



Powering Innovation That Drives Human Advancement

Driving Pre & Post processing of LS-DYNA with LS-PrePost and PyDyna

Philip Ho & LS-PrePost Team

May 20, 2025

Outlines

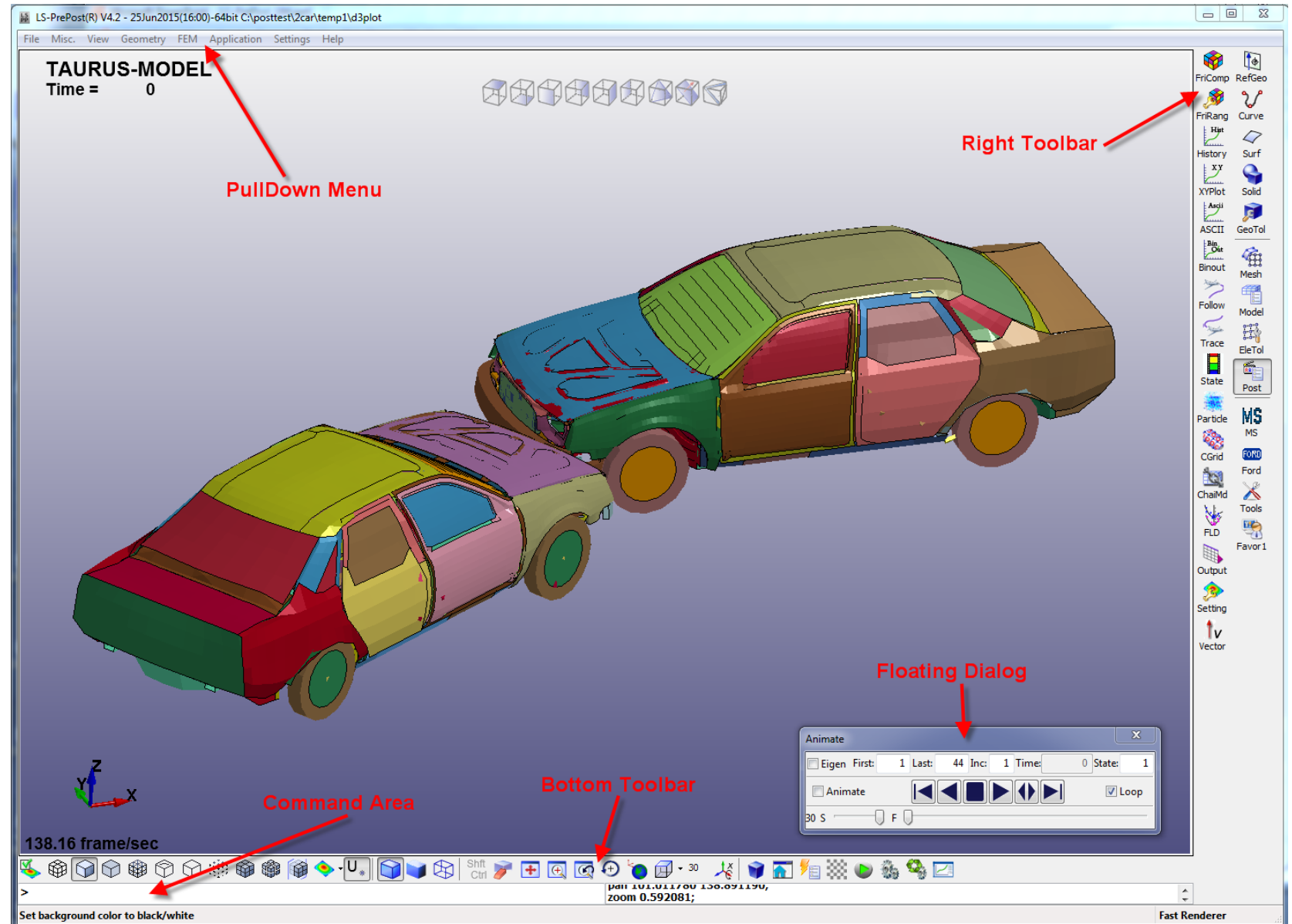
- Overview of LS-PrePost
- Solution Explorer
- Post-Explorer
- PyDyna
- PyDFP

LS-PrePost Overview

- LS-PrePost is an advanced pre and post-processor designed for LS-DYNA and is delivered free with LS-DYNA
- LS-PrePost does not need LS-DYNA license key, it can be freely download and loaded onto any computer that meets the hardware requirements to run it
- Supported systems are Windows 10/11; 64bit Linux (Centos 7/8, Opensuse 15, Ubuntu)

LS-PrePost Overview

The user interface is designed to be both efficient and intuitive. The latest OpenGL technology is utilized to achieve fast rendering for graphics and XY plotting



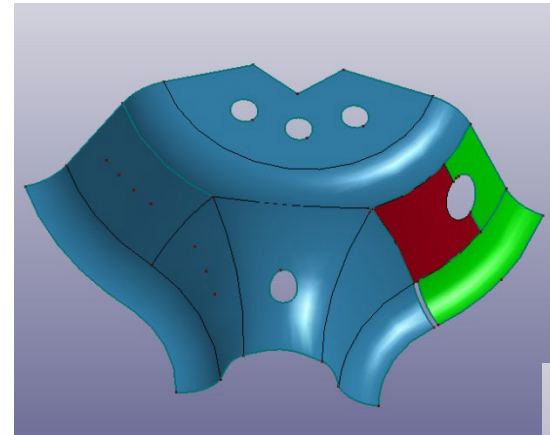
LS-PrePost Overview

There are 6 major functions in LS-Prepost

- Geometry engine
- Meshing
- Finite Element Model Building
- Keyword Manager
- Post-processing
- Applications

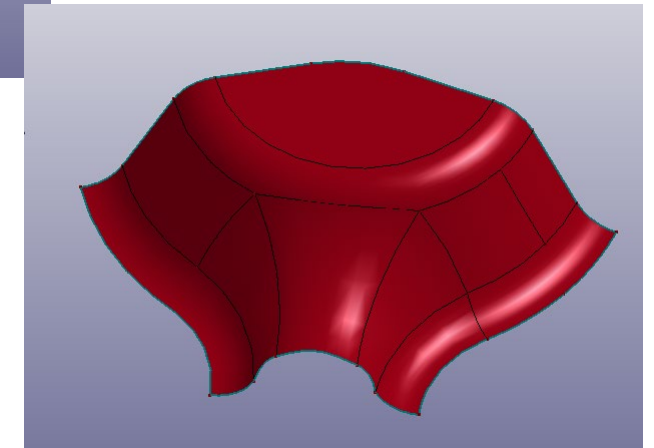
LS-PrePost – Geometry Engine

- Allows user to build geometry entities or to read geometry data from IGES or STEP formatted file
- Provides capabilities to repair and clean geometry entities before meshing, for example:
 - Stitch surfaces together
 - Trim entities against each other
 - Repair overlapping
 - Fill small holes
 - Eliminate small features like fillets surfaces
 - Suppress unwanted common edges

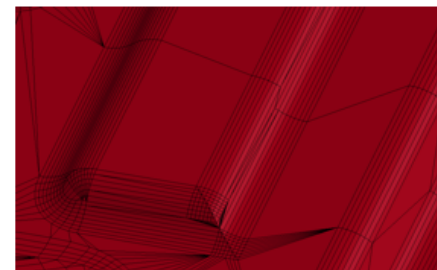
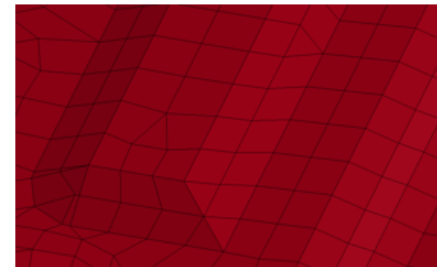
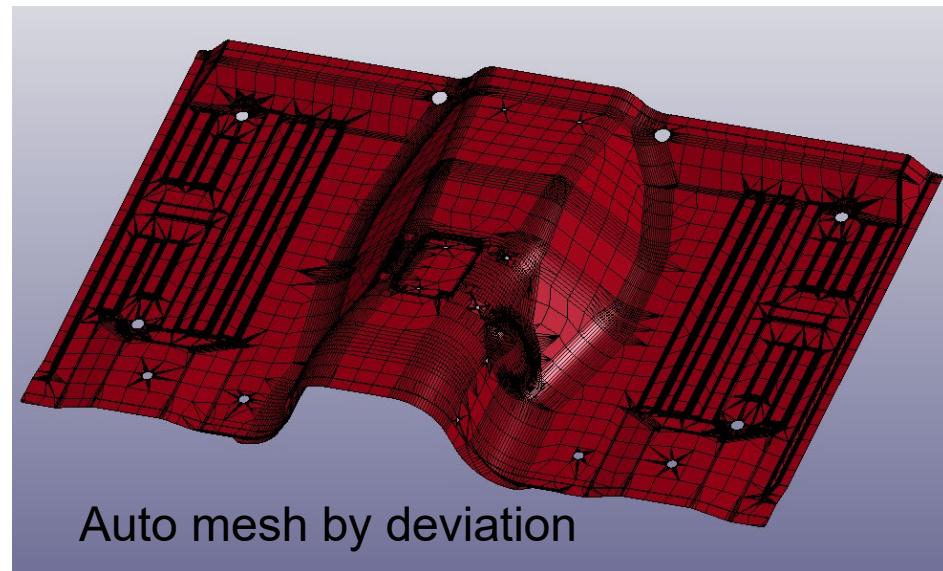
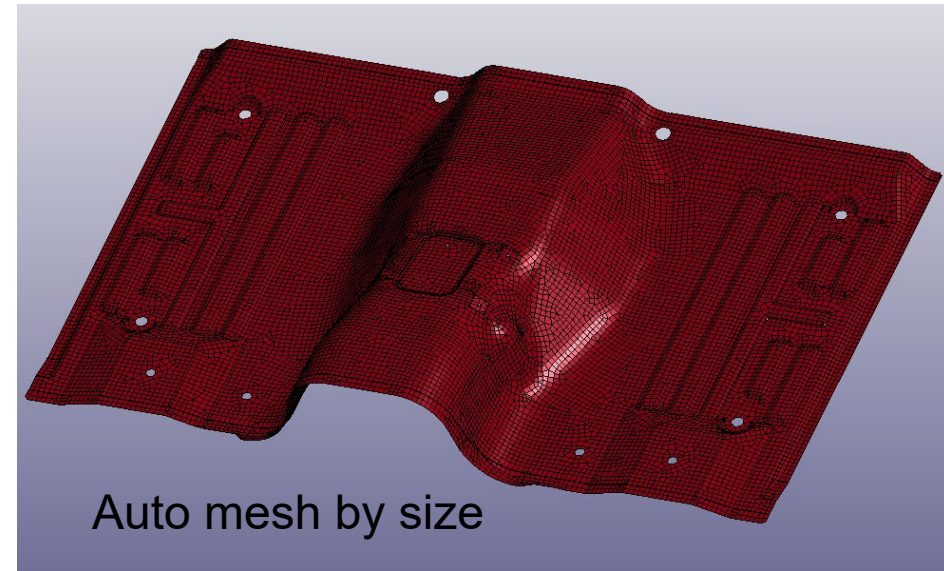
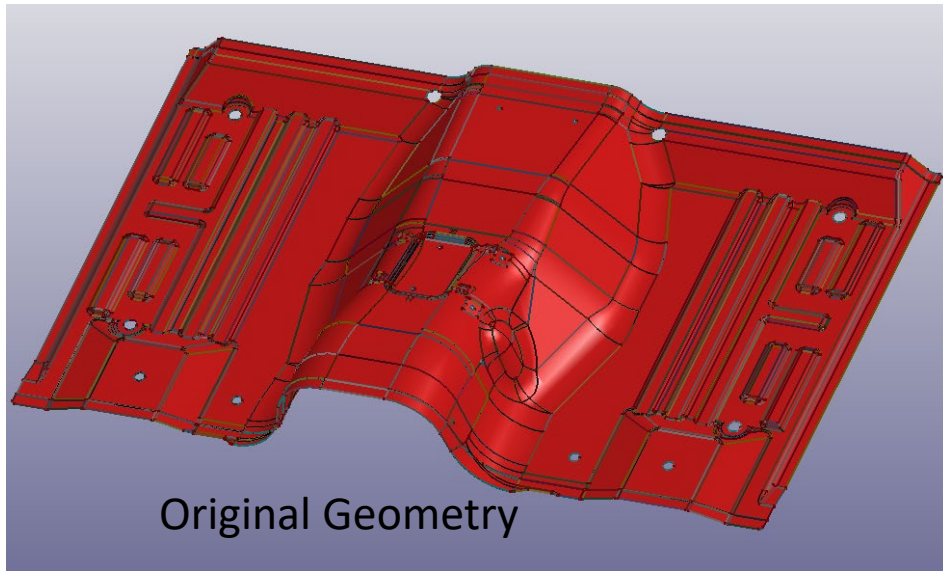


A raw geometry

A cleanup geometry



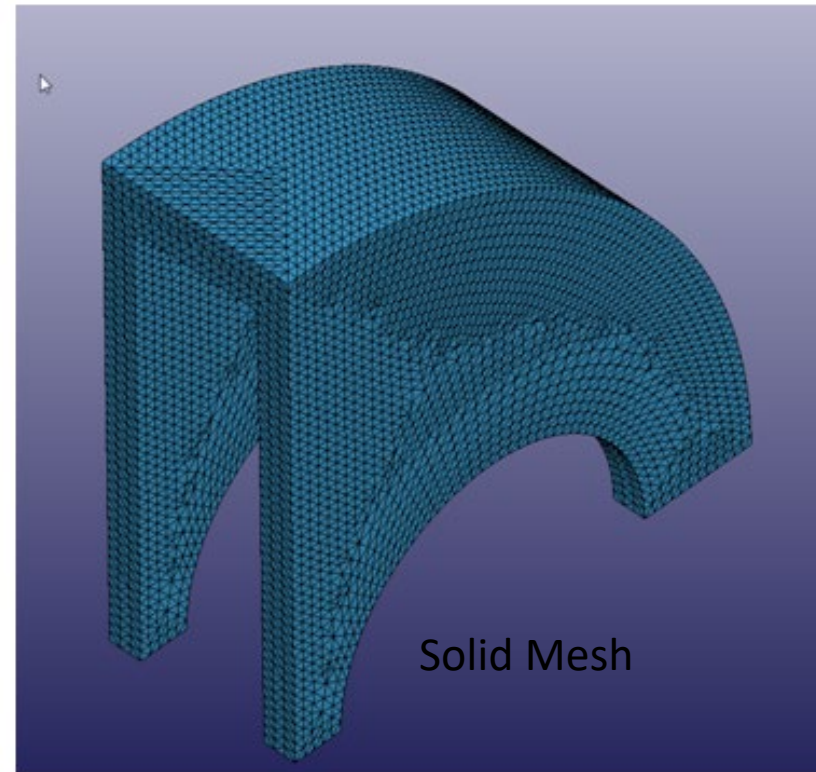
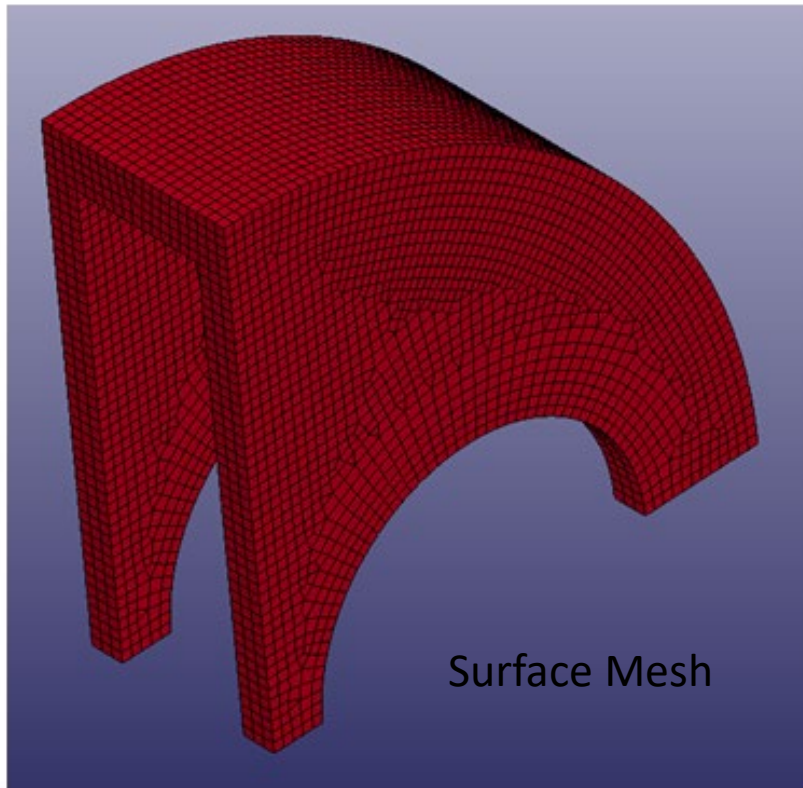
LS-PrePost – Shell Meshing



Detailed difference
at the curvature

LS-PrePost – Solid Meshing

- Mesh surface of a watertight volume with shell elements, then use automatic Tetrahedron meshing to create tetrahedron elements



