

# PRIMER 18.0

# PRIMER 18.0 – Contents

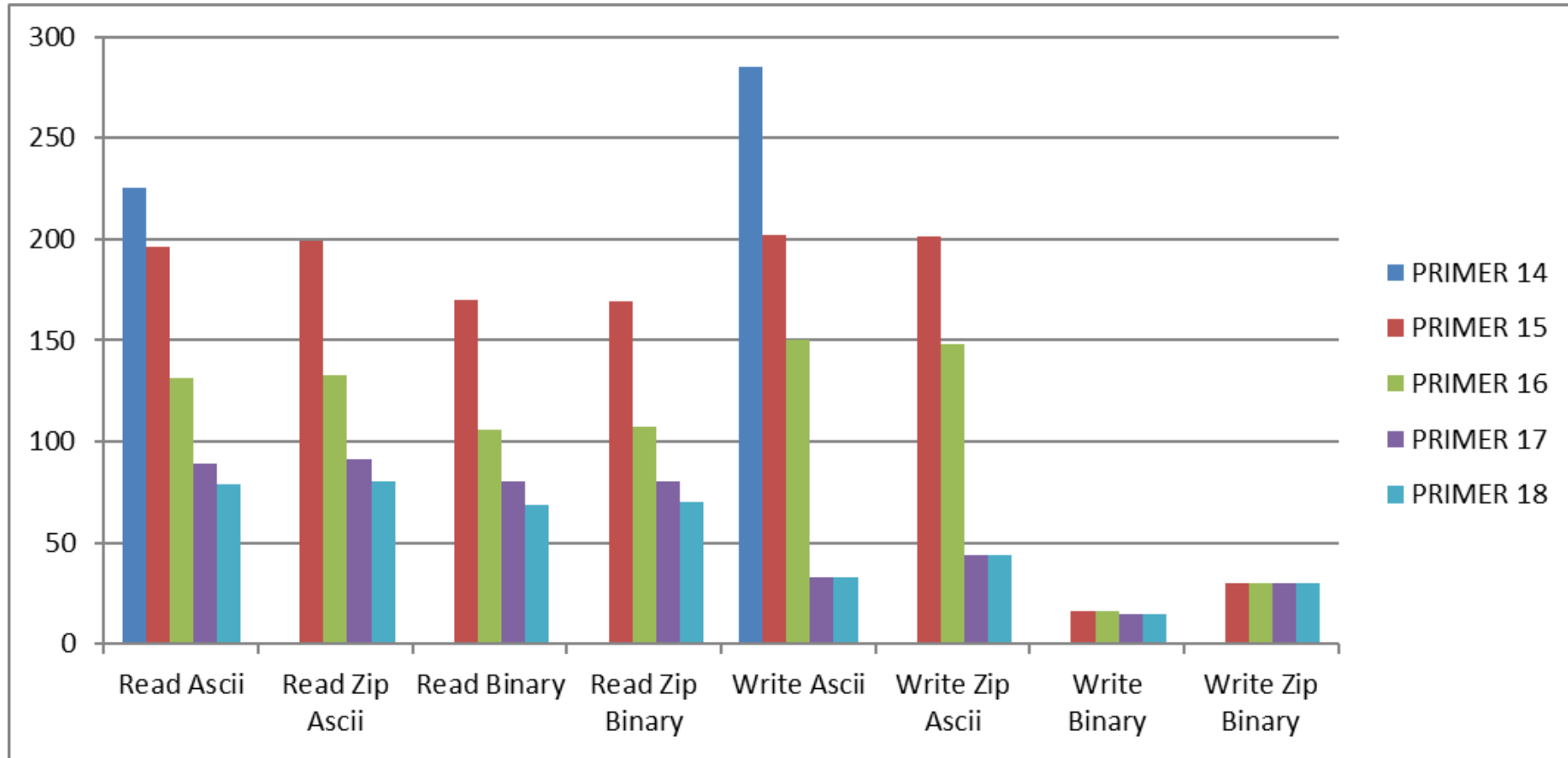
---

- [Performance Enhancements](#)
- [Changes to Input](#)
- [Rivet Connections](#)
- [Check Options](#)
- [Seatbelts](#)
- [Undocking Cut Sections Panel](#)
- [Cut Sections Improvements](#)
- [Soft = 2](#)
- [Isogeometric Analysis \(IGA\)](#)
- [Encryption Tool](#)
- [Partially Encrypted \\*AIRBAG Cards](#)
- [HIC Area Calculator](#)
- [Pedestrian Run Builder](#)
- [Implicit Analysis Setup Tool](#)
- [Running LS-DYNA from PRIMER](#)
- [Import Geometry from D3PLOT](#)
- [Checkpoint Files](#)
- [JavaScript Engine Upgrade](#)
- [JavaScript GUI Builder](#)
- [JavaScript API](#)
- [Miscellaneous](#)

# Performance Enhancements

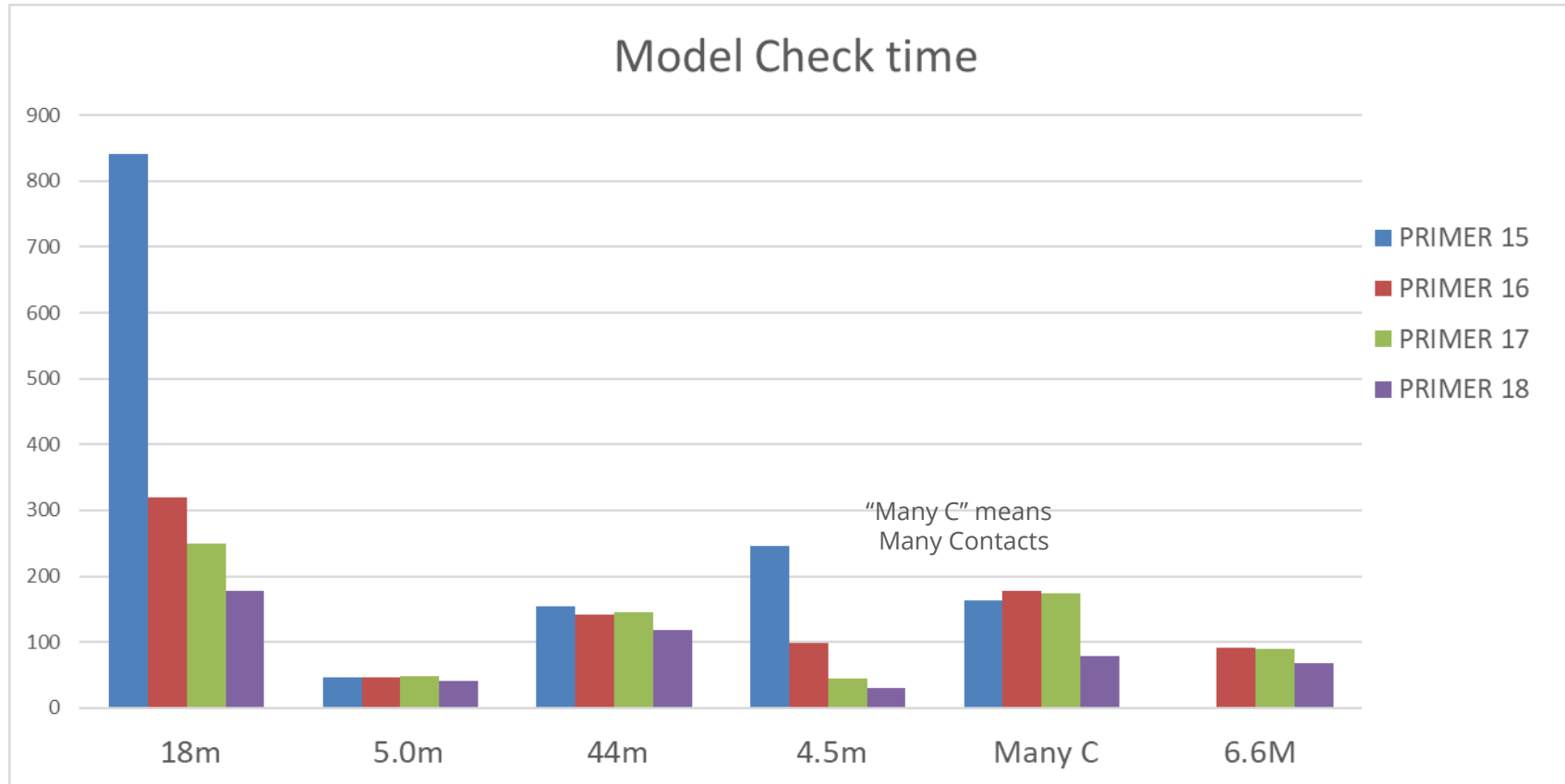
# Performance Enhancements: Time to read models

PRIMER 18 is between 5% to 10% faster than PRIMER 17 when reading models, particularly larger ones (these benchmark figures are for a 44m element model). Write speed is unchanged.



# Performance Enhancements: Time to run Model Check

Check time tends to be dominated by contacts and connections, so very model dependent. We continue to tackle bottlenecks and parallelise, the general trend is to become faster.



# Changes to Input

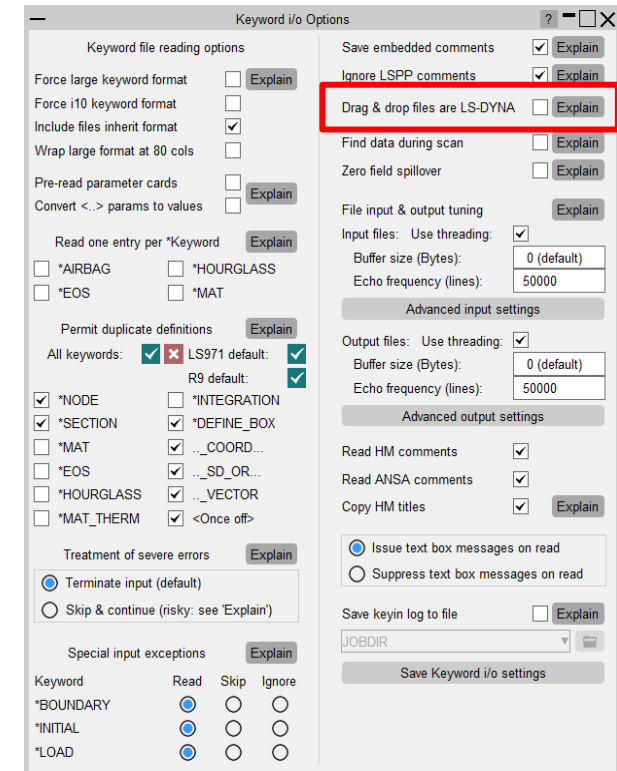
# LS-DYNA Keyword Support

---

- PRIMER 18.0 keywords:
  - LS-DYNA R12 Vol 1 and Vol 2 fully supported
  - LS-DYNA R12 Vol 3 mostly supported (only some \*DUALCESE keywords outstanding)
  - LS-DYNA R13 core \*IGA keywords supported

# Drag and Drop

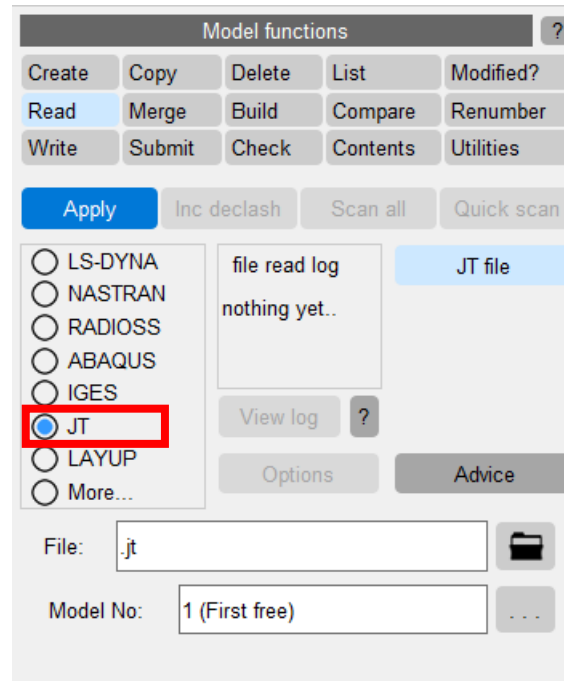
- Support for dragging and dropping of files has been extended beyond keyword files.
- You may now drag and drop other file types like Abaqus, Nastran, IGES, JT and Layup.
- This drag and drop feature only works for **Windows**.
- There is also a new option to drag and drop LS-DYNA files (\*KEYWORD format) that have a non-standard extension (\*.inp, \*.dat etc.).
- If the checkbox is ticked, all drag and drop files are assumed to be \*KEYWORD format. If unticked, files are read according to file extensions.





# JT Reader Update

- The Siemens JT file reader libraries have been updated from version 9.0 to 10.8.



# Rivet Connections

# Rivet Connections

---

- Self-Piercing Rivets (SPR) have been added as a new connection type in PRIMER.
- Rivets can be:
  - created in a similar way to other connection types;
  - modified via the connection table;
  - read from and written to XML files;
  - accessed via JavaScript.
- The following slides demonstrate these features.

# Creating a Rivet (methods similar to spot-weld)

Connections

Create Find con Read Check  
Table Find unc Write From FE  
Delete List Contact Lines  
Merge Compare Library Help

Next >> Create connections (M1)

Select elems/parts to connect

720 shells selected

All Visible Sketch

☐ Exclude rigid from selection

Connection type to create:

☒ Spotweld / Rivet  
☐ Bolt / Joint  
☐ Adhesive  
☐ Spotweld Lines

MAKE CONNECTIONS

Dismiss Apply Undo Last Help

Create spotwelds/Rivets

ATTRIBUTES

Connection element type: SPR2 Rivet  
C\_SPR2 NSID for Rivet : Elem type  
Rivet Diameter <D> : Beam  
Use Pref values: 4 Hexas  
Reverse Last 8 Hexas  
Explain 12 Hexas  
Creation method 16 Hexas  
☐ X, Y, Z coords  
☒ pick screen point  
☐ pick single node  
☐ all nodes in set  
☐ line of welds/rivets  
☐ pick connection  
☐ auto weld  
☐ pick geom point

Global Options/Settings:

Max thick: 10.0 Edge dist: 3.0  
Angle tol: 30.0 Options ...  
Optional title: Rivet\_title

- Creation methods:
  - Screen pick
  - Node pick
  - Nodes in set
  - Line of rivets
  - Pick geometry point

- An optional specified title will be stored on each connection made – this is a way of distinguishing between different SPR types in your model.

# Creating Rivet Connection

- 'Use Pref values' - SPR2 settings from pref *primer\*default\_settings\_for\_rivet\_creation: D=10, FN=999, FT=999, etc.*
- Pref supports CONSTRAINED\_SPR2 settings D, FN, FT, DN, DT, XLN, XLT, ALPHA1, ALPHA2, ALPHA3, EXPN, EXPT, INTP, DENS.
- Any newly created C\_SPR2 will import these values.
- Settings can be modified post-creation using keyword editor for selected \*CONSTRAINED\_SPR2.

MAKE CONNECTIONS

Dismiss Apply Undo Last Help

Create spotwelds/Rivets

ATTRIBUTES

Connection element type: SPR2 Rivet

C\_SPR2 NSID for Rivet : <none>

Rivet Diameter <D> : 10.0

Use Pref values: ☒

Reverse Last Explain

Match existing  
Create new SPR2  
Select SPR2

Creation method

☐ X, Y, Z coords  
☒ pick screen point  
☐ pick single node  
☐ all nodes in set  
☐ line of welds/rivets  
☐ pick connection  
☐ auto weld  
☐ pick geom point

Pick screen point

# Creating Rivet Connection

- XYZ position determines layers to be joined.
- *'Match existing'* will try to match rivet to existing C\_SPR2 considering layers – layers are unique in C\_SPR2 cards.
- With *'Use Pref values'* set, all non-zero pref values must be matched in C\_SPR2.
- If no match, reverts to *'Create new SPR2'* with pref values (if set).
- *'Create new SPR2'* always makes new C\_SPR2 with pref values for each new connection.
- *'Select SPR2'* only makes rivet if layers match between connection being created and selected C\_SPR2.

MAKE CONNECTIONS ? - □ X

Dismiss Apply Undo Last Help

Create spotwelds/Rivets

ATTRIBUTES

Connection element type: SPR2 Rivet ▼

C\_SPR2 NSID for Rivet : <none> ▼

Rivet Diameter <D> : 10.0

Use Pref values: ☒

Reverse Last Explain

Match existing  
Create new SPR2  
Select SPR2

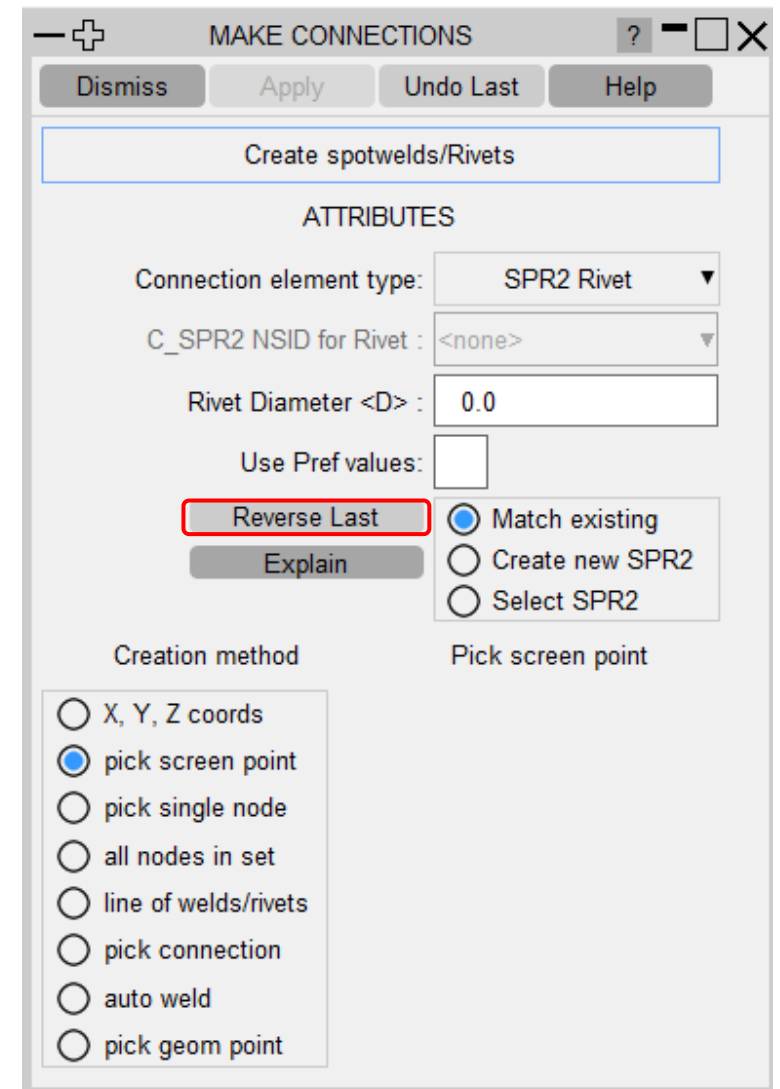
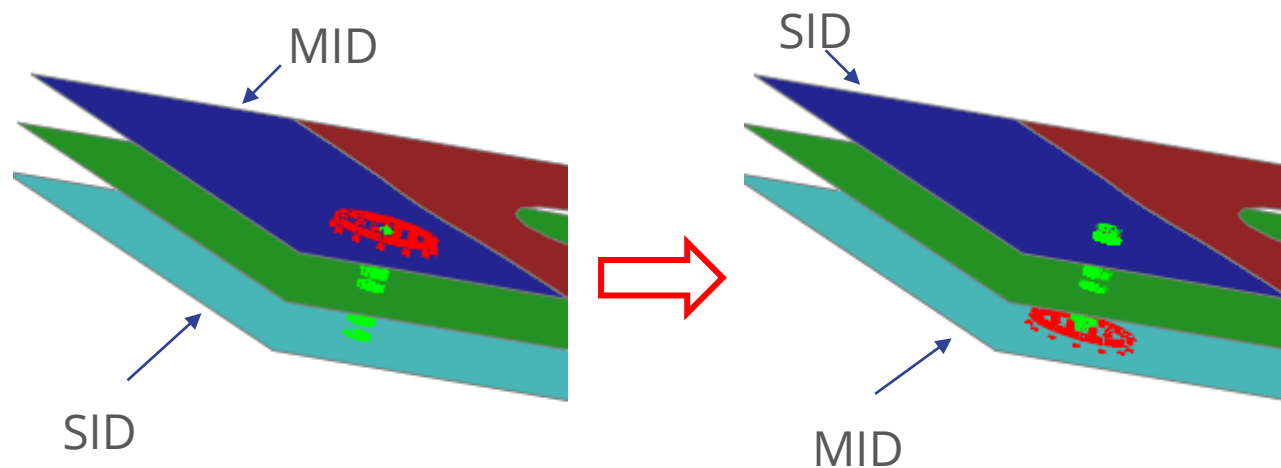
Creation method

Pick screen point

☐ X, Y, Z coords  
☒ pick screen point  
☐ pick single node  
☐ all nodes in set  
☐ line of welds/rivets  
☐ pick connection  
☐ auto weld  
☐ pick geom point

# Creating Rivet Connection

- Rivet orientation is important.
- C\_SPR2 orients rivet MID-XPID1..XPID4-SID.
- 'Reverse Last' can be used to reverse last created rivet.



# Modifying Rivet Connections

CONNECTION TABLE

Dismiss

View...

Options...

Refresh

Action: update & remake

Show all

Spotwelds

Adhesive

Rivet

Write...

?

Apply:

Undo

All

Selected

Changed

Autoscale

Clear

Sel all

Select

Show sel

Bolts/Joints

Spot lines

Set column

ID	Type	Subtype	Layer 1	Layer 2	Layer 3	SPR2 SNID	NID	SPR2 THK	SPR2 D	SPR2 FN	SPR2 FT	SPR2 DN	SPR2 DT	SPR2 DENS
CNX1	RIVET	SPR2	P3	P2	P4	1	847	6	10	999	999	0	0	0
CNX2	RIVET	SPR2	P3	P2	P4	1	848	6	10	999	999	0	0	0
CNX3	RIVET	SPR2	P1	P2	P4	2	849	7	0	0	0	0		
CNX4	RIVET	SPR2	P4	P2	P1	3	850	7	10	999	999	0		

Change SPR2 SNID

Update & remake

Sketch conx

Sketch FF

- The Connections Table displays data on \*CONSTRAINED\_SPR2 card.
- Edit via keyword editor permitted for unique selection – modifies all rivets that use this SPR2.
- Change applies to all rivets of SPR2 node set (SNID) - does not require update & remake.
- Do not change MID, XPIDn, SID on C\_SPR2 – this must be done via the connection table to retain data link between connection and C\_SPR2 card (see next slide).

- Change SPR2 SNID
- Update & remake
- Sketch conx
- Sketch FE
- Show conx & layer panels
- Show conx & attached panels
- Empty conx
- Delete conx
- Upd & remake (reposition)
- Upd & remake (swap layers)
- Upd & remake P1 <=> P2 (bolts)
- Edit SPR2 properties



# Modifying Rivet Connection on Table

CONNECTION TABLE												
Dismiss View... Options... Refresh			Action: update & remake		Show all		Spotwelds	Adhesive	Rivet	Write... ?		
Apply: Undo		All	Selected	Changed	Autoscale	Clear	Sel all	Select	Show sel	Bolts/Joints	Spot lines	Set column
ID	Type	Subtype	Layer 1	Layer 2	Layer	SPR2 SNID	NID	SPR2 THK	SPR2 D	SPR2 FN	SPR2 FT	P1 (coords)
CNX1	RIVET	SPR2	P3	P2		1	851	6	10	999	999	84.46 47.40
CNX2	RIVET	SPR2	P3	P2		1	852	6	10	999	999	82.20 12.52
CNX3	RIVET	SPR2	P1	P2	P4	2	849	7	0	0	0	11.56 21.51
CNX4	RIVET	SPR2	P4	P2		3	850	7	10	999	999	26.14 48.26

- Connection layers can be changed for unique selection. Requires *'Update & remake'*.
- Reverse by *'Update & remake (swap layers)'* for unique selection.
- Coordinate *P1* can be changed. *'Update & remake'* allowed unconditionally.

Cannot change ID
Update & remake
Sketch conx
Sketch FE
Show conx & layer panels
Show conx & attached panels
Empty conx
Delete conx
Upd & remake (reposition)
Upd & remake (swap layers)
Upd & remake P1 <=> P2 (bolts)

# Modifying Rivet Connection on Table

CONNECTION TABLE									
Dismiss	View...	Options...	Refresh	Action: update & remake		Show all	Spotwelds	Adhesive	Rivet
Apply: Undo	All	Selected	Changed	Autoscale	Clear	Sel all	Select	Show sel	Spot lines
									Write...
									Set column
ID	Type	Subtype	Layer 1	Layer 2	SPR2 SNID	NID	SPR2 THK	SPR2 D	SPR2 FN
CNX1	RIVET	SPR2	P1	P2	1	563	4	5.5	1.2
CNX2	RIVET	SPR2	P1	P2	1	564	4	5.5	1.2
CNX3	RIVET	SPR2	P1	P2	1	565	4	5.5	1.2
CNX4	RIVET	SPR2	P1	P2	1	566	4	5.5	1.2
CNX5	RIVET	SPR2	P1	P2	1	567	4	5.5	1.2
CNX6	RIVET	SPR2	P1	P2	1	568	4	5.5	1.2

Cannot change ID

Update & remake

Sketch conx

Sketch FE

Show conx & layer panels

Show conx & attached panels

Empty conx

Delete conx

Upd & remake (reposition)

Upd & remake (swap layers)

Upd & remake P1 <=> P2 (bolts)

Split out Rivet C\_SPR2s



2	565	4	5.5	1.2
2	566	4	5.5	1.2
2	567	4	5.5	1.2

- If multiple rivets share same C\_SPR2, properties cannot be changed for subset of rivets.
- Property modification requires selection of all rivets using this C\_SPR2.
- Split out selected Rivet C\_SPR2s will copy existing C\_SPR2 and make a new node set enabling modification of properties of selected rivets (single or multiple).

