...
*INTERFACE_SPRINGBACK_LSDYNA
...
*CASE_END_1
```

```
*CASE_BEGIN_2
*INCLUDE
geo001_parts_and_sections.key
*INCLUDE
case1.dynain.lsda
*INCLUDE
geo002.key
*CONTROL_TERMINATION
...
Other specific controls
*LOAD_...
*INTERFACE_SPRINGBACK_LSDYNA
...
*CASE_END_2
.
.
.
*CASE_END_N
*END
```

The *CASE - concept

- Keyword template for requesting dynain.lsda

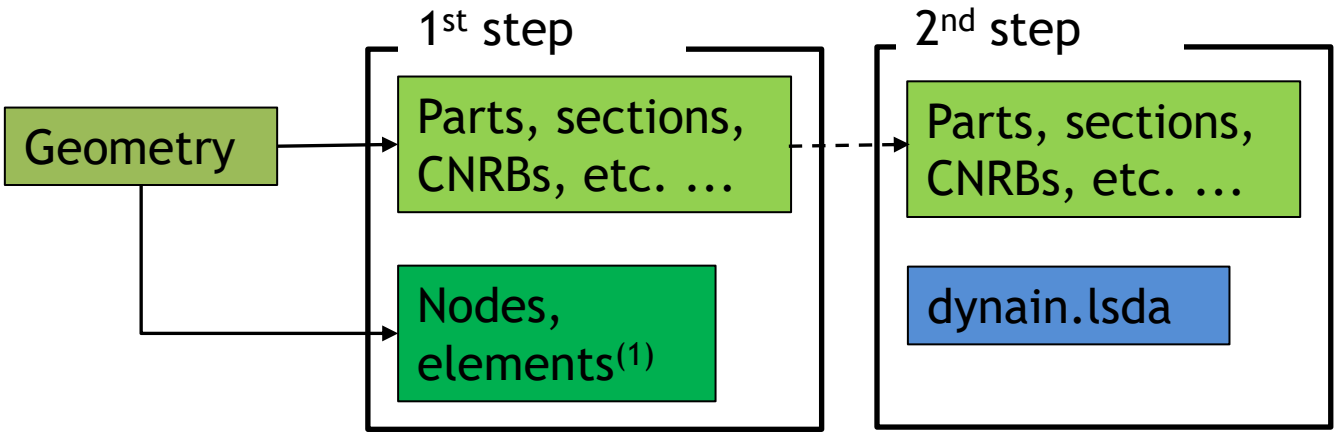
```
*SET_PART_LIST_GENERATE_TITLE
All parts for dynain.lsda
       338
         1  99999999
*INTERFACE_SPRINGBACK_LSDYNA
$#    psid       nshv      ftype          _     ftensr     nthhsv      rflag    intstrn
       338        999          3          0          0          0          1          0
$#    optc       sldo       ncyc     fsplit     ndflag      cflag      hflag
OPTCARD             0          0          0          1          1          1
*INTERFACE_SPRINGBACK_EXCLUDE
$#                                                                          kwdname
BOUNDARY_SPC_NODE
```

=3 gives dynain.lsda

supress output of boundary_spc cards to maintain flexibility in following steps