LS-PrePost – LS-DYNA Model Builder

Create and modify a complete LS-DYNA model

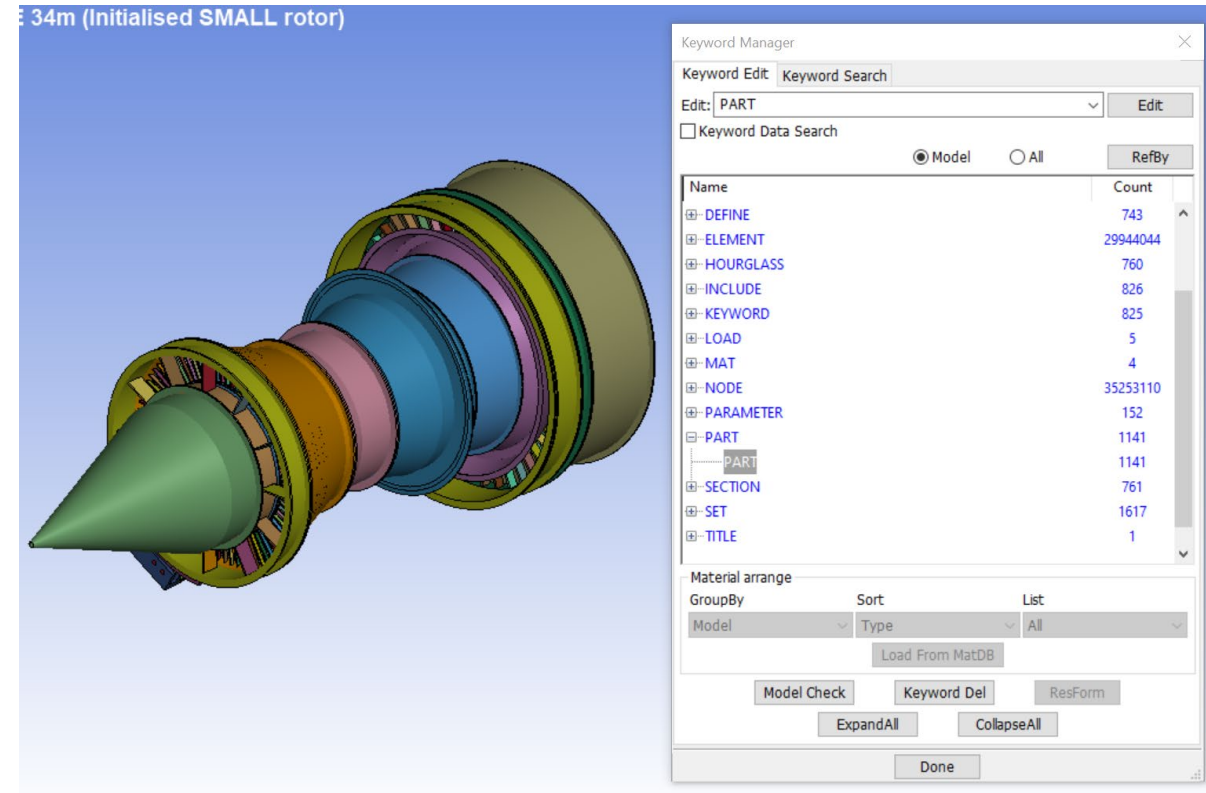
- LS-DYNA entities data creation and modification
 - Coordination systems
 - Set data
 - Constrained data
 - Contact definitions
 - Initial conditions
 - Define curves
 - Rigid Walls
 - Loading

LS-PrePost – LS-DYNA Model Builder

- LS-DYNA non-graphical keyword data creation and modification
 - Material data
 - Execution control data
 - Physical properties
 - Misc. parameters
 - Output control
- Keyword data deck checking – to ensure model is acceptable for LS-DYNA

LS-PrePost – Keyword Manager

- Most completed and up-to-date LS-DYNA keyword support in the industry
- Keyword data is represented by tree structure, and each has its own form
- Each Keyword field has its own description which eliminates the need of user manual most of the time
- Extensive keyword data checking to ensure model is acceptable by LS-DYNA



LS-PrePost – A Typical Keyword Form

Keyword Input Form

Clear Accept Delete Default Done

☐ Use *Parameter (Subsys: 1 b.key) Setting

*CONTROL_OUTPUT (1)

1	NPOPT	NEECHO	NREFUP	IACCOP	OPIFS	IPNINT	IKEDIT	IFLUSH
	1	3	0	0	0.0	0	0	0
2	IPRTF	IERODE	TET10	MSGMAX	IPCURV	GMDT	IP1DBLT	EOCS
	0	0	2	50	0	0.0	0	0
3	TOLEV	NEWLEG	FRFREQ	MINFO	SOLSIG	MSGFLG	CDETOL	
	2	0	1	0	0	0	10.0000000	

COMMENT:

Click this field to get the description

IACCOP:=Averaged accelerations from velocities in file NODOUT and the time history database file d3thdt:
EQ.0: no average (default),
EQ.1: averaged between output intervals.
EQ.2: Built-in, user-defined filtering. With this option the keyword parameter DT2MS on *CONTROL_TIMESTEP must be defined. All data points between output intervals are stored and used to obtain the filtered output values. The user defined filter must be provided and linked. The procedure for handling is not yet defined.

Keyword Data Search Manager

- Popup “Keyword Data Search Manager” when “Keyword Data Search” is checked, or else open “Keyword Input Form”

The screenshot displays the 'Keyword Data Search Manager' interface. On the left, a table lists parts with columns: Draw, PID, PartType, TITLE, SECID, MID, EOSID, and HGID. The table contains 20 rows of data. A search dialog is open over the table, showing a list of parameters to search by: SECID, MID, EOSID, HGID, GRAV, ADPOPT, and TMID. The 'Keyword Data Search' checkbox is checked. On the right, the 'Keyword Manager' window is visible, showing the 'Keyword Search' tab. It includes a search bar, a list of results with counts, and buttons for 'Model Check', 'Keyword Del', 'ResForm', 'ExpandAll', and 'CollapseAll'.

Draw	PID	PartType	TITLE	SECID	MID	EOSID	HGID
☐	1000001	Solid	Fan Case	1000001	1001		1000001
☐	1832001	Solid	Mount Ring	1832001	1001		1832001
☐	1832002	Solid	Mount Ring Lugs	1832001	1001		1832001
☐	1832003	Solid	Mount Ring Pins	1832001	1001		1832001
☐	1832004	Solid	Mount Ring Spherical Brgs	1832001	1001		1832001
☐	2642001	Solid	Rear Fan Case	2642001	1001		2642001
☐	3398001	Solid	FBH Hub	3398001	1001		3398001
☐	5062001	Solid	FBH Splitter	5062001	1001		5062001
☐	6789001	Solid	FBH ESS	6789001	1001		6789001
☐	6794001	Solid	FBH ESS	6794001	1001		6794001
☐	6799001	Solid	FBH ESS	6799001	1001		6799001
☐	6804001	Solid	FBH ESS	6804001	1001		6804001
☐	6809001	Solid	FBH ESS	6809001	1001		6809001
☐	6814001	Solid	FBH ESS	6814001	1001		6814001
☐	6819001	Solid	FBH ESS	6819001	1001		6819001
☐	6824001	Solid	FBH ESS	6824001	1001		6824001
☐	6829001	Solid	FBH ESS	6829001	1001		6829001
☐	6834001	Solid	FBH ESS	6834001	1001		6834001

Search parameter by name

- ☒ SECID
- ☒ MID
- ☒ EOSID
- ☒ HGID
- ☐ GRAV
- ☐ ADPOPT
- ☐ TMID

Count

Count
743
29944044
760
826
825
5
4
35253110
152
1141
1141
761
1617
1

Sort: Model Type List

Load From MatDB

Model Check Keyword Del ResForm

ExpandAll CollapseAll

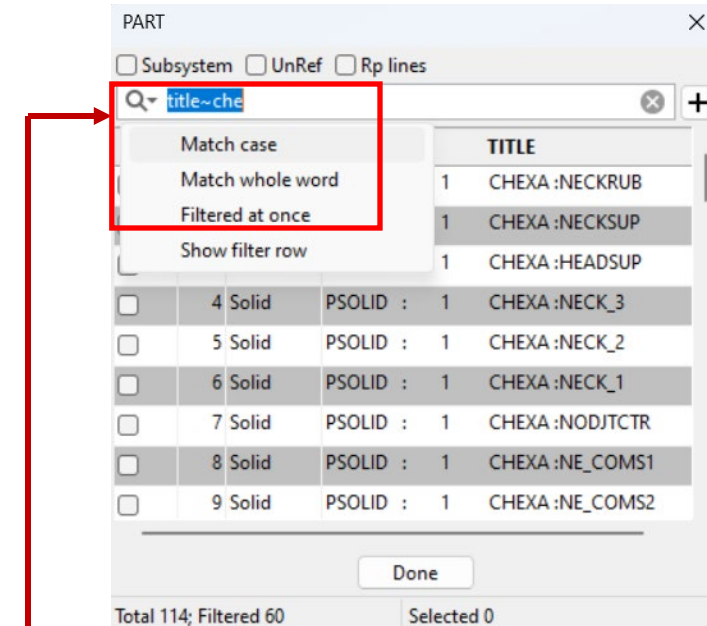
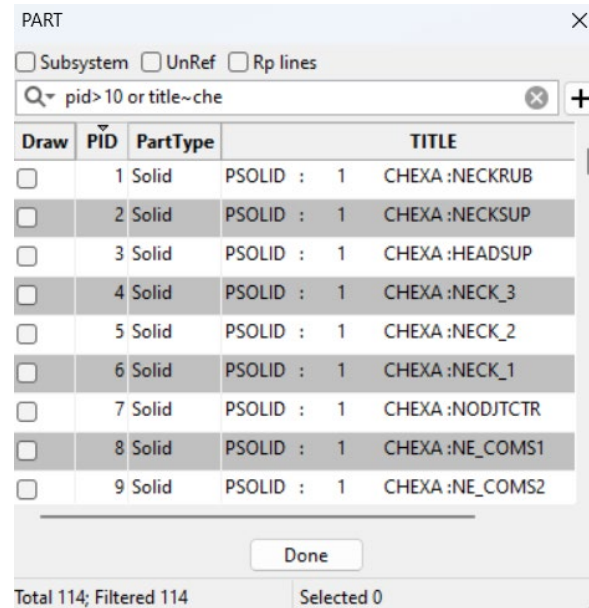
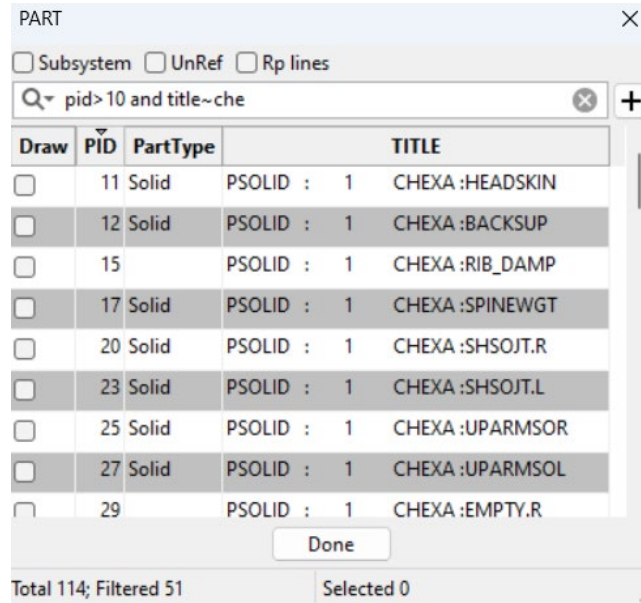
Done

Keyword Data Search Manager

- Supported operations in “Keyword Data Search Manager”:
 - List information of subsystem that the keyword belongs to
 - List if the keyword is unreferenced
 - List keyword data of repeatable card
 - List toggled parameters
 - List part information
 - Display keywords as filtered parameter values
 - Sort displayed keywords
 - Draw keyword entities
 - Right click menus

Supported operations on “Keyword Data Search Manager”

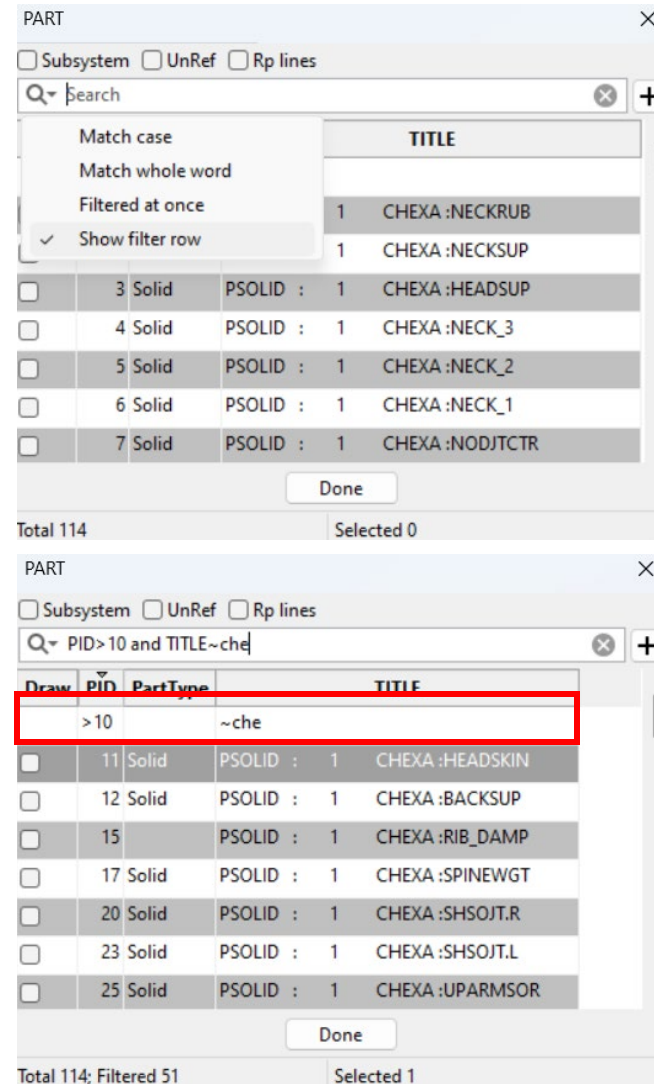
- Display keywords as filtered parameter values



- Click at the front area of the search widget to set match conditions
 - “Filtered at once” means filtered keywords once the text of search widget changed

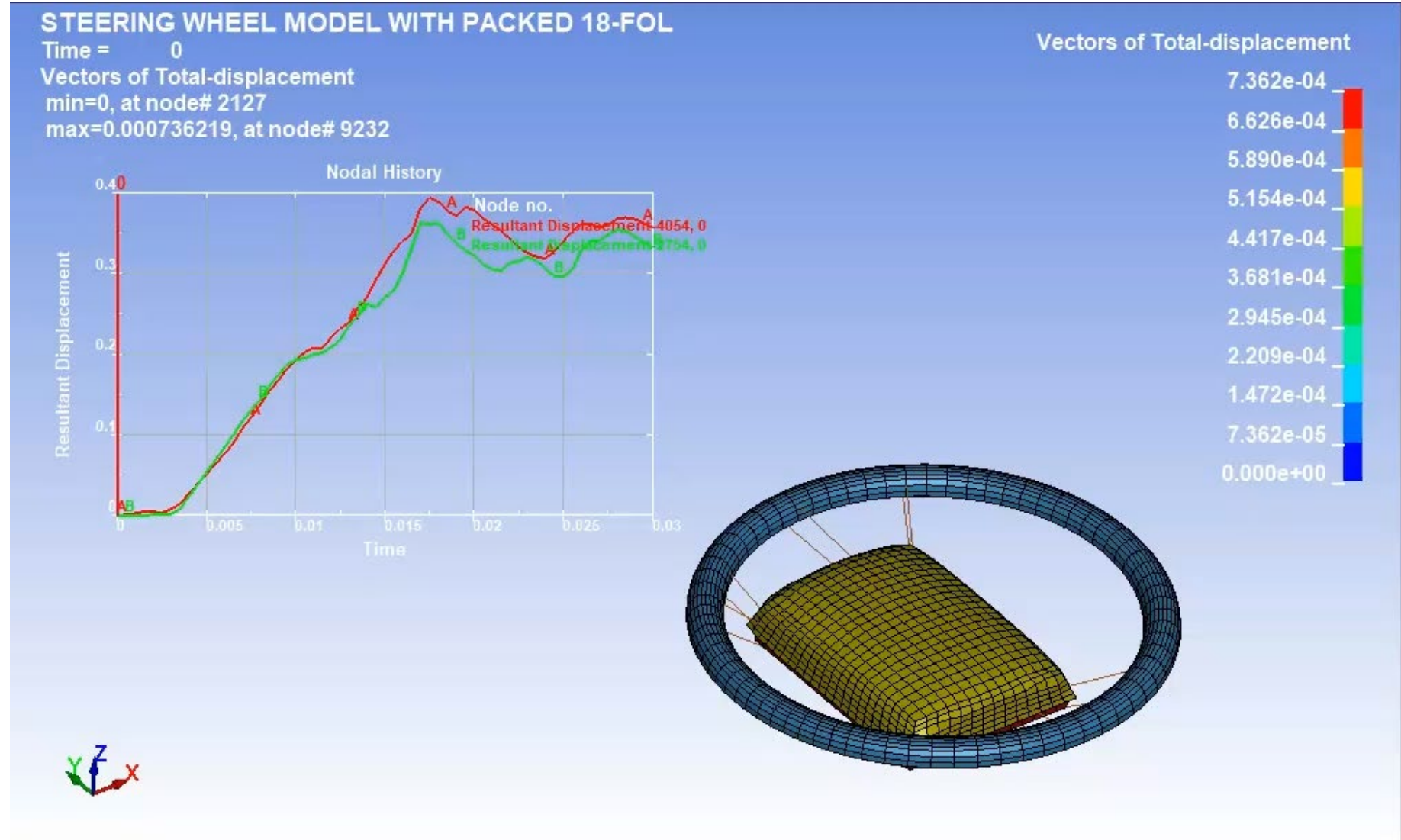
Supported operations on “Keyword Data Search Manager”