# Writing Rivet Connection to XML

Connections ? X

Create Find con Read Check  
Table Find unc Write From FE  
Delete List Contact Lines  
Merge Compare Library Help

Apply Write spotwelds

File:

Spotwelds Spot lines Rivets  
Bolts/Joints Adhesive

☒ all connections  
☐ by connection id  
☐ by layer panels  
☐ by attached panels  
☐ by spotweld part  
☐ by spotweld beam  
☐ by spotweld solid  
☐ by adhesive part  
☐ by multiple seams  
☐ by single seam  
☐ by connection title

☐ spotweld file  
☒ xml connection file  
☐ UG format  
☐ IGES format  
☐ User script  
☐ master connection file

- Xml Rivet Option -  
☐ Write SNID  
☒ Write D, FN, FT, etc

Default is to write SPR2 settings <d><fn><ft>,etc with every rivet - SPR2 cards will be constructed on read.

```
<connection type="rivet">  
  <method>spr2</method>  
  <title></title>  
  <id>1</id>  
  <coord x="14.734613" y="48.934669" z="0.000000" />  
  <rdiam>5</rdiam>  
  <fn>1.23</fn>  
  <ft>0.34</ft>  
  <dn>0</dn>  
  <dt>0</dt>  
  <xln>0</xln>  
  <xlt>0</xlt>  
  <alpha1>0</alpha1>  
  <alpha2>0</alpha2>  
  <alpha3>0</alpha3>  
  <density>7.7e-06</density>  
  <intp>1</intp>  
  <expn>0</expn>  
  <expt>0</expt>  
  <layer type="PART_ID">  
    <part id="1" />  
  </layer>  
  <layer type="PART_ID">  
    <part id="2" />  
  </layer>  
</connection>
```

Alternate is to write only SPR2 node set id SNID – each rivet will connect to extant SPR2 cards.

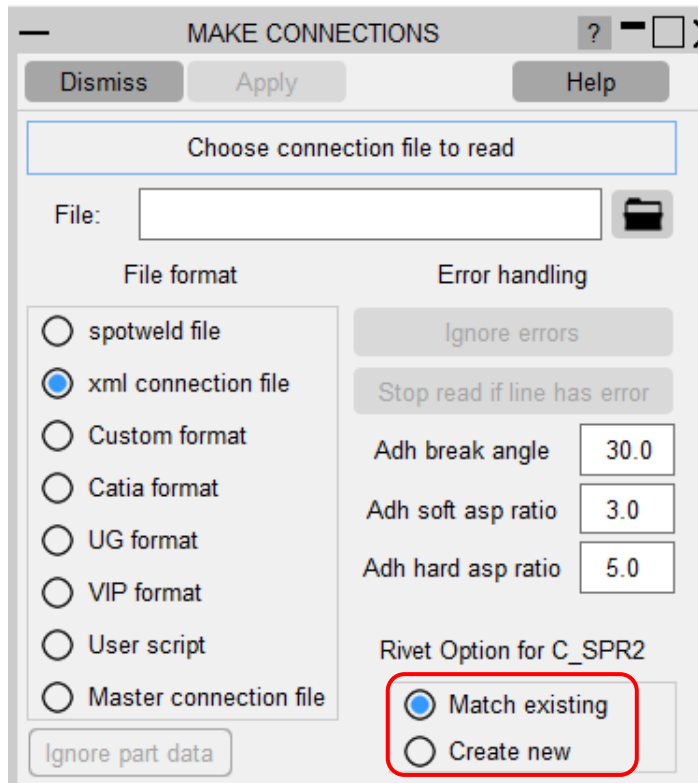
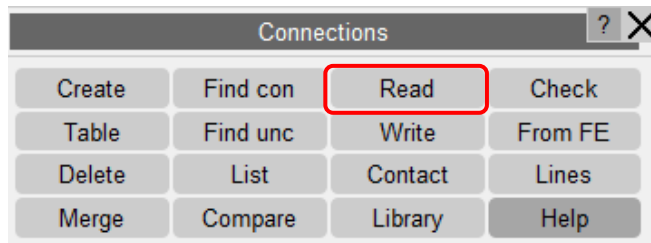
- Xml Rivet Option -

☒ Write SNID

☐ Write D, FN, FT, etc

```
<connection type="rivet">  
  <method>spr2</method>  
  <title></title>  
  <id>1</id>  
  <coord x="14.723724" y="50.547028" z="0.000000" />  
  <spr2nsid>1</spr2nsid>  
  <layer type="PART_ID">  
    <part id="1" />  
  </layer>  
  <layer type="PART_ID">  
    <part id="2" />  
  </layer>  
  <layer type="PART_ID">  
    <part id="4" />  
  </layer>  
</connection>
```

# Reading Rivet Connection from XML



- *'Match existing'* (default) will create minimum number of C\_SPR2 cards.
- *'Create new'* will generate new C\_SPR2 for each rivet.
- If pref *default\_settings\_for\_rivet\_creation* set, any undefined setting will be set from them.

# Creating Rivet Connection in JavaScript

---

- It is possible to create and modify rivets via JavaScript.
- The following slide shows an example.

# Creating Rivet Connection in JavaScript

---

```
var c1 = new Conx(m, 10, 48, 0); // create rivet connection c1

c1.type = Conx.RIVET;

var c2 = new Conx(m, 30, 48, 0);

c2.type = Conx.RIVET;

c2.spr2_match = true; // c2 will match C_SPR2 card for c1 (assuming it joins same layers)

var c3 = new Conx(m, 50, 48, 0);

c3.type = Conx.RIVET;

c3.spr2_match = false; // c3 will have new C_SPR2

Conx.UseSPR2Pref(true); // use pref settings for new SPR2

Conx.RealizeAll(m);

if(c3.spr2_unshared) // option to check C_SPR2 it is not shared
{
    var spr2 = Spr2.GetFromID(m, c3.spr2_id); // change C_SPR2 property
    spr2.fn = 1.2;
}
```

# Check Options

# Check Options: Element quality

- CHECK OPTIONS panel
  - The “Max solid spotweld/adhesive warpage” criterion is used in the “Adhesive” and “Spotweld” categories.
  - This has now been included in the “Quality” category too.

CHECK OPTIONS

Dismiss Help

Select checking category

Category: Quality

☐ Element Quality Checks (ALL) Reset Save element quality settings to oa\_pref

Element Quality Criteria	Shells	Solids	TShells
<input type="checkbox"/> Min length	-5.0	-5.0	-5.0
<input type="checkbox"/> Max aspect ratio	5.0	5.0	5.0
<input type="checkbox"/> Max warpage(*)	20.0	20.0	20.0
<input checked="" type="checkbox"/> Max adhesive warpage(*)		20.0	
<input type="checkbox"/> Max skew	60.0	60.0	60.0
<input type="checkbox"/> Min Tria angle	30.0	30.0	30.0
<input type="checkbox"/> Max Tria angle	120.0	120.0	120.0
<input type="checkbox"/> Min Quad angle	40.0	40.0	40.0
<input type="checkbox"/> Max Quad angle	140.0	140.0	140.0
<input type="checkbox"/> Min Jacobian	0.7	0.7	0.7
<input type="checkbox"/> Max Taper	0.5		
<input type="checkbox"/> Min Tetra Collapse		0.1	

Jacobian calculation method: GAUSS POINTS

☐ Min timestep  
Element : 1.0E-6 Spotweld: 1.0E-6

☐ Max added mass  
Element : <off> Spotweld: <off>

☒ Ignore rigid elements

☒ Ignore null elements

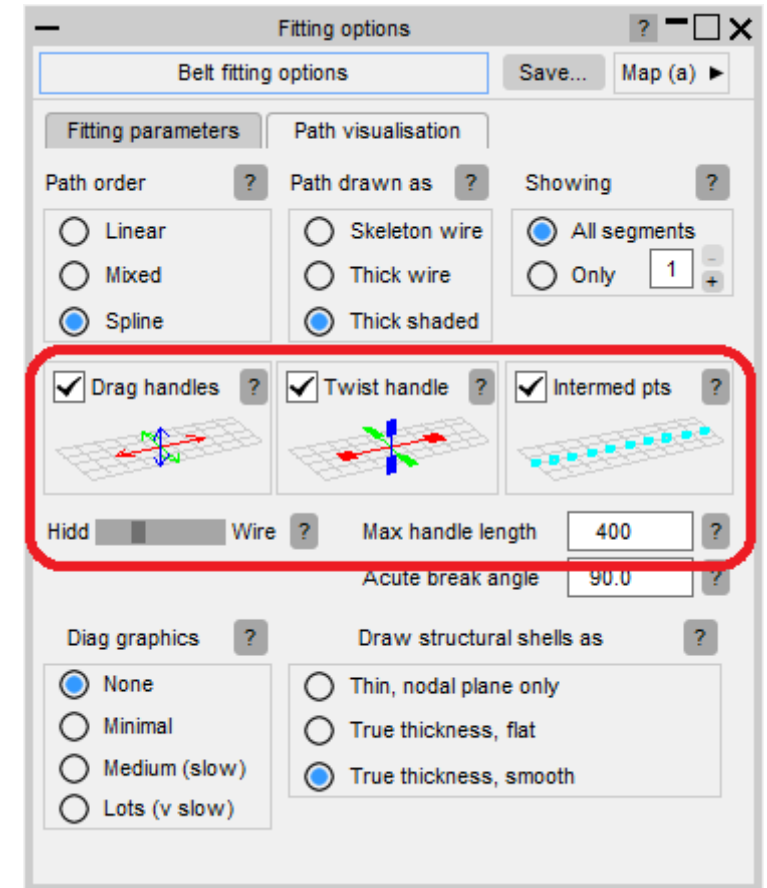
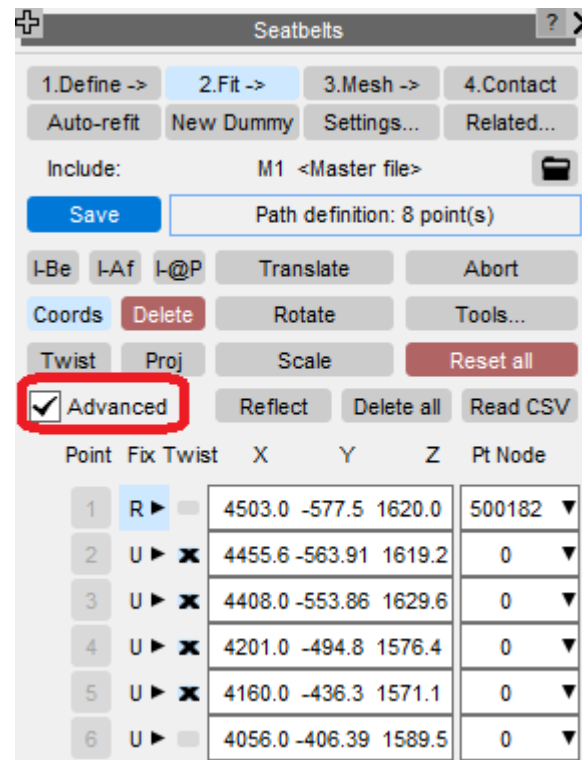


# Seatbelts

# New “Advanced” belt path editor mode: basic controls

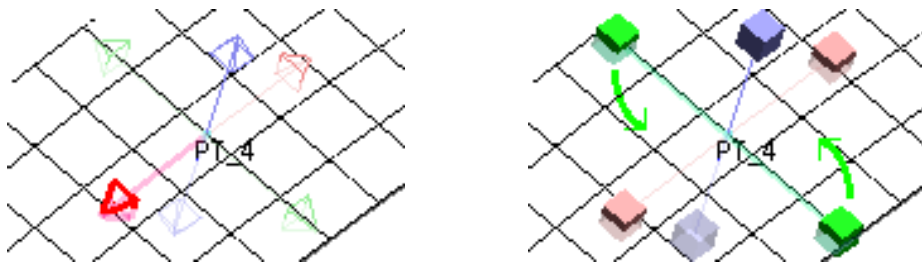
The “Advanced” path editor makes difficult geometries far easier to fit.

- Original editor still fully functional, now called “basic”.
- New “advanced” editor is switchable in the path point editor panel. You can swap modes back and forth at will.
- Its attributes can be controlled in the new “Path Visualisation” tab in the floating Fitting options panel.

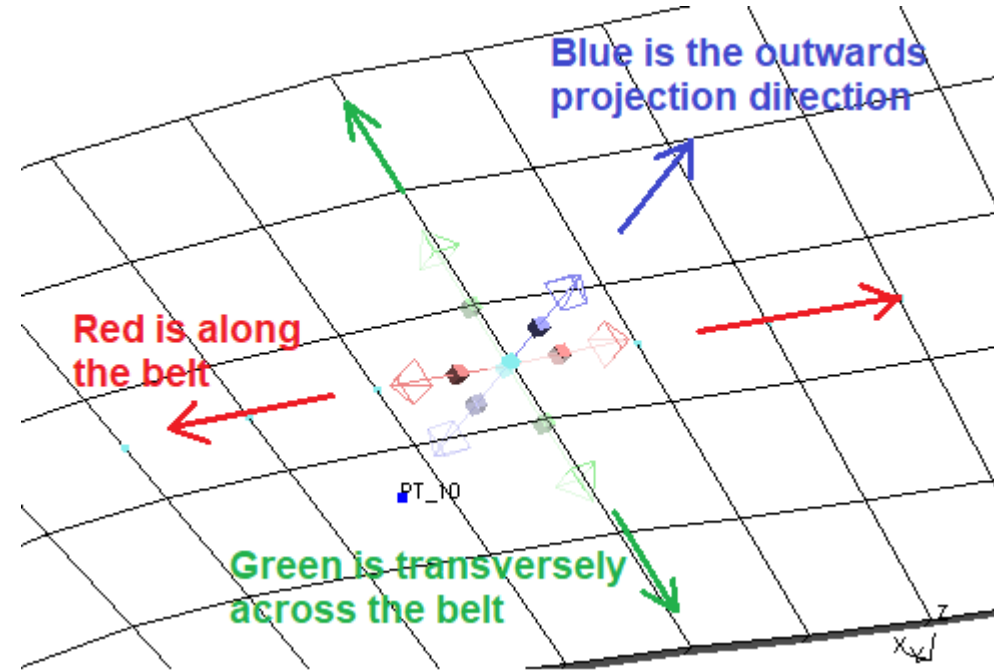


# Advanced Seatbelt fitting editor: Mouse controls

- Each basic path point now has a triad which allows the mouse to drag and rotate the point.
- Arrows drag (move) the point in local directions, Cube handles twist it, for example:

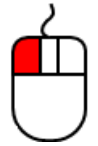
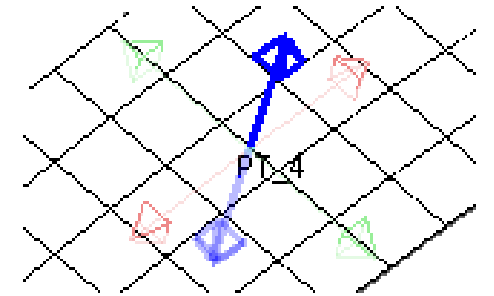
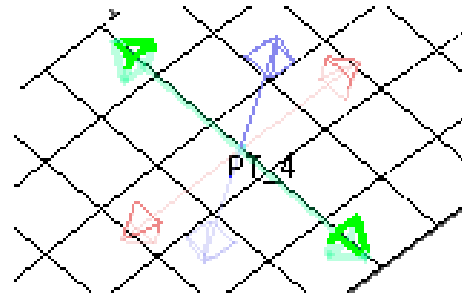
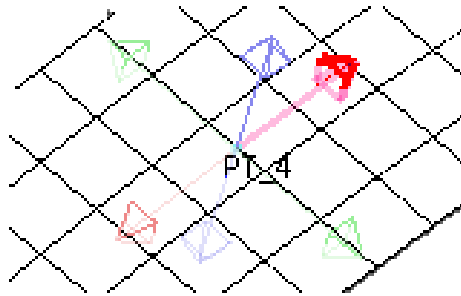


Red: Drag along belt    Green: twist transversely



Local axis system for drag & twist

# Advanced Seatbelt fitting editor: Dragging points

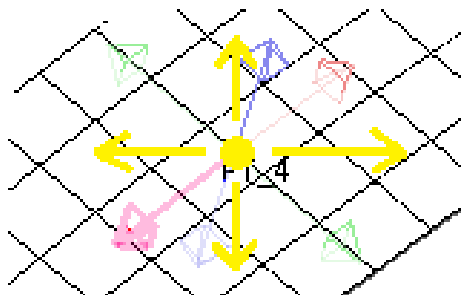


**Red:** drag forwards and backwards

**Green:** drag sideways

**Blue:** drag up/down

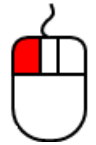
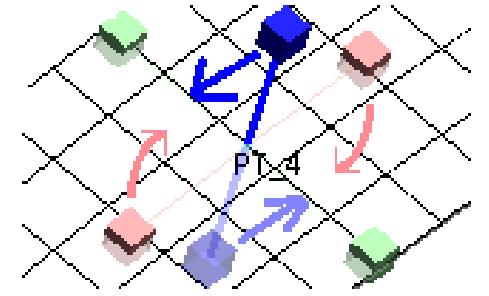
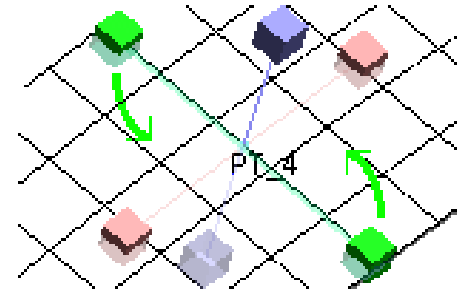
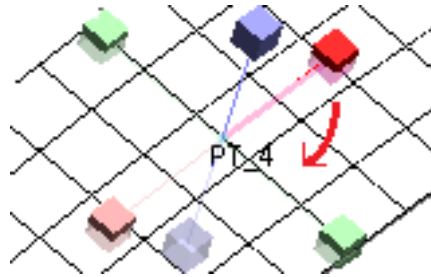
These are all in “path local” directions, as shown by the arrow vectors.



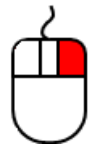
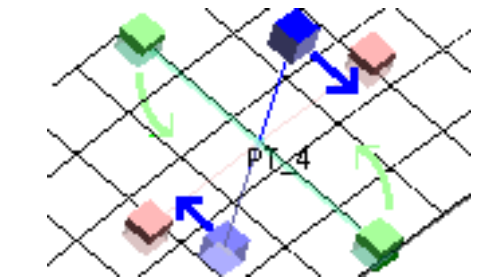
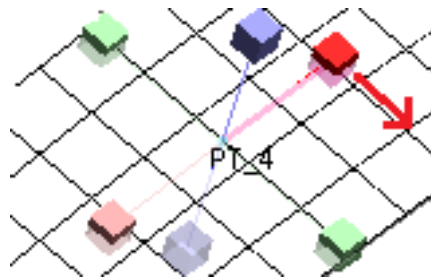
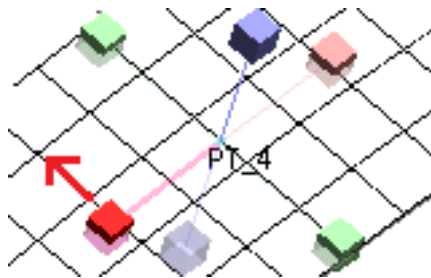
Points may also be “free” dragged in the current screen space using their central circle symbol as a handle.

In all cases dragging a point separates it from any underlying node used to define it.

# Advanced Seatbelt fitting editor: Twisting and skewing points

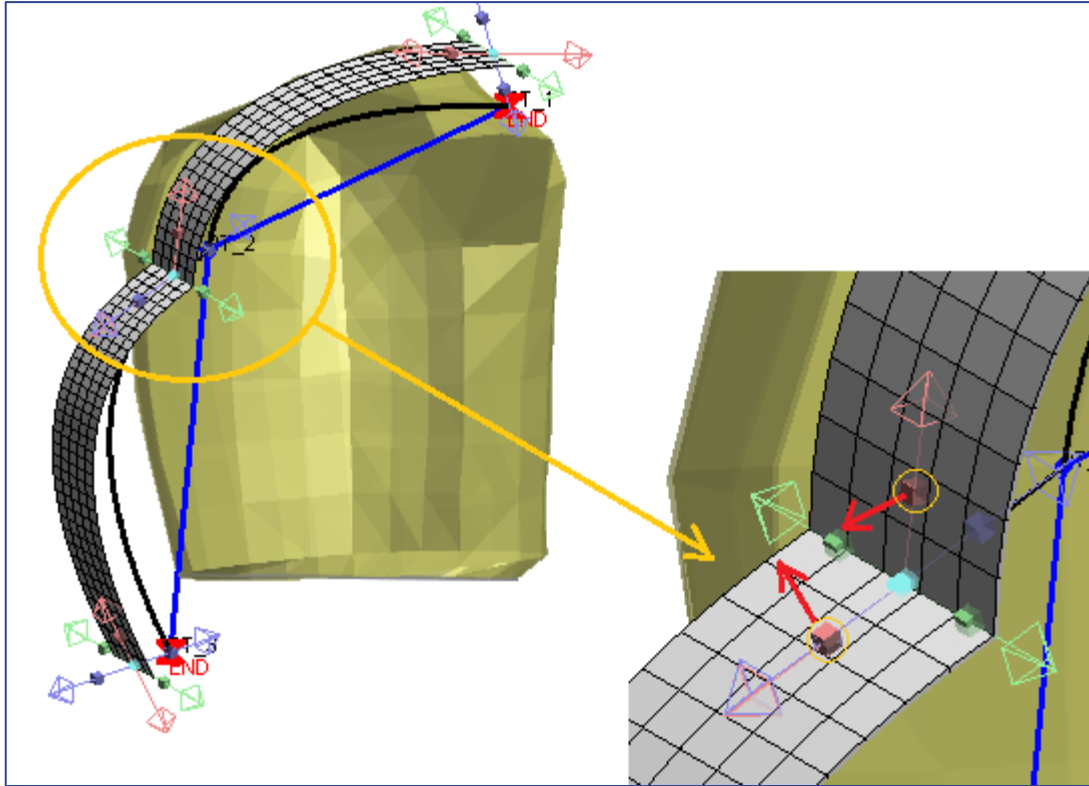


Left mouse actions rotate and twist the path

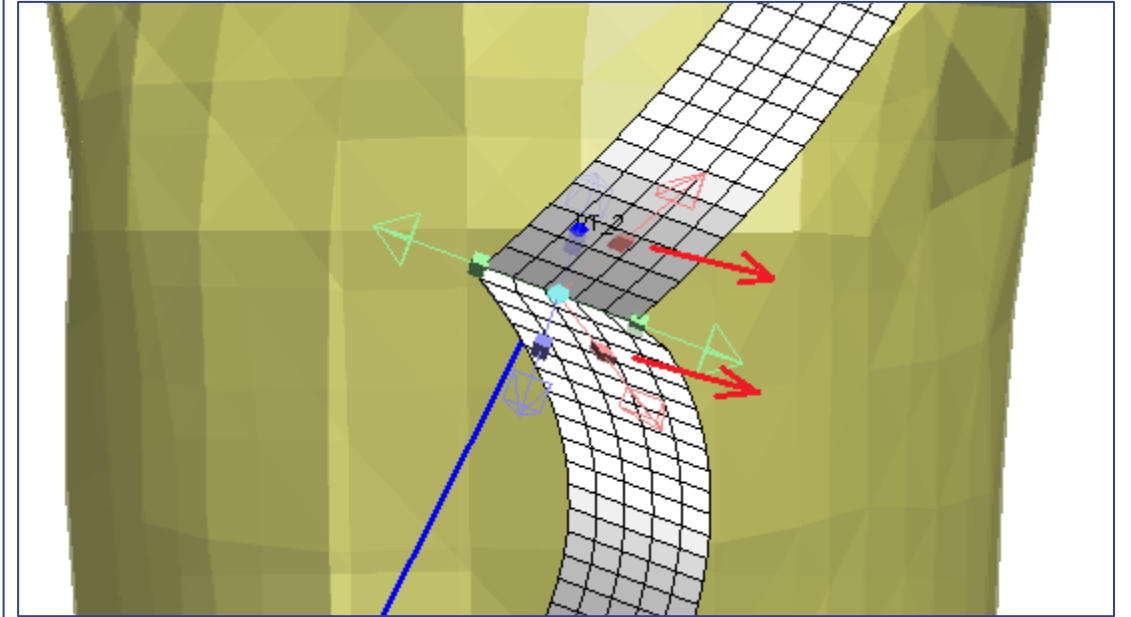


Right mouse actions skew the path.

# Advanced Seatbelt fitting editor: Example path movements



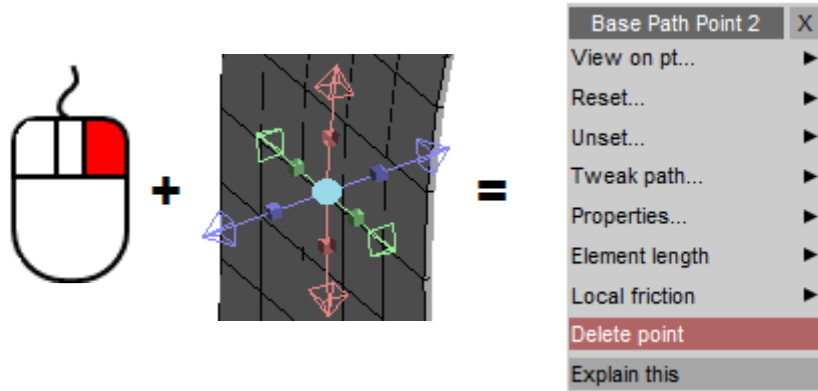
The spline path curvature can be “broken” at a point to give a sharp change of direction.



“Skewing” the path in plane is also possible, making it easy to thread the belt through difficult geometries.