The *CASE - concept - Model partitioning

- The dynain.lsda file will contain node and element definitions for the deformed configuration
  - Initial stresses, plastic strain and history variables
  - Contact state etc.
- Double definition of nodes and elements is not allowed
- It will be required to partition the geometry information



Notes: (1) For example *ELEMENT_MASS will not be output to dynain

The *CASE - concept - Model partitioning

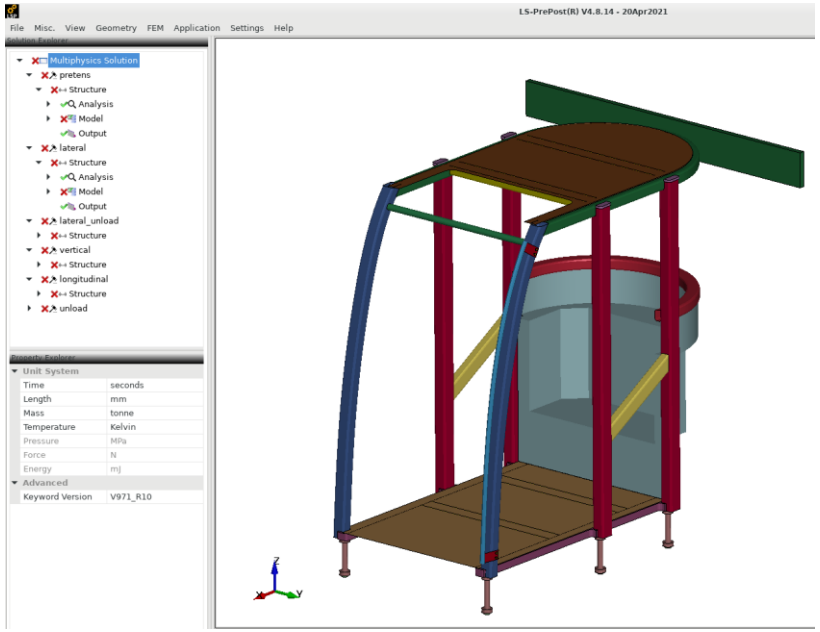
- The dynain.lsda file will contain node and element definitions for the deformed configuration
  - Initial stresses, plastic strain and history variables
  - Contact state etc.

Output to dynain.lsda	Not output to dynain
*NODE, *ELEMENT_... ,	*ELEMENT_MASS
	*CONSTRAINED_...
*BOUNARY(1)	*LOAD_...
*INITIAL_STRESS_...	*MAT_...,
*INITIAL_FOAM_REF...	
Contact state	*CONTACT_
	*PART, *SECTION
	*SET_...

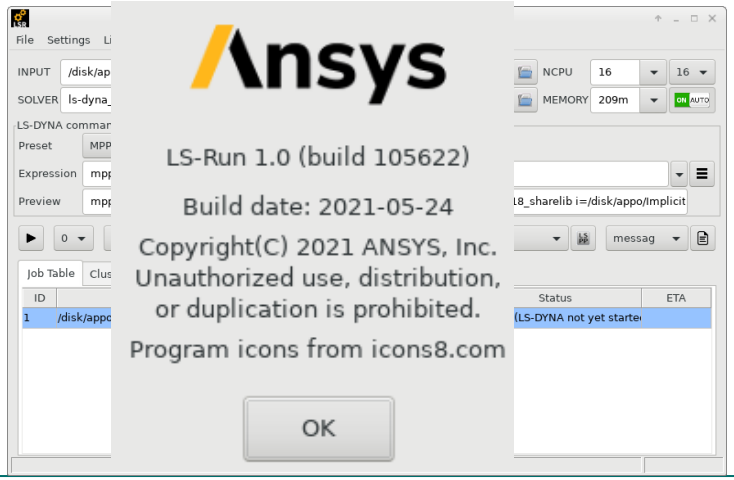
Notes: (1) Output of boundary cards in the dynain file is in many cases not desired (limits flexibility) and can be supressed

The *CASE / dynain.lsda - concept