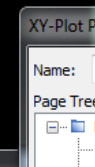
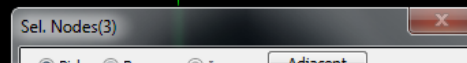
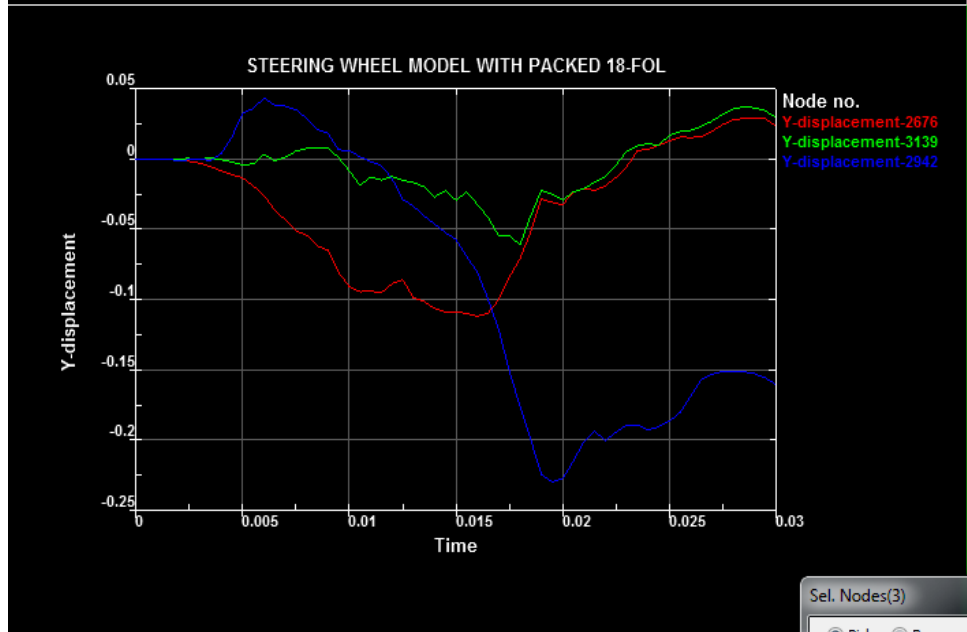
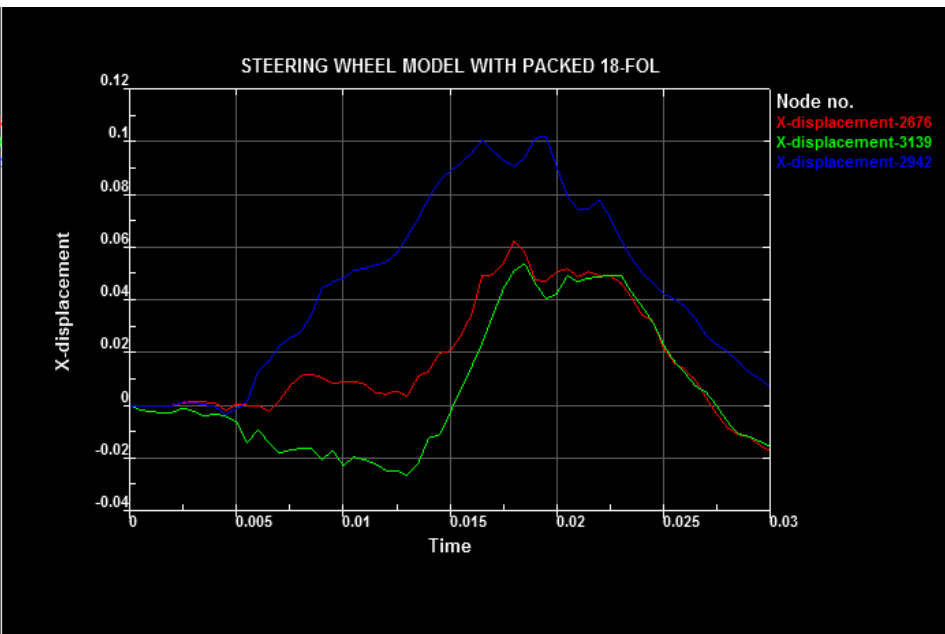
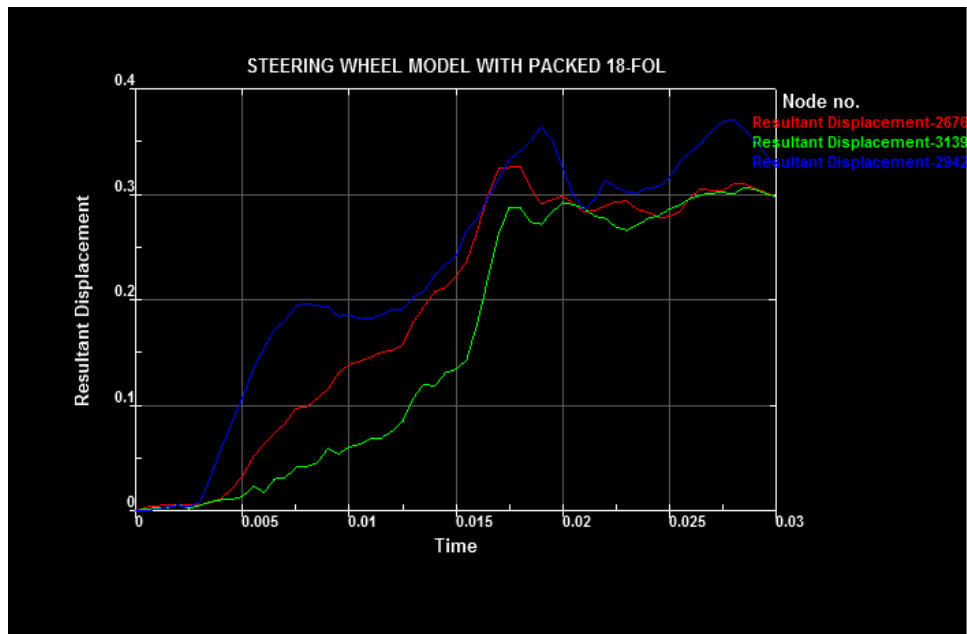
- Display keywords as filtered parameter values
 - Toggle the menu “Show filter row” to display the filter row in the grid
 - Input filter value of the parameter
 - Supported symbols of parameter value filter: >, GT, >=, GE, <=, LE, !=, NE, =, EQ, <, LT, ~
 - Supported symbols of combined conditions: “and”/“&&”, “or”/“||”



Post-Processing

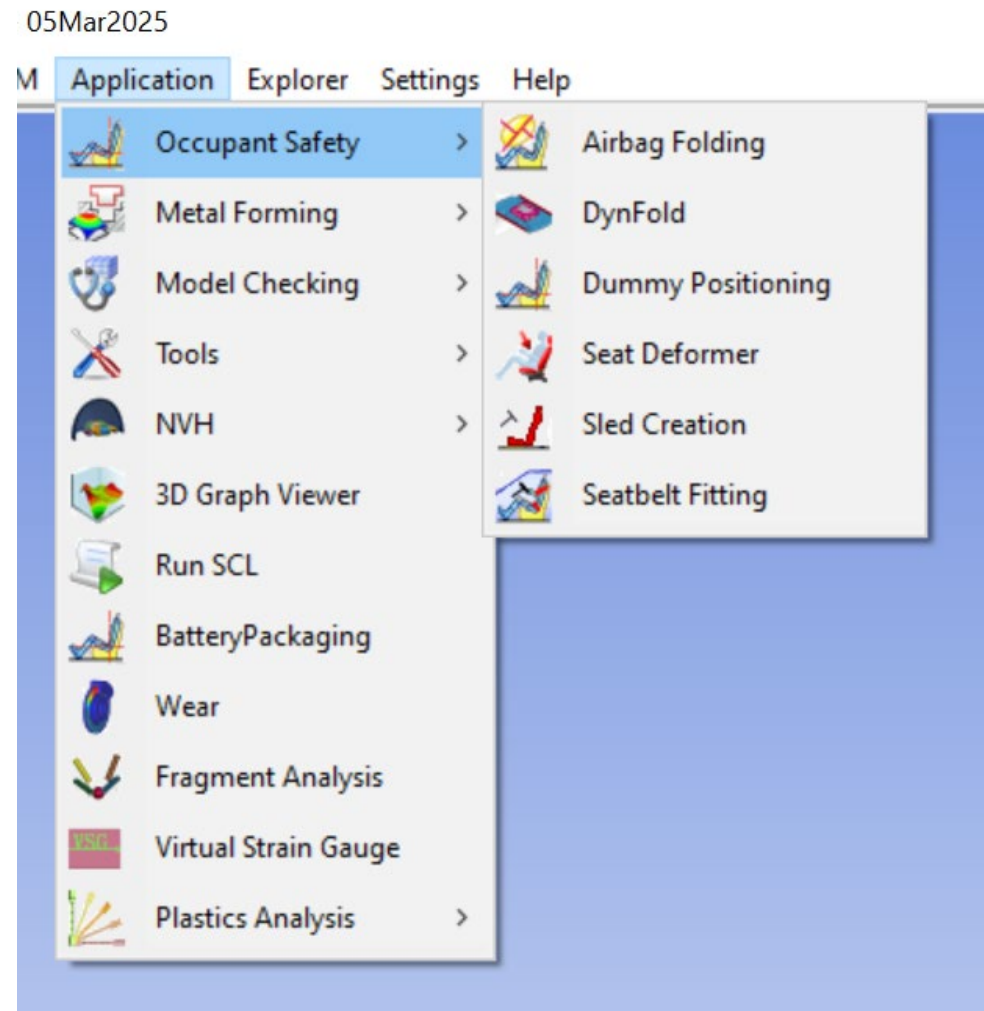
- Animation
- Vector plot
- Fringe contour
- XY history





LS-PrePost – Applications

- An application - Many operations being grouped together into one interface for a special purpose
 - Airbag folding
 - Dummy positioning
 - Seatbelt fitting
 - Seat Deformer
 - Sled Creation
 - Battery Packaging
 - Wear Analysis
 - Fragment Analysis
 - Virtual Strain Gauge