# Advanced seatbelt fitting editor: Right click options at point

Right click at base path point gives a menu of options.



**View on point**

**Reset**

**Unset**

**Tweak path**

**Properties**

**Element length**

**Local friction**

**Delete Point**

Centres current view on point.

Reverts to pre-edited state.

Undoes rotation and twist at point.

Gives fine adjustments.

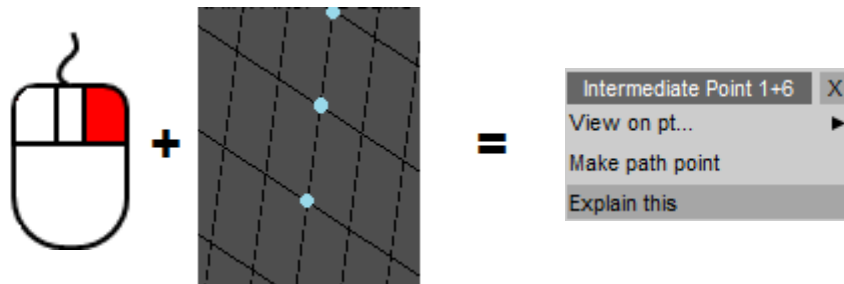
Sets attributes at point (slipring, etc).

Sets local element length.

Sets local friction coefficient.

Deletes this point.

Right click at intermediate point has further options.



**View on point**

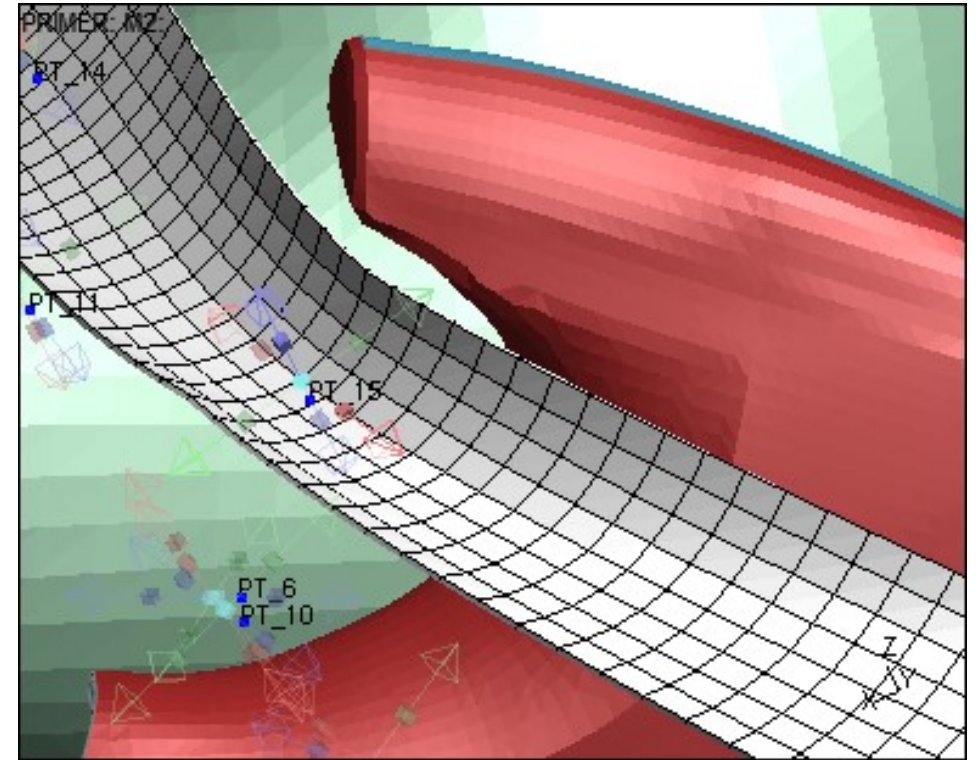
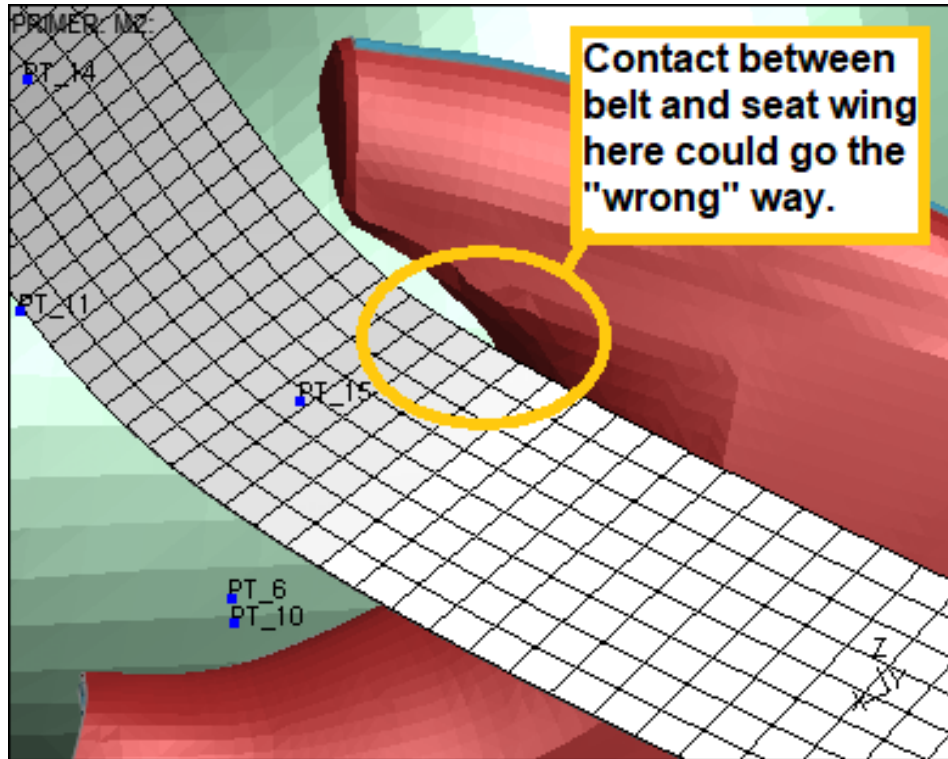
**Make path point**

Centres current view on point.

Makes a new basic path point at this intermediate point location.

# Advanced seatbelt fitting editor: Adding “curl” at a point.

Example of a Tweak: “Curling” the belt path helps it to negotiate difficult geometry.

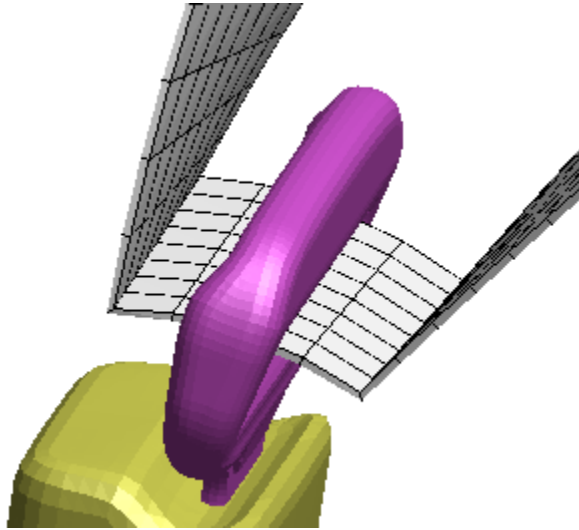


This example shows how adding “curl” can help the belt to fit to the wing of a child booster seat.

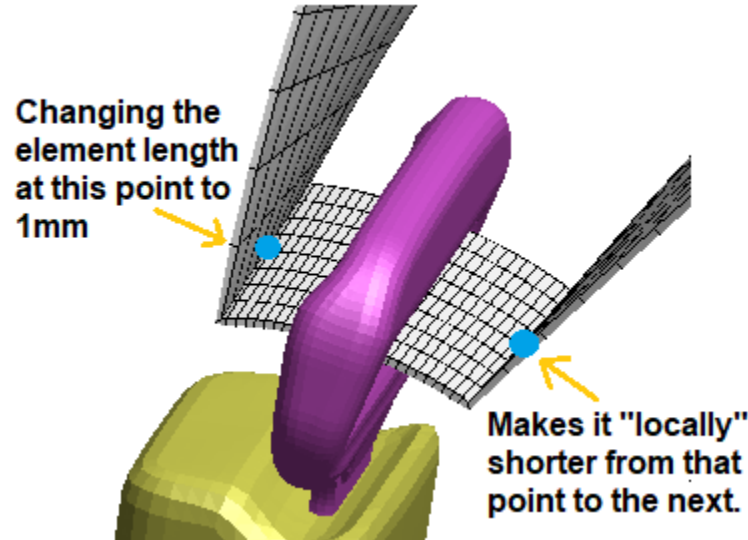


# Advanced seatbelt fitting editor: “local” element lengths

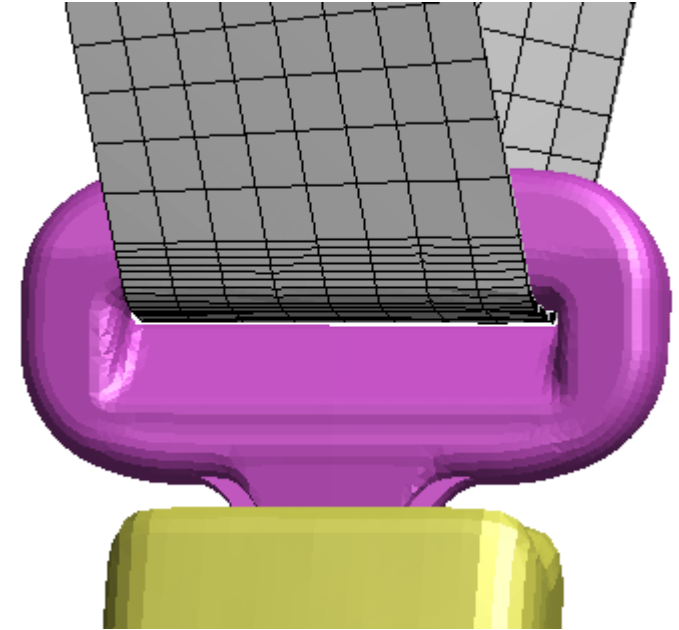
Setting a “local” element length allows the belt path to fit to tight geometries.



Meshing through a pelvis buckle with a coarse element length works, but is not satisfactory.



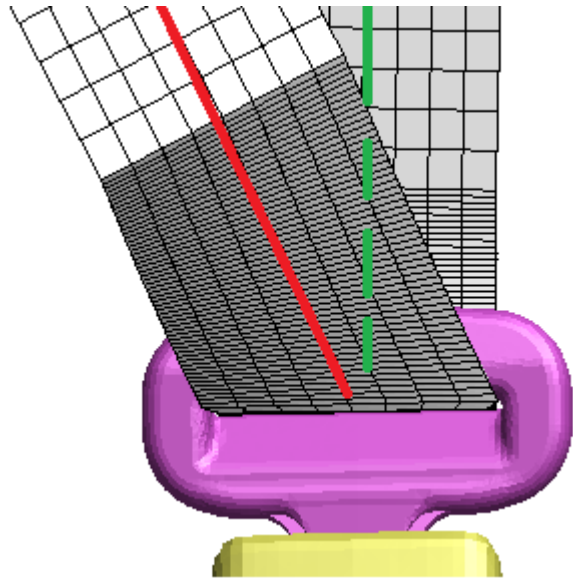
The option to shorten the elements locally from point N to N+1 has been used here.



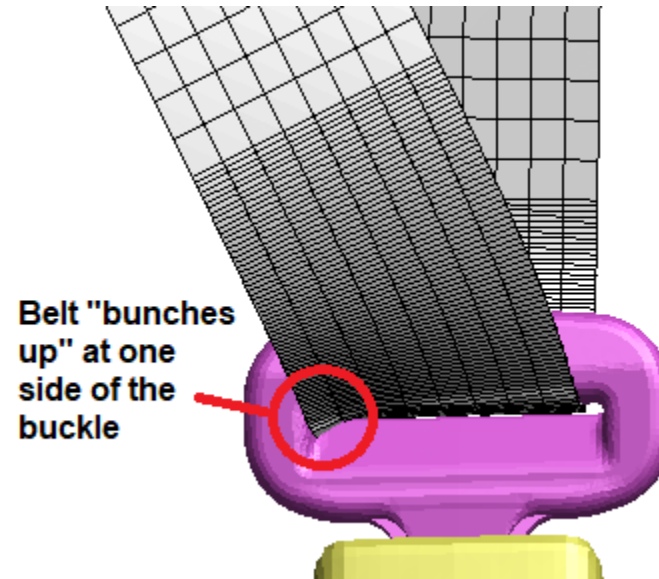
This gives a well conditioned mesh during fitting that will also work better during the actual analysis.

# Advanced seatbelt fitting editor: local friction coefficient

Using a “local” friction coefficient to solve the “bunching up” problem at a buckle.

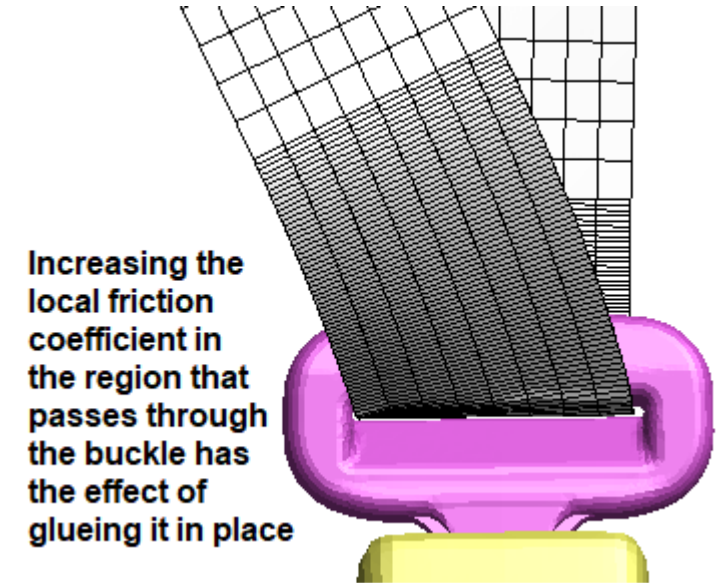


The path through this buckle is not symmetric which can cause problems during fitting.



Belt "bunches up" at one side of the buckle

The path (correctly) gets pulled to one side, causing bunching up in that corner

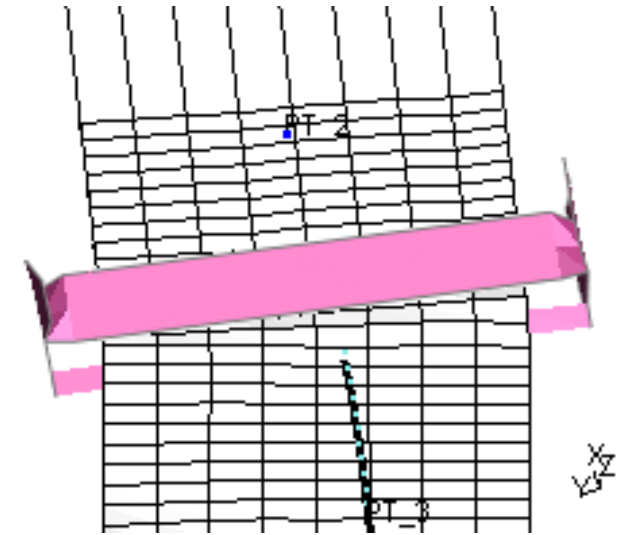
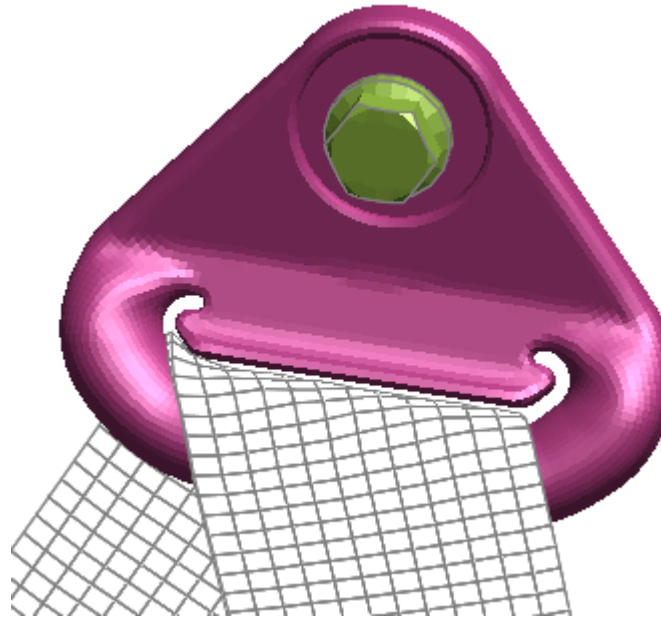
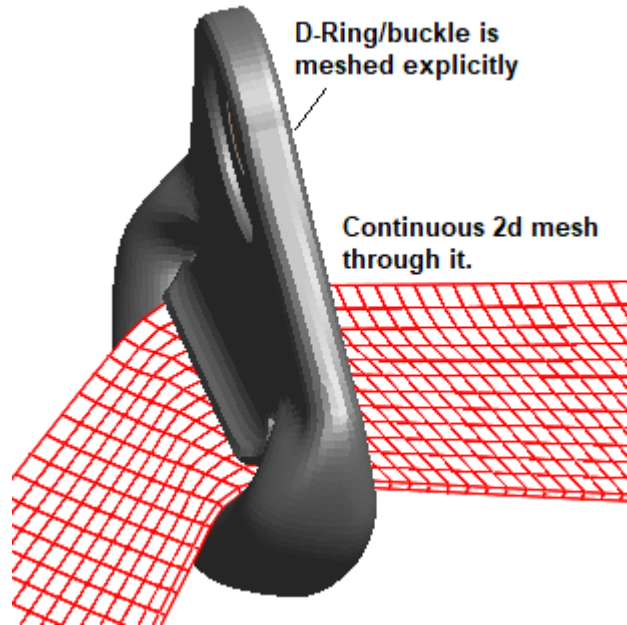


Increasing the local friction coefficient in the region that passes through the buckle has the effect of glueing it in place

A higher local friction makes the belt “stick to” the buckle, solving the problem.

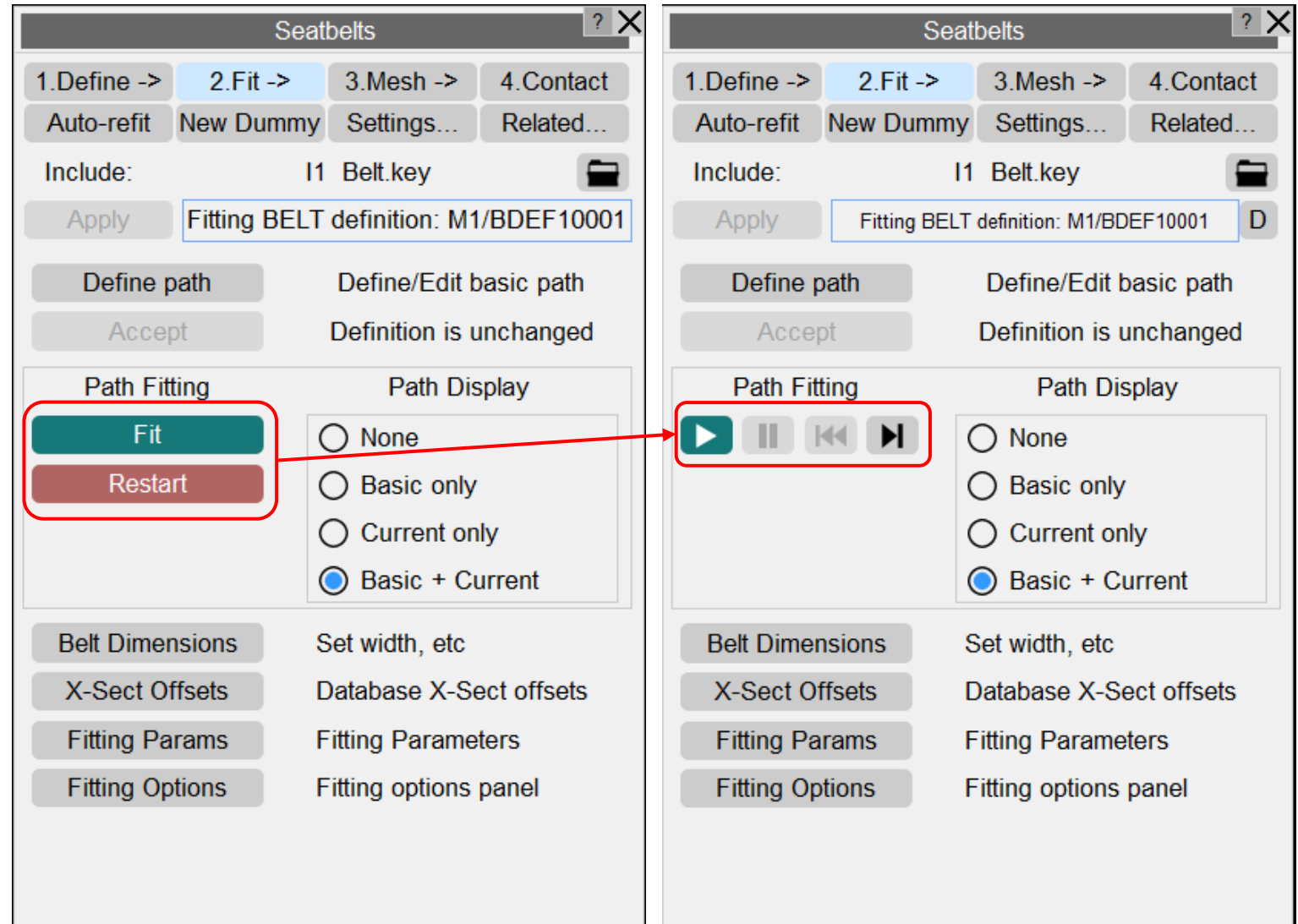
# Advanced seatbelt fitting editor: explicitly meshed sliprings

The improvements in the advanced editor make it possible to thread the belt through practically any geometry, so that \*ELEMENT\_SEATBELT\_SLIPRING is no longer necessary. Here are some examples of explicitly meshed buckles and guides.



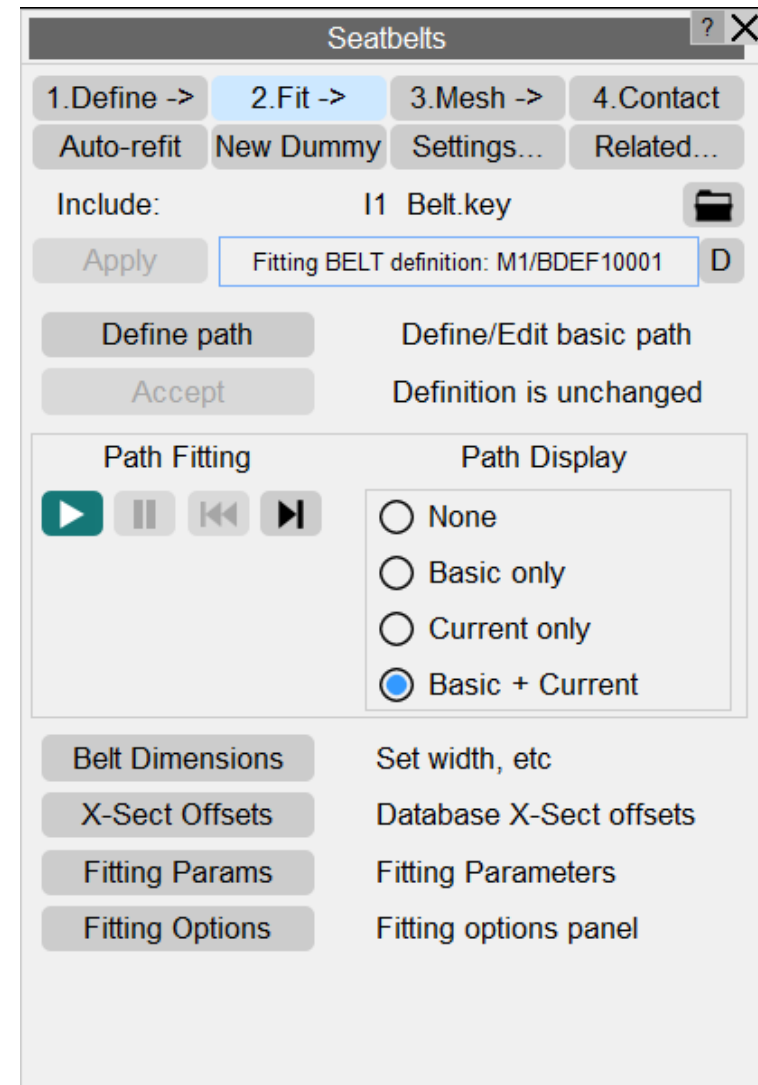
# "Fit" Panel changes - Path Fitting buttons

- More path fitting buttons added to give greater control.