- GUI support for the *CASE / dynain.lsda - concept is available
  - The Solution Explorer in LS-PrePost
  - From V4.6
  - LS-PrePost will auto-generate the required keywords, and partition model information accordingly
  - Including recommended implicit control card setting

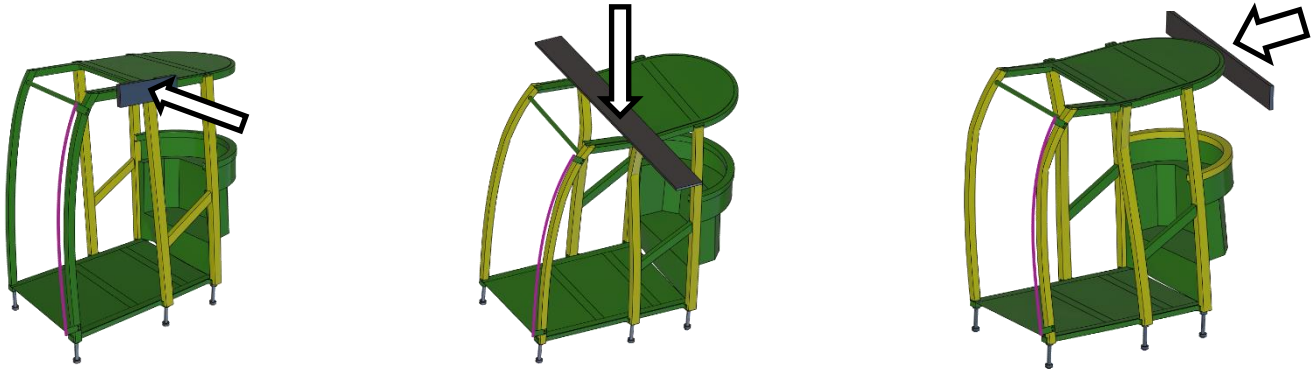


- GUI support for job submission
  - LS-Run, from v. 1.0.105622 including selection of cases

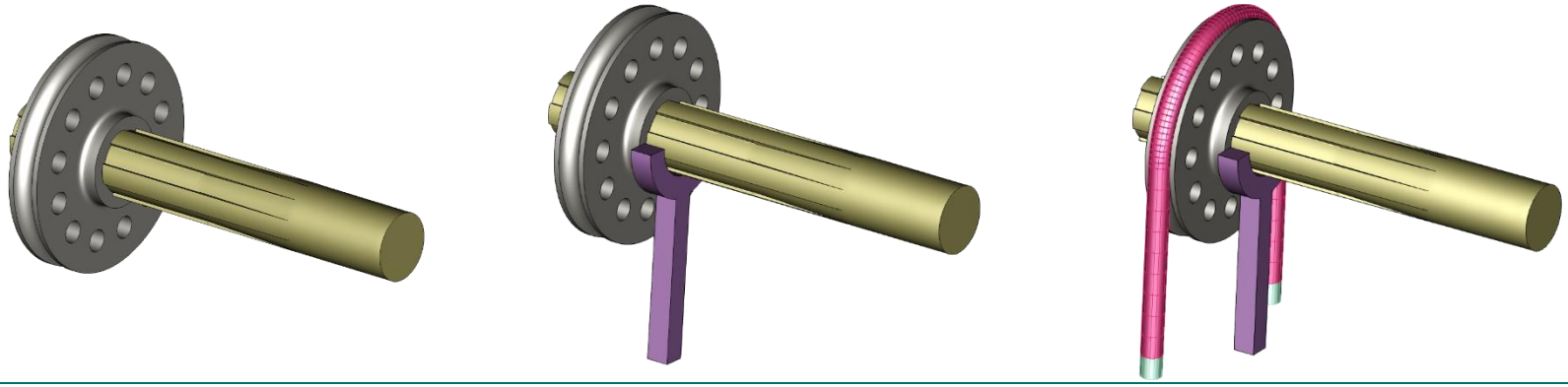


Examples

- A ROPS subjected to standard testing protocol
  - Set-up using the Solution Explorer in LS-PrePost

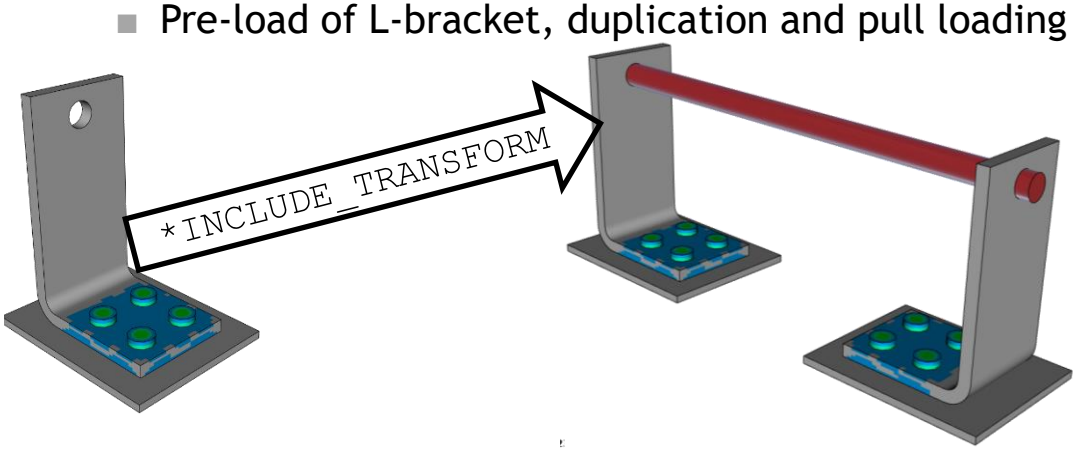


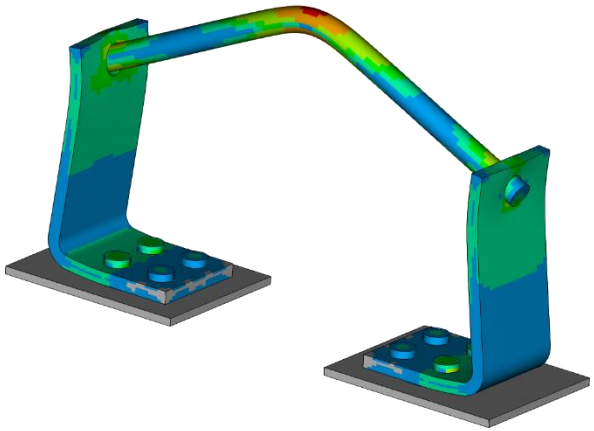
- A spline shaft assembly
  - Example of adding parts, and setting up different load cases based on one assembly



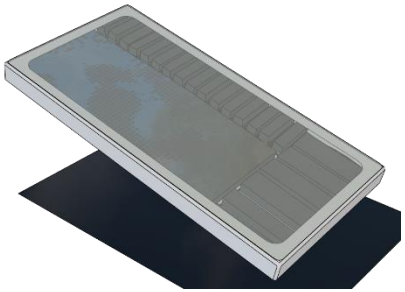
Examples

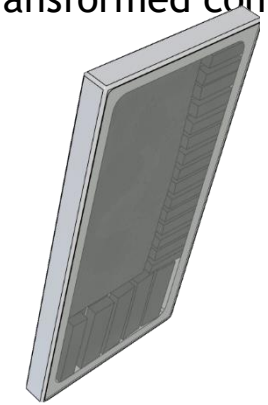
- Duplication of pre-tensioned sub-assembly
  - Pre-load of L-bracket, duplication and pull loading





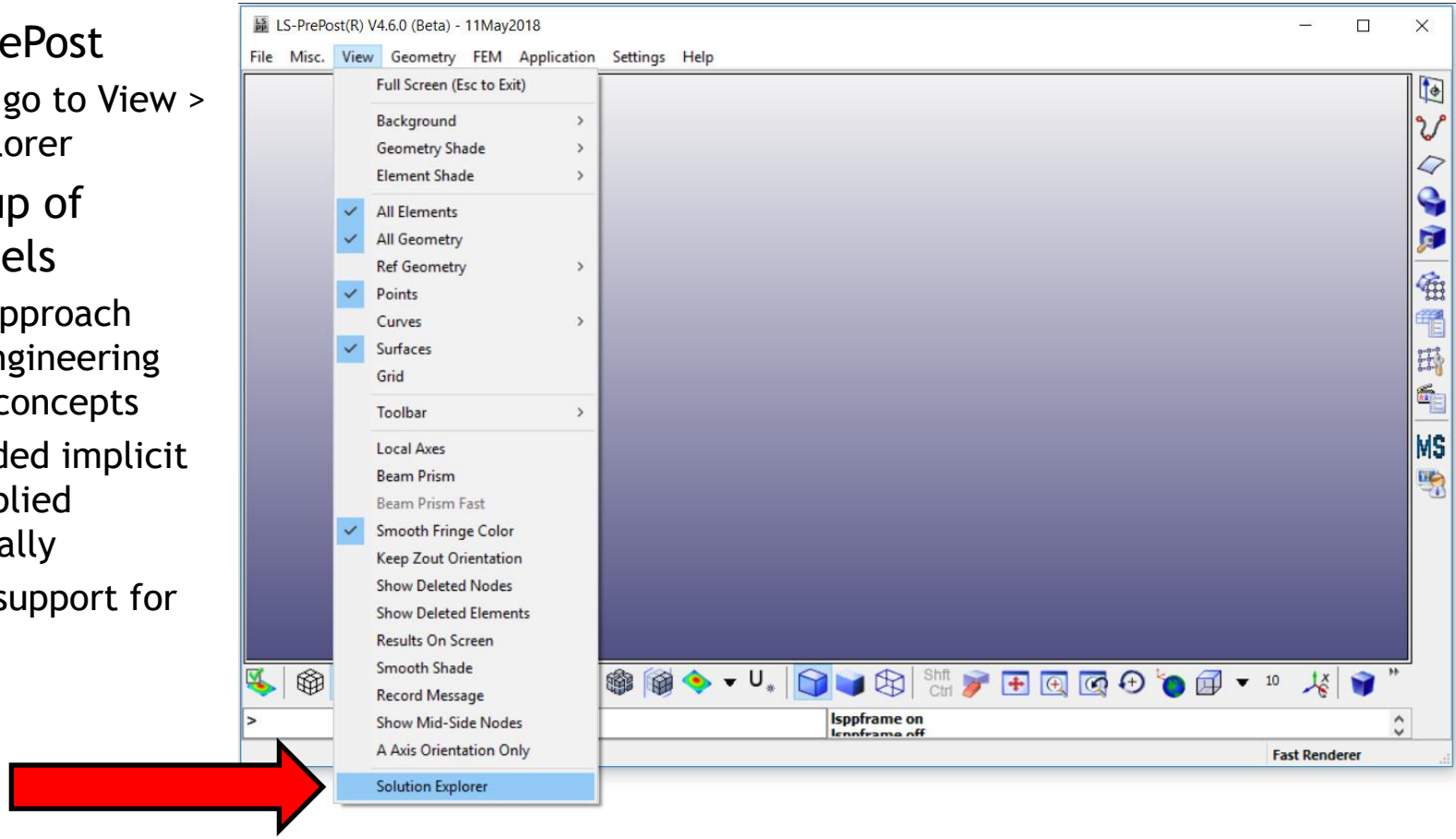
- Springback after drop test
  - Explicit drop test analysis, followed by springback in a transformed configuration