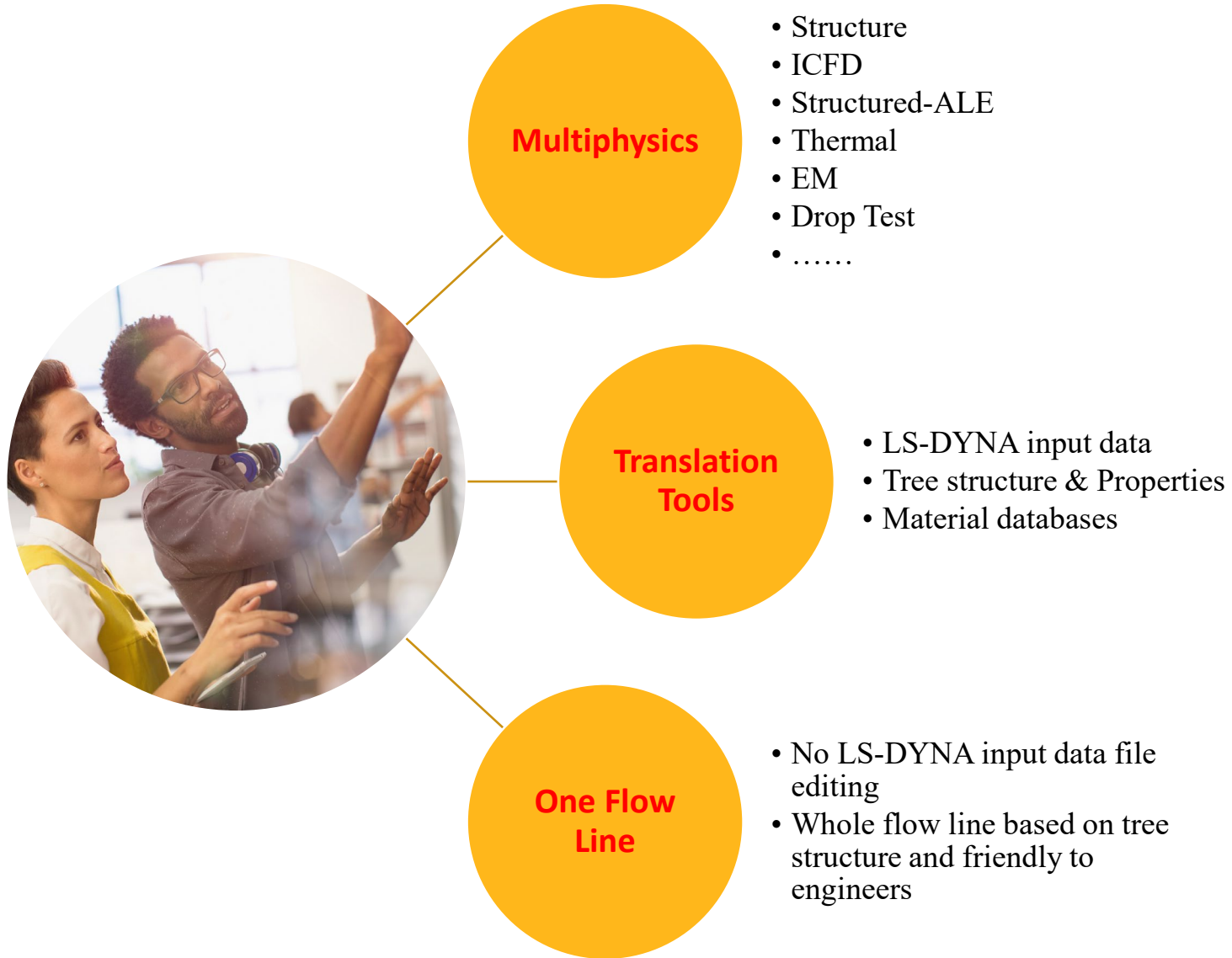




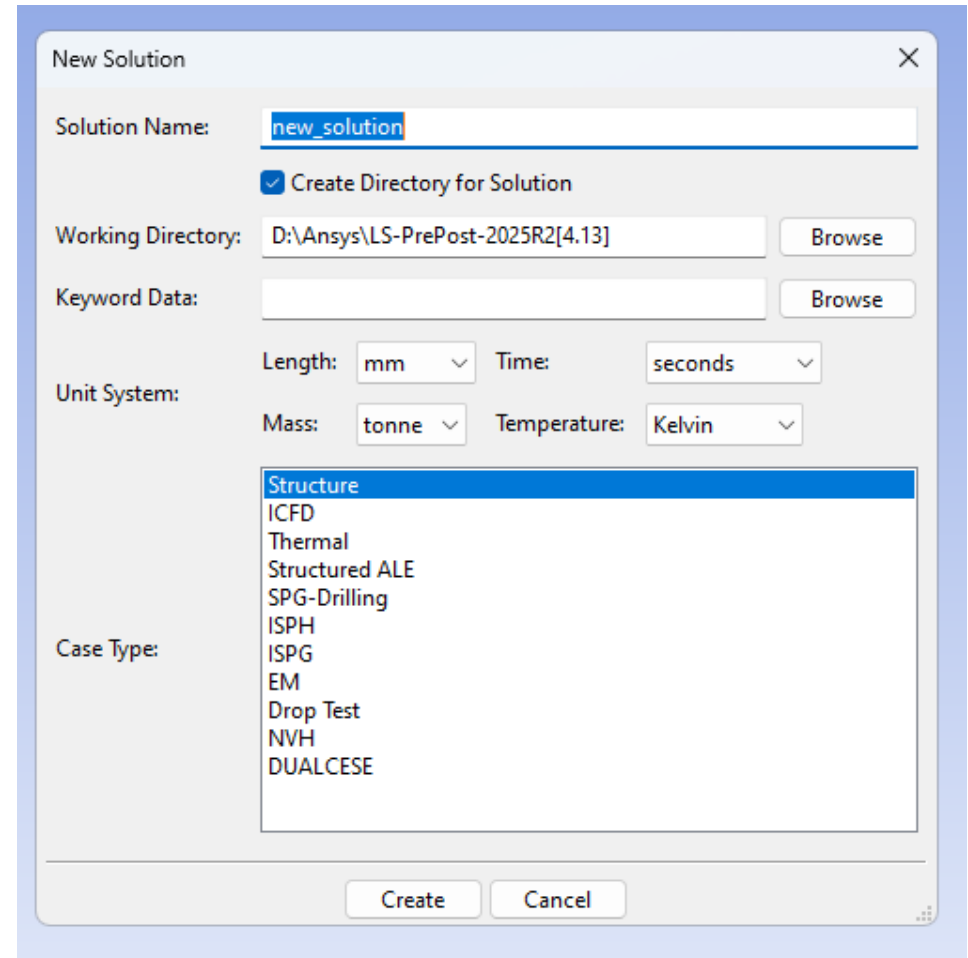
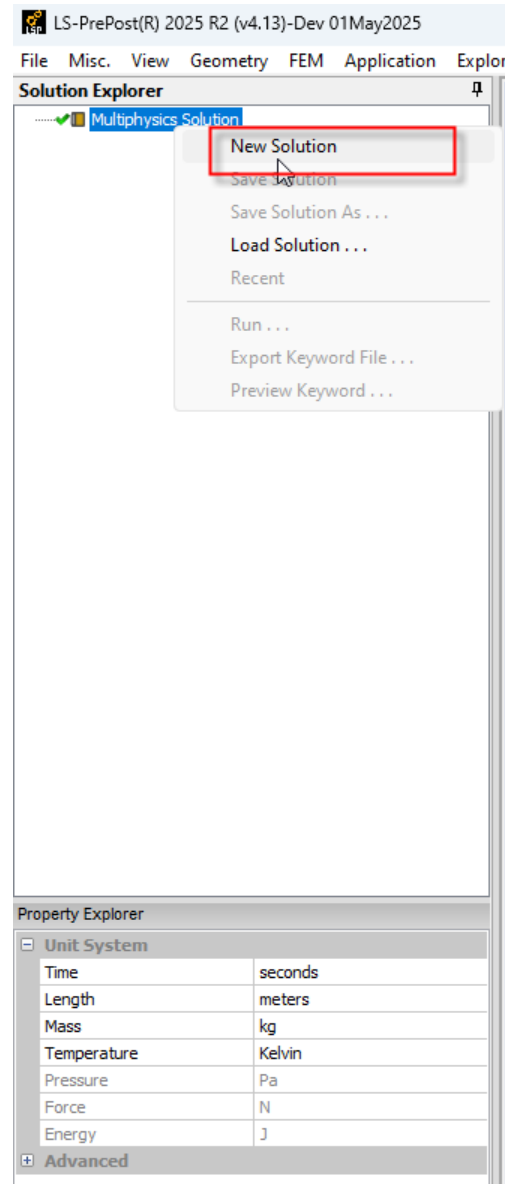
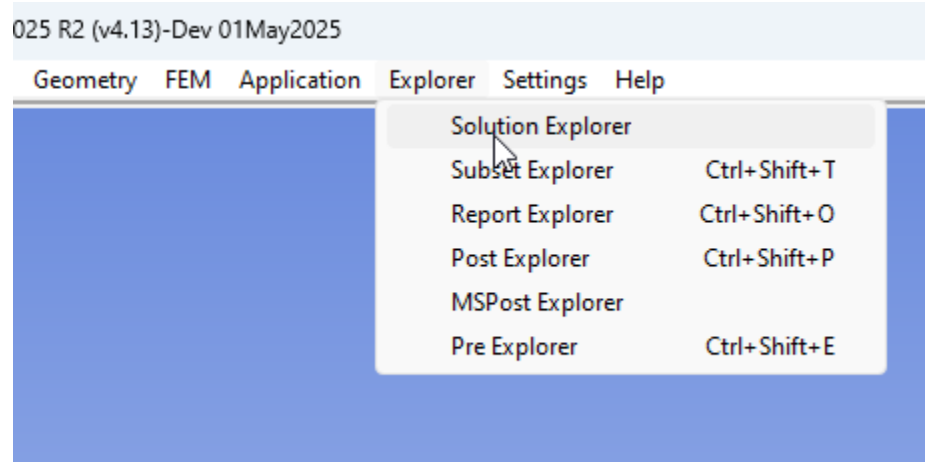
Powering Innovation That Drives Human Advancement

Solution Explorer

Solution Explorer

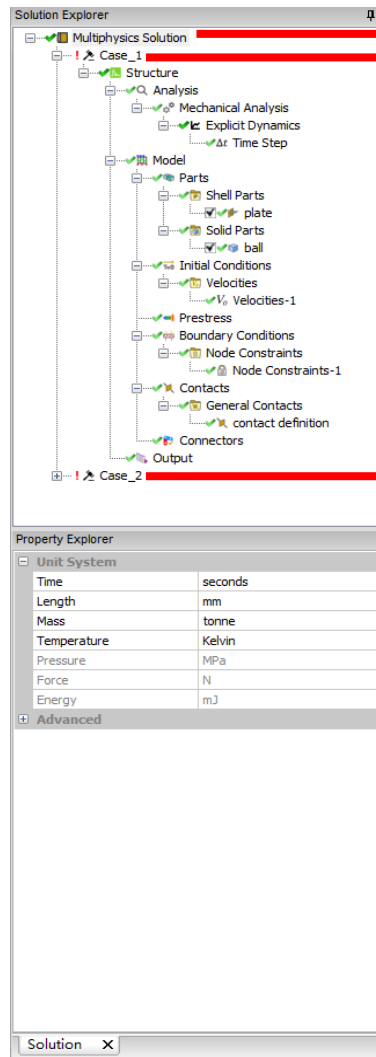


Solution Explorer Entrance



Folder	LS-DYNA Input	Case
• Solution Name • Working Directory	• Mesh data (Part, Element data, Node data, Material...) • Unit System	• Structure • ICFD • ... • EM • Drop Test

Solution Explorer Tree Structure



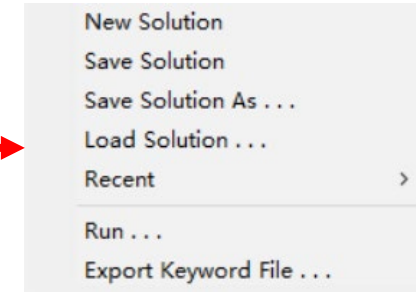
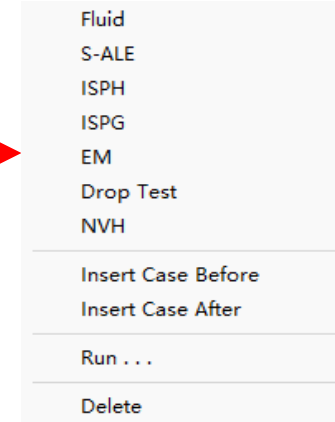
Top Level

Case Level

Structural Tree Structure

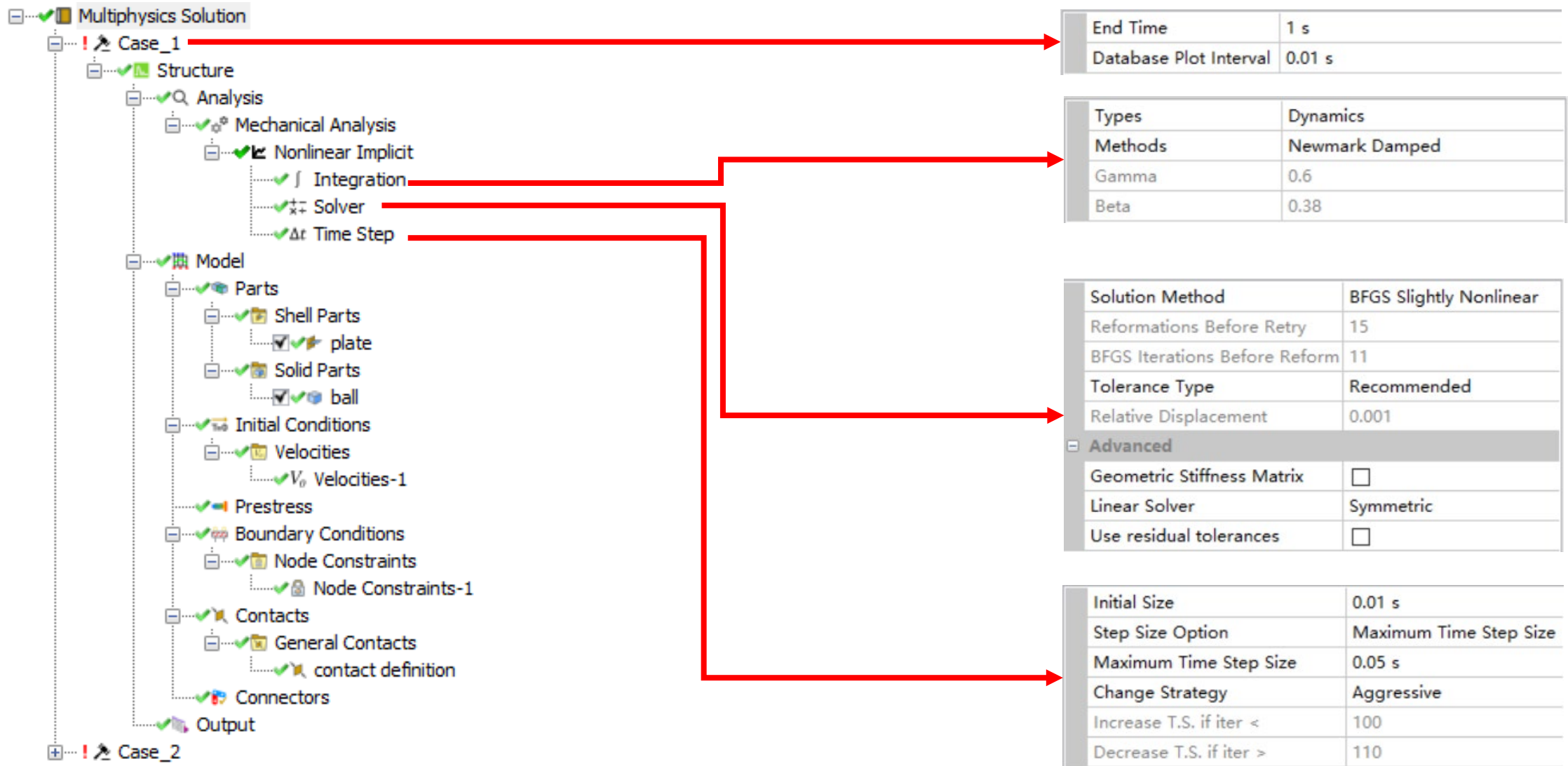
Case Level

Property Explorer

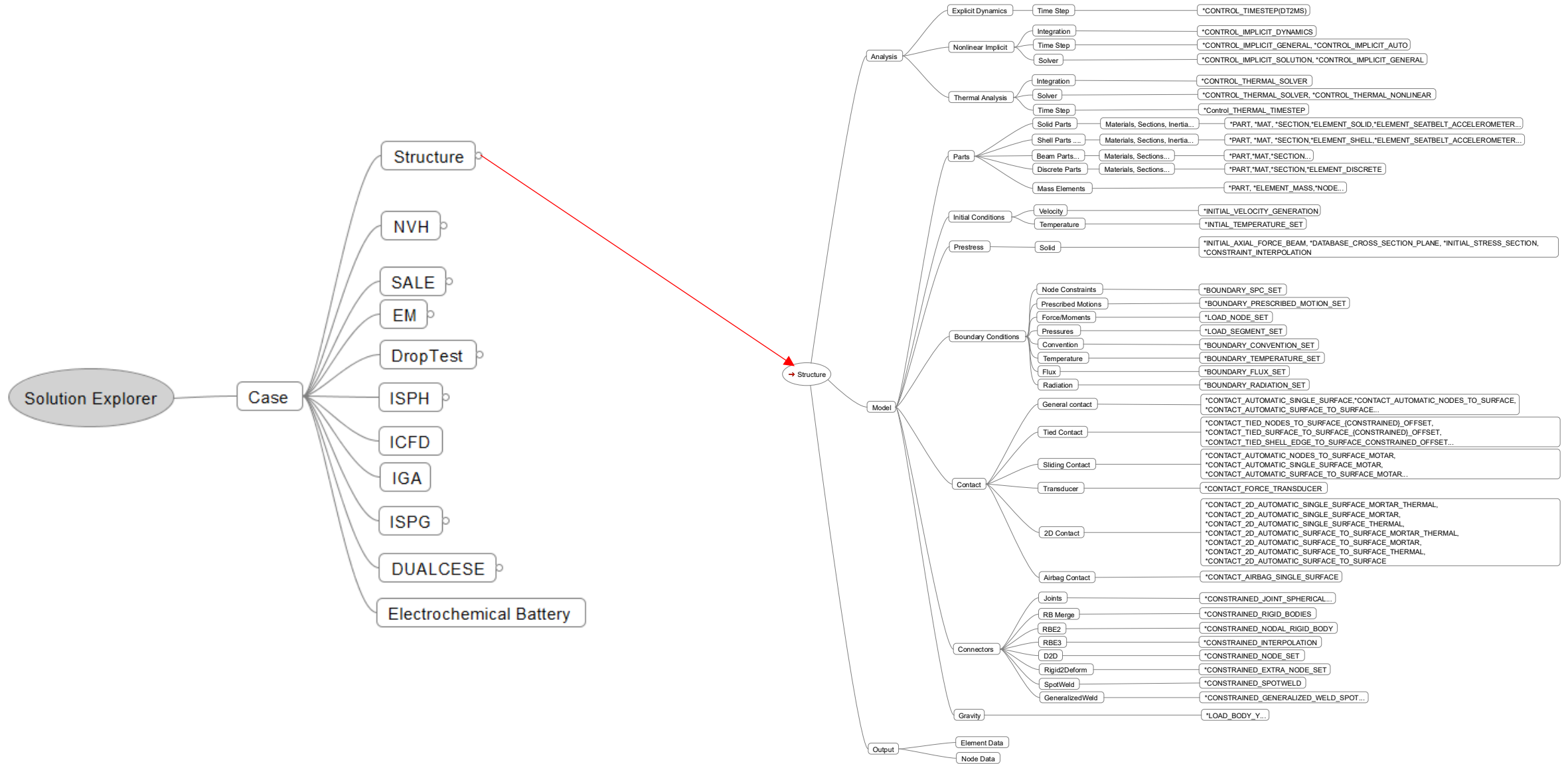


1. Different tree item has different properties.
2. Edit the properties to change analysis data.
3. Many tools can be designed to the properties.
4. The descriptions of the properties are friendly to engineers.

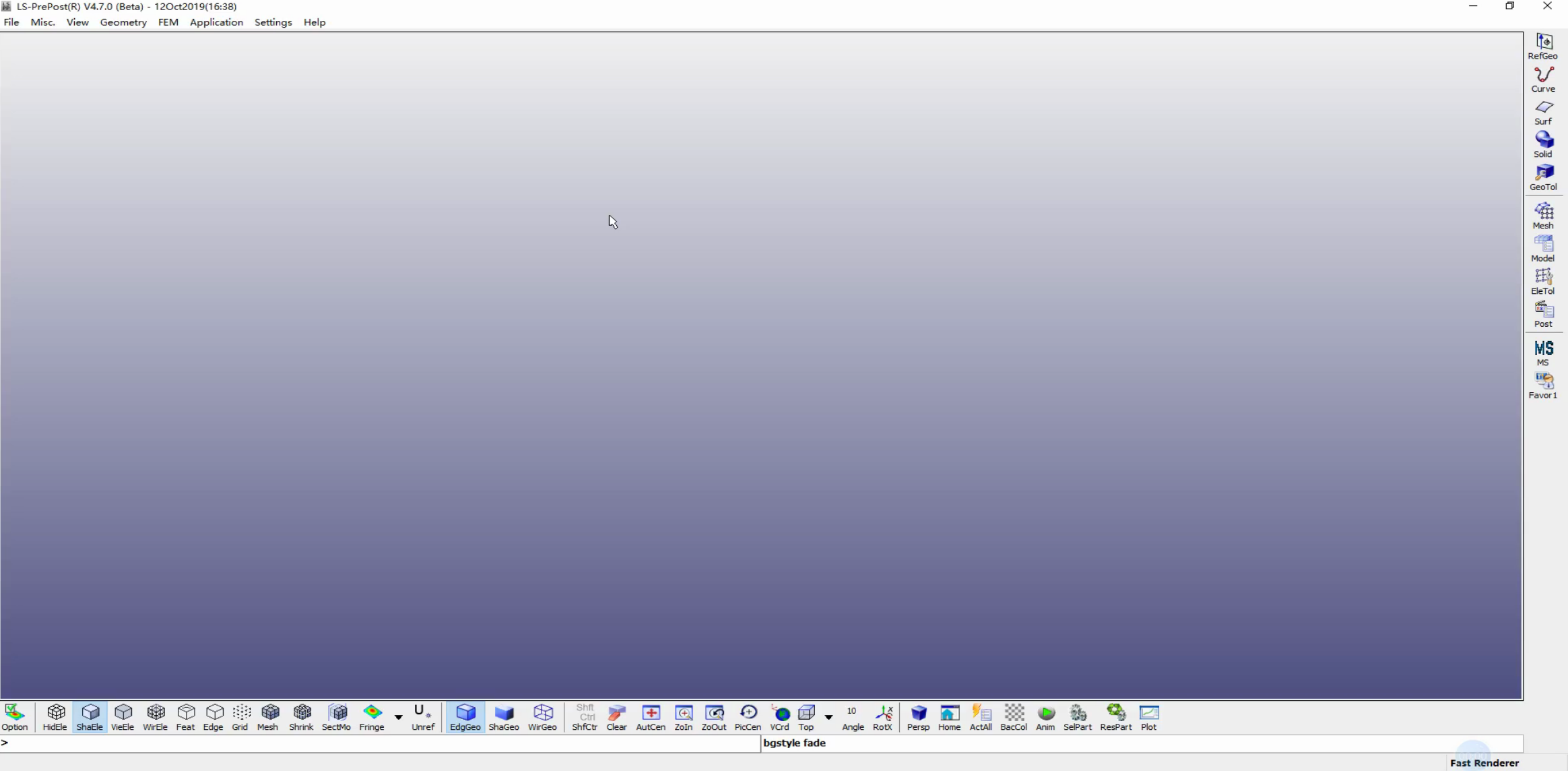
Solution Explorer Tree Structure



Maps of LS-DYNA input data from tree objects



Solution Explorer Demo Video



Solution Explorer Example Highlights (NVH)