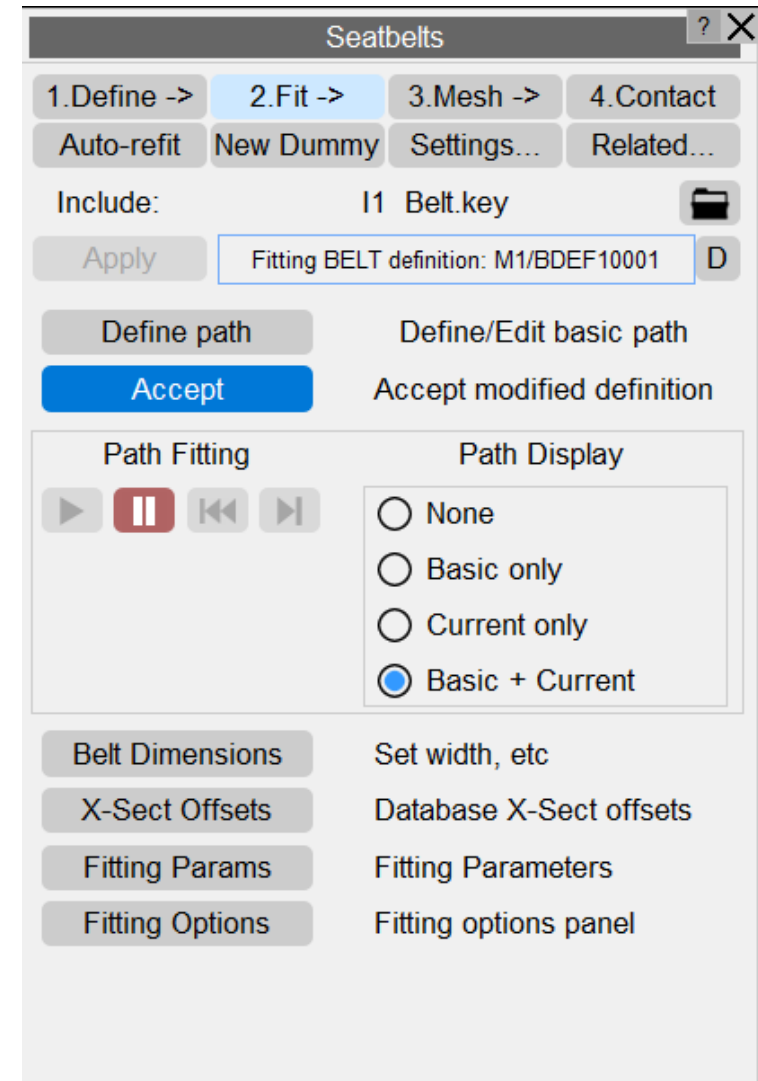
# "Fit" Panel changes – Initial State

- Initial (or Start) state of panel is shown
  - Before "Fit"
  - After "Reset".
- The button with the "Play" icon is the new "Fit" button.
  - Hover text has been modified for clarification.
  - Button alias has been added for macros.



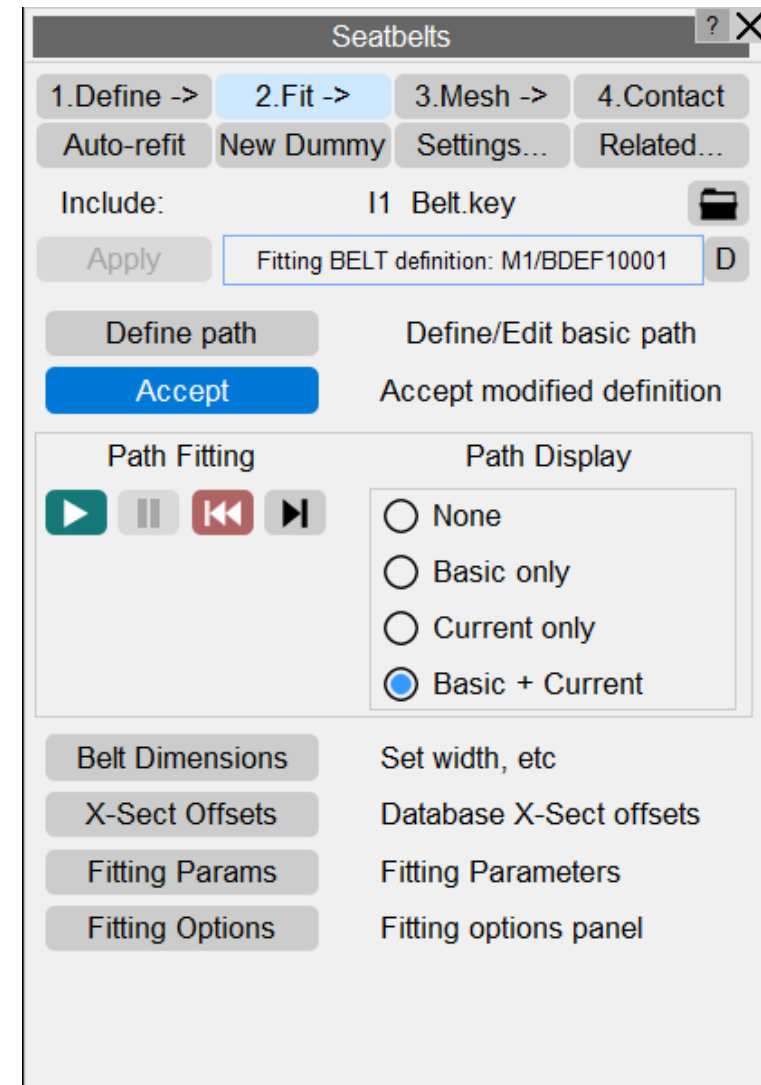
# "Fit" Panel changes – Processing State

- Processing (or Running/ Fitting) state is shown
  - During "Fit".
- A "Pause" button has been added.
  - This is a duplicate of the "Stop" button previously used to Stop/Pause the fitting process.



# “Fit” Panel changes – Paused State

- Paused state is shown
  - After “Pause”
  - After “Step”.
- The “Reset” button replaces the previous “Restart” button.
  - The path fitting will revert to the initial state and stay there till further actions are taken.
  - Note: “Restart” = “Reset” + “Fit”.
- A “Step” button has been added
  - The path fitting will be stepped forwards by 1 iteration.

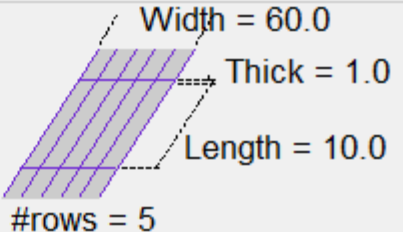


# Mesh Density Control

- Mesh density controls have been reorganised to provide easier workflow.

## PRIMER 17.0

Belt width:	<input type="text" value="60.0"/>
Belt length:	<input type="text" value="10.0"/>
Belt thickness:	<input type="text" value="1.0"/>
#rows:	<input type="text" value="5"/>
Length param:	<input type="text" value="&lt;none&gt;"/>



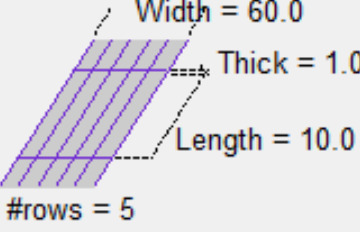
Width = 60.0  
Thick = 1.0  
Length = 10.0  
#rows = 5

☐ \_Sn per segm parameter



## PRIMER 18.0

Seatbelt Mesh Density:	
<input type="text" value="Coarse (Default)"/>	
Max curve angle:	<input type="text" value="30.0"/>
Belt length:	<input type="text" value="10.0"/>
#rows:	<input type="text" value="5"/>
Belt width:	<input type="text" value="60.0"/>
Belt thickness:	<input type="text" value="1.0"/>
Length param:	<input type="text" value="&lt;none&gt;"/>



Width = 60.0  
Thick = 1.0  
Length = 10.0  
#rows = 5

Tolerance:	<input type="text" value="1.0E-5"/>
Projection:	<input type="text" value="35.0"/>
Trans Friction:	<input type="text" value="0.0"/>

☐ \_Sn per segm parameter



# Mesh Density Control – 4 levels

- New dropdown menu to control “Max curve angle”, “Belt length” & “#rows”.

Seatbelt Mesh Density:

**Coarse (Default)** ▼

Max curve angle: 30.0

Belt length: 10.0

#rows: 5

Belt width: 60.0

Belt thickness: 1.0

Length param: <none>

Width = 60.0

Thick = 1.0

Length = 10.0

#rows = 5

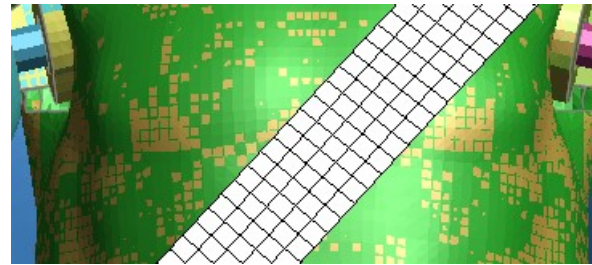
Tolerance: 1.0E-5

Projection: 35.0

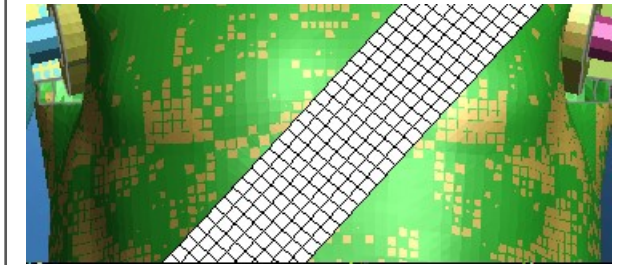
Trans Friction: 0.0

\_Sn per segm parameter ☐ ?

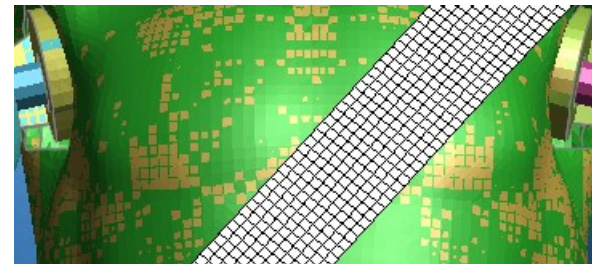
Coarse (Default)



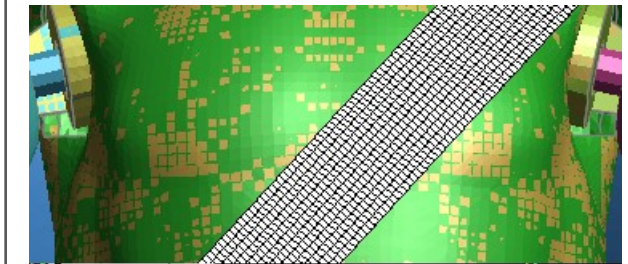
Medium



Fine



Very Fine



# Duplicated Fitting Parameters

- Original Panel

Fitting Parameters: Done Save... Explain

Tol:	1.0E-5	Convergence tolerance
#lter	25	#Iterations per pass
Over	5.0E-2	Facet overlap value
Pene	5.0	Max penetration dist
Proj	35.0	Init projection dist
Curve	30.0	Max curvature angle
Friction	0.0	Transverse friction
Acute ang	90.0	'Acute' angle limit

☒ Use parallel fitting Explain



- Values duplicated in pop-up panel

Seatbelt Mesh Density:

Coarse (Default) ▼

Max curve angle: 30.0

Belt length: 10.0

#rows: 5

Belt width: 60.0

Belt thickness: 1.0

Length param: <none>

Width = 60.0  
Thick = 1.0  
Length = 10.0  
#rows = 5

Tolerance: 1.0E-5

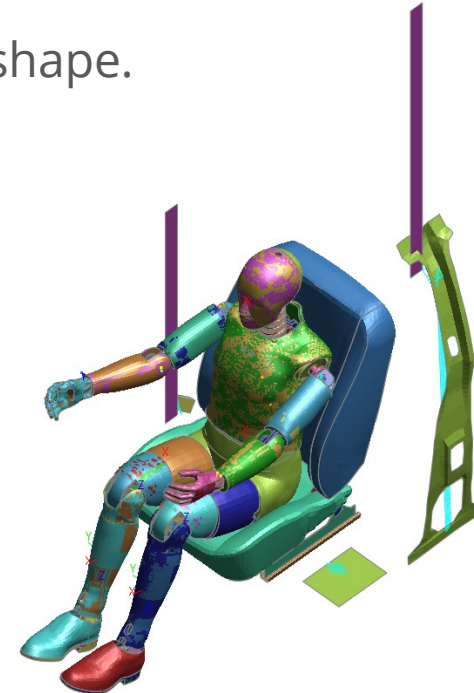
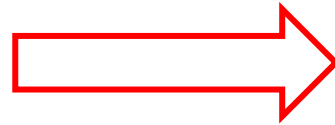
Projection: 35.0

Trans Friction: 0.0

\_Sn per segm parameter ☐ ?

# Reference Geometry – Belt Meshing (SHELLs only)

- Initial distortion of the fabric shell belt mesh can now be removed during the analysis with the aid of \*AIRBAG\_REFERENCE\_GEOMETRY.
- PRIMER now has the option to create \*AIRBAG\_REFERENCE\_GEOMETRY automatically during the belt meshing phase.
- When MAT FABRIC is used for the shell belt with |FORM|  $\geq 12$ , the TSRFAC loadcurve can be used to gradually pull the mesh back to its reference geometry shape.



# Reference Geometry – Panel changes

“Mesh” panel:

- Added tick box.
- Added reference geometry label.

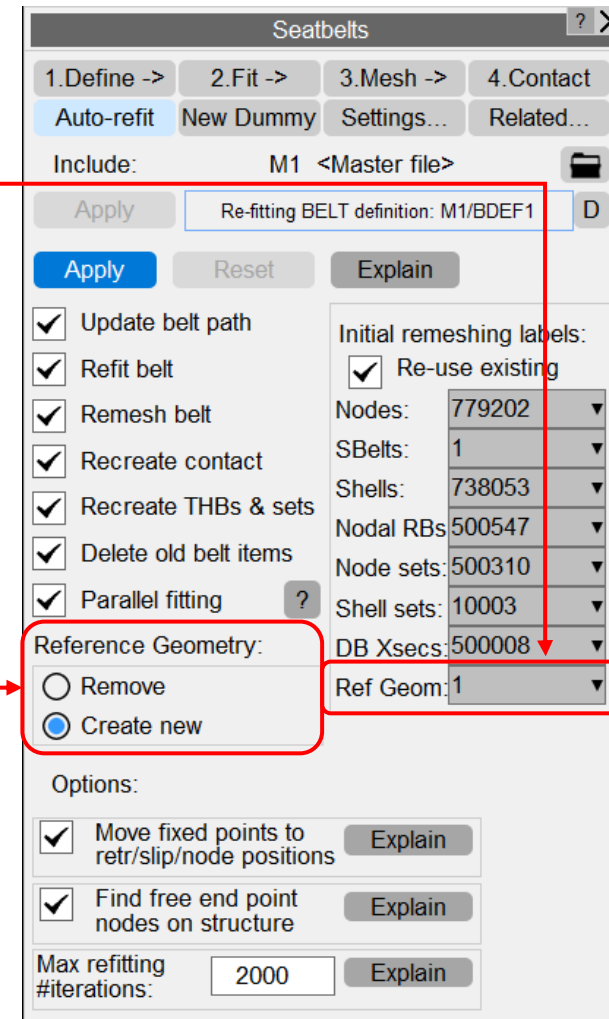
The screenshot shows the 'Seatbelts' software interface. The '3.Mesh ->' tab is selected. The 'Include:' field shows 'M1 <Master file>'. The 'Current BELT defn:' is 'M1/BDEF1'. The 'Mesh type' is set to 'O / N'. The 'Reset mesh' button is visible. The 'Pt 1: END' and 'Node 203472' are shown. The '#Belt elems = 89' and 'Sb1 part: 10001' are displayed. The 'Node <auto>' and 'Pt 2: Fixed' are shown. The 'Create reference geometry' checkbox is checked and highlighted with a red box. The 'Ref Geometry' dropdown menu is also highlighted with a red box. The 'Meshing start labels' section shows various parameters like Nodes, Shells, SBelts, Retractors, Sliprings, 2D slip nset, Node sets, Shell sets, Nodal RBs, DB XSections, Part Sets, and Ref Geometry.

Meshing start labels			
Nodes:	780495	Node sets:	500316
Shells:	739033	Shell sets:	10004
SBelts:	106	Nodal RBs:	500551
Retractors:	1	DB XSections:	500008
Sliprings:	1	Part Sets:	500010
2D slip nset:	500316	Ref Geometry:	1

# Reference Geometry – Panel changes

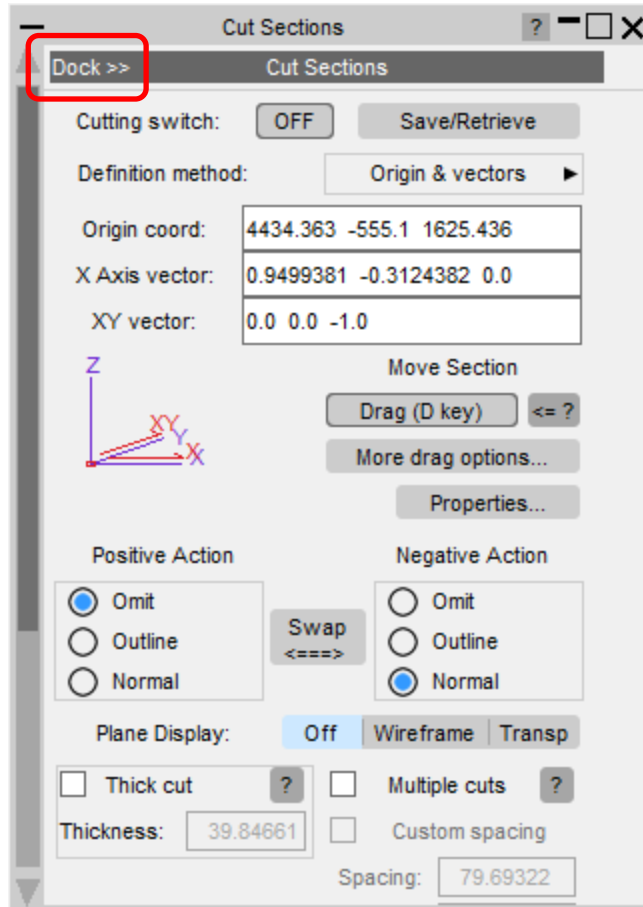
“Auto-refit” panel:

- Added reference geometry label.
- Added radio buttons.

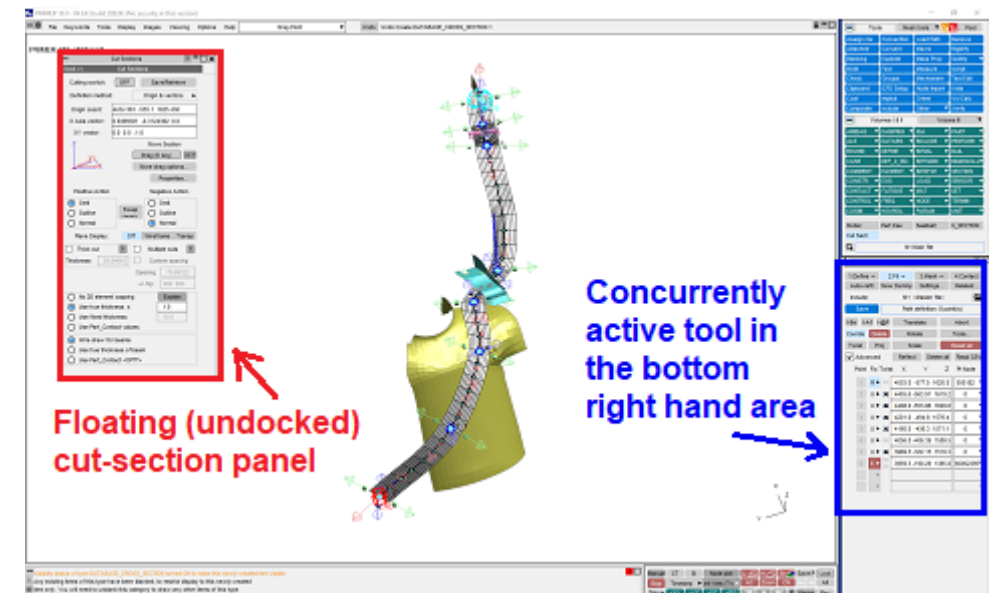


# Undocking Cut Sections Panel

# Cut section panel can be undocked



- Undocking the Cut-section panel allows it to be active at the same time as other tools but – crucially – not occupying the tabbed area at the bottom right hand side of the screen.
- This means that it can be used at the same time as other graphical operations, such as seatbelt fitting.



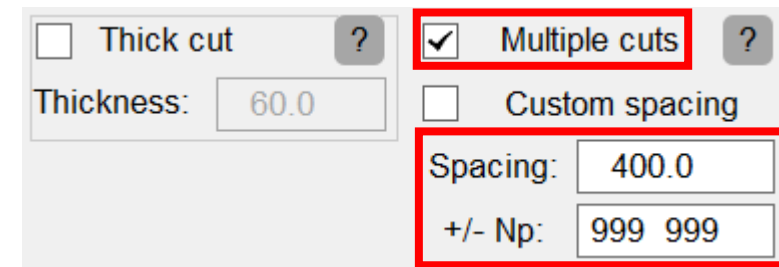
# Cut Sections Improvements



# Multiple parallel cuts

---

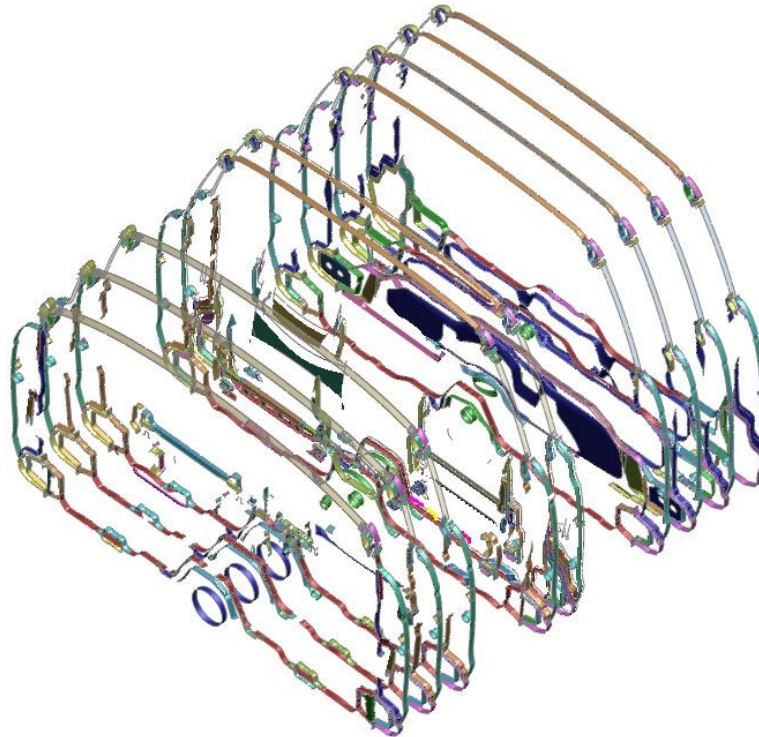
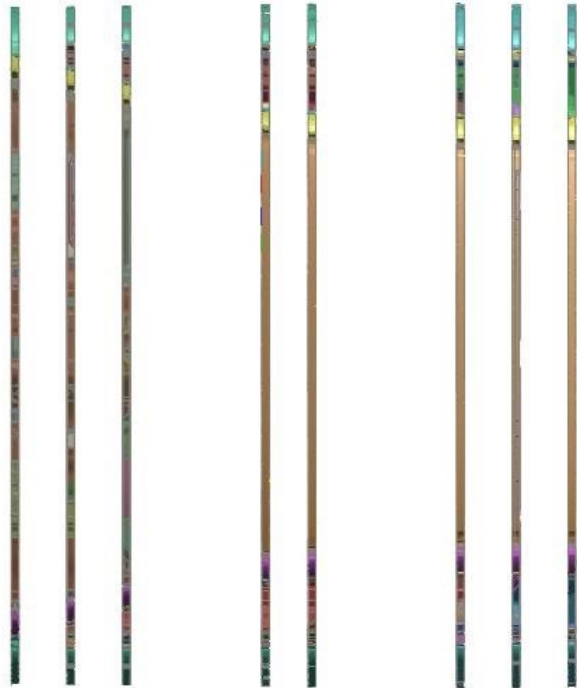
- For a given cut plane multiple parallel cuts can be defined by a (uniform) spacing and numbers of planes in each direction.



<input type="checkbox"/> Thick cut	?	<input checked="" type="checkbox"/> Multiple cuts	?
Thickness:	60.0	<input type="checkbox"/> Custom spacing	
		Spacing:	400.0
		+/- Np:	999 999

# Custom spacing

- In PRIMER 18 the spacing can now be customised to not necessarily uniform spacing.