Simulation of a ROPS – set-up using Solution Explorer

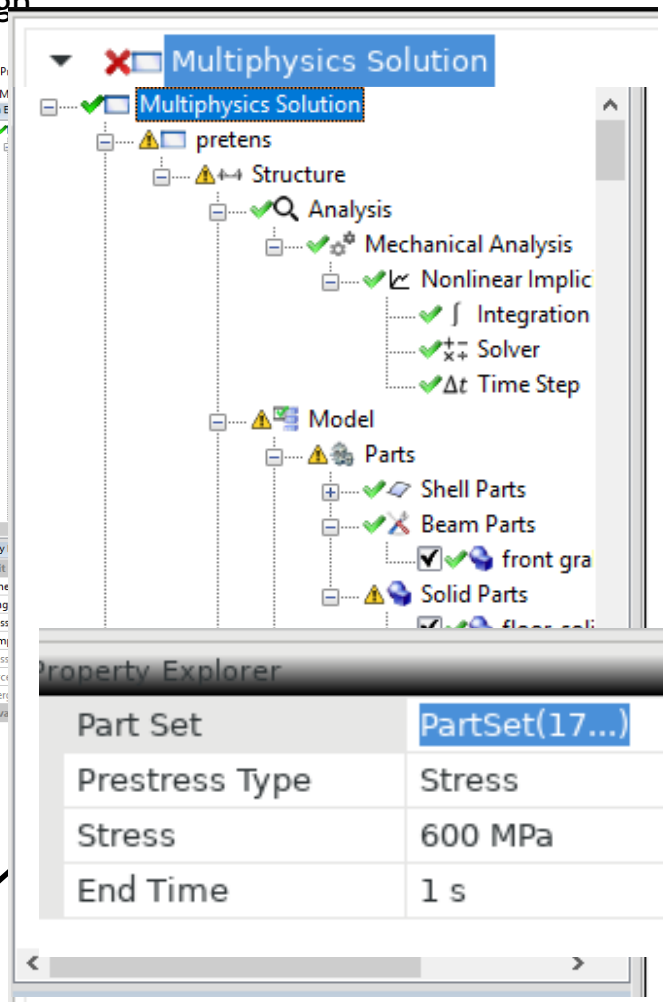
- Part of LS-PrePost
  - From V4.6, go to View > Soltion Explorer
- GUI for set-up of implicit models
  - Top-down approach based on engineering simulation concepts
  - Recommended implicit settings applied automatiacally
  - Integrated support for load steps

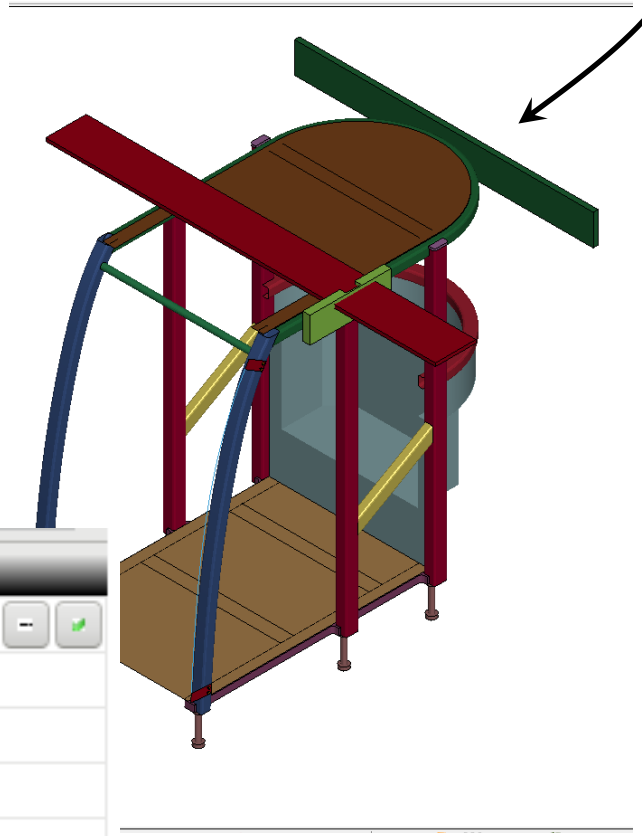


Simulation of a ROPS – set-up using Solution Explorer

Solution tree for navigating through model content

Main canvas displaying the model

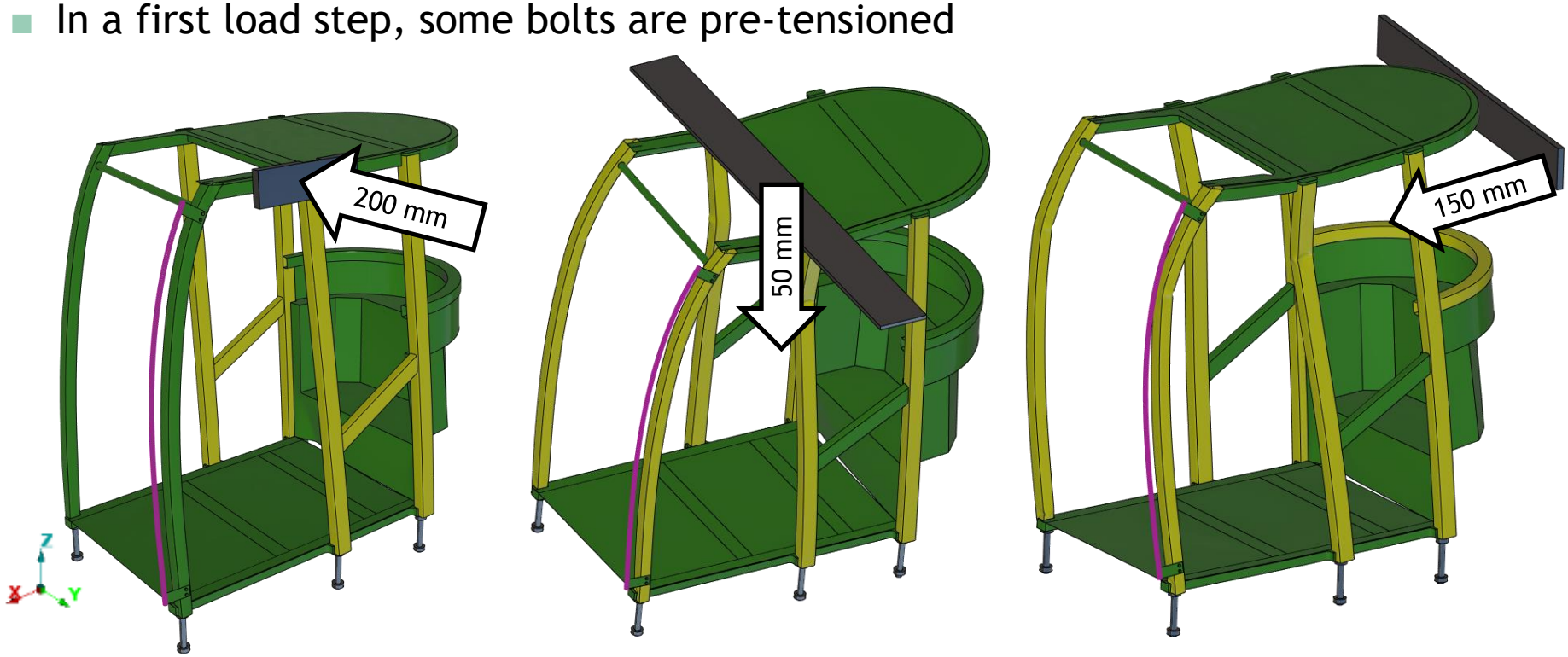




Parameter editor for completing settings of tree branchs

Load case description for the ROPS case

- All loadings are applied using prescribed displacements.
- Rigid impactors are used
  - The lateral loading is applied as a Y-displacement of 200 mm
  - The vertical loading is applied as a Z-displacement of 50 mm
  - The longitudinal loading is applied as a X-displacement of 150 mm
- In a first load step, some bolts are pre-tensioned



Simulation set up using the Solution Explorer