New Solution

Solution Name: new_solution

☒ Create Directory for Solution

Working Directory: C:\TestData\NVH\Crash2NVH\Crash2NVH Browse

Keyword Data: C:\TestData\NVH\Crash2NVH\sa Browse

Unit System:

Length: mm Time: seconds

Mass: tonne Temperature: Kelvin

Case Type:

- Structure
- ICFD
- Thermal
- Structured ALE
- ISPH
- ISPG
- EM
- Drop Test
- NVH**
- DUALCESE

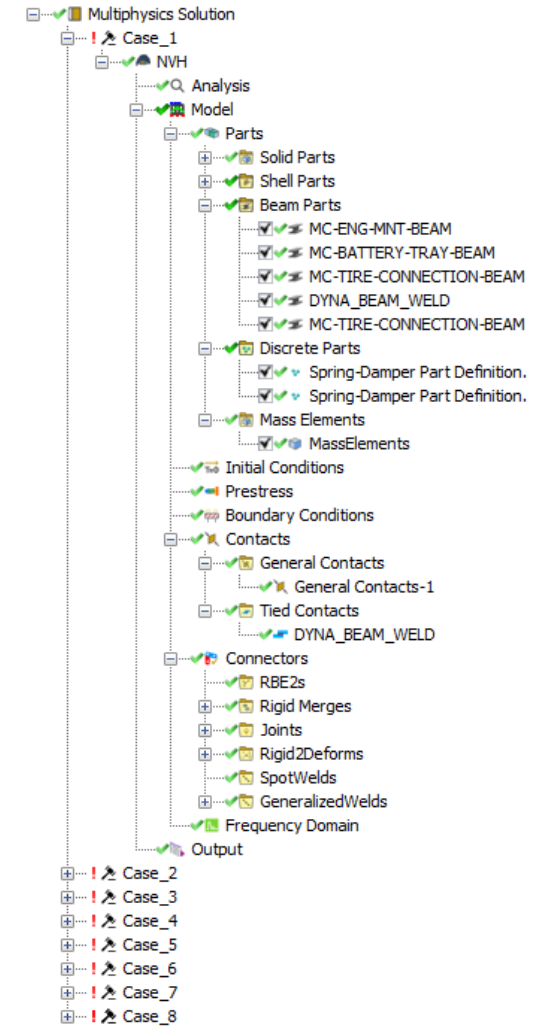
Sub Case:

- ☒ Frequency Response Function
- ☒ Steady State Dynamics
- ☒ Random Vibration
- ☒ Response Spectrum
- ☒ DDAM
- ☒ Acoustic FEM
- ☒ Acoustic BEM

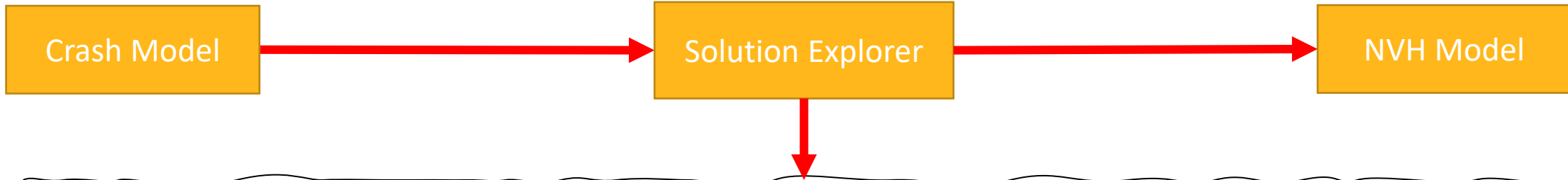
Create Cancel

Click “One Button” to create multiple solution trees at one time:

- Case_1: Eigen Value
- Case_2: Frequency Response Function
- Case_3: Steady State Dynamics
- Case_4: Random Vibration
- Case_5: Response Spectrum
- Case_6: DDAM
- Case_7: Acoustic FEM
- Case_8: Acoustic BEM



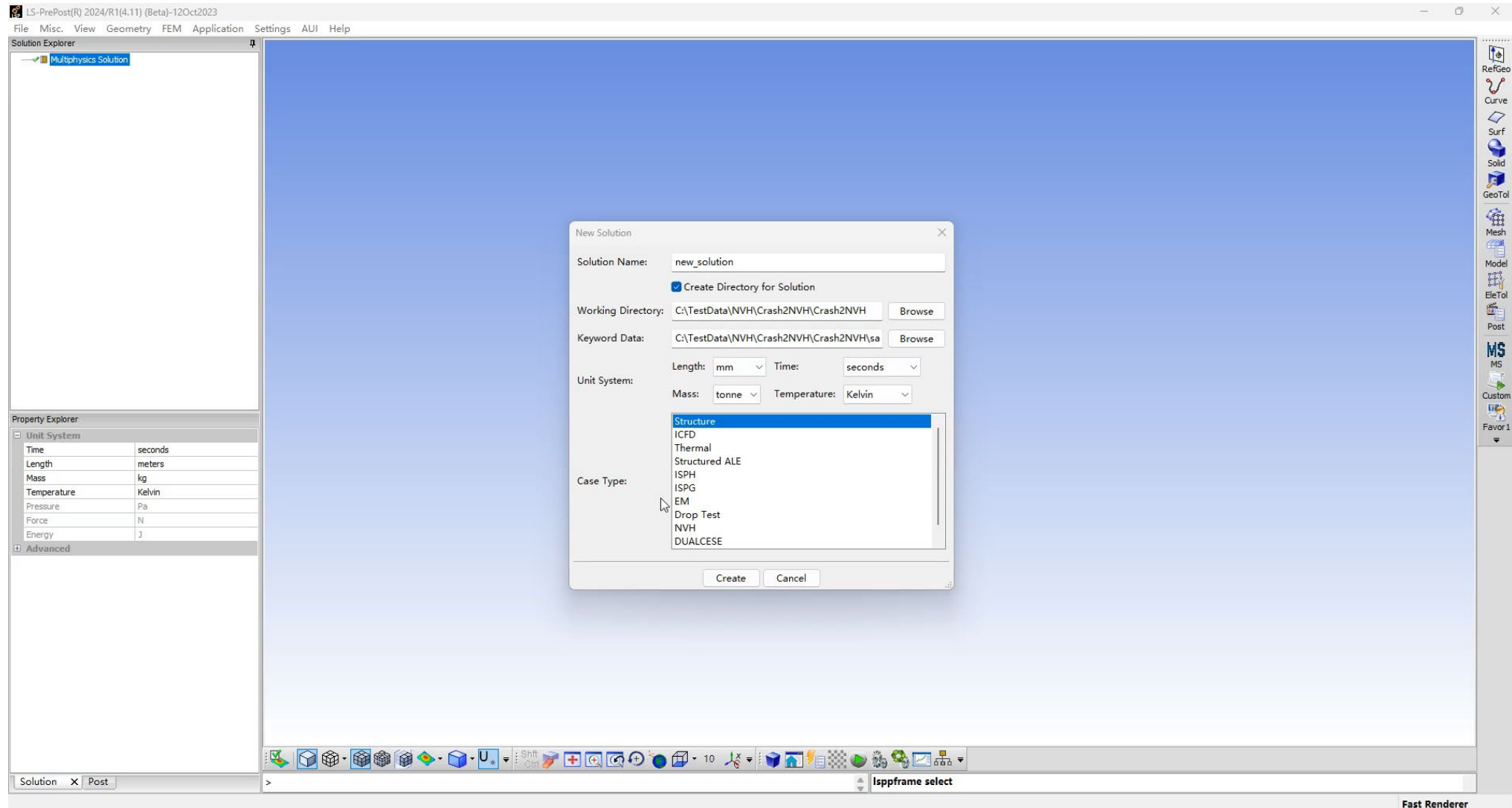
Solution Explorer Example Highlights (NVH: conversion rules)



1. **Disable:** Airbag, Gravity...
2. **Contact:** tied contact
 - 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
3. **Materials:** changing to linear material
Keep : *MAT_ELASTIC, *MAT_BLATZ-KO_RUBBER, *MAT_NULL, *MAT_RIGID, *MAT_HONEYCOMB,
*MAT_CRUSHABLE_FOAM,*MAT_SPOTWELD,*MAT_SPRING_ELASTIC,*MAT_DAMPER_VISCOUS...
Change:*MAT_PIECEWISE_LINEAR_PLASTICITY to MAT_ELASTIC
4. **Section:** changing to linear elements
Linear element type: **Shell** 18,20, and 21; **solid** 18.
5. **Keep:** "Boundary Conditions" and "Connectors" like joints, spotwelds, rbe2s, rbe3s and so on.

Solution Explorer Example Highlights (NVH)

Click “One Button” to convert crash model to nvh model.



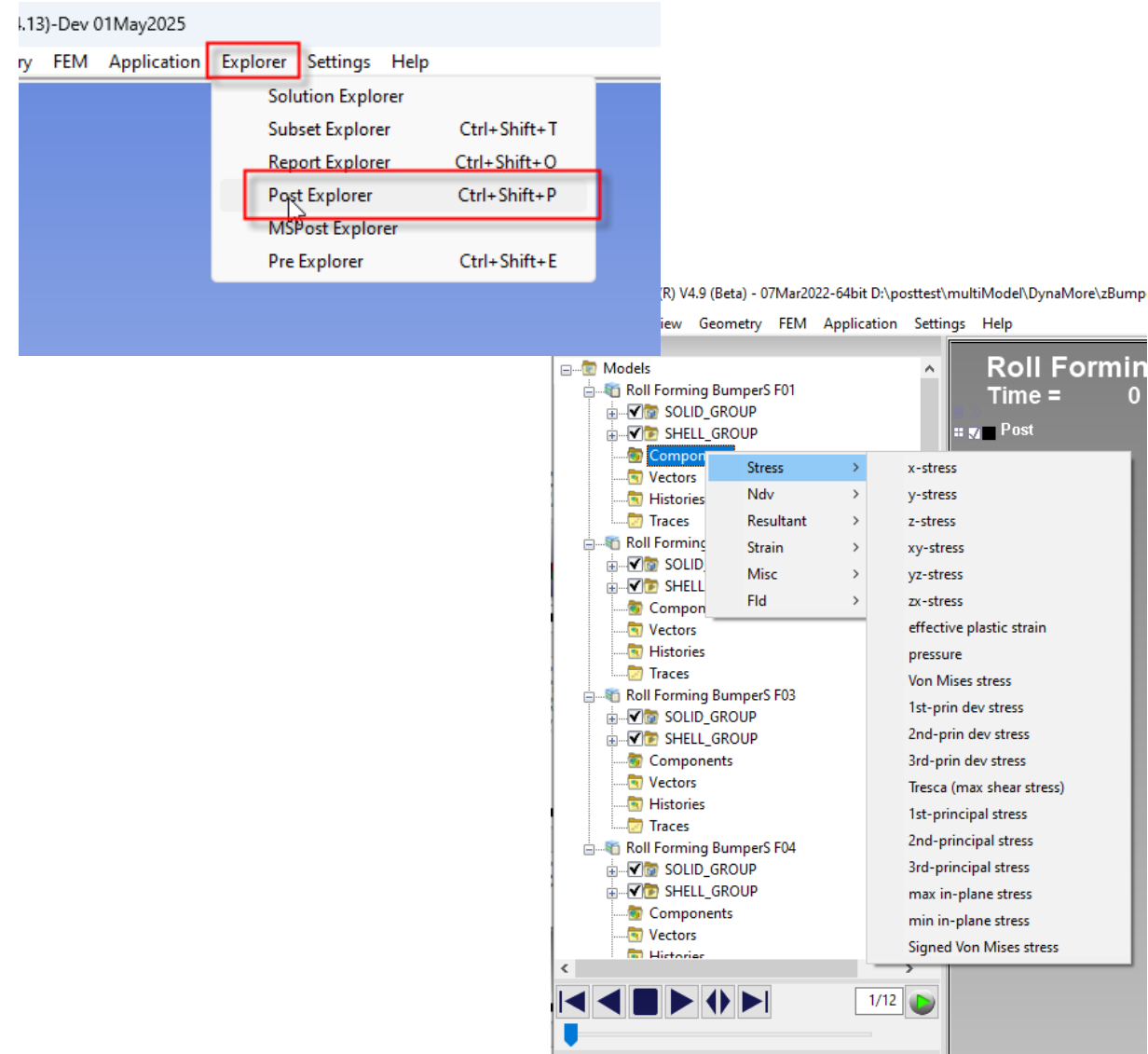


Powering Innovation That Drives Human Advancement

Post-Processing with Post Explorer

Post-Processing – Introduction of Post Explorer

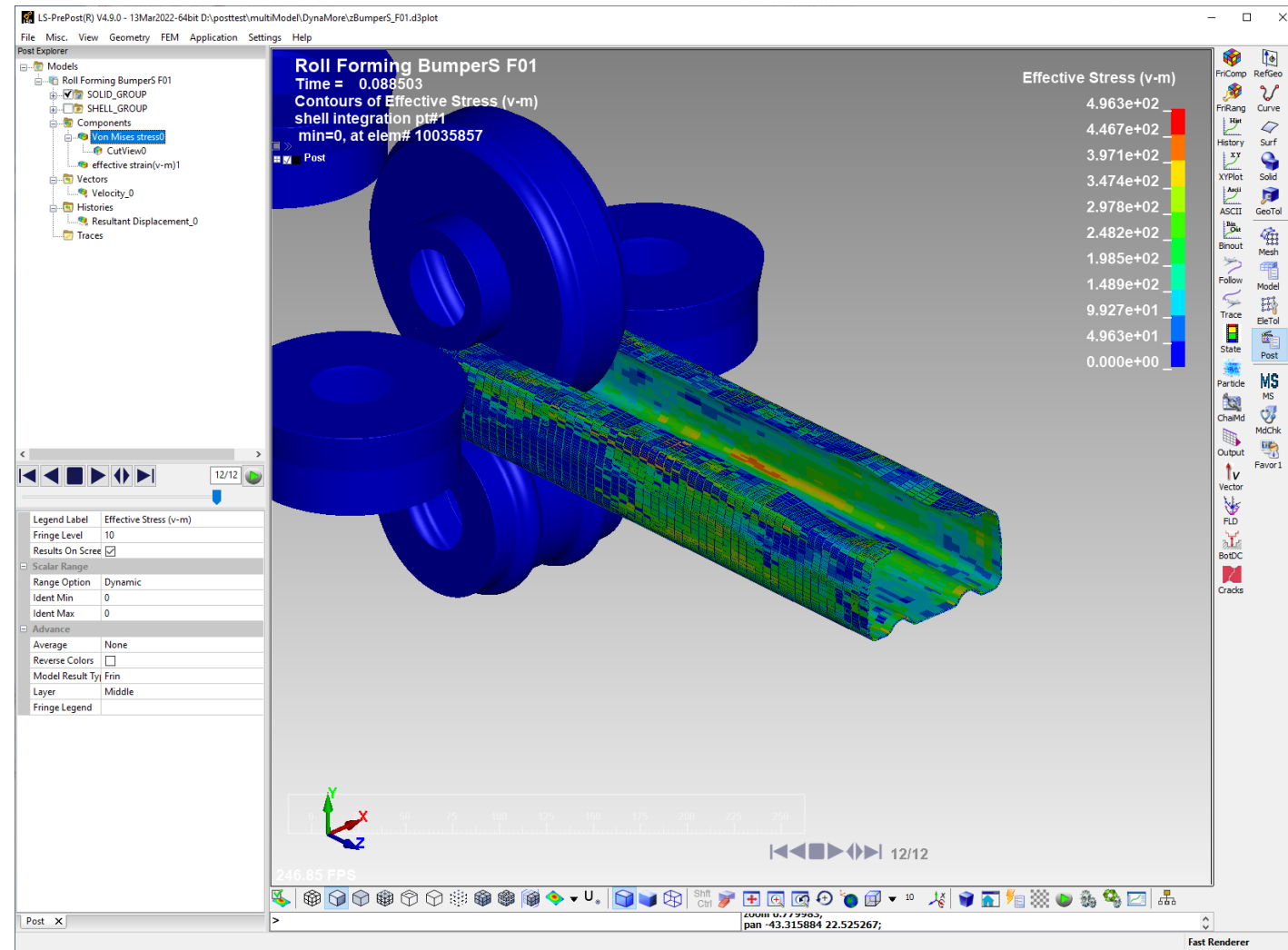
- A new paradigm of LS-DYNA post-processing in LS-PrePost
- It is feature tree-based operations
- To activate: pull down menu “Explorer”->”Post Explorer”
- Multiple models can be loaded and listed
- Each model has its own Components, Vectors, Histories, and Traces main objects
- Right click on these objects to select sub-object