☒ Thick cut ?  
Thickness: 20.0

☒ Multiple cuts ?  
☒ Custom spacing  
Edit spacing

CUT SECTION CUSTOM SPACING

Positions relative to local cut origin Done

Number of parallel planes 9

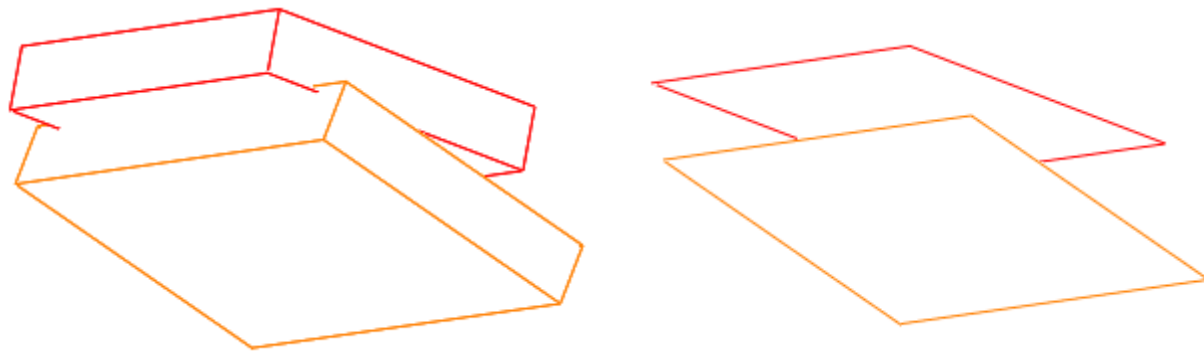
Add position:

1 ▶	-800.0
2 ▶	-650.0
3 ▶	-500.0
4 ▶	-125.0
5 ▶	0.0
6 ▶	400.0
7 ▶	550.0
8 ▶	700.0
9 ▶	850.0
10 ▶	undefined

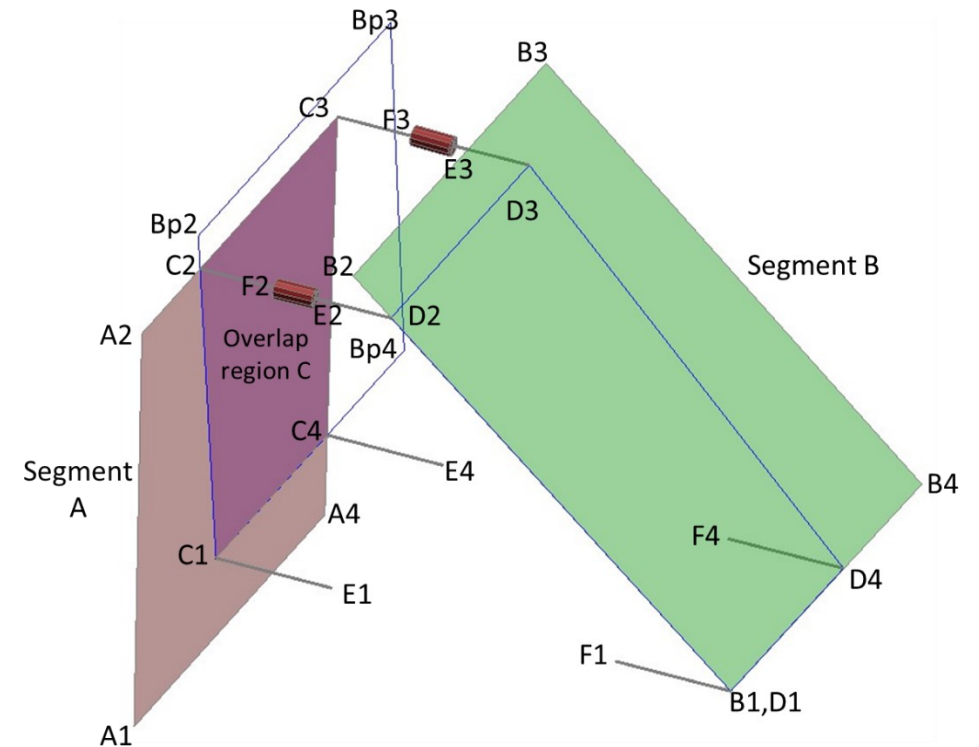
SOFT = 2

# SOFT = 2 supported for contact analysis

Surface to surface and single surface contacts in LS-DYNA have option to use segment based contact (soft = 2). This method gives more accurate edge to edge behaviour. Without soft = 2 shell edges can penetrate if there is no 's'-node to 'm'-segment contact.



With soft = 2 LS-DYNA uses a projection algorithm to calculate worst case penetration which PRIMER v18 simulates.



# Isogeometric Analysis (IGA)

# IGA Enhancement

---

- Isogeometric Analyses can be run in LS-DYNA using the keyword `*ELEMENT_SHELL_NURBS_PATCH` and `*DEFINE_NURBS_CURVE` to define the surfaces.
- The `*ELEMENT_SHELL_NURBS_PATCH` edit panel allows you to manipulate knot values, change the basis function degree, visualize knot grids, and sketch trimming curves.

# Visualise Knot Grid

- “Display Knot” will draw all the knot values (NURBS elements).

MODIFY SHELL\_NURBS\_PATCH 1

Update Reset All Check Sketch Only Cancel Copy In X-Refs Text Edit

Include: M1 <Master file>

Modify SHELL\_NURBS\_PATCH element 1 (model 1)

Title: <No name given>

NPID	PID	NPR	PR	NPS	PS	PERIR	PERIS
1	1	52	3	72	3	0	0
WFL	FORM	INT	NISR	NISS	IMASS		IDFNE
0	0	0	3.0	3.0	0		0

Screen Pick Knot

r-Knot Value: <none> Insert r-Knot

s-Knot Value: <none> Insert s-Knot

RK1	RK2	RK3	RK4	RK5	RK6	RK7	RK8
0.75510198	0.77551	0.79591799	0.81632698	0.83673501	0.85714298	0.87755102	0.89795899
0.91836703	0.93877602	0.95918399	0.97959203	1.0	1.0	1.0	1.0

SK1	SK2	SK3	SK4	SK5	SK6	SK7	SK8
0.884058	0.89855099	0.91304302	0.92753601	0.942029	0.95652199	0.97101402	0.98550701
1.0	1.0	1.0	1.0				

N1	N2	N3	N4	N5	N6	N7	N8
3725	3726	3727	3728	3729	3730	3731	3732
3733	3734	3735	3736	3737	3738	3739	3740
3741	3742	3743	3744				

W1	W2	W3	W4	W5	W6	W7	W8

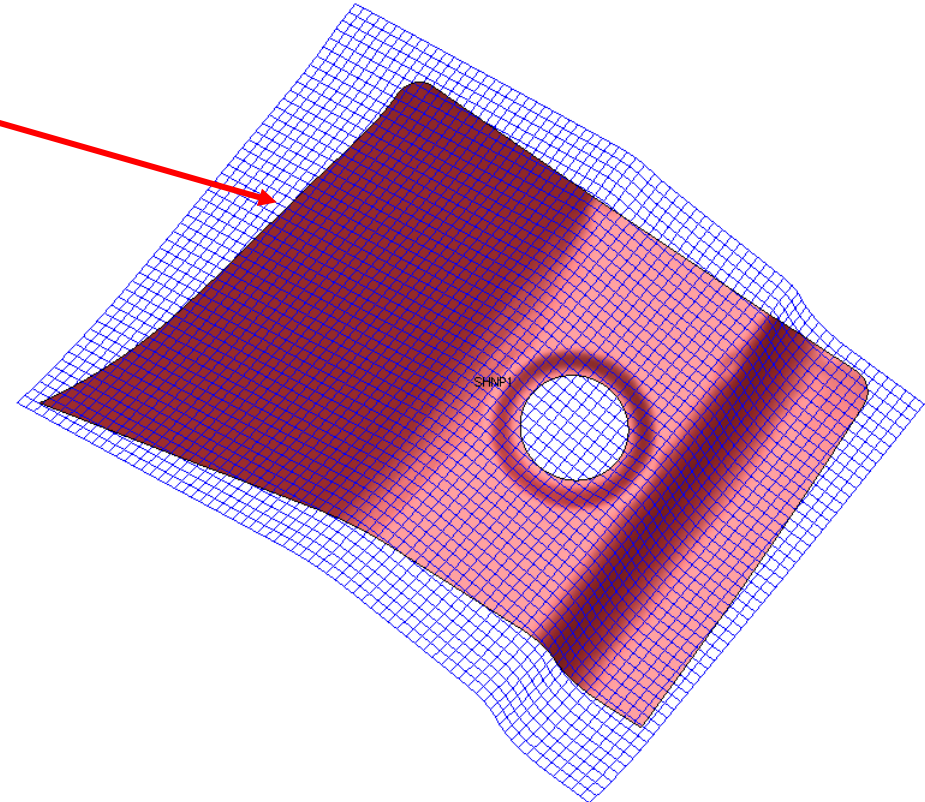
\_TRIMMED

Trimming loop 1 of 2 Previous Next Delete loop Add loop Sketch loop Sketch all loops

TITLE

trimming loop 1

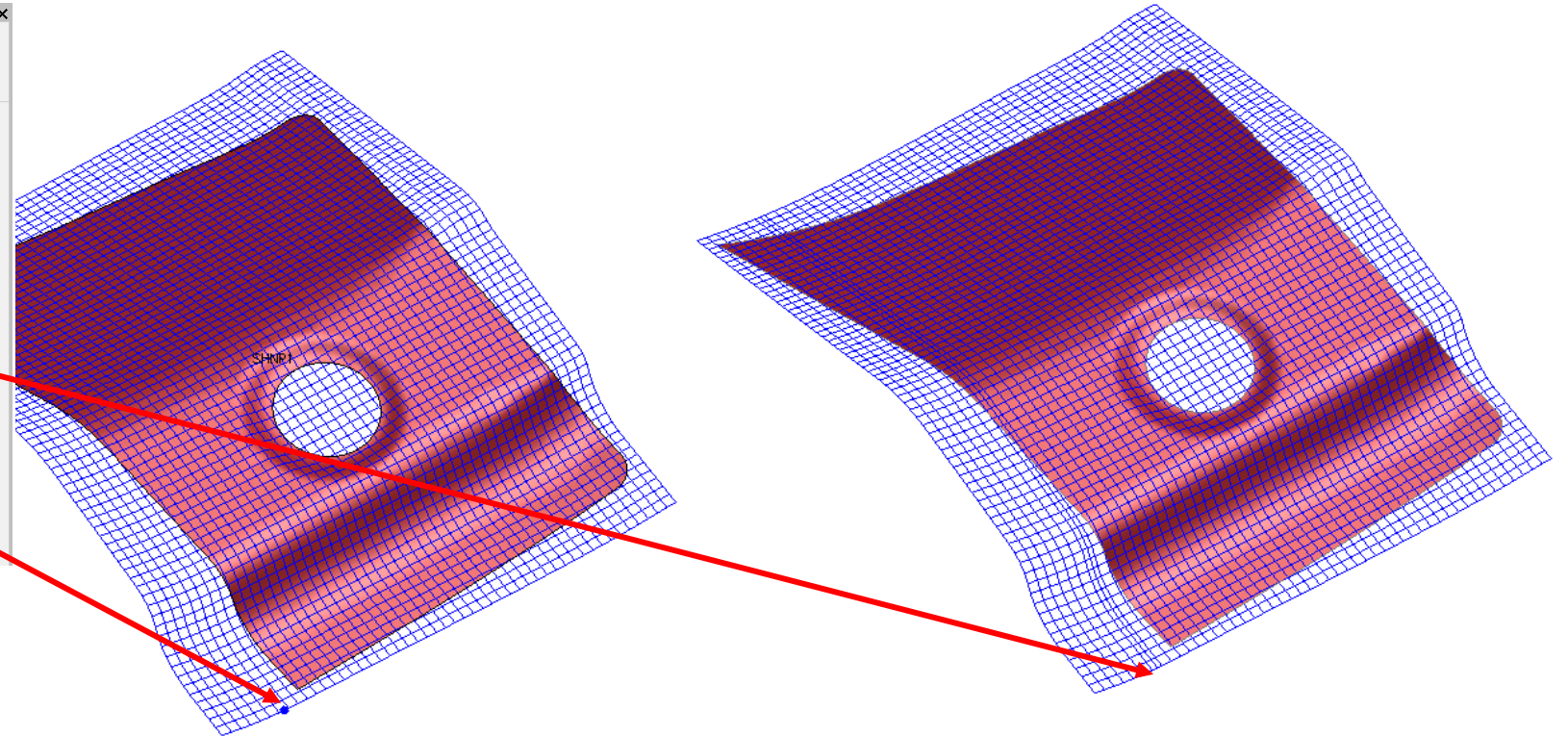
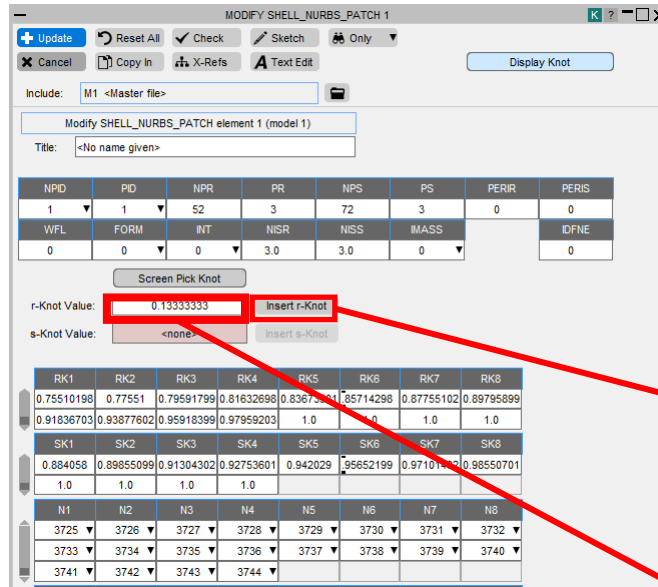
C1	C2	C3	C4	C5	C6	C7	C8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	





# Insert Knot Values (h-refinement) - First Method

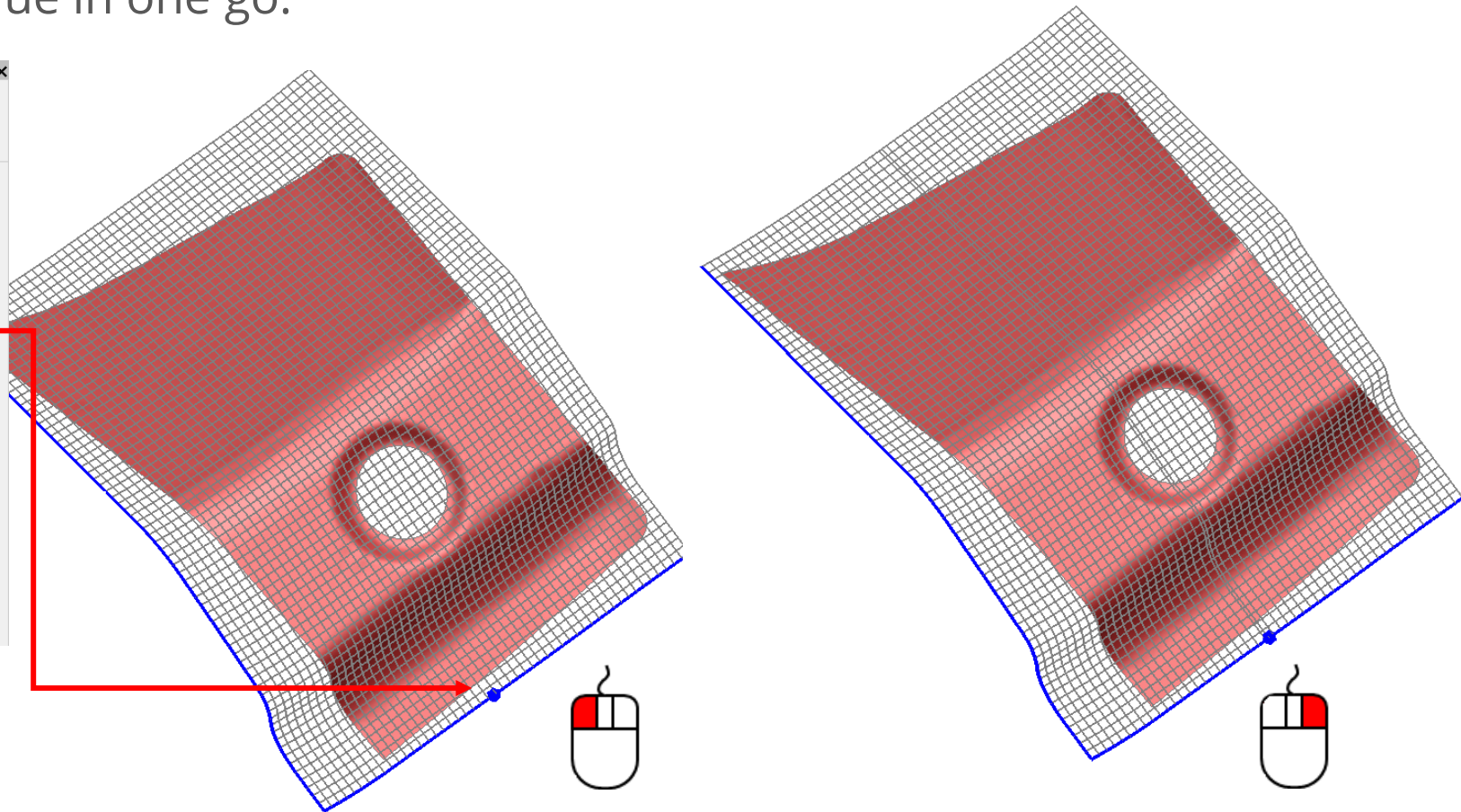
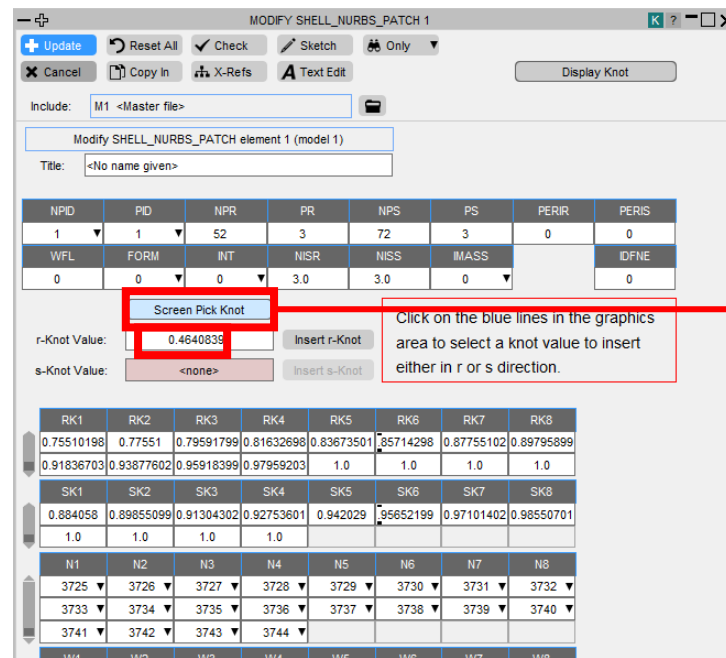
- First method: type in new knot values in “r-Knot Value/s-Knot Value” textboxes and press “Insert r-knot/Insert s-knot”.





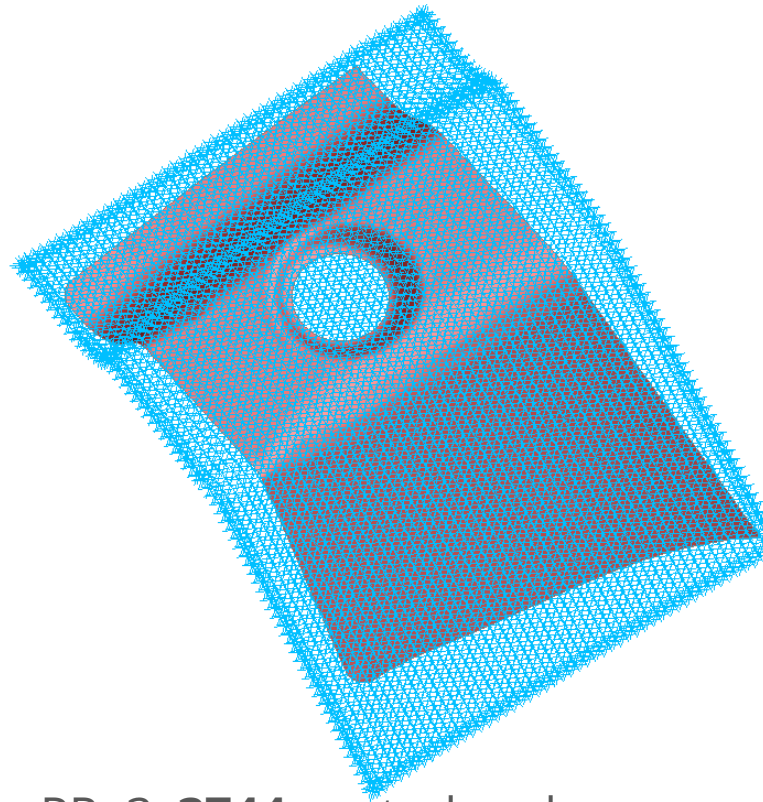
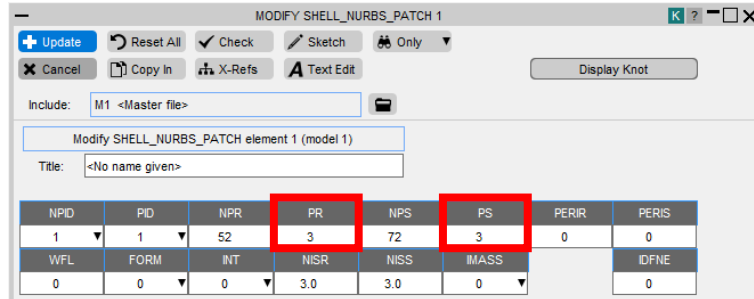
# Insert Knot Values (h-refinement) - Second Method

- Second method: select a knot value by left clicking on the blue solid lines in the graphics area and press "Insert r-knot/Insert s-knot". Alternatively, use right click to select and insert a knot value in one go.

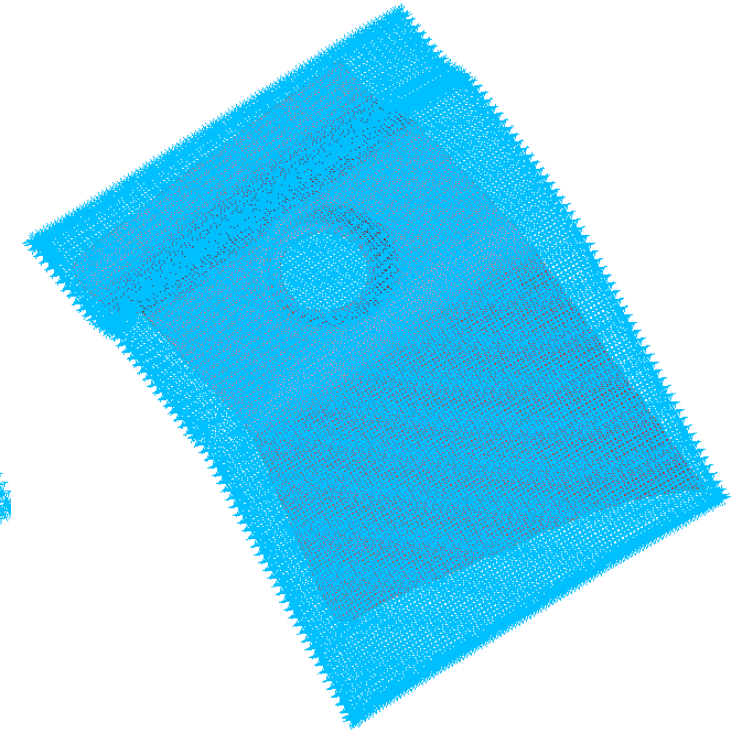


# Change the basis function order (p-refinement)

- The order can be changed by typing in the new value in **PR/PS** textboxes. It will add new or delete existing control points.



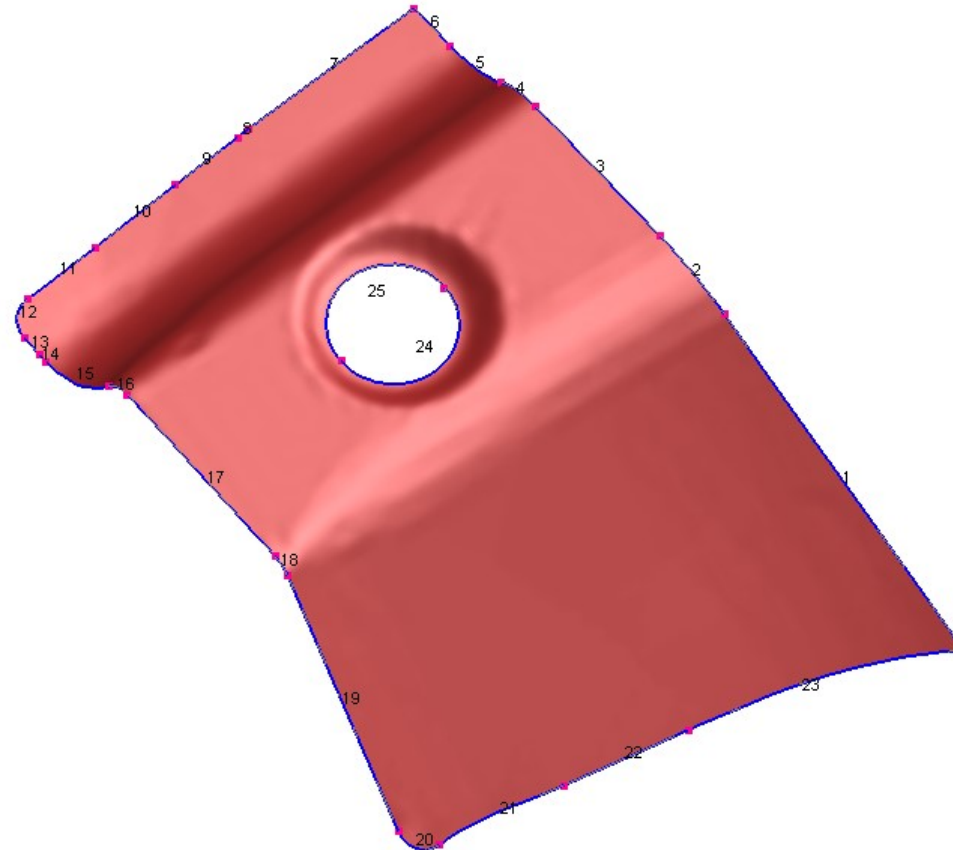
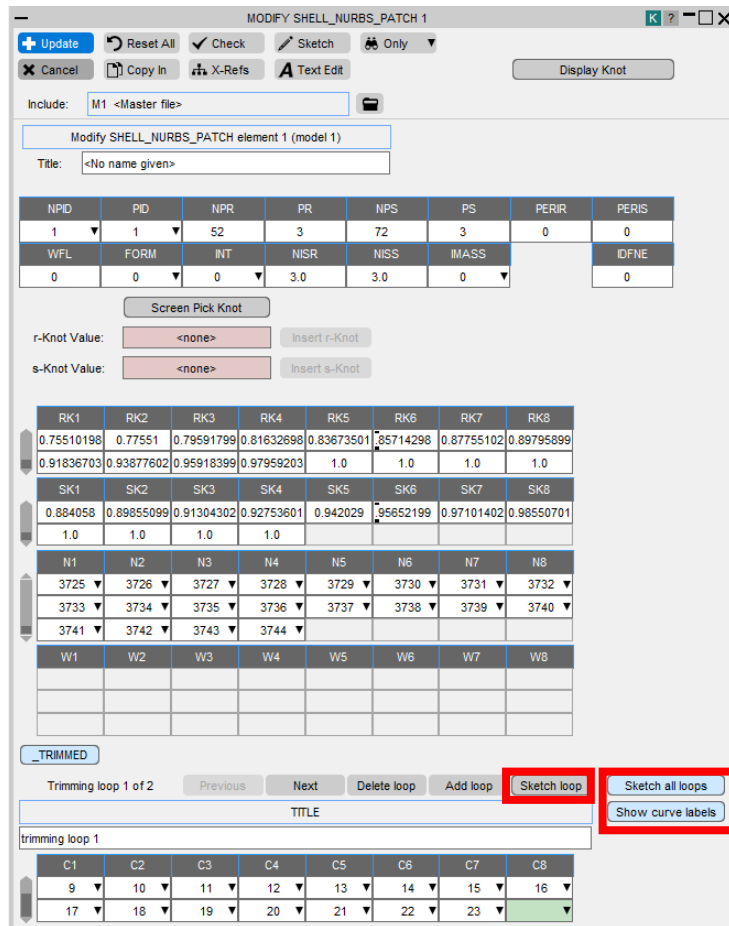
PR=3, **3744** control nodes



PR=4, **7272** control nodes

# Sketch Trimming Curves

- Enable “Sketch loop” or “Sketch all loops” toggle buttons to sketch trimming curves.