- The loading sequence was set up using the `*CASE` functionality
  - Totally 6 cases were defined
- By this, the loading sequence is performed as a series of separate, consecutive analyses, where one impactor at the time is active

Case #	Action	Active impactor	Duration
1	Bolt pre-tensioning	None	1 s
2	Lateral loading	Lateral	10 s
3	Lateral unloading	Lateral	10 s
4	Vertical loading and unloading	Vertical	20 s
5	Longitudinal loading	Longitudinal	10 s
6	Final unloading	Longitudinal	10 s

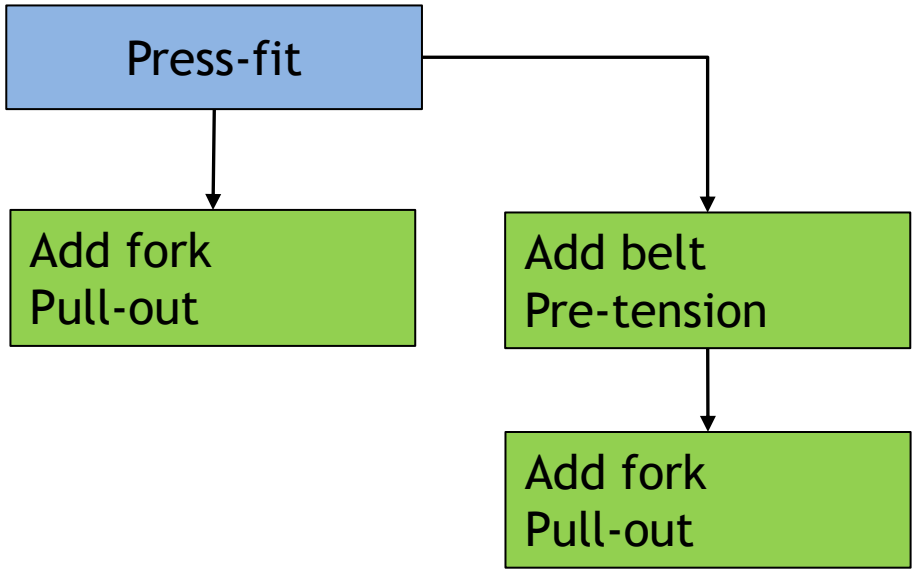
- Different solver settings were used in order to optimize the solution time
  - Full-Newton for the loading sequences (and whole case 4)
  - Moderately non-linear for the bolt pre-tensioning and the unloading sequences
  - It would also be possible to use the full-Newton settings for all cases

Some results: deformation

0:Z_cprd3plot : 1900201ROPS Cab Test 11-All loading,simple LDD,grabhandle+fundament pret : STATE 2 ,TIME 1.00000000E+01

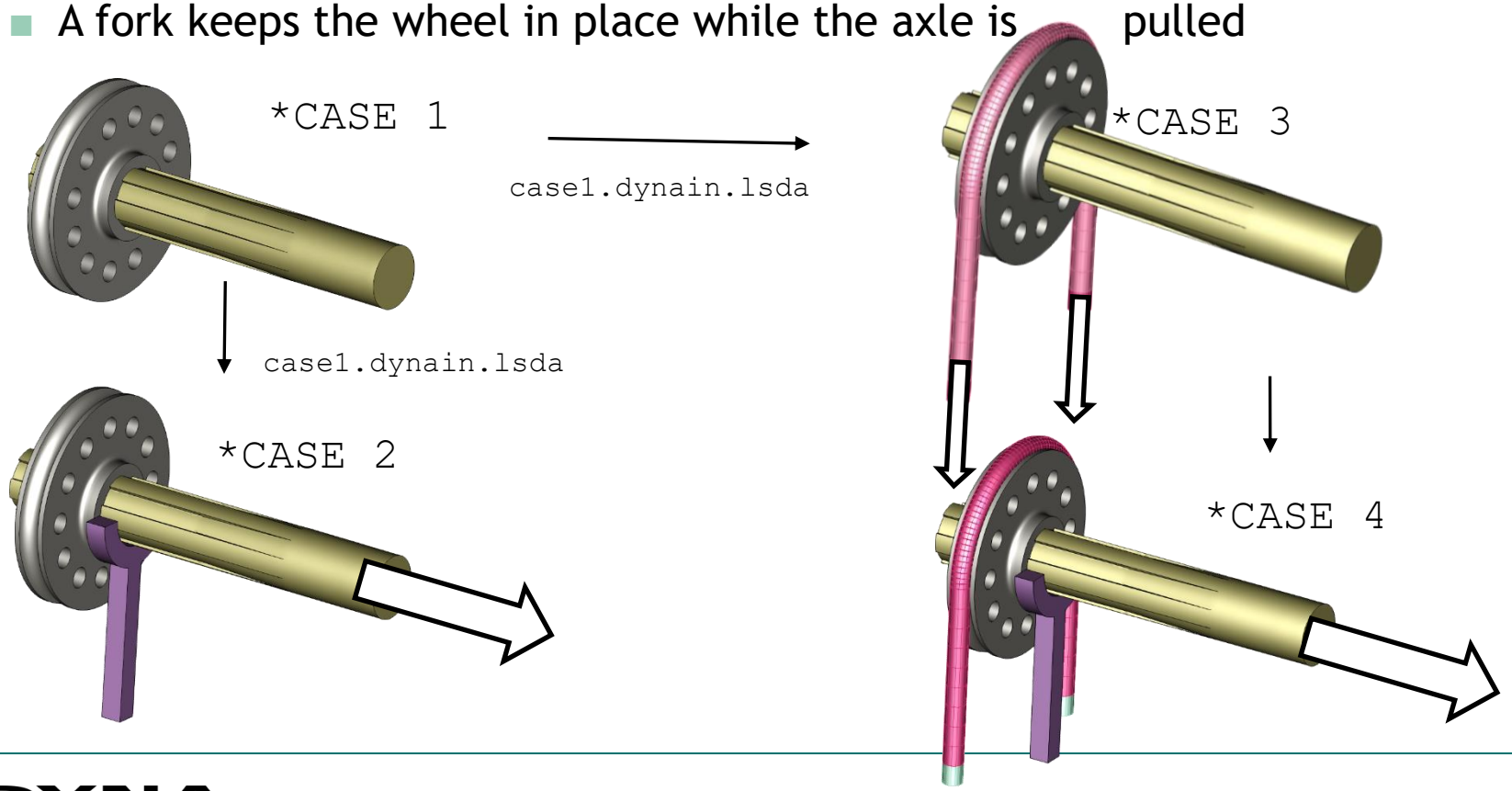
Example – A spline shaft assembly