Post-Processing – Introduction of Post Explorer

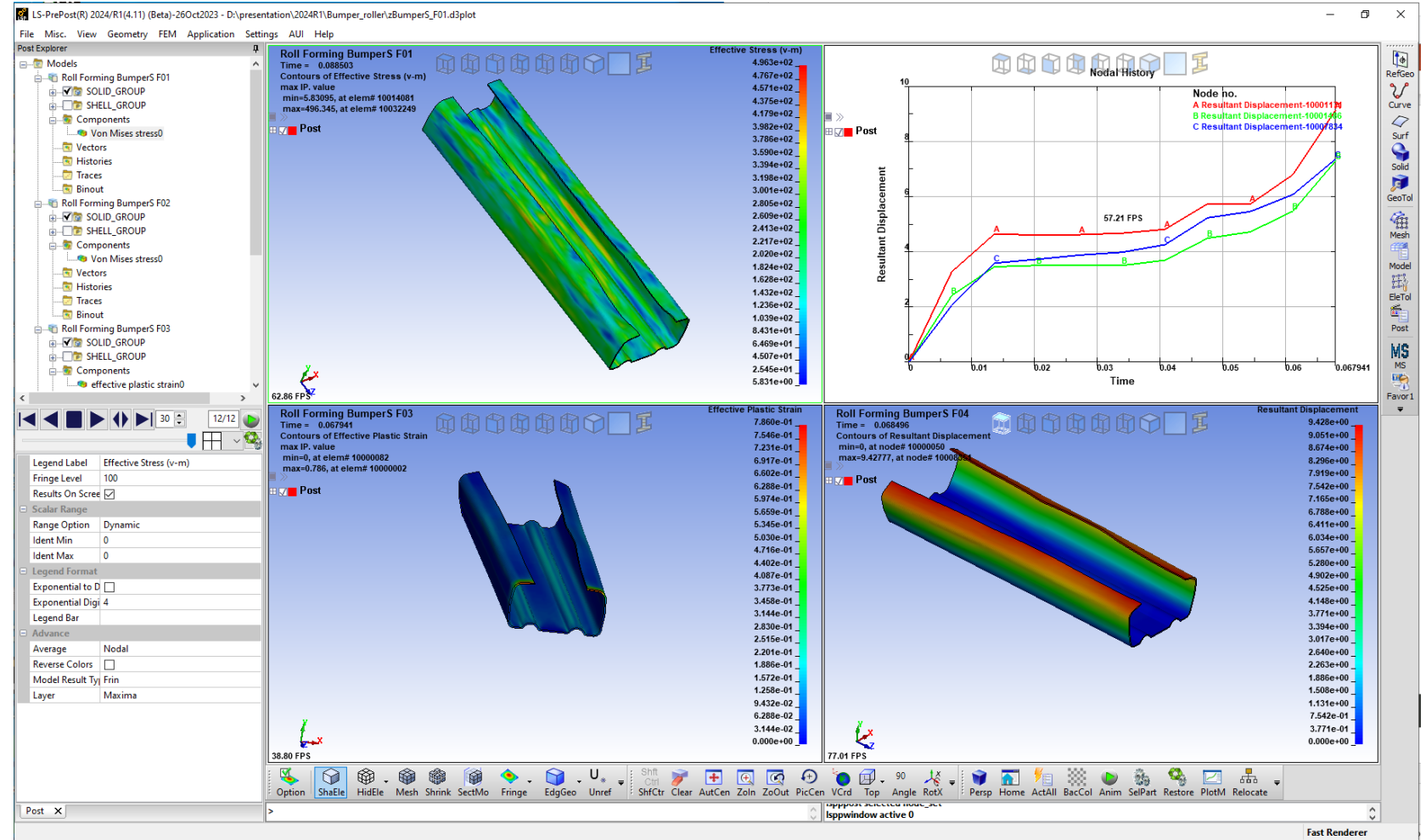
- Each selected object has its own property tree
- Many quantities associated with this object can be changed on the property tree

Legend Label	Effective Stress (v-m)
Fringe Level	11
Results On Screen	☒
Scalar Range	
Range Option	Dynamic
Ident Min	0
Ident Max	0
Advance	
Average	None
Reverse Colors	☐
Model Result Ty	Frin
Layer	Middle
Fringe Legend	Lower
	Middle
	Upper
	Maxima
	Average
	Minima



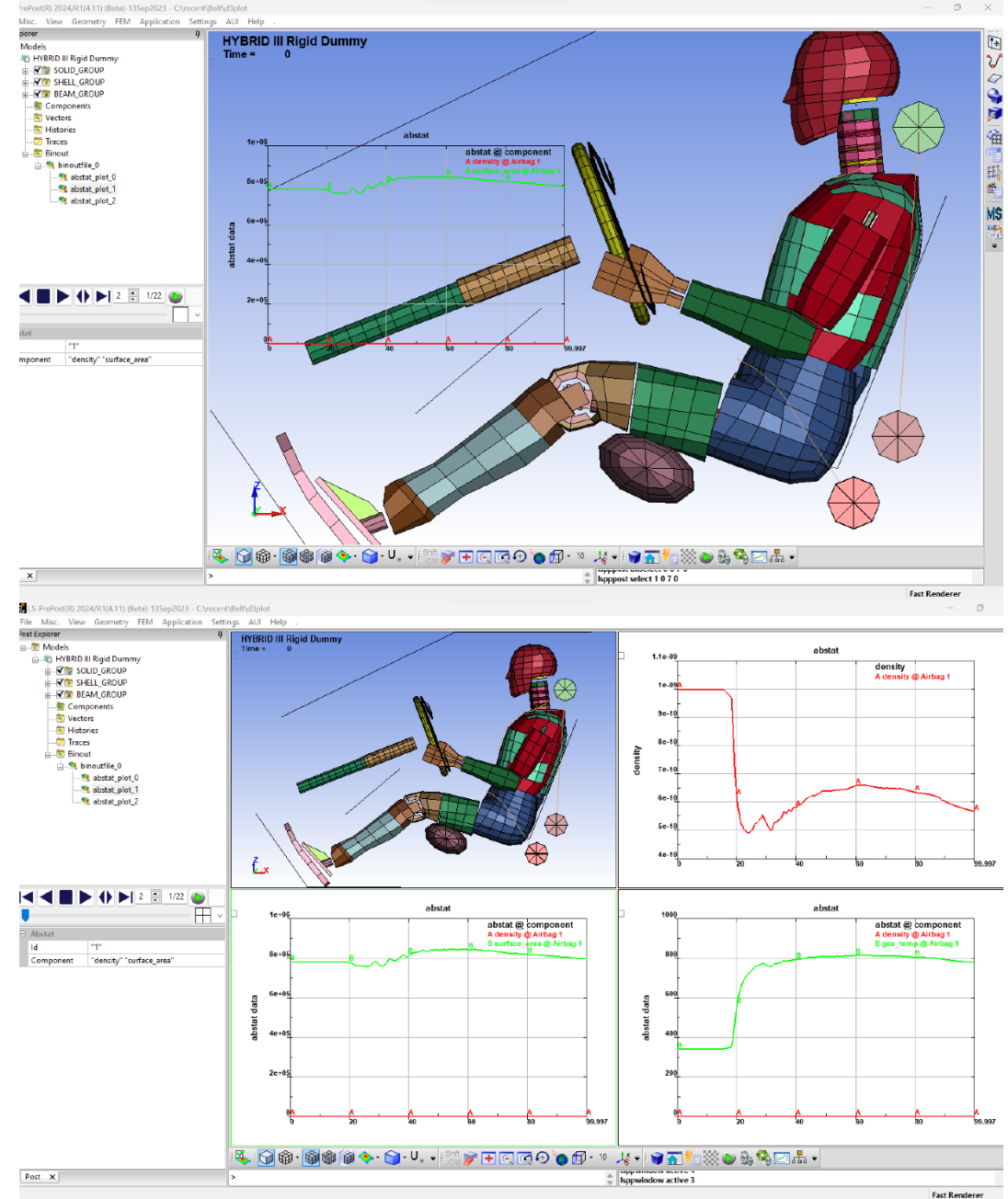
Post-Processing – Introduction of Post Explorer

- Multiple models can be shown on Split windows
- History data (XY plot) can also be drawn on one of the windows



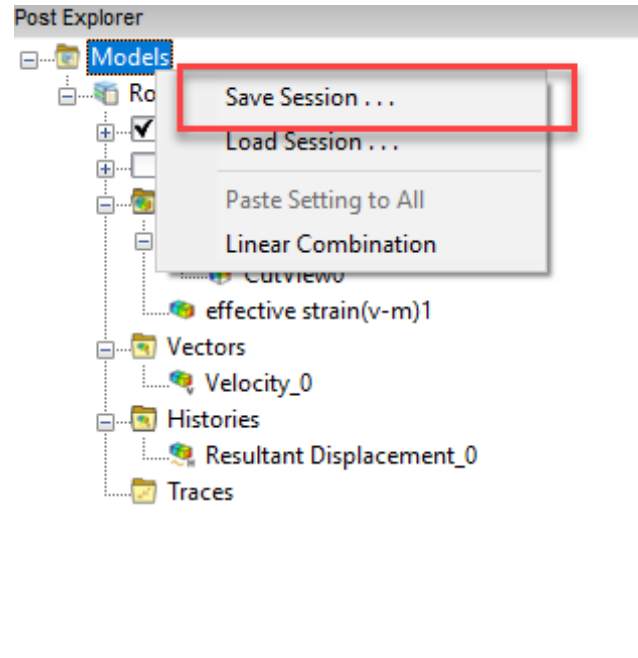
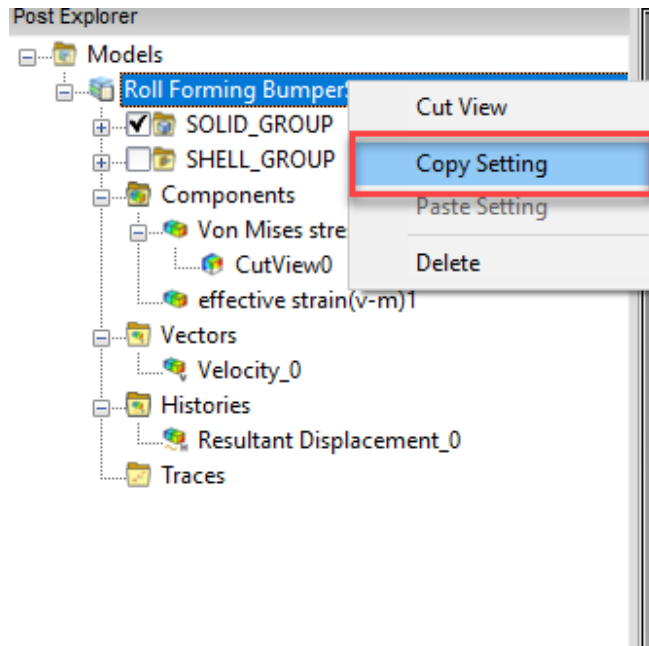
Post-Processing – Post Explorer

- Binout Support
- Binout history data is one of the object on the Feature tree
- Not all Binout branches are supported for now
- ASCII history data interface will come in the future



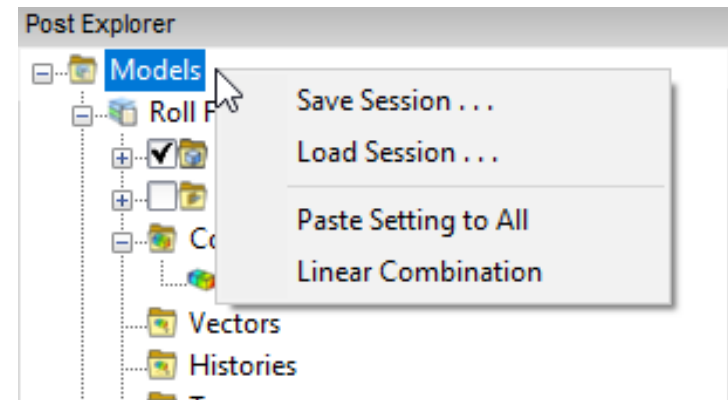
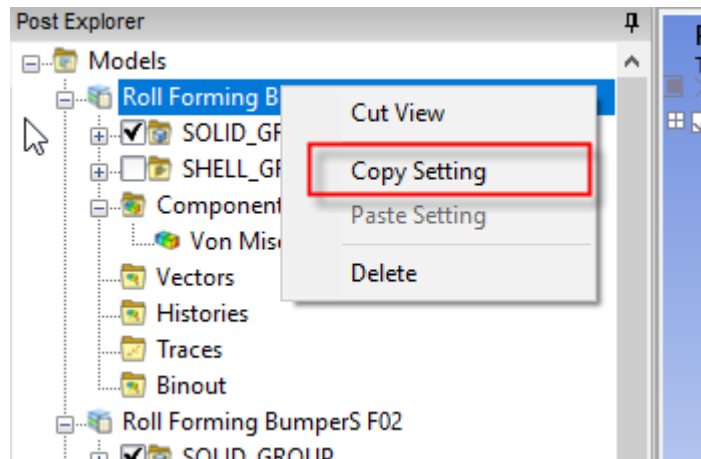
Post-Processing – Introduction of Post Explorer

- Many objects with their properties can be setup in the feature tree
- The Settings for one model can be copied and pasted to other model (if multiple modes have been loaded)
- All settings for the session can be saved to file and be loaded back in the future

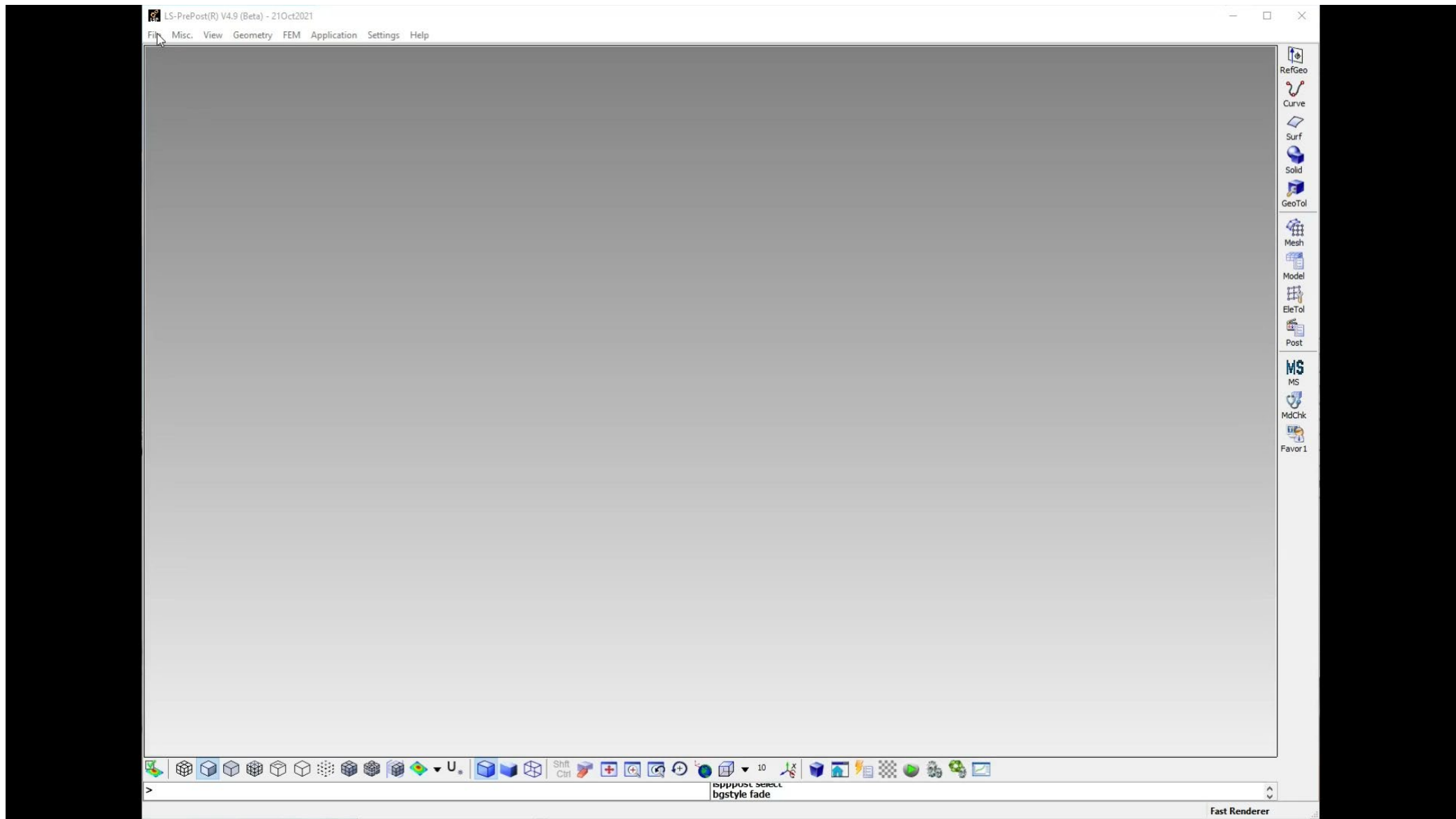


Post-Processing – Introduction of Post Explorer

- Many objects with their properties can be setup in the feature tree
- The Settings for one model can be copied and pasted to other model (if multiple modes have been loaded)
- All settings for the session can be saved to file and be loaded back in the future