# Other IGA related enhancements

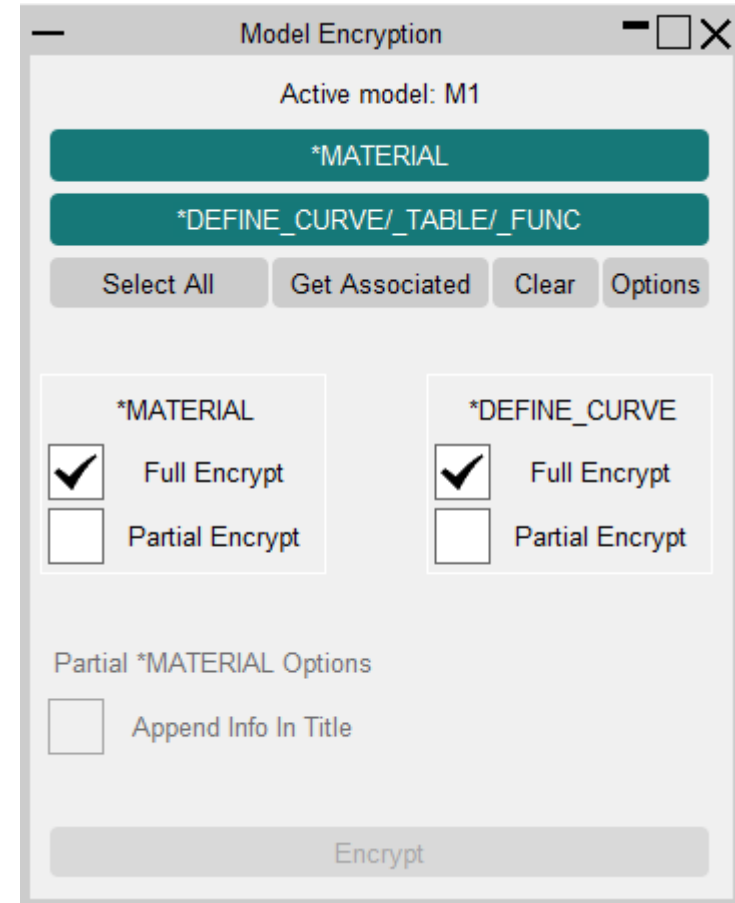
---

- Added warning if a node defined on \*ELEMENT\_SHELL\_NURBS\_PATCH is used in \*BOUNDARY\_SPC, \*BOUNDARY\_PRESCRIBED\_MOTION or \*LOAD\_NODE, as it will be a non-structural node.
- Added support for mass, CofG and inertia tensor calculation for \*ELEMENT\_SHELL\_NURBS\_PATCH.
- Added warning for very high order basis function (PR and PS value) in \*ELEMENT\_SHELL\_NURBS\_PATCH.
- Added warning if NISS/NISR are less than PS/PR in \*ELEMENT\_SHELL\_NURBS\_PATCH.
- Various new \*IGA keywords added.
- Added keyword editor for \*ELEMENT\_SOLID\_NURBS\_PATCH similar to \*ELEMENT\_SHELL\_NURBS\_PATCH.

# Encryption Tool

# Automatic PGP Encryption

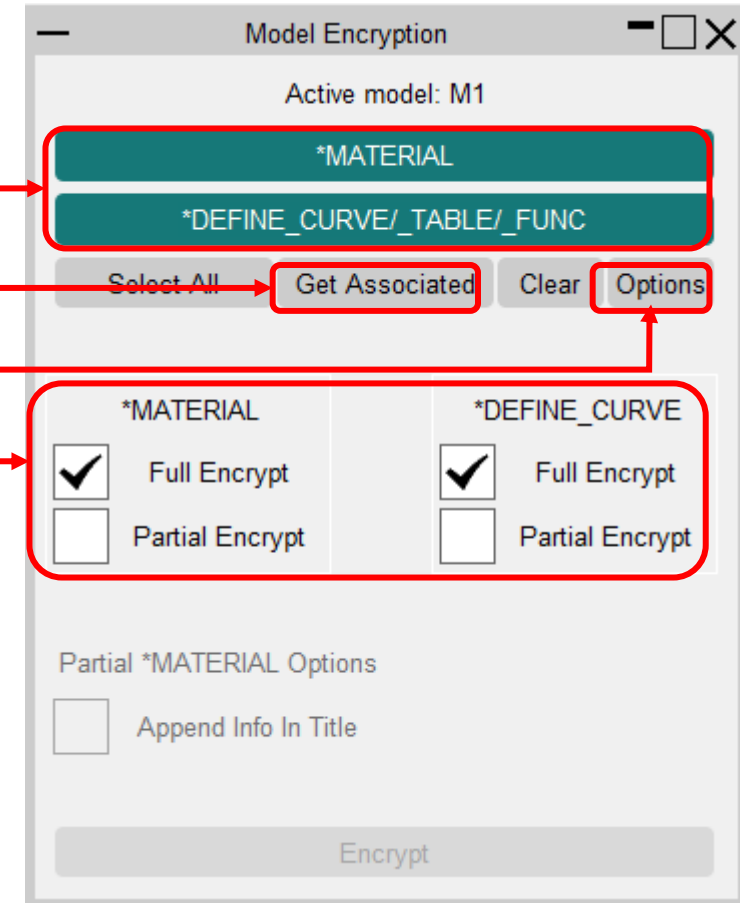
- This Oasys PRIMER tool is for encrypting \*MATERIAL and \*DEFINE\_CURVE keywords for LS-DYNA.
- The tool allows you to choose between partially and fully encrypting a keyword.
- Encrypted keywords can be used by both LS-DYNA and PRIMER.





# Easy Encryption

- Easily select different Keywords
- Select \*DEFINE\_CURVES associated to a \*MATERIAL
- Switch between 3 LSTC keys
  - 2048bit key
  - 1024bit key
  - Legacy R6 key
- Customise your encryption options



# Partially Encrypted \*AIRBAG Cards



# Partially Encrypted PGP data in \*AIRBAG cards

---

As with \*MAT and \*DEFINE\_CURVE it is now possible to define partially encrypted \*AIRBAG cards in order to protect proprietary data.

<b>*AIRBAG_xxx_ID</b>	<b>Where xxx is the airbag type. A new *AIRBAG header is required for each definition</b>
<label> <title>	The first line of the definition giving its label and an optional title must be supplied.  (The title is not parsed in any way so it can contain anything.)
-----BEGIN PGP MESSAGE-----  [Encrypted data]  -----END PGP MESSAGE-----	The encrypted data may start at any line thereafter.  It is normally the case that it will start immediately after the [label, title] row above, but PRIMER will "remember" the line at which it starts, so further lines of data in clear could be supplied if desired.

# HIC Area Calculator

# Main Panel Updates

- The main panel has been updated to include a table of results.
- A new comparison tab has been added.
- Additional options have been included and some have moved location.

HIC Area Calculator

Read Data File Read Second File

REFINE PERIMETER

Tight Slack

220 Read Perimeter File

DISPLAY OPTIONS

☒ Show Input Points Text: Not Shown

☒ Show Perimeter Text Size: ▲ ▼

☒ Banded Input Y Offset: 0

☒ Banded Output Projection: Off

Change Symbol Size

Calculate Area

Calculation Points Analysis Utilities

CALCULATION PARAMETERS

Regulation: GTR Scale Factor: 1 Res

Low HIC: 1000 High HIC: 1700

Grid Size: 10

v17

HIC Area Calculator

Read Data File Read Second File

DISPLAY OPTIONS

☒ Show Input Points Text: Not Shown

☒ Show Perimeter Text Size: ▲ ▼

☒ Coloured Input Y Offset: 0

Change Symbol Size Project to Mesh

Show Plot Show Contour Bar

Read data to calculate

	# of Points	Low HIC % by	>High	
	<Low	>Low	# Points	Area
Child				
Adult				
Combined				

Calc Points Analysis Utilities Compare

CALCULATION PARAMETERS

Reg: GTR Scale Factor: Res

Refine Perim: Tight 220 Slack

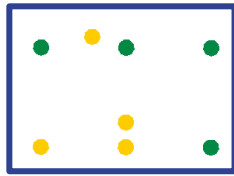
High HIC: 1700 Fine Grid Size: 10

Low HIC: 1000 Calculate Area

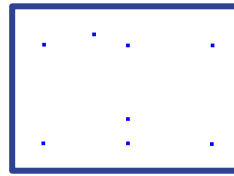
v18

# Display Option Changes

- **Coloured Input** | option was previously 'Banded Input' and toggles between a simple blue square icon or colour coded circle.

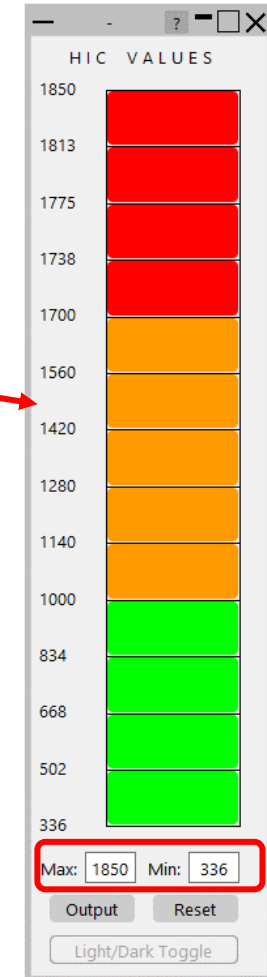
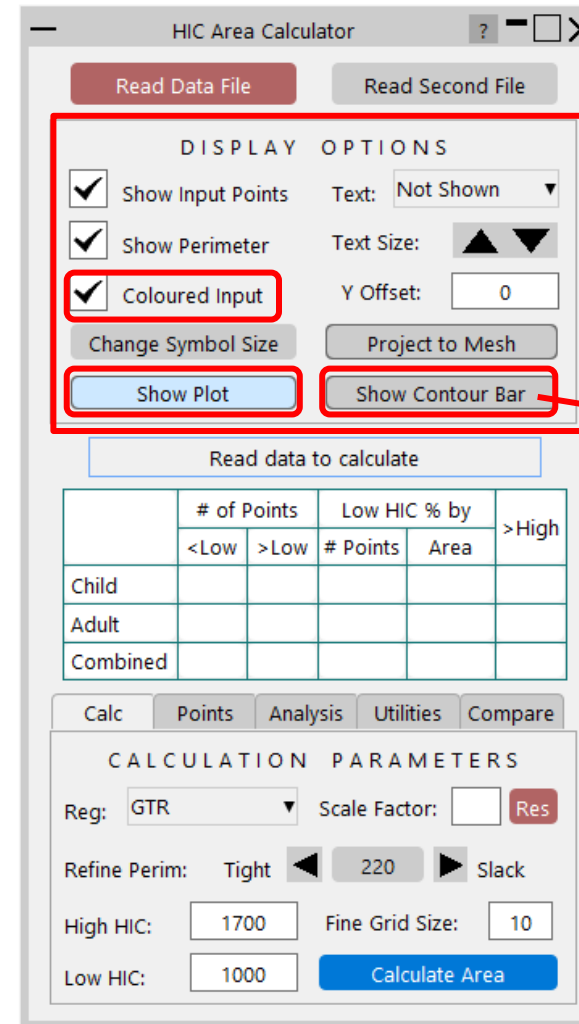


Coloured Input



Not Coloured Input

- **Show Plot** | for showing/hiding all of the on screen graphics at once, which may be useful when working with multiple data sets.
- **Show Contour** | for showing/hiding the contour bar, which is no longer the main location for the results.



Options for changing contour bar limits now more obvious.

# Table of Results

- The new table displays the result information for the current regulation:

## GTR

	# of Points		Low HIC % by		>High
	<Low	>Low	# Points	Area	
Child	36	37	49.3 %	63.3 %	0
Adult	79	41	65.8 %	71.1 %	5
Combined	115	78	59.6 %	68.5 %	5

Determined by point names

# of Points with HIC > High HIC Limit

# of Points with HIC less than and greater than Low HIC Limit.

Note: Points > Low also includes points > High

Area % as determined by point count ratio (# Points) and area calculation (Area).

Area calculations are only displayed when the specific area calculation is requested.

Note regarding Child/Adult Areas: by default the script 'shrink wraps' a perimeter to the relevant points. It is possible to change this behaviour (detailed in later slide).

## NCAP

	Points	ENCAP v8 Scores				
		G	Y	O	B	Total
Child	73	30	22.5	6.5	0	59.00
Adult	61	36	18.75	0	0	54.75
Cyclist	59	6	19.5	11	0	36.50
Total	193	72	60.75	17.5	0	18.684 / 24

Determined by point names

Number of impact points read

Scores per colour band:  
Green, Yellow, Orange, Brown

Absolute total score per zone

Overall score

18.684 / 24
ENCAP v8 Normalised Score = 18.684 (out of 24)
Calculated as follows:
Total score = 150.25. (150.25)
Max achievable = number of impacts = 193. (193 input + 0 additional)
Score ratio = total score / max achievable = 150.25 / 193 = 77.850%.
Max score awarded for head impacts (ENCAP v8) = 24.
Final normalised score = regulation head impact maximum * score ratio
= 24 * 77.850%
= 18.684 (3dp)

Hover text presents the full calculation

Buttons allow you to enter additional impact points and scores per zone.

# Calc Tab

- Additional new regulations:
  - EuroNCAP v9 (DRAFT)
  - C-NCAP 2021
- Maximum NCAP point allocation is now editable. Defaults are:
  - EuroNCAP v8 = 24
  - EuroNCAP v9 = 18
  - C-NCAP 2021 = 10
- Perimeter control moved from top of main panel to Calc tab.
- Area calculation has been updated to provide smoother interpolation when input blobs are arranged in regular grids.

**Note: EuroNCAP v9 has not been officially released.**  
**Settings are correct at time of release based on DRAFT protocol.**

Calc Points Analysis Utilities Compare

**CALCULATION PARAMETERS**

Reg: ENCAP v8 Scale Factor:  Res

Yellow HIC:  Orange HIC:

Brown HIC:  Red HIC:

Max Points:  [Recalculate Score](#)

Calc Points Analysis Utilities Compare

**CALCULATION PARAMETERS**

Reg: GTR Scale Factor:  Res

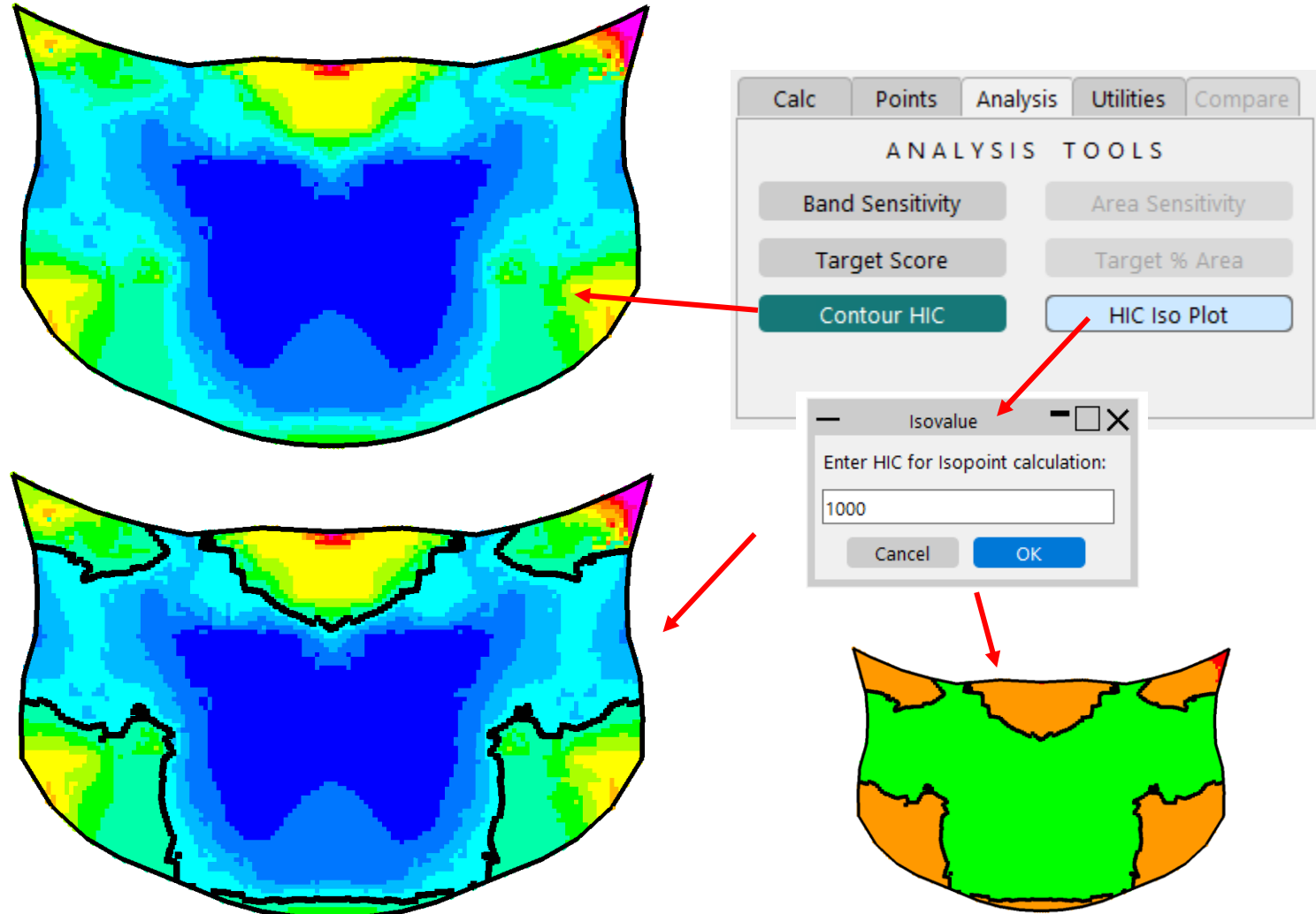
Refine Perim: Tight 185 Slack

High HIC:  Fine Grid Size:

Low HIC:  [Calculate Area](#)

# Analysis Tab

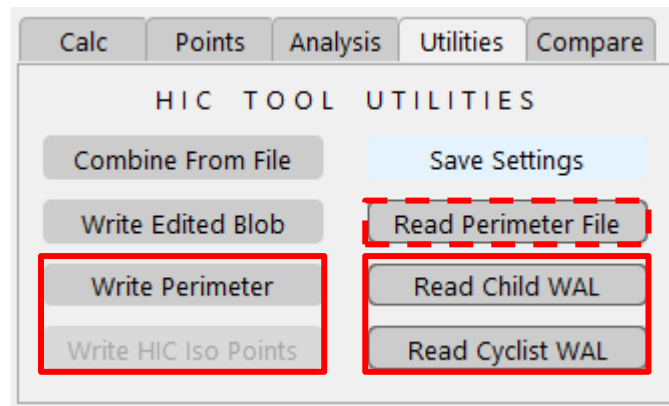
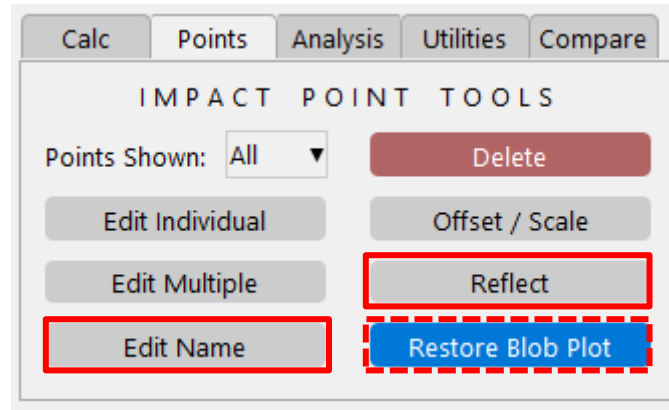
- New functionality:
  - **Contour HIC** | Produces a contour plot of HIC values. Previously this functionality was termed 'Banded Output'.
  - **HIC Iso Plot** | Marks on area/contour plot regions of constant HIC.
  - HIC Iso Points can be output from Utilities tab.



# Points & Utilities Tabs

## Points

- **Edit Name** | The first letter of point names dictate the type of point:
  - C = Child
  - A = Adult
  - B or W = Cyclist
- **Reflect** | reflected points have \_R appended to their name.
- **Restore Blob Plot** has moved from the Utilities tab to the Points tab.



## Utilities

- New functionality to **write**:
  - **Perimeter** data.
  - **HIC Iso Points**
- New functionality to **read**:
  - **Child Wrap Around Line**
  - **Cyclist Wrap Around Line**
- **Read Perimeter File** has moved from the top of the main panel to the utilities tab.

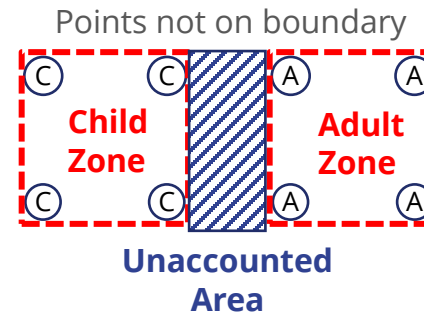


# Wrap Around Lines & Area Calculation Zones

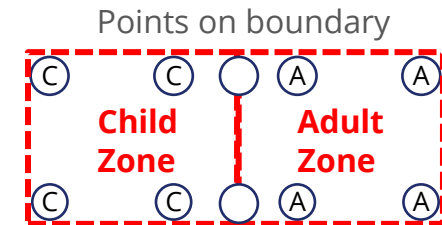
- The HIC Area tool now calculates adult and child areas individually, as well as the combined HIC area, as before.

	# of Points		# Points	Area	>High
	<Low	>Low			
Child	36	37	49.3 %	63.3 %	0
Adult	79	41	65.8 %	71.1 %	5
Combined	115	78	59.6 %	68.5 %	5

- The area considered for each individual zone is found by 'shrink wrapping' the relevant points. This works well when points are located along boundary wrap around lines but not as well if they do not, since this can result in an 'unaccounted area'.

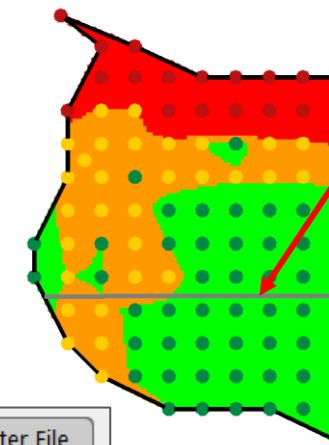
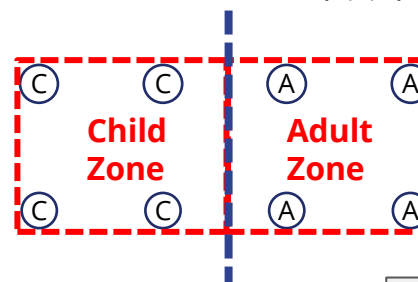


**'Shrink Wrapped' Areas**

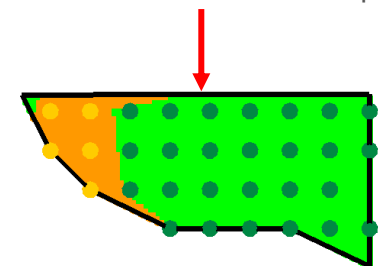


- To facilitate the case where points are not on boundary lines, the tool can now read Child (and Cyclist) wrap around lines (WAL). When present the WAL is used to determine the boundary, not shrink wrapping.

Child WAL Defined (X,Y,Z)



WAL shown on full plot and used to define area (rather than shrink wrap).