- A wheel is mounted on a splined axle with a press-fit
- The force to pull loose the wheel is to be determined for two configurations
  - Without the drive belt
  - With pre-tensioned drive belt
- A fork keeps the wheel in place while the axle is pulled



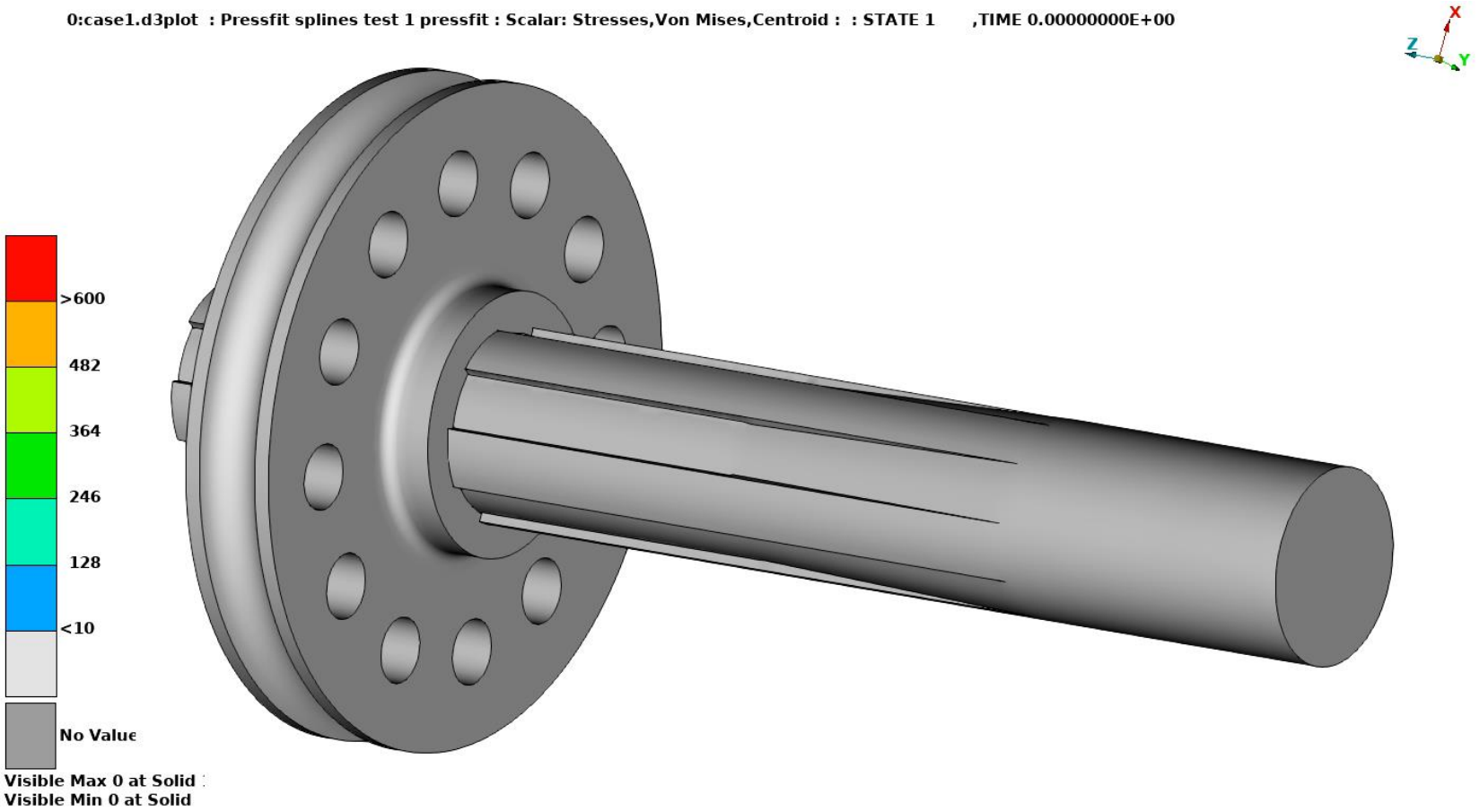
Example – A spline shaft assembly

- A wheel is mounted on a splined axle with a press-fit
- The force to pull loose the wheel is to be determined for two configurations
  - Without the drive belt
  - With pre-tensioned drive belt
- A fork keeps the wheel in place while the axle is pulled



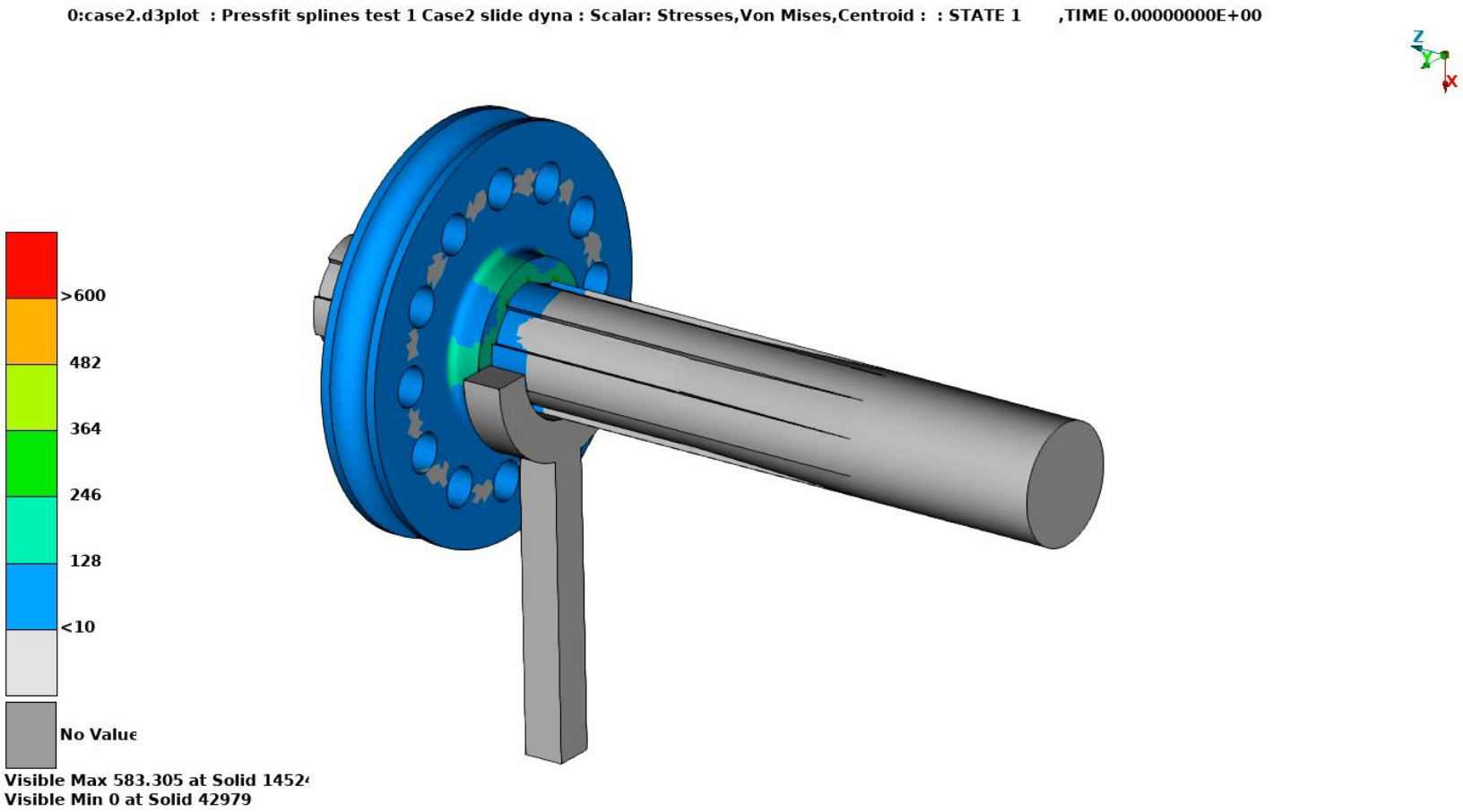
Some results

- Press-fit



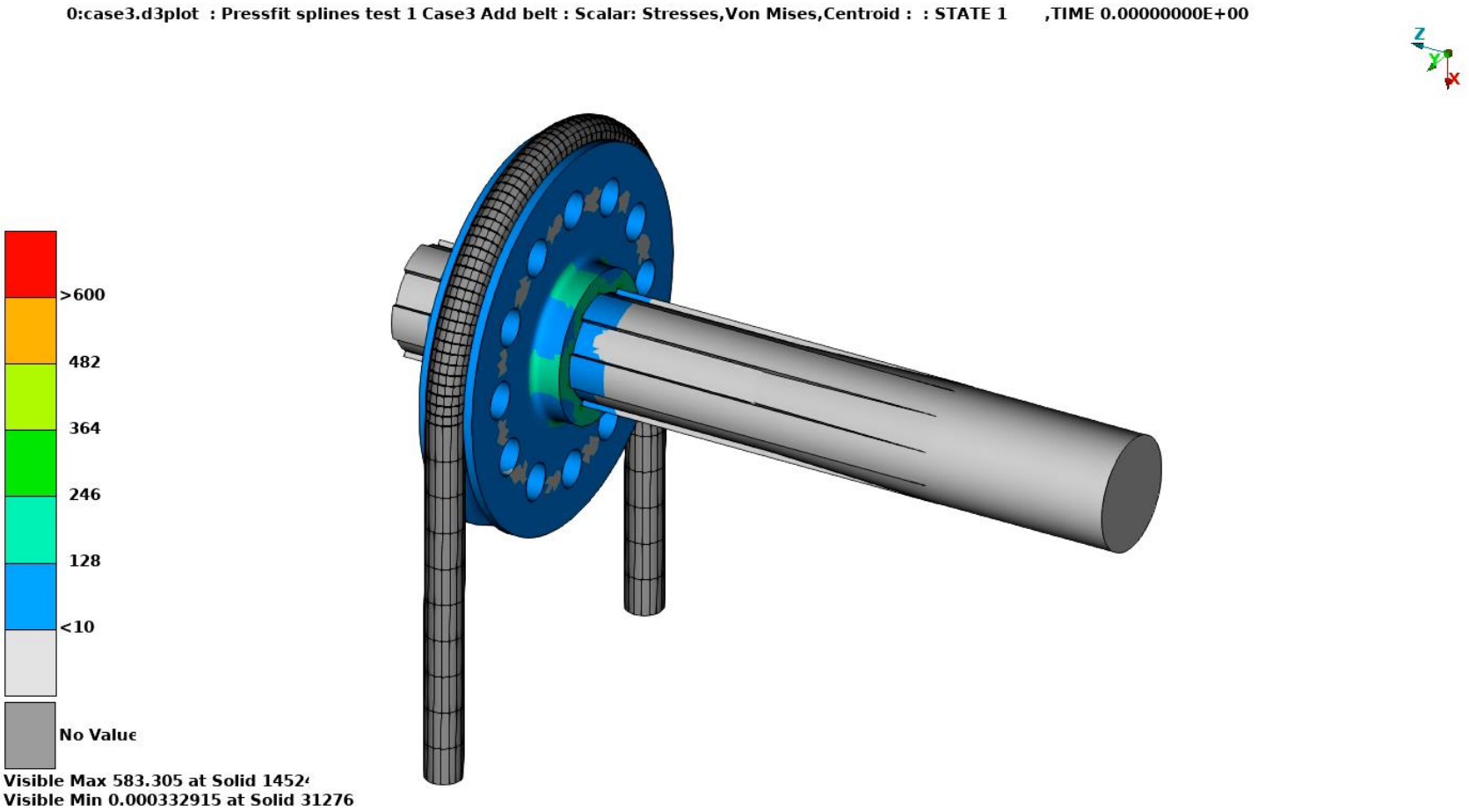
Some results

- Add fork and pull out



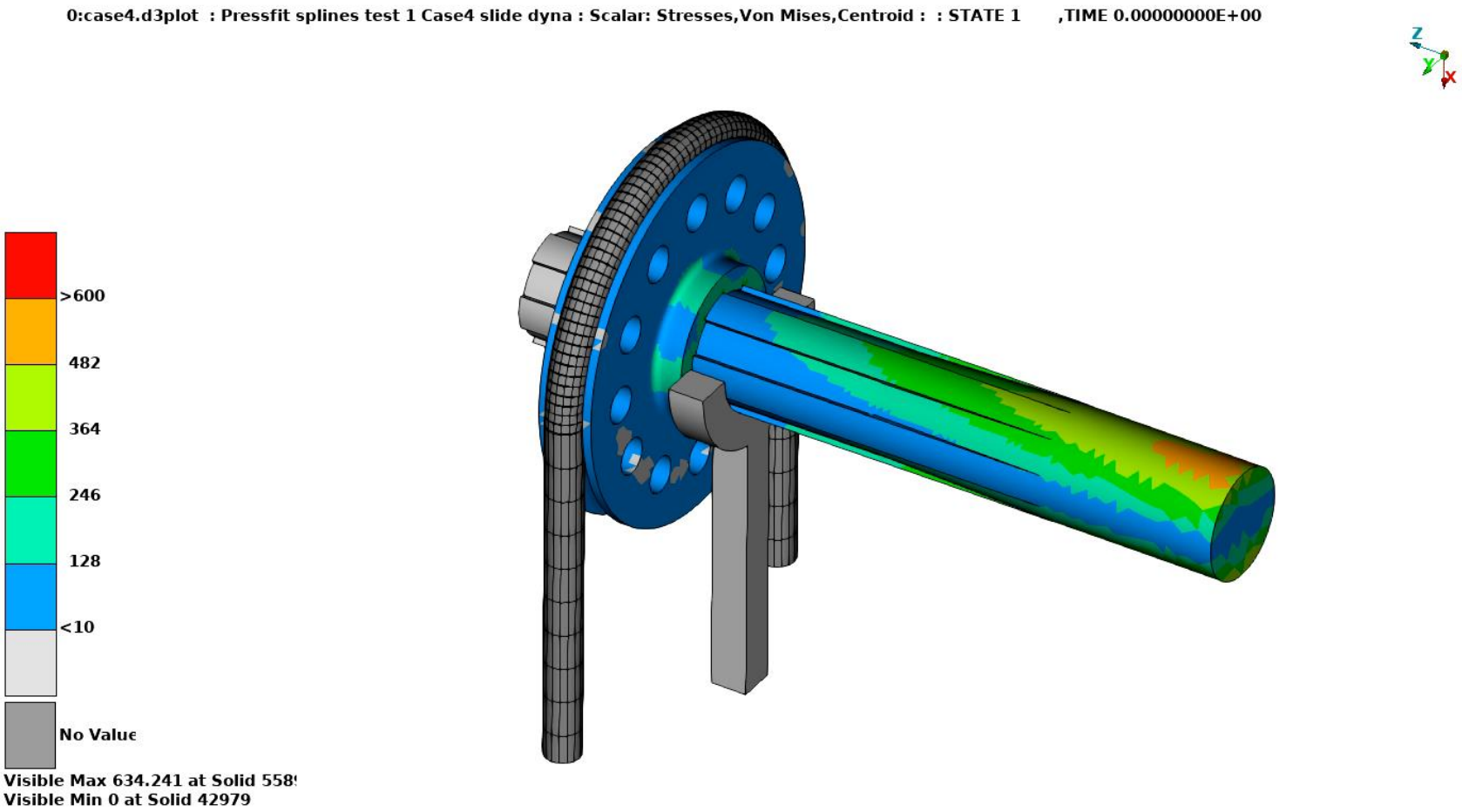
Some results

- Add belt and tenion



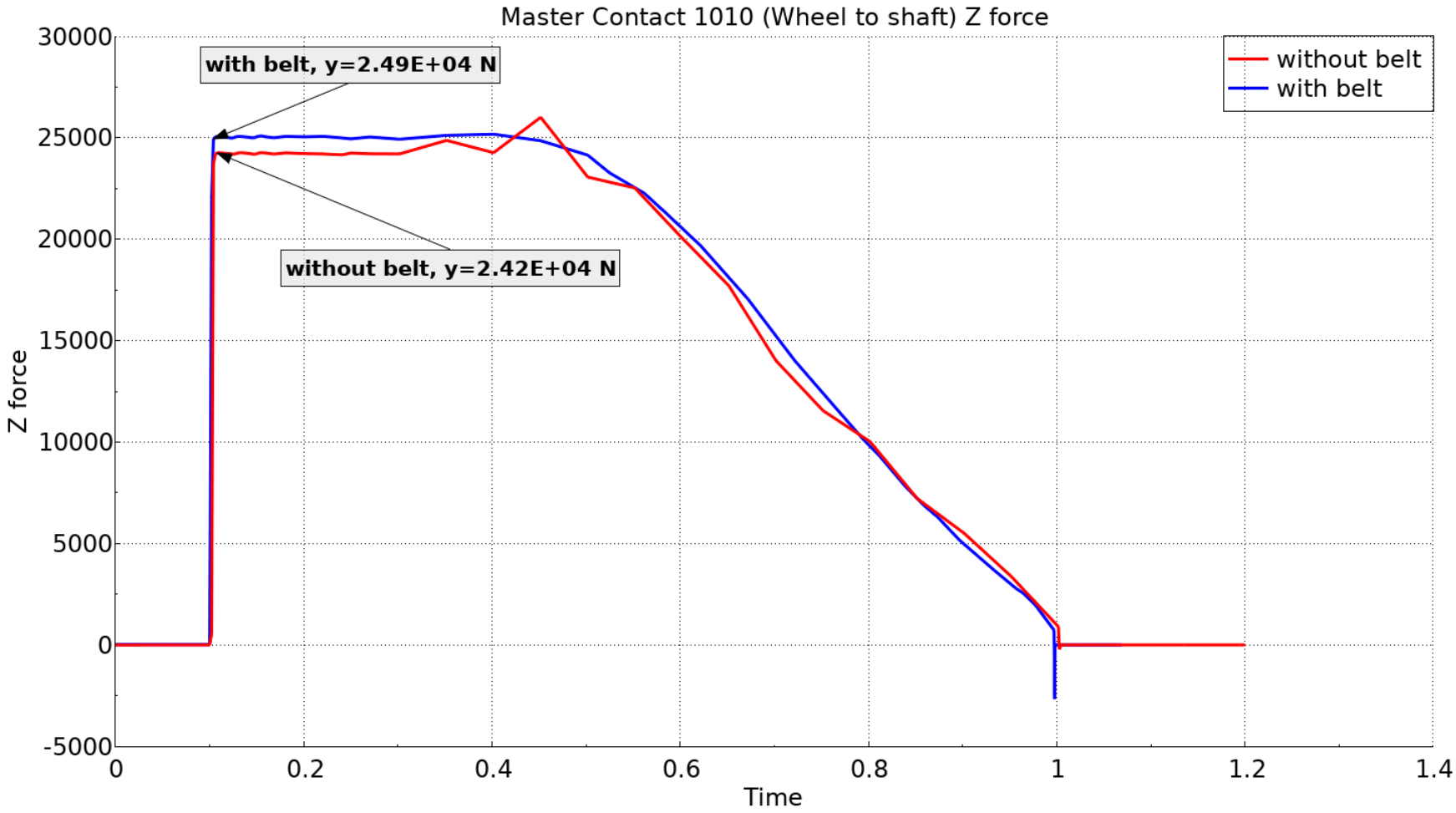
Some results

- Add fork and pull out



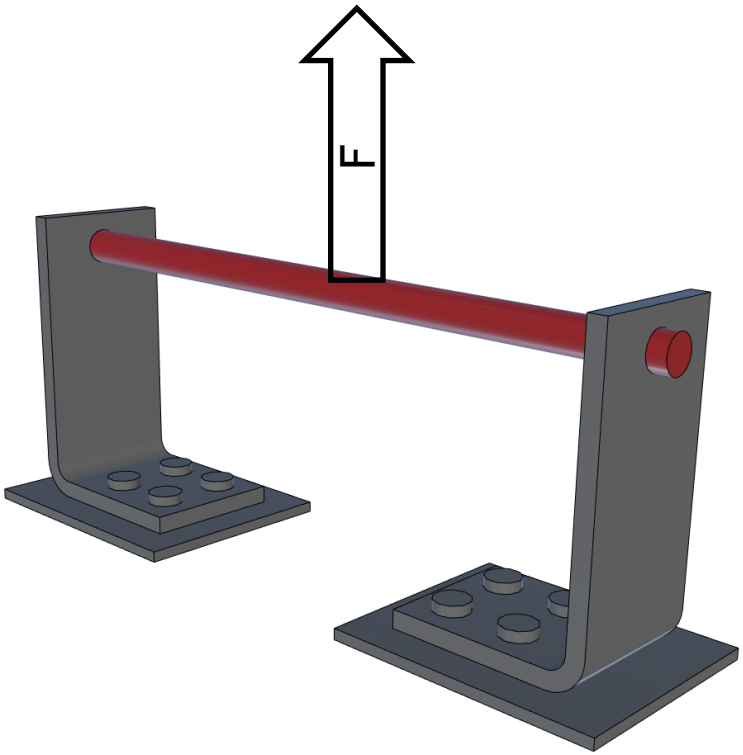
Some results

- Pull-out force comparison



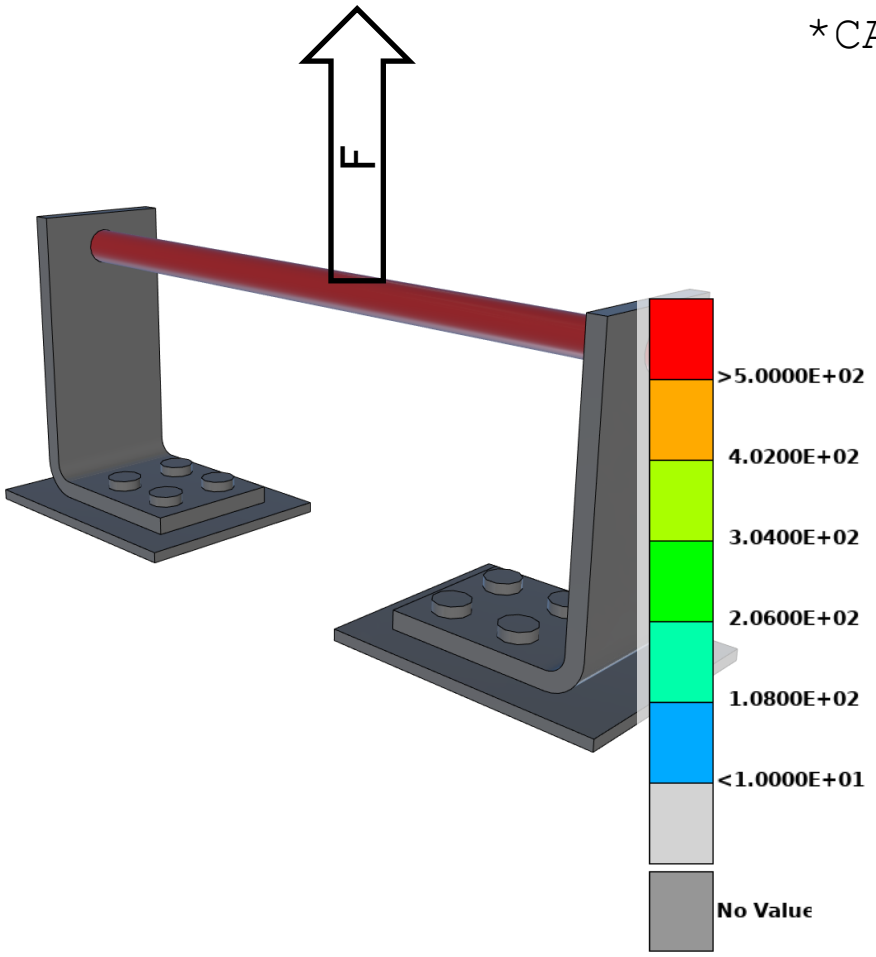
Duplication of pre-loaded subassembly

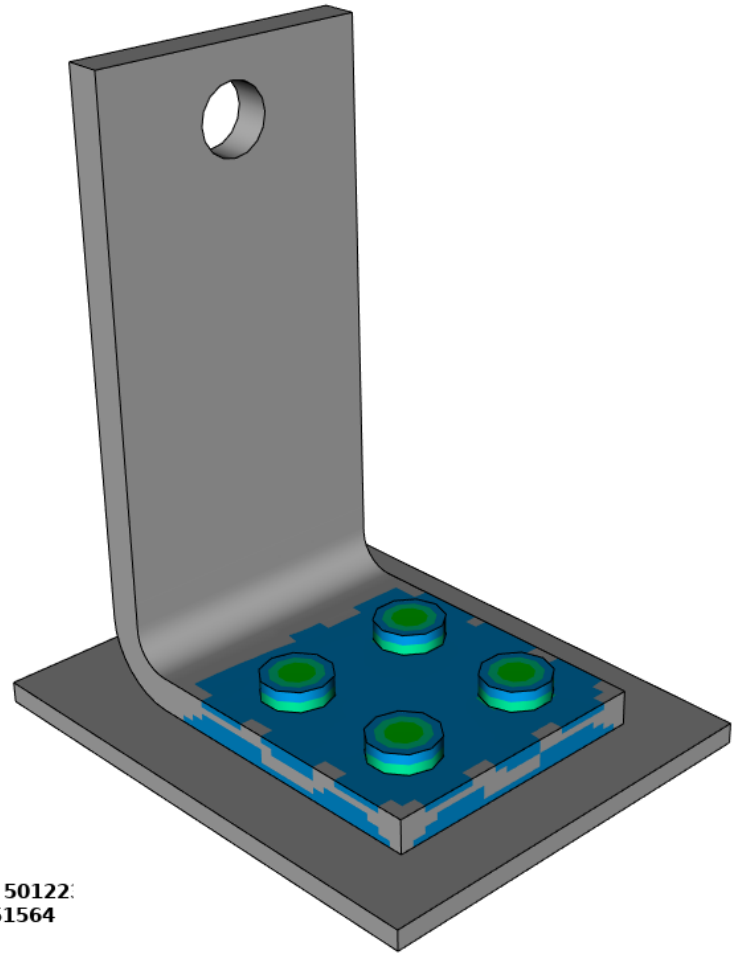