Post-Explorer Demo





Powering Innovation That Drives Human Advancement

Work progress of PyDYNA & PyDPF

pyDyna

PyDYNA is a Pythonic package for providing a more convenient and complete way to build an Ansys LS-DYNA input deck, submit it to the Ansys LS-DYNA solver, and finally postprocess the results

<https://github.com/ansys/pydyna>
<https://dyna.docs.pyansys.com>

PyDyna

```
git clone https://github.com/pyansys/pyDyna
cd pyDyna
pip install -e .
```

```
unzip ansys-dyna-core-v0.3.5-wheelhouse-
Linux-3.8.zip -d wheelhouse
pip install ansys-dyna-core -f wheelhouse --no-
index --upgrade --ignore-installed
```

PyDPF

<https://github.com/ansys/pydpf-post>

Pre

```
from ansys.dyna.core.pre.dynasolution import DynaSolution
solution = DynaSolution(hostname)
solution.open_files(fns)
solution.set_termination(termination_time=10)
ballplate = DynaMech(AnalysisType.NONE)
solution.add(ballplate)
solution.download(serveroutfile,downloadfile)
```

Solver

```
import ansys.dyna.core.solver as solver
dyna=solver.DynaSolver(hostname,port)
dyna.push("./output/ball_plate.k")
dyna.start(4)
dyna.run("i=ball_plate.k memory=10m ncycle=20000")
```

Post

```
from ansys.dpf import core as dpf
ds = dpf.DataSources()
ds.set_result_file_path(r'./d3plot', 'd3plot')
resultOp = dpf.Operator("lsdyna::d3plot::stress_von_mises")
result = resultOp.outputs.stress_von_mises()
```

Docker Service

Pre-Service

```
docker compose up -d
```

gRPC Architecture

Solver-Service

```
docker compose up -d
```

DynaSolution

DynaMech
DynaIGA
DynaICFD
DynaSALE
DynaEM
DynaAirbag

dyna.docs.pyansys.com

My AppsPHOIndex of /anonymo...Settings - PasswordsBookmarks bar

PyAnsys

Getting startedUser guideAPI referenceExamplesContributeMore ▾

Search the docs ...

Ctrl + K

0.8 (stable) ▾

PyAnsys >

PyDYNA documentation 0.8.0

Py

Ansys

python

3.10 | 3.11 | 3.12 | 3.13

pypi

v0.8.0

CI

passing

codecov

unknown

License

MIT

code style

black

Overview

PyDYNA is a Pythonic package for providing a more convenient and complete way to build an Ansys DYNA input deck, submit it to the Ansys LS-DYNA solver, and finally postprocess the results.

PyDYNA contains three submodules, `ansys.dyna.core.pre`, `ansys.dyna.core.solver`, and `ansys.dyna.core.run`.

- `pre`: This module provides highly abstracted APIs for creating and setting up DYNA input decks. There are many classes supported, namely, DynaMech, DynaIGA, DynaICFD, DynaSALE, DynaEM, DynaNVH, DynaMaterial, DynaISPH, DynaICFD and DynaAirbag. Each of these classes can be used to generate LS-DYNA keywords. Since these classes have high-level abstraction, each function call generates groups of keywords needed to define an input in LS-DYNA.
- `solver`: This API provides features to interact directly with the Ansys LS-DYNA solver. LS-DYNA is primarily a batch solver with very limited interactive capabilities, the `solver` service provides a way to push input files to the LS-DYNA solver, monitor the state of the running job, change the value of a load curve and finally retrieve result files back from the server
- `run`: This module provides the ability to start the LS-DYNA solver. This does not require any client-server library or Docker container.

Once you have results, you can use the Ansys Data Processing Framework (DPF), which is designed to provide numerical simulation users and engineers with a toolbox for accessing and transforming simulation data. DPF can access data from Ansys solver files and from several files with neutral formats, including CSV, HDF5, and VTK. Using DPF's various operators, you can manipulate and transform this data.

The [ansys-dpf-post package](#) provides a simplified Python interface to DPF, thus enabling rapid postprocessing without ever leaving a Python environment. For more information on DPF-Post, see the [DPF-Post documentation](#).

On this page

PyDYNA documentation 0.8.0

Overview

Documentation and issues

License

Edit on GitHub

[Show Source](#)

40

©2025 ANSYS, Inc. / Proprietary. Do Not Share.

Powering Innovation That Drives Human Advancement

Documentation and issues

Documentation for the latest stable release of PyDyna is hosted at [PyDYNA documentation](#).

For examples on how to use PyDYNA, see [Examples](#) in the PyDYNA documentation.

In the upper right corner of the documentation's title bar, there is an option for switching from viewing the documentation for the latest stable release to viewing the documentation for the development version or previously released versions.

On the [PyDYNA Issues](#) page, you can create issues to report bugs and request new features. On the [PyDYNA Discussions](#) page or the [Discussions](#) page on the Ansys Developer portal, you can post questions, share ideas, and get community feedback.

To reach the project support team, email pyansys.core@ansys.com.

License

PyDYNA is licensed under the MIT license.

PyDYNA makes no commercial claim over Ansys whatsoever. This library extends the functionality of Ansys LS-DYNA by adding a Python interface to LS-DYNA without changing the core behavior or license of the original software. The use of the interactive control of PyDYNA requires a legally licensed local copy of LS-DYNA.

For more information on LS-DYNA, see the [Ansys LS-DYNA](#) page on the Ansys website.

PyDYNA Workflow



- build an Ansys DYNA input deck
- submit it to the Ansys LS-DYNA solver

interact directly with the Ansys LS-DYNA solver

postprocess the results

For more information about PyDYNA: [PyDYNA documentation 0.8.0 PyDYNA](#)

PyDYNA - Start Server locally

Update server package automatically

Start pydyna locally	Windows	Linux
ansys-pydyna-pre-server.zip	Download server package automatically	Download server package automatically
ansys-pydyna-solver-server.zip	Installed the ANSYS locally Download server package automatically	Installed the openmpi package Download server package automatically

PyDYNA - Start Server in Docker

Setup pydyna-pre docker.

Ensure that Docker is installed(Linux: Docker Engine, Windows: Docker Desktop)

File name	Description
Dockerfile	Build docker image
Linux-binaries.zip	Include library and .proto file,dowload from: https://github.com/ansys/pydyna/releases/download/v0.3.1/linux-binaries.zip
Docker-compose.yml	“docker compose up -d”
README.rst	How to start pydyna-pre container

For more information about PyDYNA: [PyDYNA documentation 0.8.0 — PyDYNA](#)

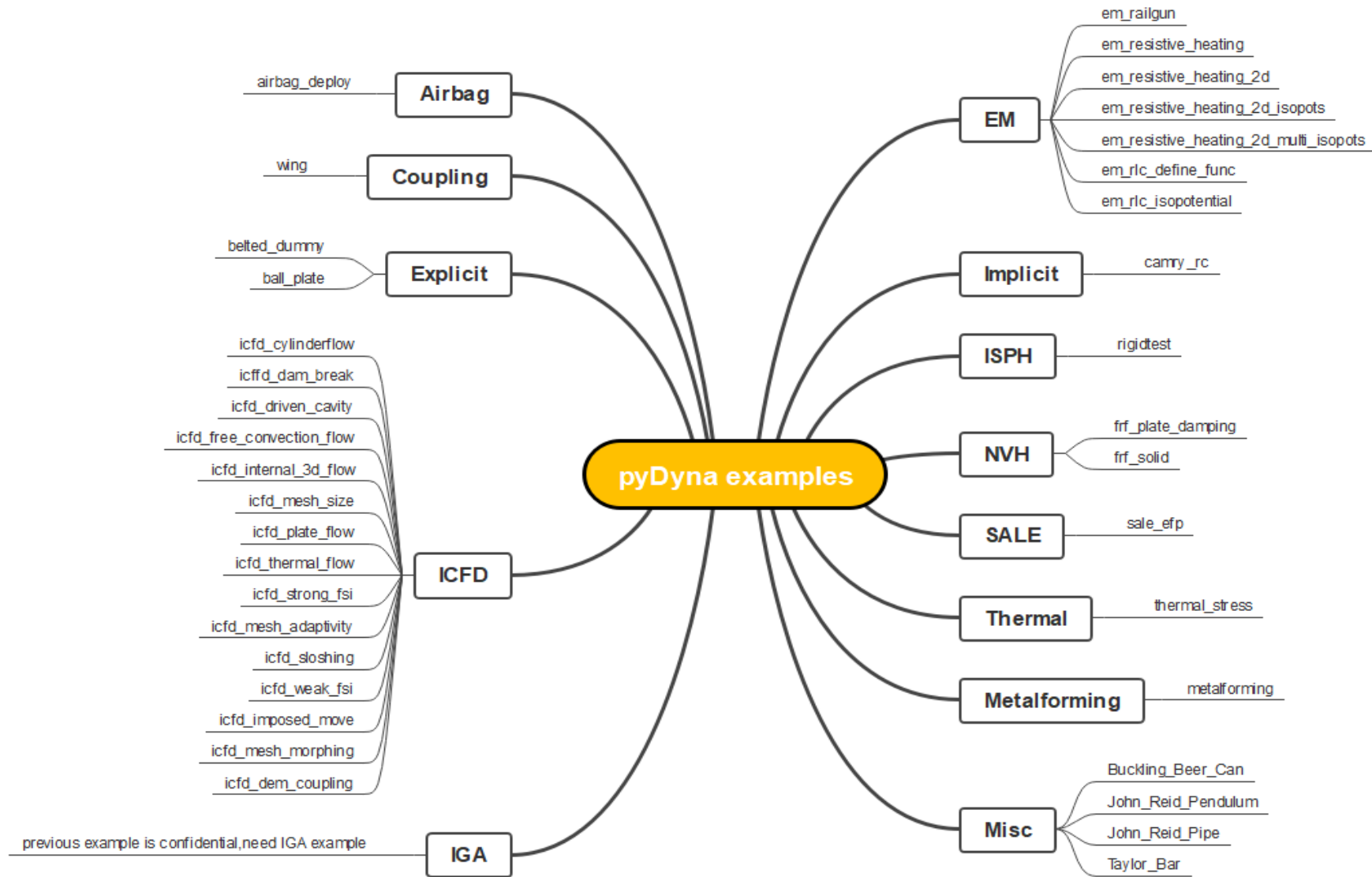
PyDYNA - Start Server in Docker

Setup pydyna-solver docker.

Ensure that Docker is installed(Linux: Docker Engine)

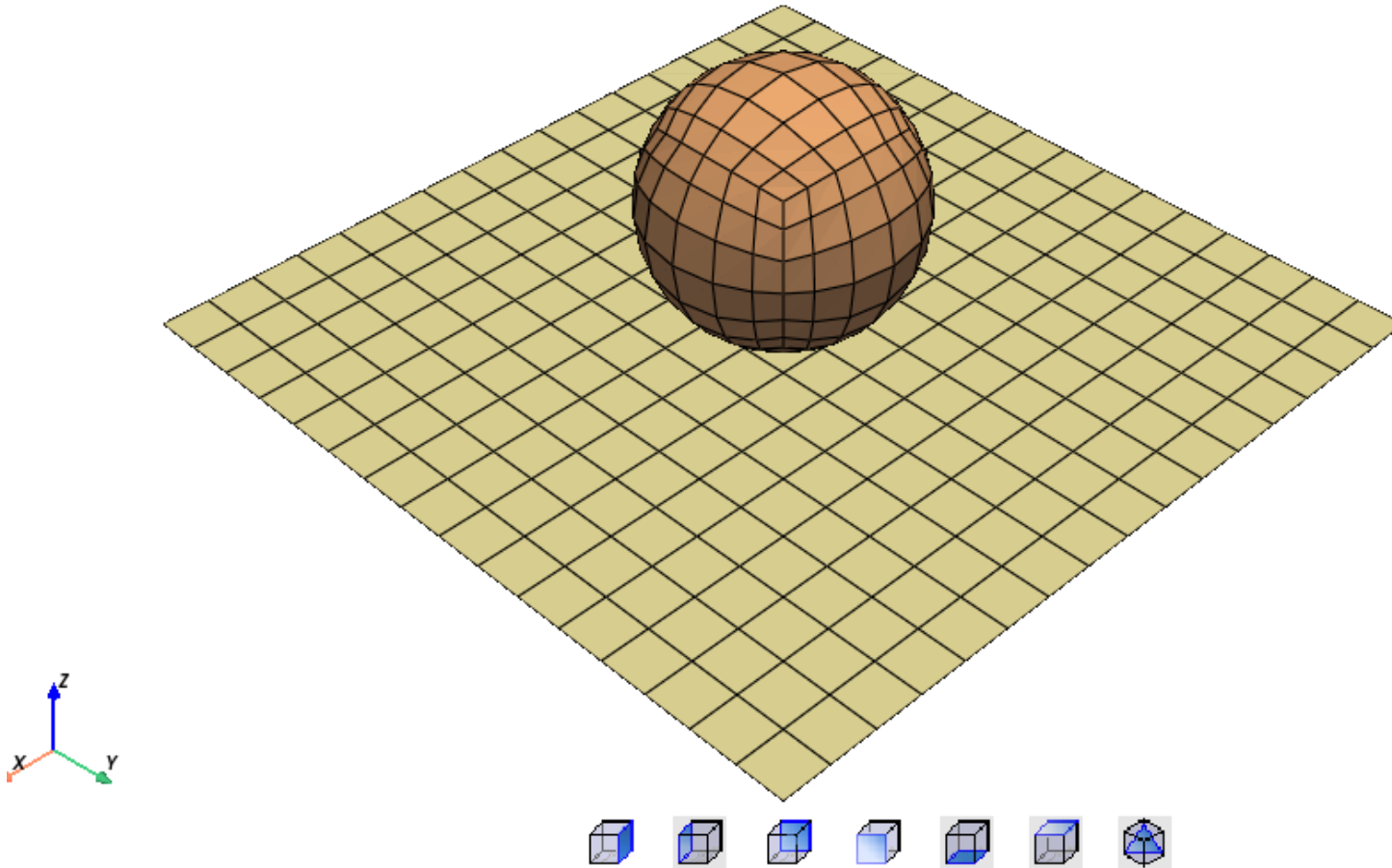
File name	Description
Dockerfile	Build docker image
mppdyna_docker_centos7.zip	Include libraries and .proto file, download from: https://github.com/ansys/pydyna/releases/download/v0.3.1/linux-binaries.zip
docker-compose.yml	“docker compose up -d”
do_build	./do_build
README.rst	How to start pydyna-solver container