Read Perimeter File

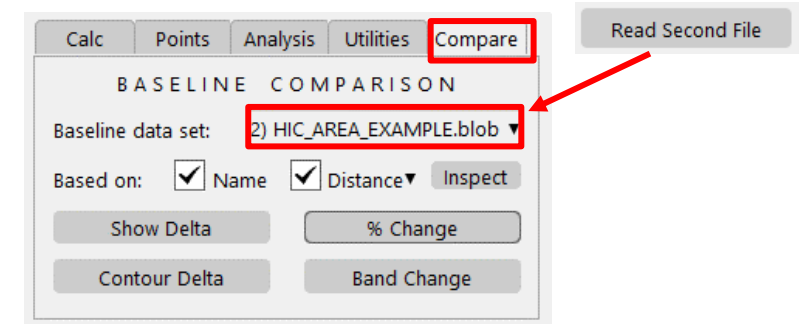
Read Child WAL

Read Cyclist WAL

Points Shown: Child ▼

# Compare Tab

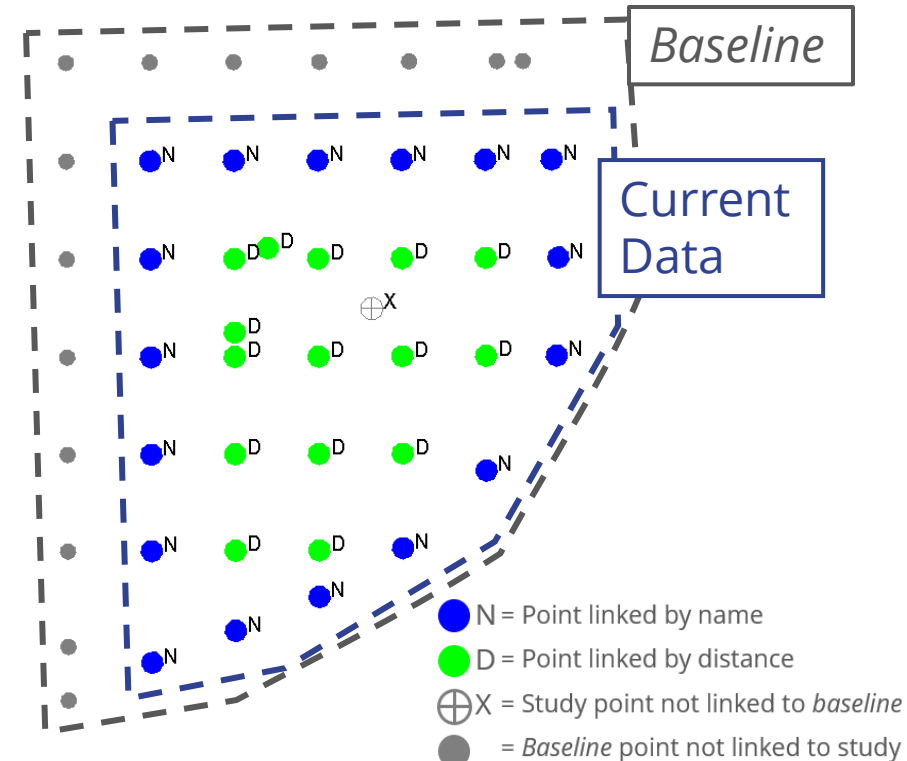
- New tab for comparing one set of HIC results with another.
- You must first read a second file before selecting it from the menu.
- Note: the *baseline* data is assumed to be the previous result.



Baseline → Current HIC Data  
Data selected from Compare menu      Main data read in to session

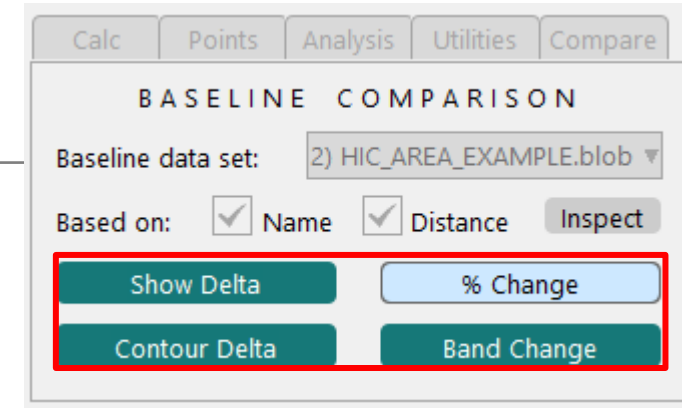
- The current and *baseline* data are compared based on their names and proximity to each other (name takes precedence over distance).
- The **Inspect** button allows you to check how the points have been matched.

**Note: the tool does not currently check for duplicate point names, it is the users responsibility to ensure point names are unique.**

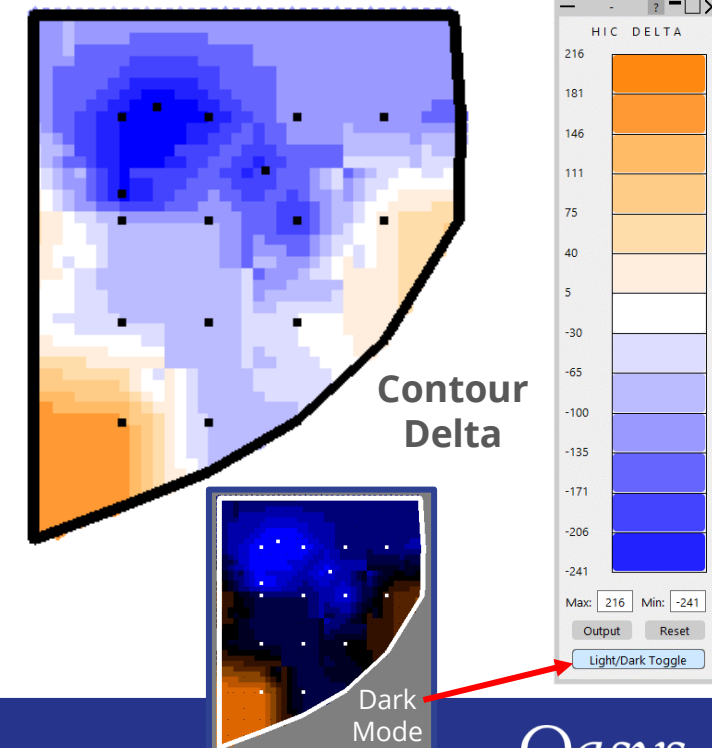
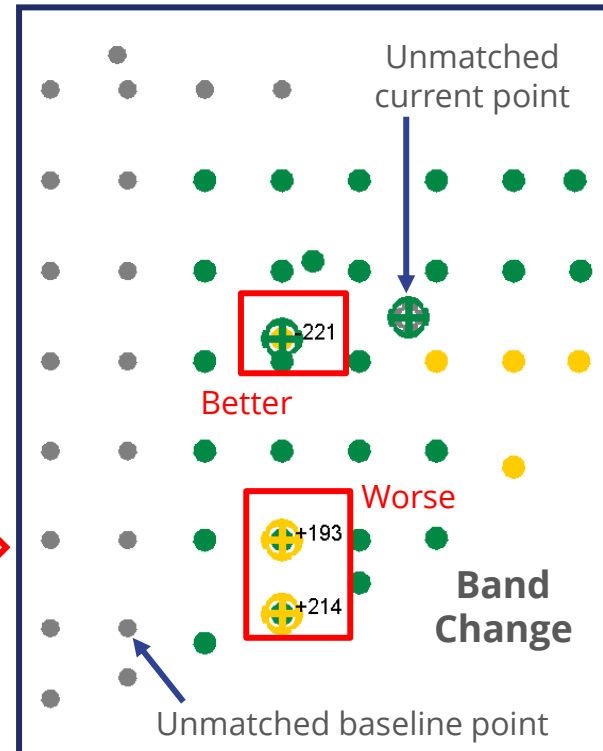
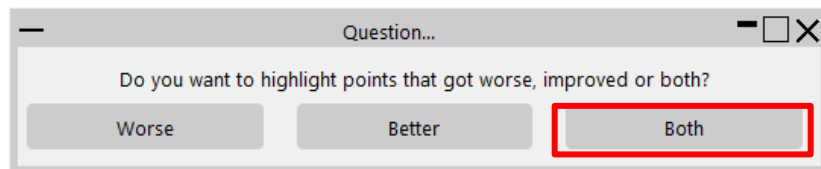


# Compare Tab

- When linked, the difference (delta) between the current and *baseline* can be calculated (current – *baseline*).
- **Show Delta** | plots the HIC difference on screen.
- **% Change** | presents the difference as a %.
- **Contour Delta** | maps the difference as a contour surface
- **Band Change** | highlights points that have changed banding.



**Show Delta** plot (absolute values). Points shown in grey were not matched to *baseline*.



# Pedestrian Run Builder

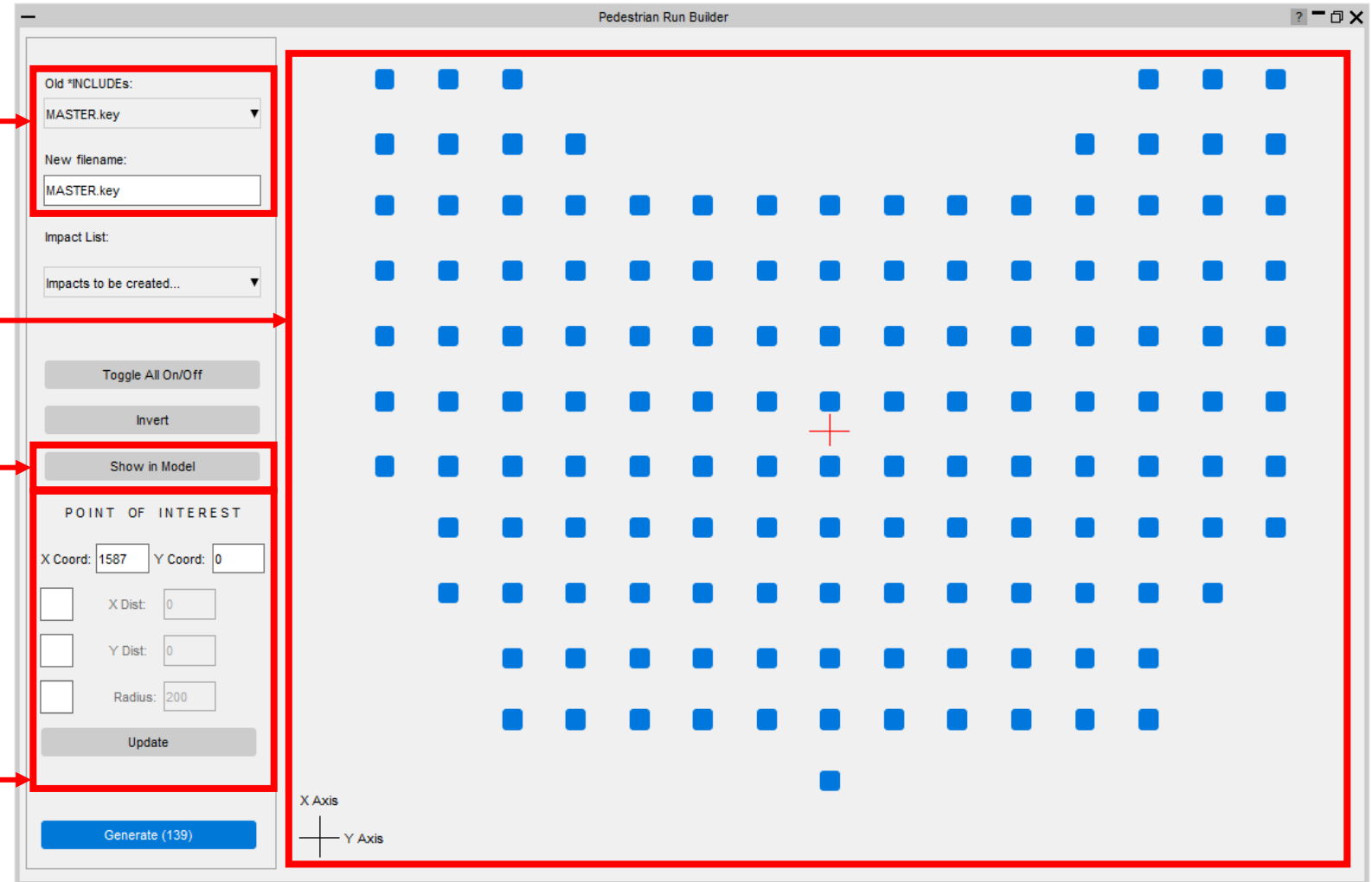
# Impact Substitution

---

- This tool creates a new set of impactor analysis models based on an existing set.
- Allows you to easily update include files referenced in the original models in an interactive way.
- Removes the requirement to manually update individual include references across many files.

# Interactive Impact Point Selection

- Easy changing of include file names.
- Interactive Impact points.
  - All of the impacts displayed in a grid according to X/Y coordinate.
- Displays the impact points in a model.
- Select points via their coordinates.

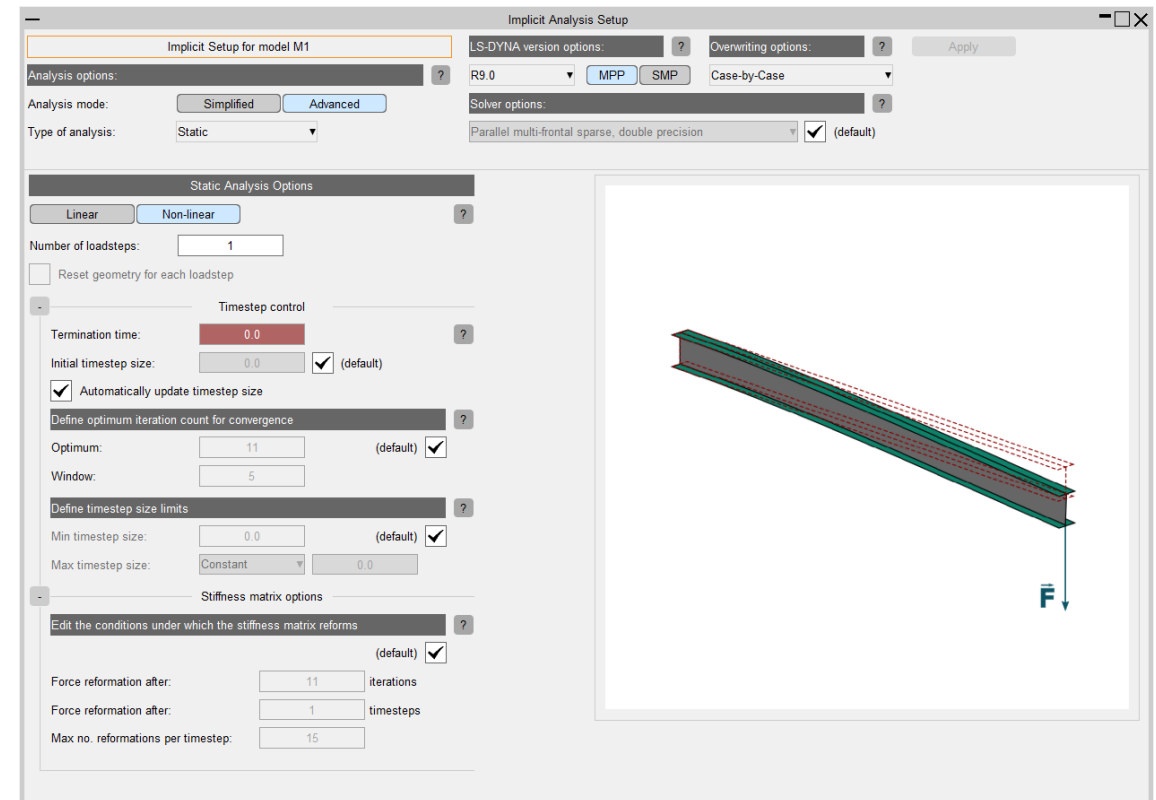


# Implicit Analysis Setup Tool

# Implicit Analysis Setup Tool

Generate the required \*CONTROL cards for a selection of Implicit analyses:

- **Static;**
- **Transient Direct;**
- **Transient Modal;**
- **Buckling** (standalone or intermittent);
- **Eigenvalue** (standalone or intermittent);
- **Frequency Domain FRF.**





# Implicit Analysis Setup Tool

---

The setup tool has two configuration modes:

## **Simplified:**

- Requires less input.
- Based on a default analysis.



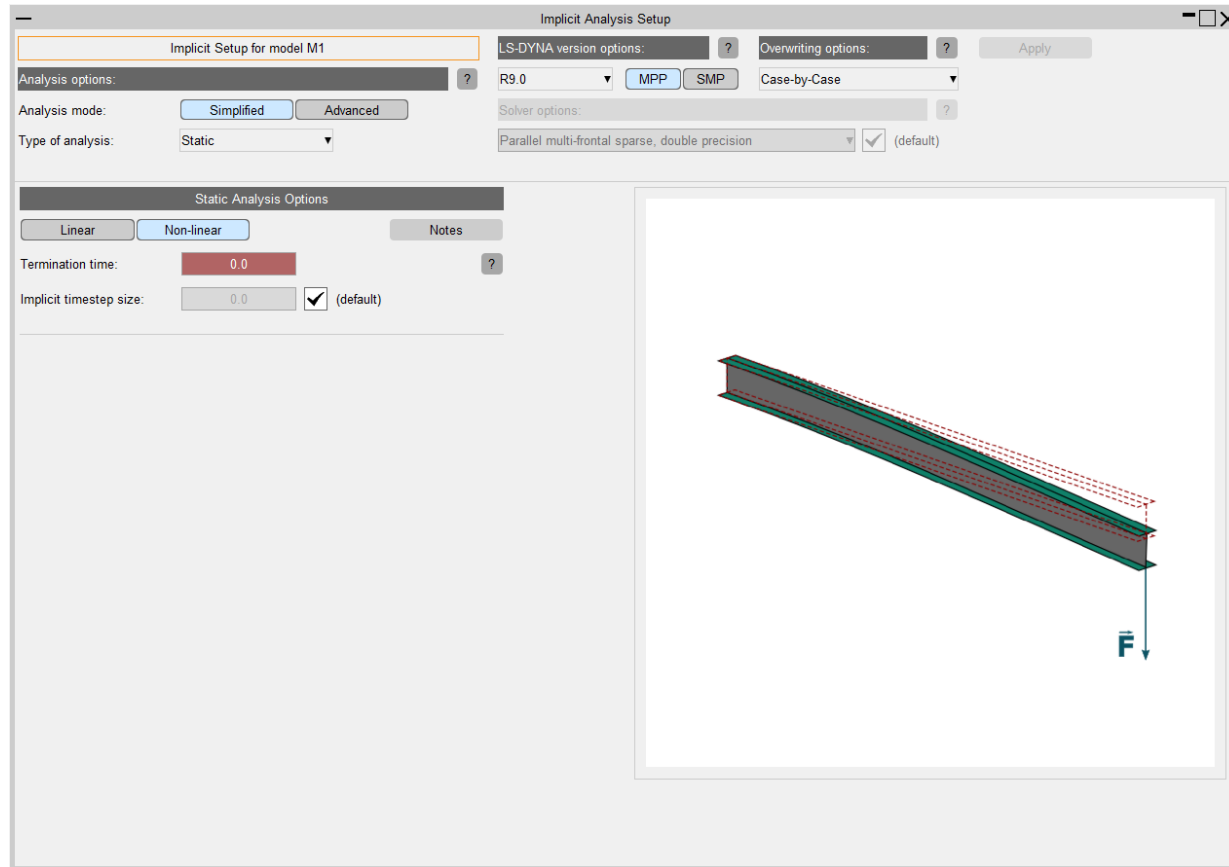
The screenshot shows the 'Implicit Analysis Setup' window. At the top, it says 'Implicit Setup for model M1'. Below this, there are two tabs: 'Simplified' (which is selected and highlighted in blue) and 'Advanced'. To the right of the tabs is a dropdown menu for 'Analysis options' with a question mark icon. Below the tabs, there is a label 'Analysis mode:' followed by the 'Simplified' button and the 'Advanced' button. Below that is a label 'Type of analysis:' followed by a dropdown menu showing 'None selected'. On the right side of the window, there is a section for 'LS-DYNA version options:' with a dropdown menu showing 'R9.0' and two buttons, 'MPP' (highlighted in blue) and 'SMP'. Below this is a section for 'Solver options:' with a text box containing 'Parallel multi-frontal sparse, double precision'.

## **Advanced:**

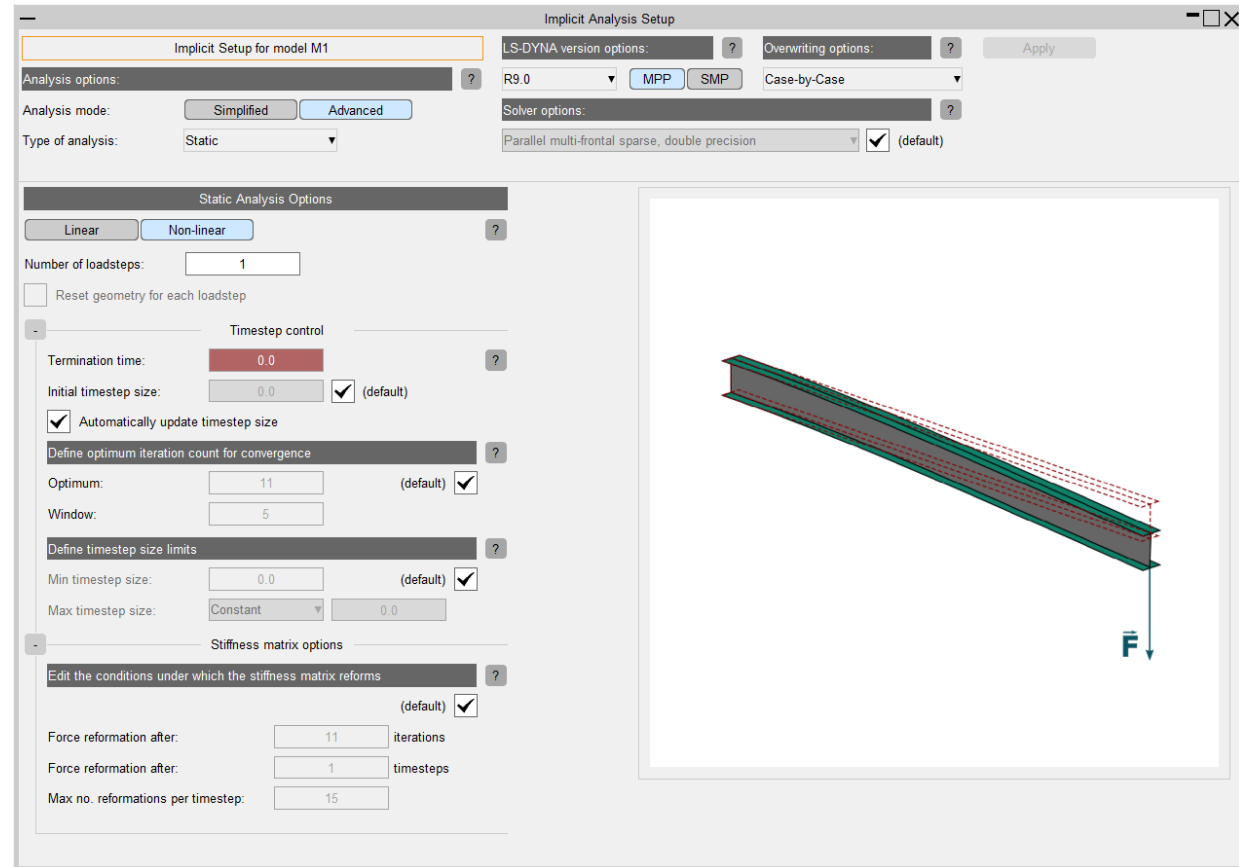
- Allows for more customised analysis settings.

# Implicit Analysis Setup Tool

Simplified mode:

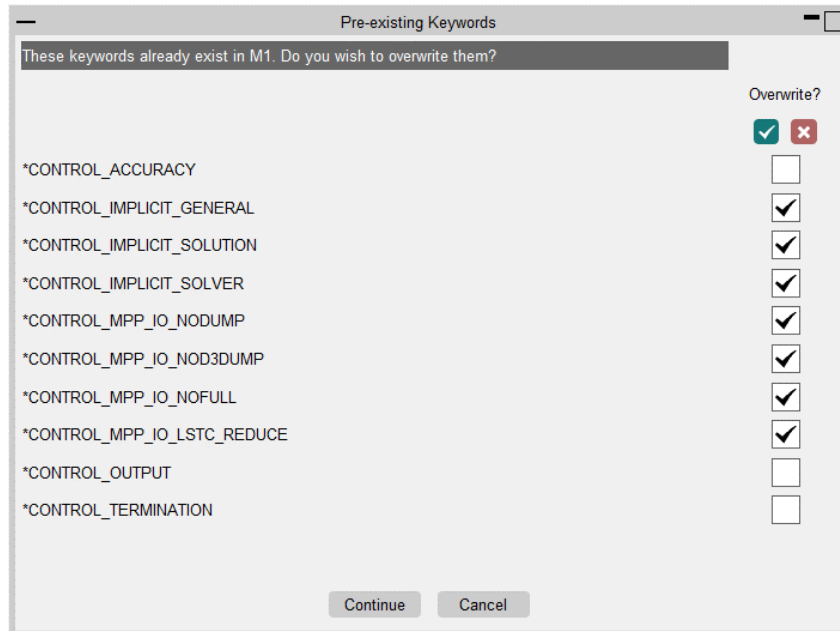


Advanced mode:



# Implicit Analysis Setup Tool

- Choose which analysis type to setup for and then work through the prompts, providing input where required (red textboxes highlight invalid/missing input).
- When finished, click **Apply** to write the changes to the selected model. If some keywords already exist they can be chosen to be overwritten or ignored (Overwriting options – Case-by-Case).

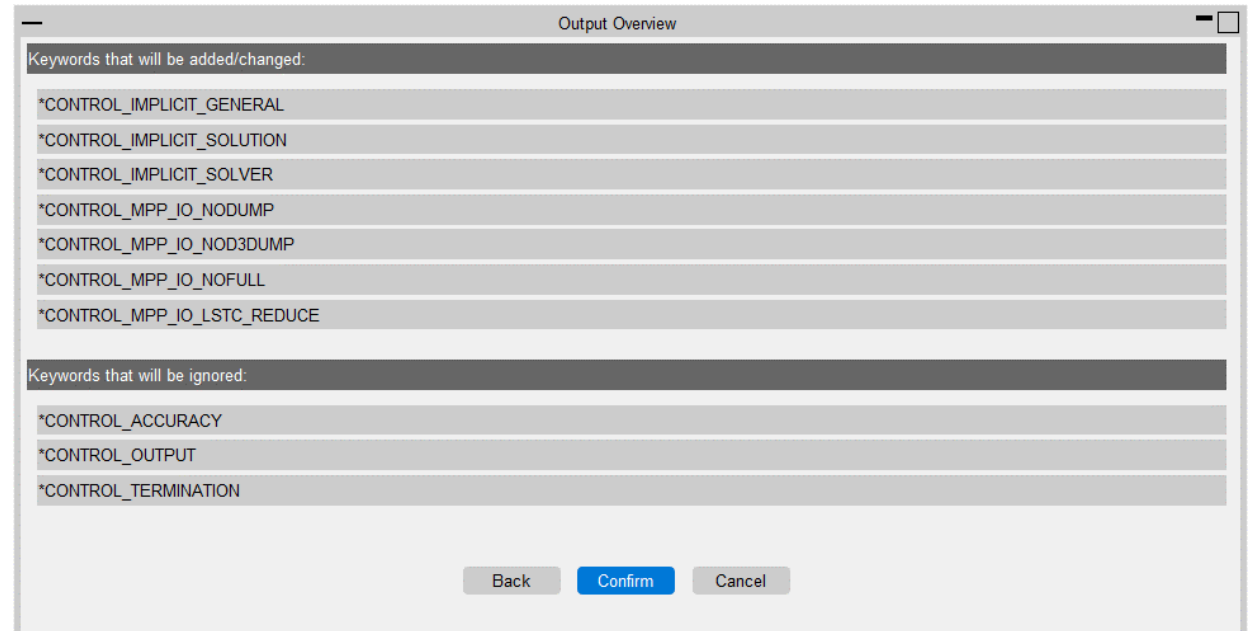


Pre-existing Keywords

These keywords already exist in M1. Do you wish to overwrite them?