- A crossbar is attached by two L-brackets
- A subassembly occurs in multiple instances
  - L-bracket with pre-tensioned bolts
- `*INCLUDE_TRANSFORM` makes it possible to duplicate pre-loaded subassemblies



Duplication of pre-loaded subassembly

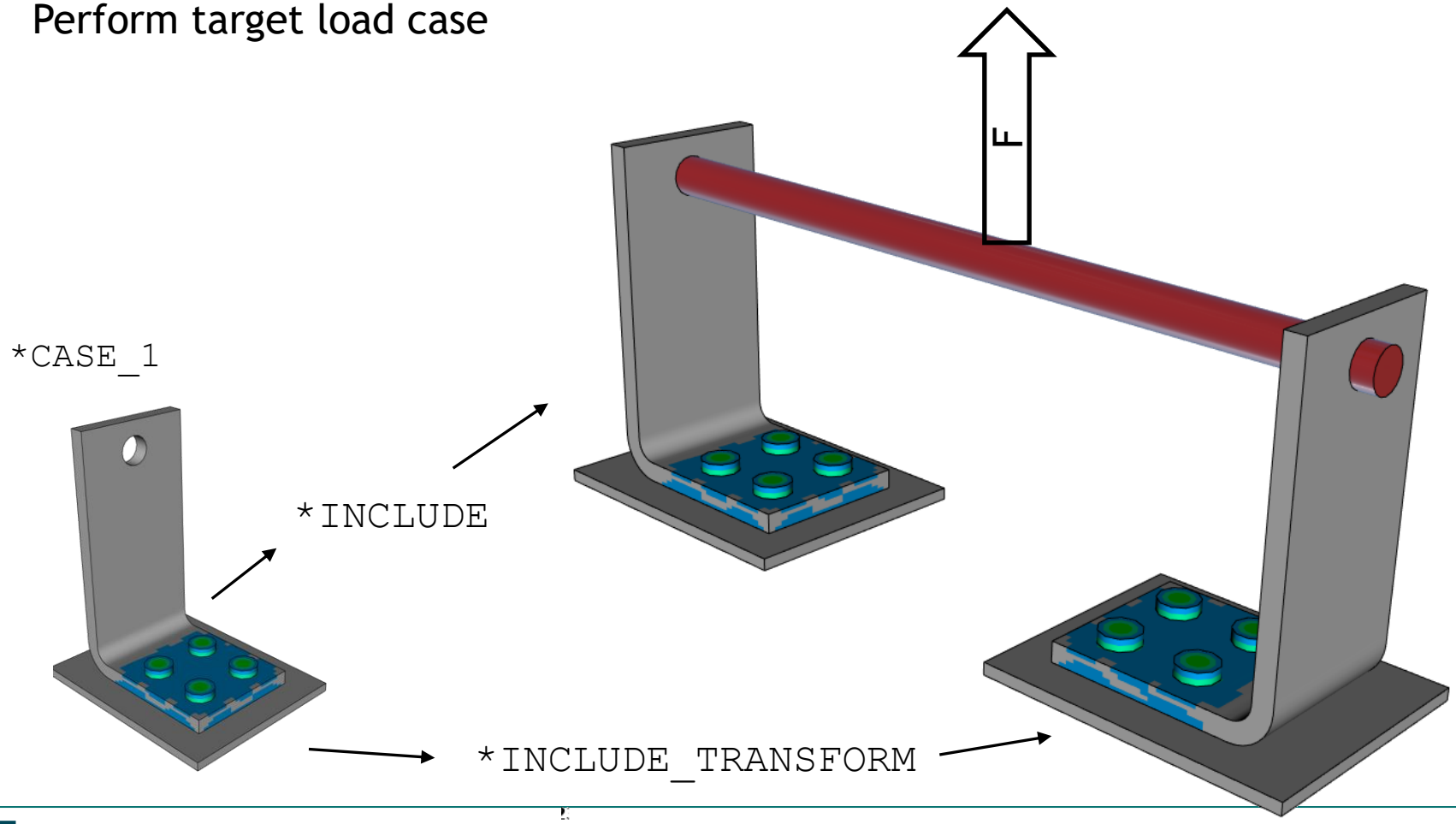
*CASE_1 : Pretension of one sub-assemby



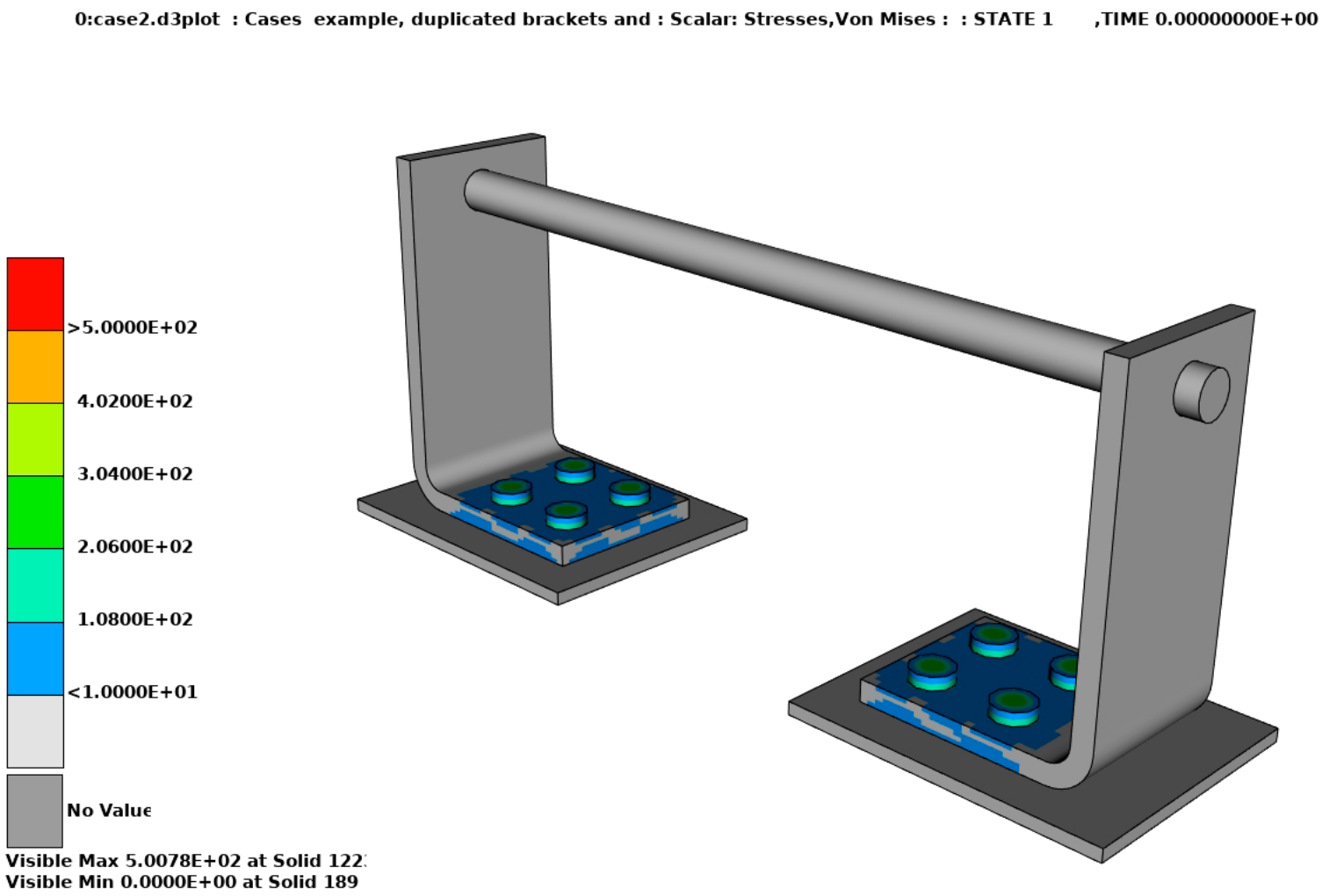


Duplication of pre-loaded subassembly

*CASE_2 : Include crossbar and pre-tensioned attachment
Duplicate using *INCLUDE_TRANSFORM
Perform target load case

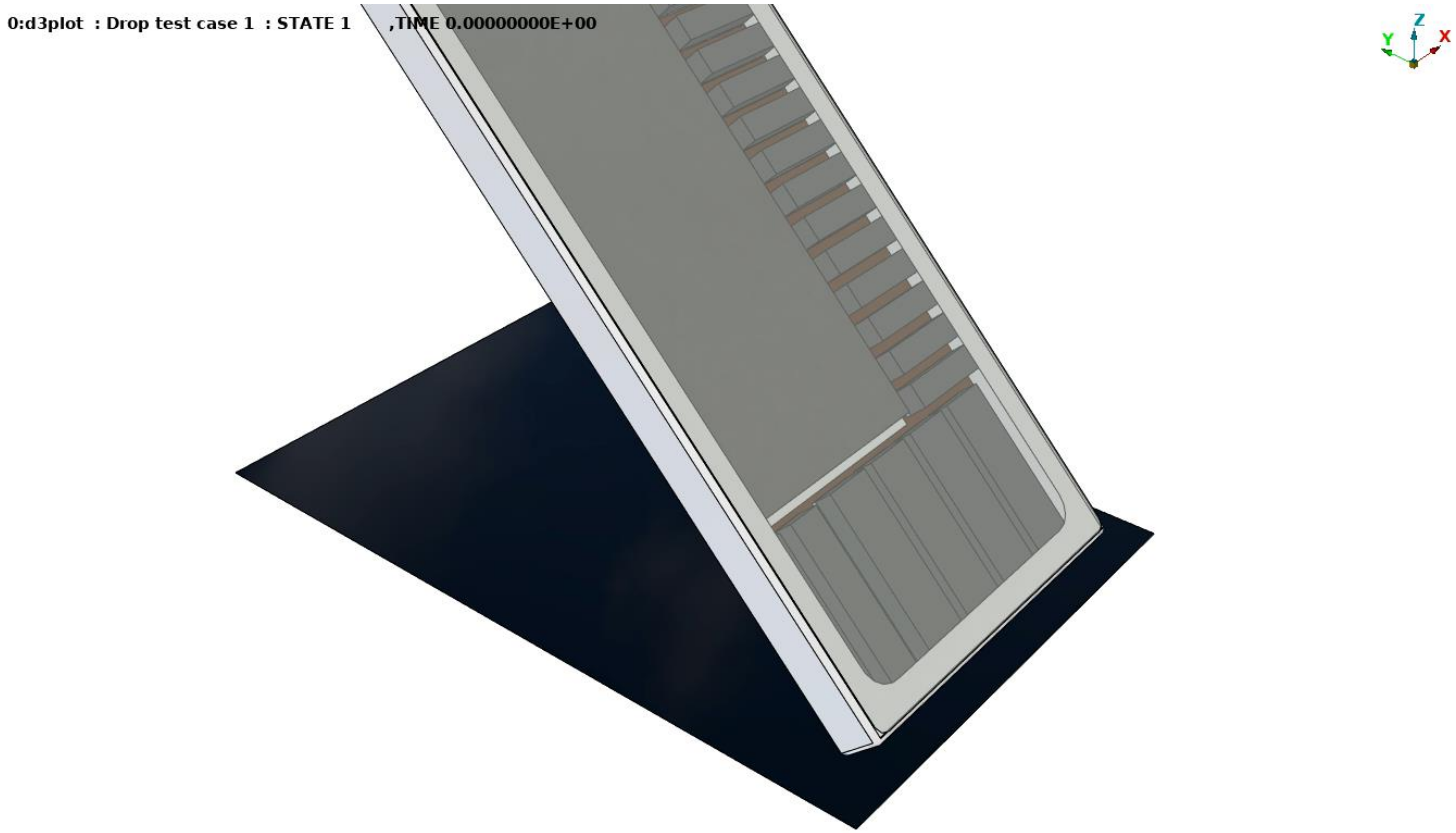


Duplication of pre-loaded subassembly



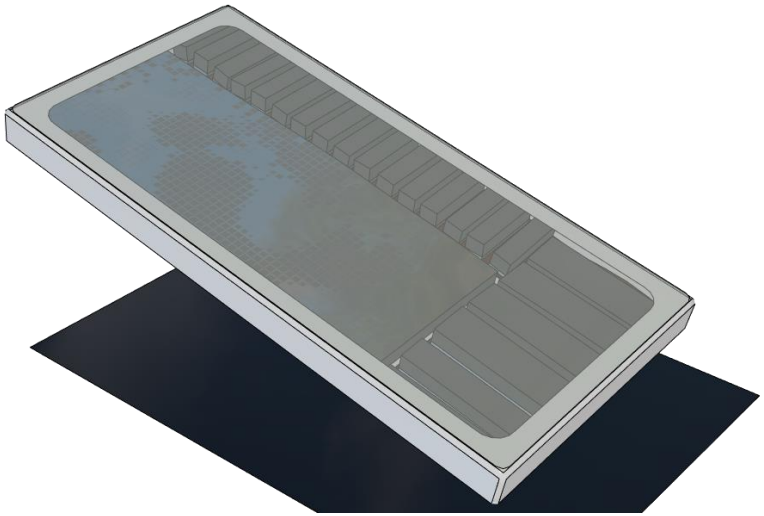
Implicit springback after explicit drop test

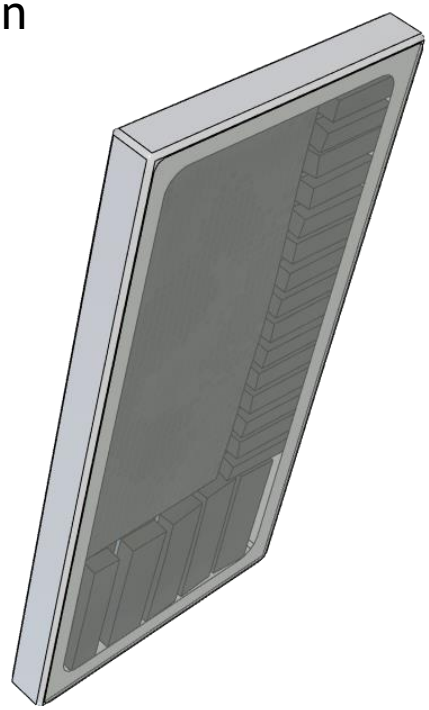
- An electronic device is subjected to a drop test
- Springback is performed in a transformed configuration



Implicit springback after explicit drop test

- An electronic device is subjected to a drop test
- Springback is performed in a transformed configuration

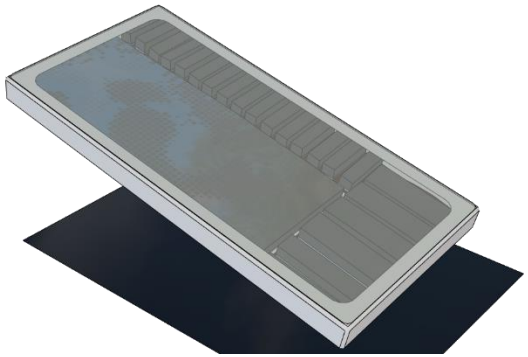




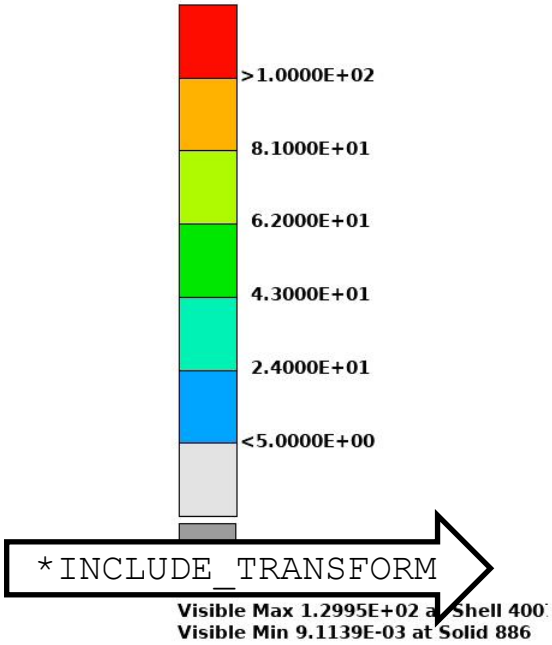