PyDYNA – Pre Examples

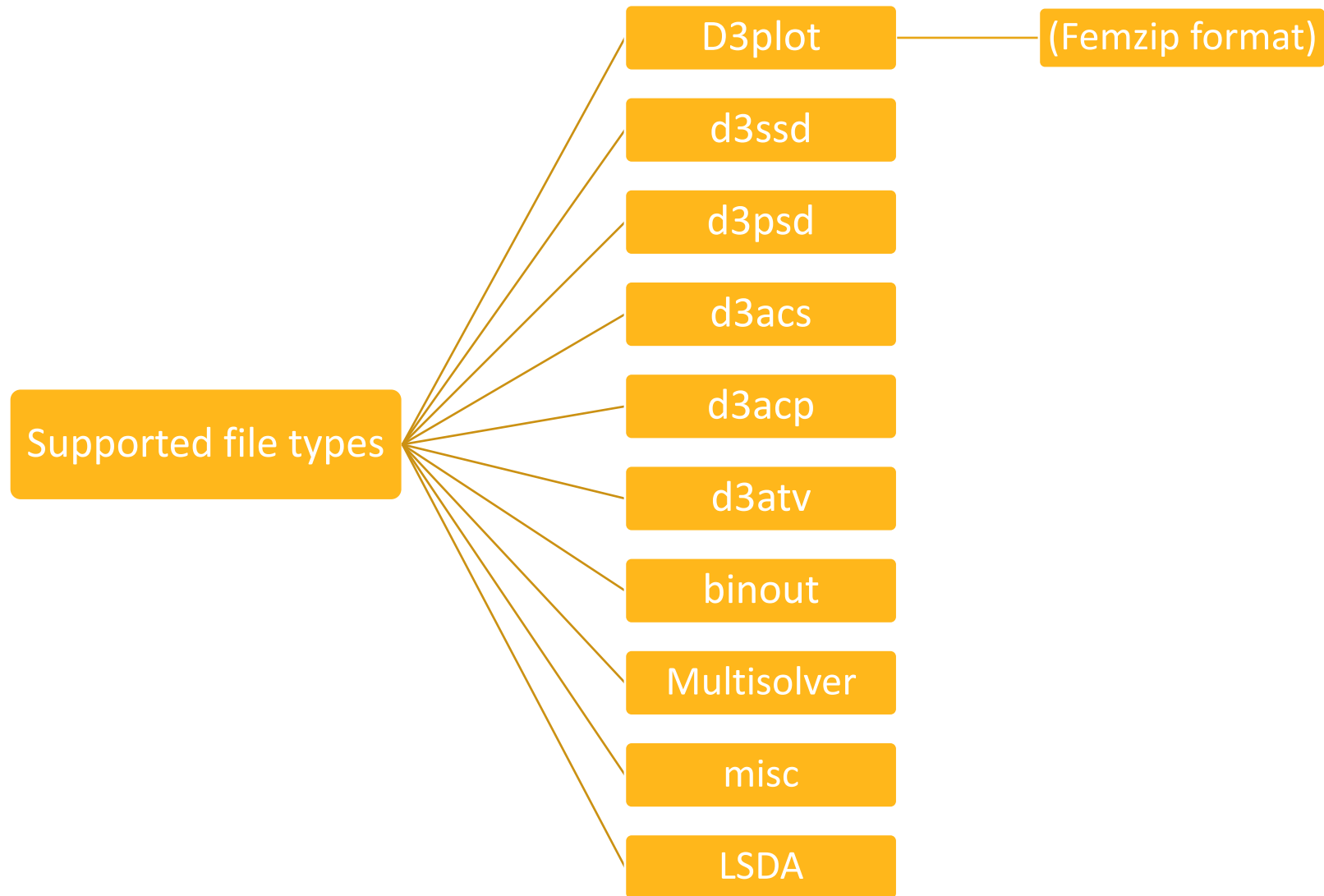


PyDYNA

Draw model: Support draw model with pyvista

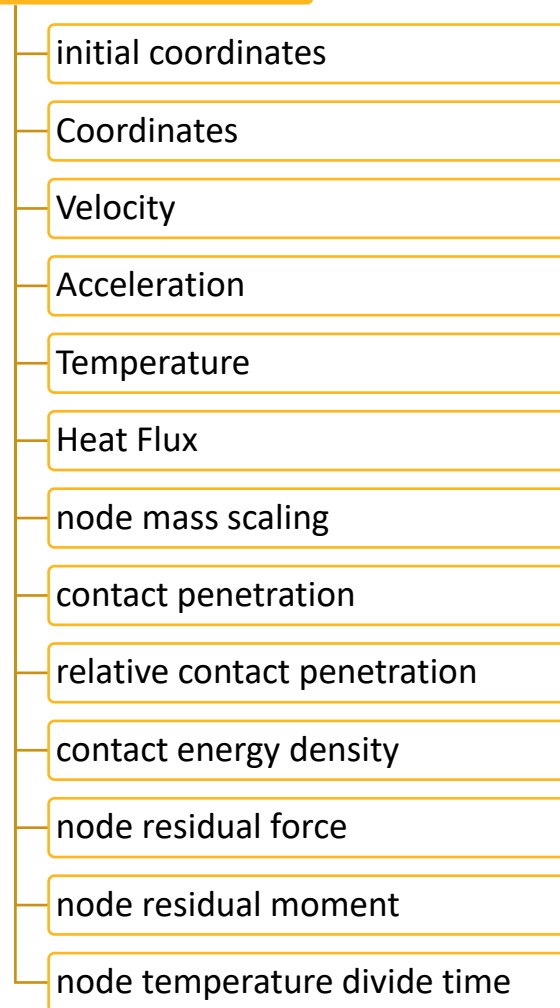


PyDPF LSDYNA plugin

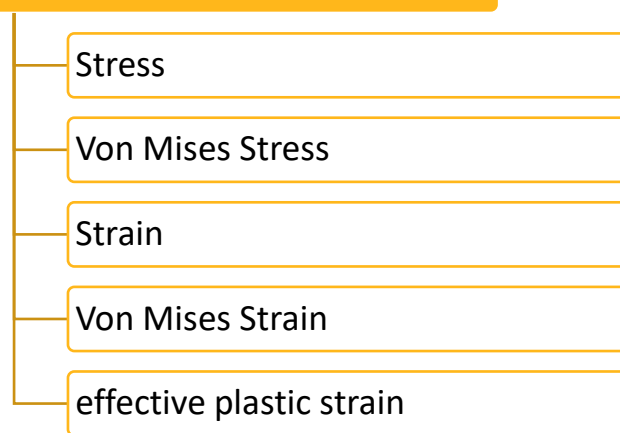


PyDPF LSDYNA plugin – Node, Element, global, Shell

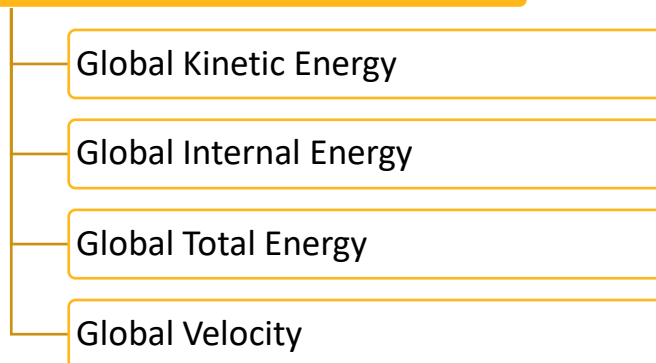
Node variable



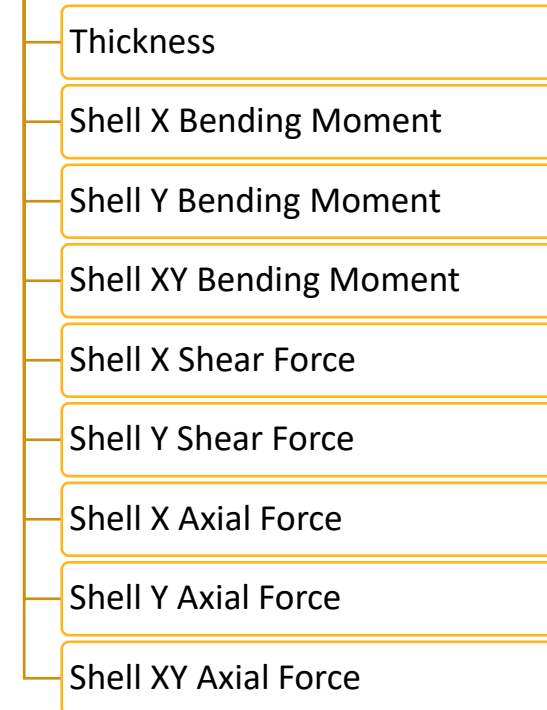
Element variable



Global variable



Shell variable



PyDPF LSDYNA plugin – Shell, Beam, Beam Ipt

Shell variable

- Thickness
- Shell X Bending Moment
- Shell Y Bending Moment
- Shell XY Bending Moment
- Shell X Shear Force
- Shell Y Shear Force
- Shell X Axial Force
- Shell Y Axial Force
- Shell XY Axial Force

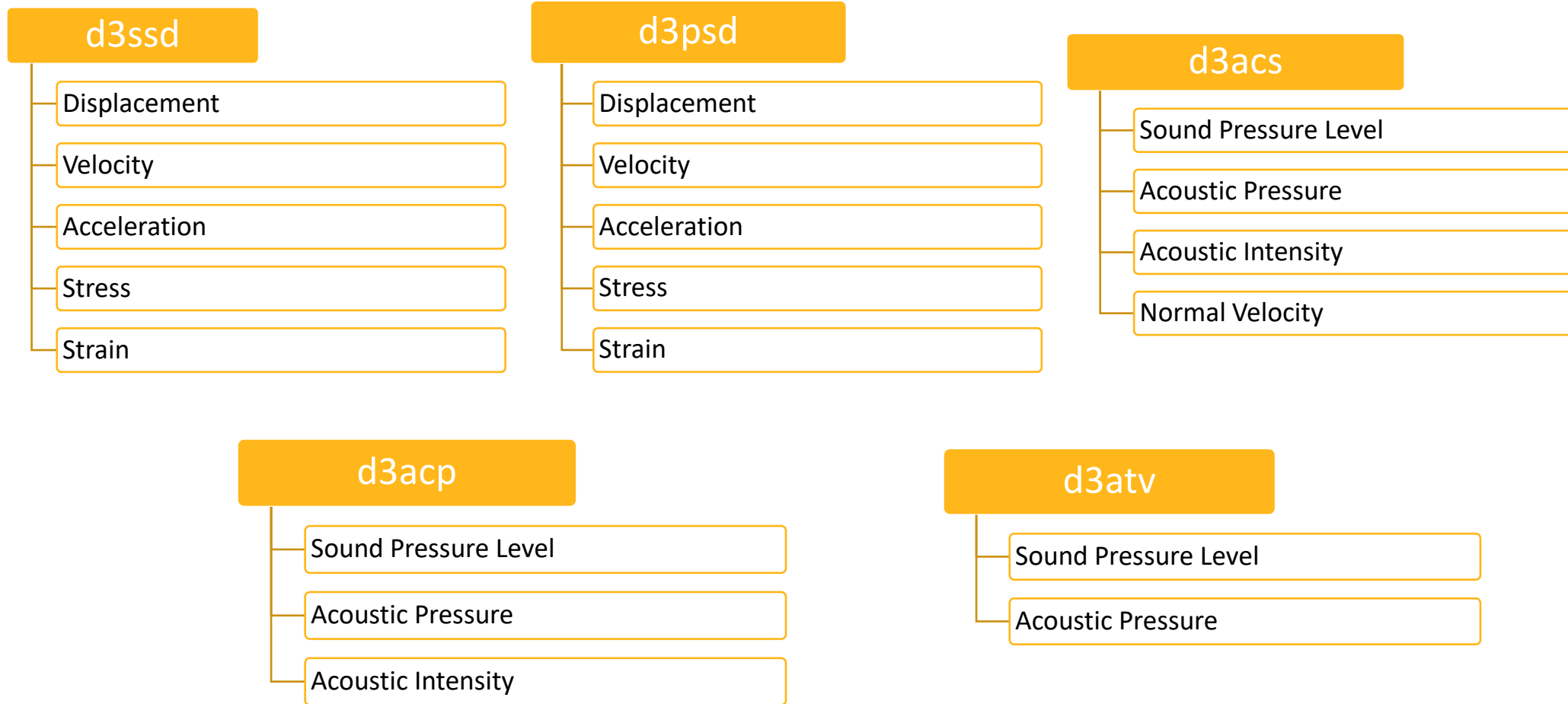
Beam variable

- Beam Axial Force
- Beam S Shear Force
- Beam T Shear Force
- Beam S Bending Moment
- Beam T Bending Moment
- Beam Torsional Moment

Beam Ipt variable

- Beam Axial Stress
- Beam RS Shear Stress
- Beam TR Shear Stress
- Beam Axial Plastic strain
- Beam Axial Total strain

PyDPF LSDYNA plugin – d3ssd, d3psd, d3acs, d3acp, d3atv



PyDPF LSDYNA plugin – Multi-Solver,Misc

Multisolver (EM,ICFD)

- Fluid velocity
- Scalar potential
- Cur density
- EM Cur
- Level Set
- Viscosity
- Fluid temperature

Misc

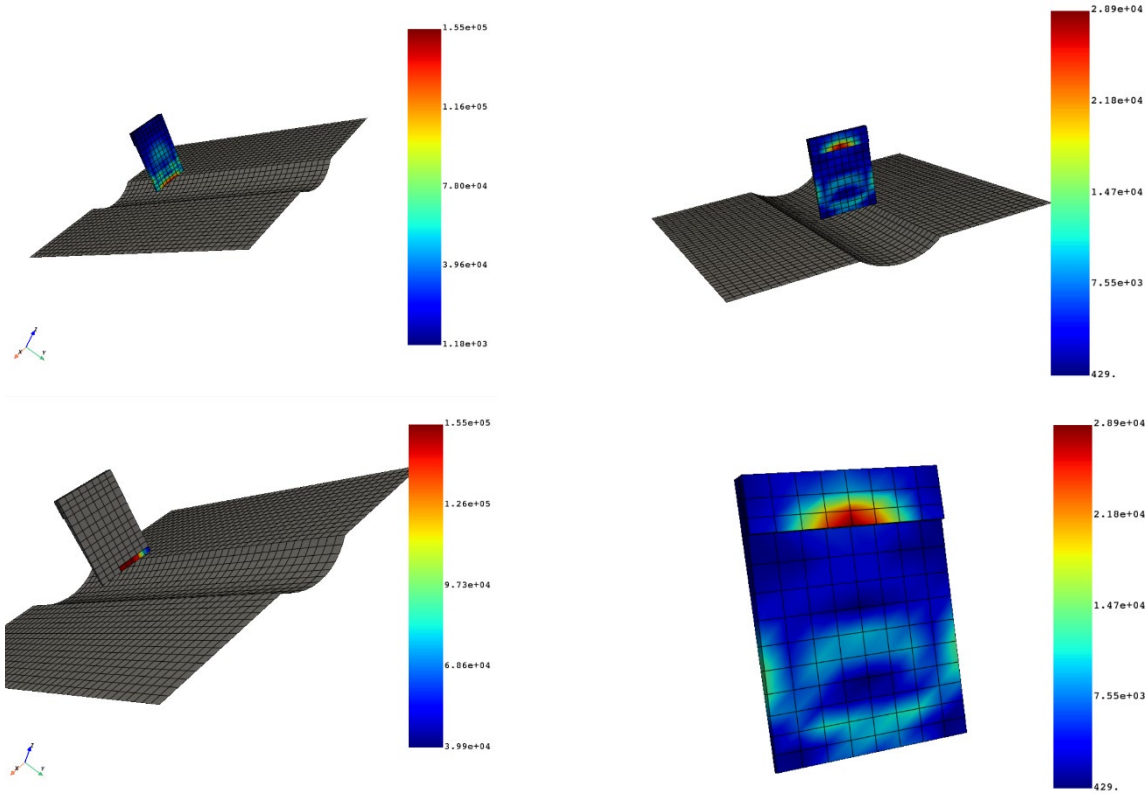
- History variable
- Erosion data
- Part IDS

PyDPF LSDYNA plugin - Binout branch

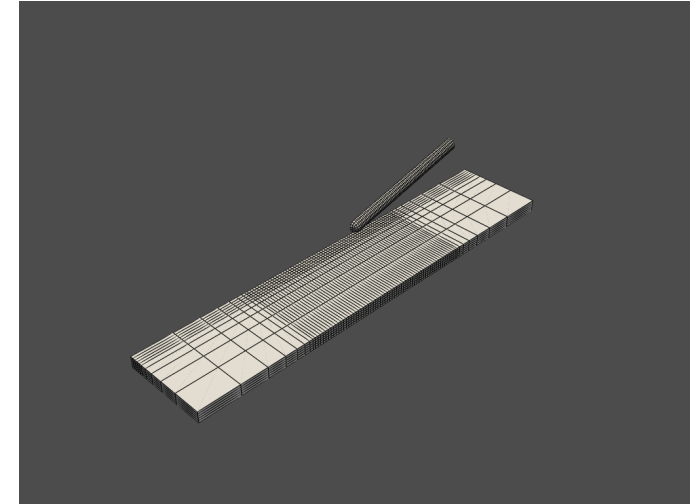
glstat	matsum	reforc	icvout
rwforc	secforc	swforc	nodout
abstat	sbtout	jntforc	spcforc
bndout	rbdout	sleout	Elout (under developing)

DPF (Data Processing Framework) LS-DYNA PlugIn

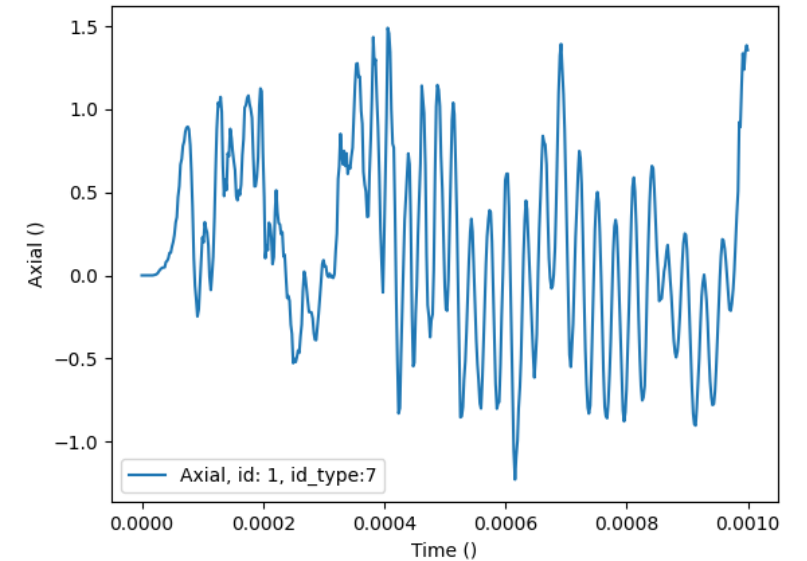
- Fringe Results (scoping, by part)



- Adaptive Model



- Binout



The background features a light gray hexagonal grid pattern. On the left side, there are two prominent diagonal stripes: a black one on the far left and a yellow one next to it, both running from the top-left towards the bottom-right.

Thanks!