Explicit drop test → `*INCLUDE_TRANSFORM` → Implicit springback

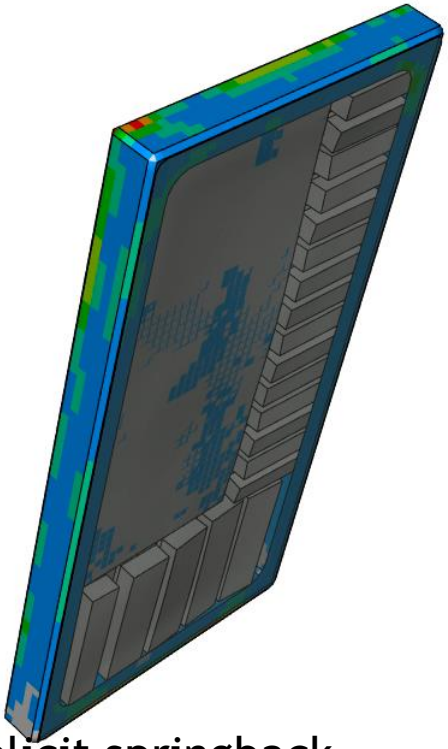
Implicit springback after explicit drop test

- An electronic device is subjected to a drop test
- Springback is performed in a transformed configuration



Explicit drop test





Implicit springback

Summary

- Using the new functionality of `dynain.lsda` (possibly combined with `*CASE`) can make it easier and more intuitive to set up multi-step simulations
  - Reduced need for birth/death times on contacts, boundary conditions etc.
  - Simpler load curves / ramps
  - Possible to add (or remove) parts between each step
  - Possible to have implicit pre-loading followed by explicit transient loading
  - “Restart” file available after each step
- Available from R12.0.0
- Relatively new feature still under development
- Some known issues in R12.0.0
  - 2nd order shells (ef 23, 24) not fully supported – will be output as 1st order shells in `dynain.lsda`
  - 1st order tet elform 13 may be problematic in some cases

Thank you!

DYNA MORE

Your LS-DYNA distributor and more