	Overwrite?
*CONTROL_ACCURACY	<input type="checkbox"/>
*CONTROL_IMPLICIT_GENERAL	<input checked="" type="checkbox"/>
*CONTROL_IMPLICIT_SOLUTION	<input checked="" type="checkbox"/>
*CONTROL_IMPLICIT_SOLVER	<input checked="" type="checkbox"/>
*CONTROL_MPP_IO_NODUMP	<input checked="" type="checkbox"/>
*CONTROL_MPP_IO_NOD3DUMP	<input checked="" type="checkbox"/>
*CONTROL_MPP_IO_NOFULL	<input checked="" type="checkbox"/>
*CONTROL_MPP_IO_LSTC_REDUCE	<input checked="" type="checkbox"/>
*CONTROL_OUTPUT	<input type="checkbox"/>
*CONTROL_TERMINATION	<input type="checkbox"/>

Continue Cancel



Output Overview

Keywords that will be added/changed:

- \*CONTROL\_IMPLICIT\_GENERAL
- \*CONTROL\_IMPLICIT\_SOLUTION
- \*CONTROL\_IMPLICIT\_SOLVER
- \*CONTROL\_MPP\_IO\_NODUMP
- \*CONTROL\_MPP\_IO\_NOD3DUMP
- \*CONTROL\_MPP\_IO\_NOFULL
- \*CONTROL\_MPP\_IO\_LSTC\_REDUCE

Keywords that will be ignored:

- \*CONTROL\_ACCURACY
- \*CONTROL\_OUTPUT
- \*CONTROL\_TERMINATION

Back Confirm Cancel

# Running LS-DYNA from PRIMER

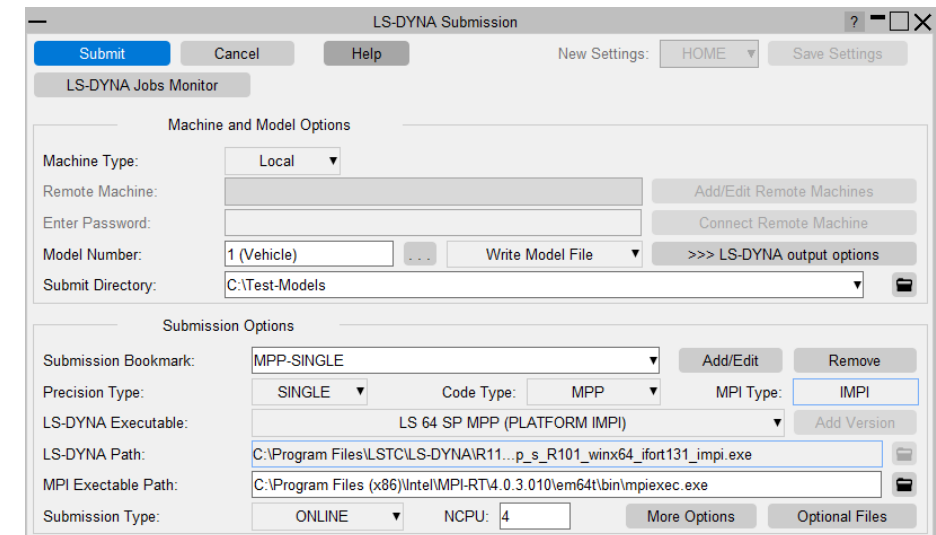
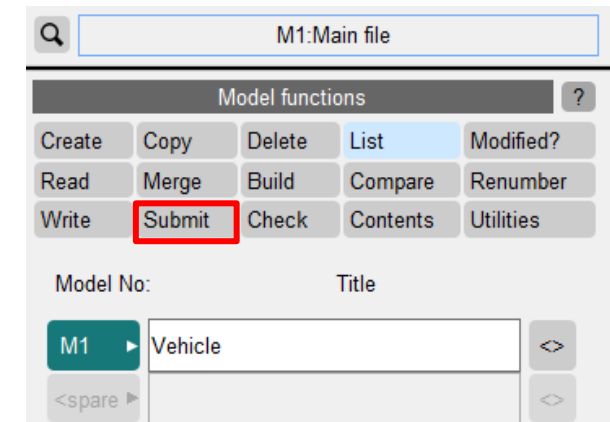
# Running LS-DYNA from PRIMER

---

- This tool allows you to easily submit a model to LS-DYNA directly from PRIMER. The model can be submitted locally or onto a remote machine/cluster
- As well as speeding up the submission process, the functionality allows easy initialisation and model checking with a few clicks
- To aid the above, this functionality has been integrated with PRIMER's LS-DYNA output checking tool to speed up the process of initialisation when checking models and visualising decomposition/load profiles
- This tool uses functionality that exists within SHELL – the Oasys LS-DYNA Environment's submission tool

# Running LS-DYNA from PRIMER

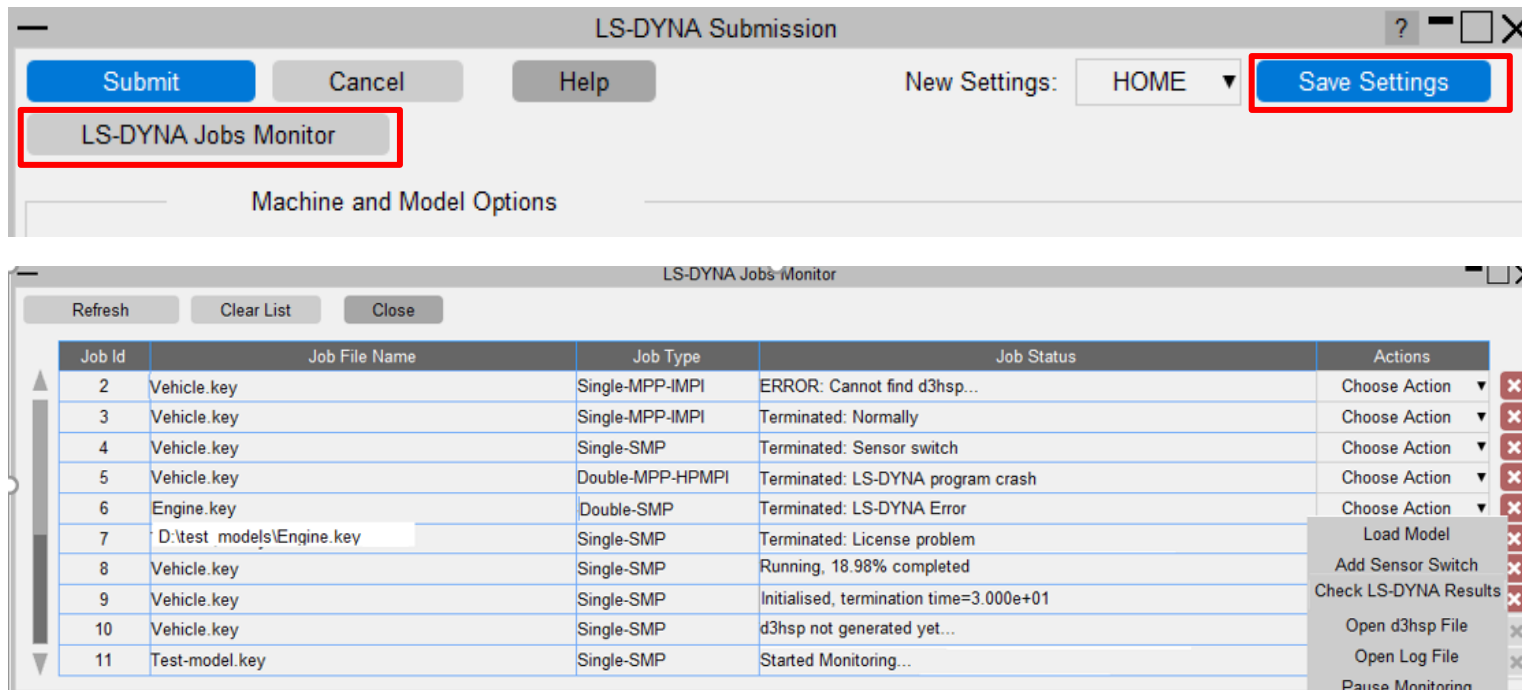
- This tool can be accessed by clicking the Model->Submit button to perform LS-DYNA runs directly from PRIMER
  - “Local” machine: The same windows or the same Linux machine from where the PRIMER session is launched
  - “Remote” machine: Linux machine on a network where LS-DYNA is configured to run
- You can also monitor the progress of “ONLINE” (real time) LS-DYNA runs on a "Local" machine
- You can perform LS-DYNA initialisation of the model in a PRIMER session and view LS-DYNA results relating to errors/warnings/load profile and decomposition via the “LS-DYNA Output Reader” tool in the PRIMER
- Note: A valid LS-DYNA license is required to perform LS-DYNA runs on “Local”/“Remote” machine



# Monitor LS-DYNA jobs

# Monitor LS-DYNA jobs

- PRIMER monitors the progress of ONLINE LS-DYNA runs on a Local machine
- The progress of such jobs can be viewed in the 'LS-DYNA Jobs Monitor' panel
- You can also save the details of such an LS-DYNA run inside the HOME area settings file by pressing the **"Save Settings"** button
  - Later sessions of PRIMER will automatically pick up these jobs from the settings file and you can monitor these jobs by pressing the **"LS-DYNA Jobs Monitor"** button





# Initialise LS-DYNA Analysis

# Initialise LS-DYNA Analysis

- You can now choose to “initialise” a model in a PRIMER session via the “Initialise in LS-DYNA” option under “LS-DYNA Results”
- The model initialisation in LS-DYNA happens via the “LS-DYNA Submission” tool in PRIMER by pressing the “Submit” button.
- Various initialisation options are available
- After the model is submitted to LS-DYNA, PRIMER monitors the LS-DYNA job progress:
  - After the LS-DYNA run is terminated, PRIMER automatically updates this panel with a list of the LS-DYNA output files.
- Once the “Apply” button has been pressed PRIMER will open the “DYNA output tree viewer” and this will display all the errors/warnings and load profiles/decomposition information – prepared from the LS-DYNA output files associated with the input model

Read Dyna

Apply Help

Read LS-DYNA output

Apply to model: 1

LS-DYNA Results: Initialise in LS-DYNA Submit

LS-DYNA directory: C:\Test-Models

Additional search:

Compressed search

Initialisation Options

Decomposition ☒ NUMPROC: 4

Initialise with: NCYCLE NCYCLE: 10

Output Categories ☒ ☒

Error/Warnings ☒ Load Profile ☒

Contact Profile ☒ Mes Profile ☒

Decomposition ☒

Output files found:

☒ Select/Deselect all

☒ cont\_profile.csv

☒ decomp\_parts.ses

☒ load\_profile.csv

☒ mes0000

☒ mes0001

☒ mes0002

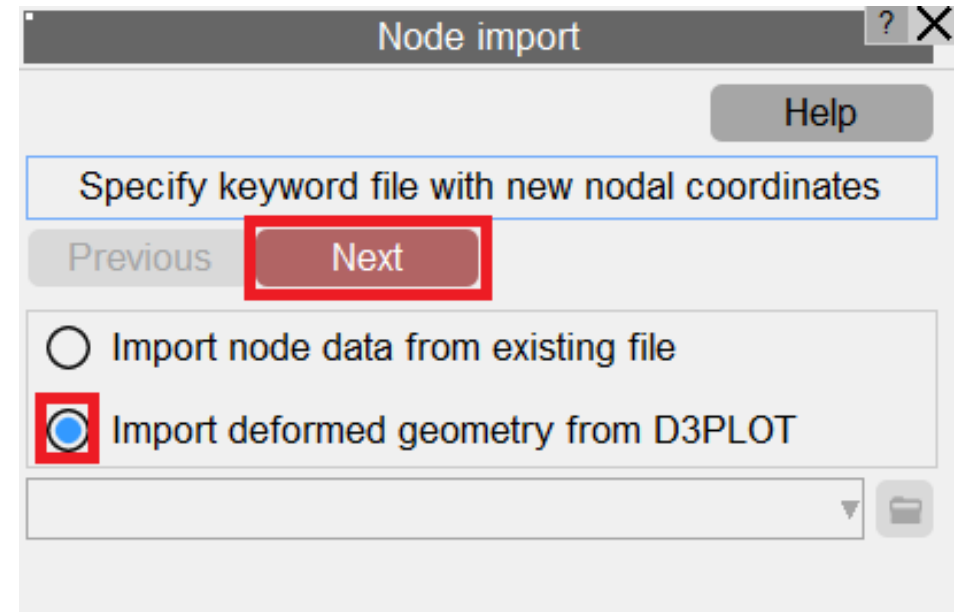
☒ mes0003

☒ Vehicle.otf

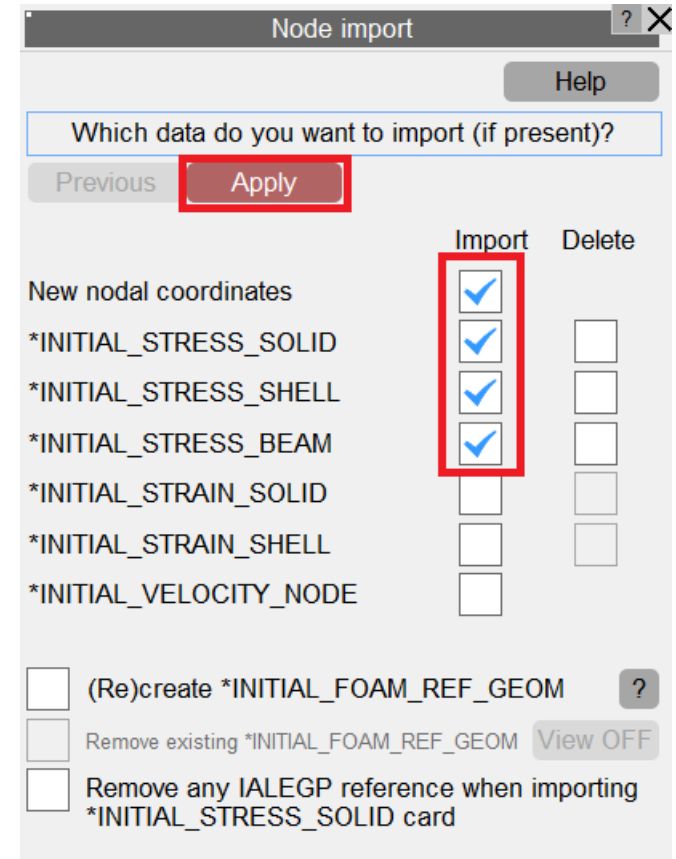
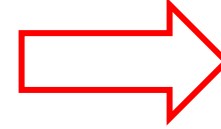
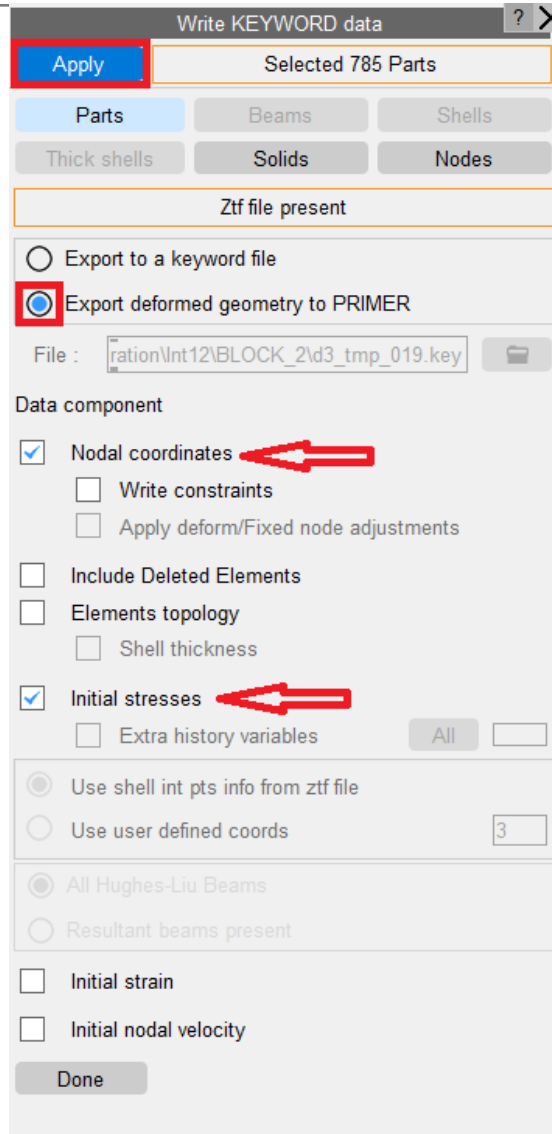
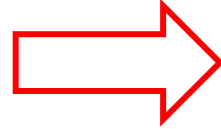
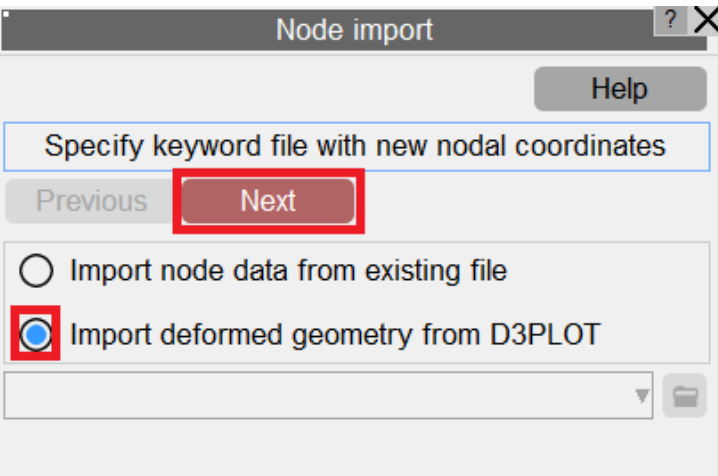
# Import Geometry from D3PLOT

# Import Geometry from D3PLOT

- The “Node import” tool in PRIMER now has an option to import deformed geometry directly from D3PLOT.
- Select the **Import deformed geometry from D3PLOT** option and click **Next**. A linked D3PLOT session will open and a ‘Write’ panel will be displayed in D3PLOT.
- Select the required options for the data that needs to be imported (nodal coordinates, initial stresses, initial strains, etc) from the panel and click **Apply**.
- The data will be sent to PRIMER and the “Node import” panel will display the selection of data from D3PLOT. Click **Apply** to import the data.

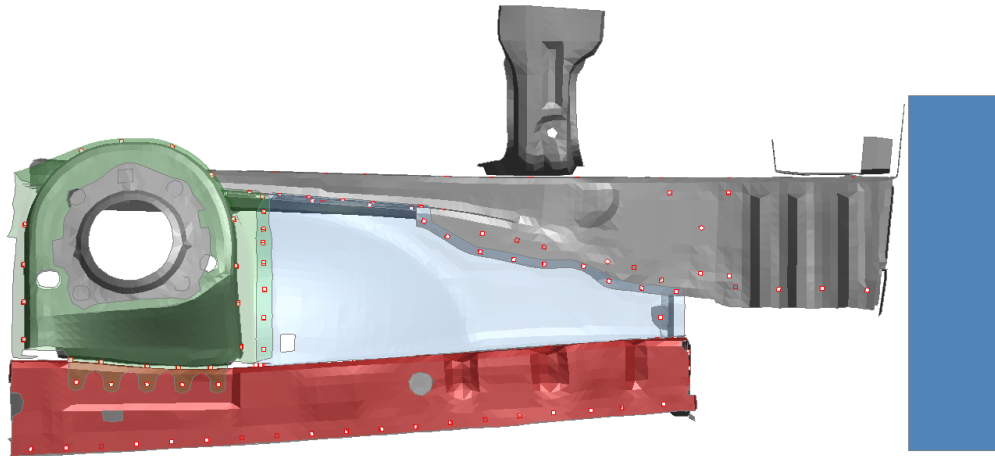


# Import Geometry from D3PLOT

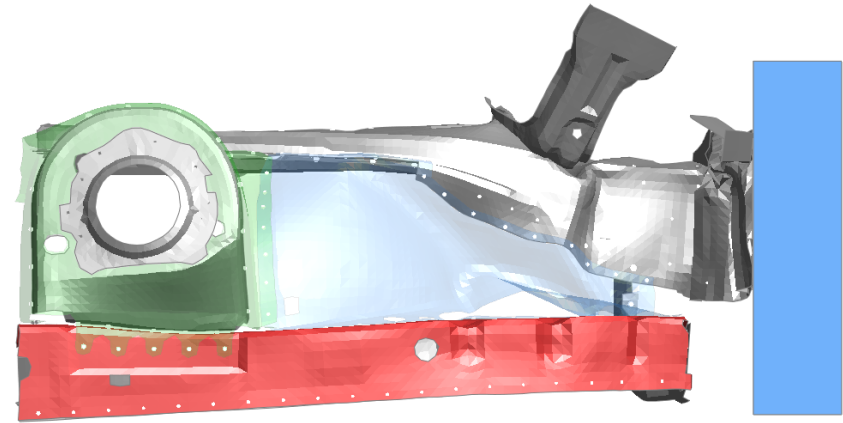
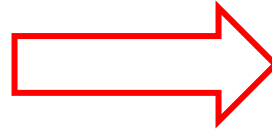


# Import Geometry from D3PLOT

---



Original model in PRIMER



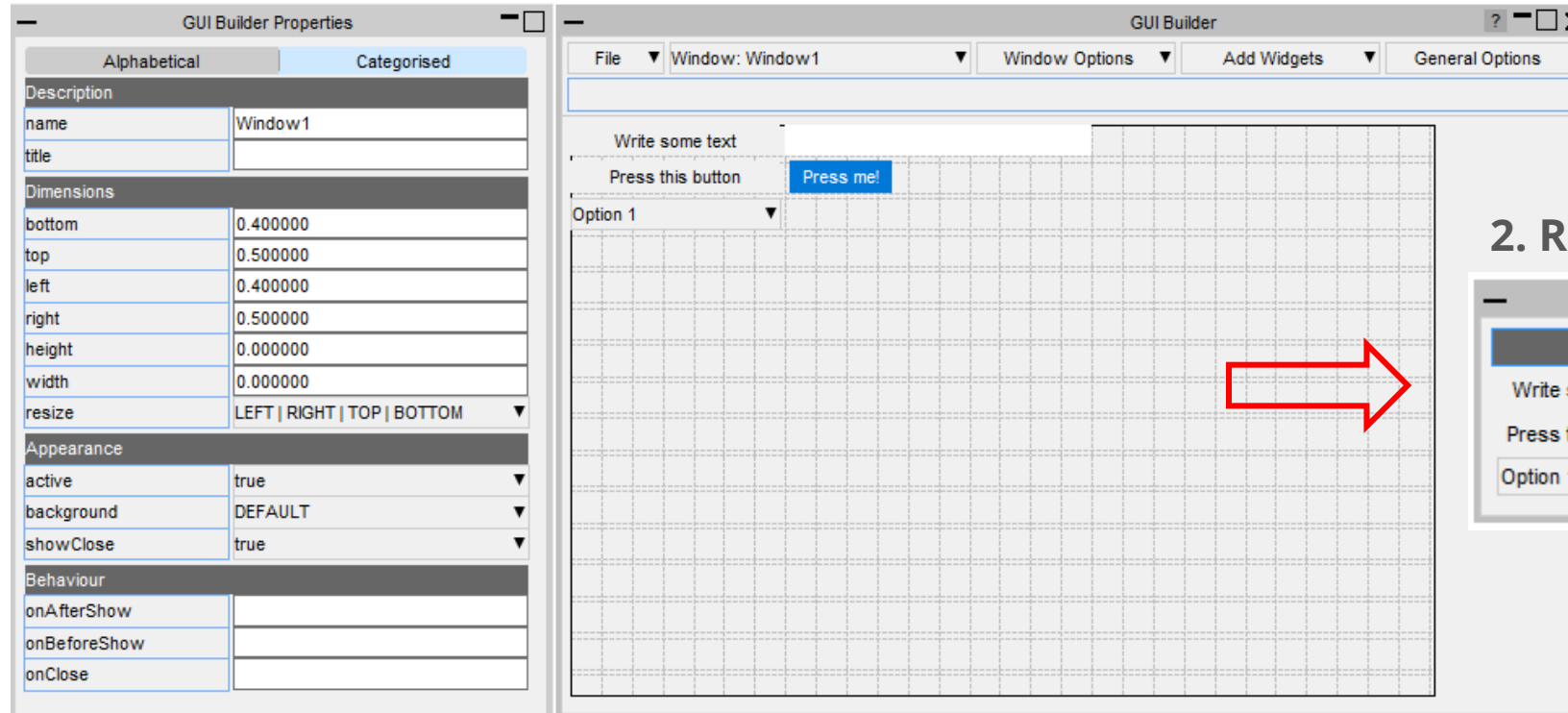
Imported deformed geometry in PRIMER  
(original model is modified)

# JavaScript GUI Builder

# JavaScript GUI Builder

An interactive GUI Builder has been added to PRIMER, D3PLOT and T/HIS to make it easier to build JavaScript GUIs, removing the need to write code to create windows and widgets.

## 1. Design and Save your GUI to a file



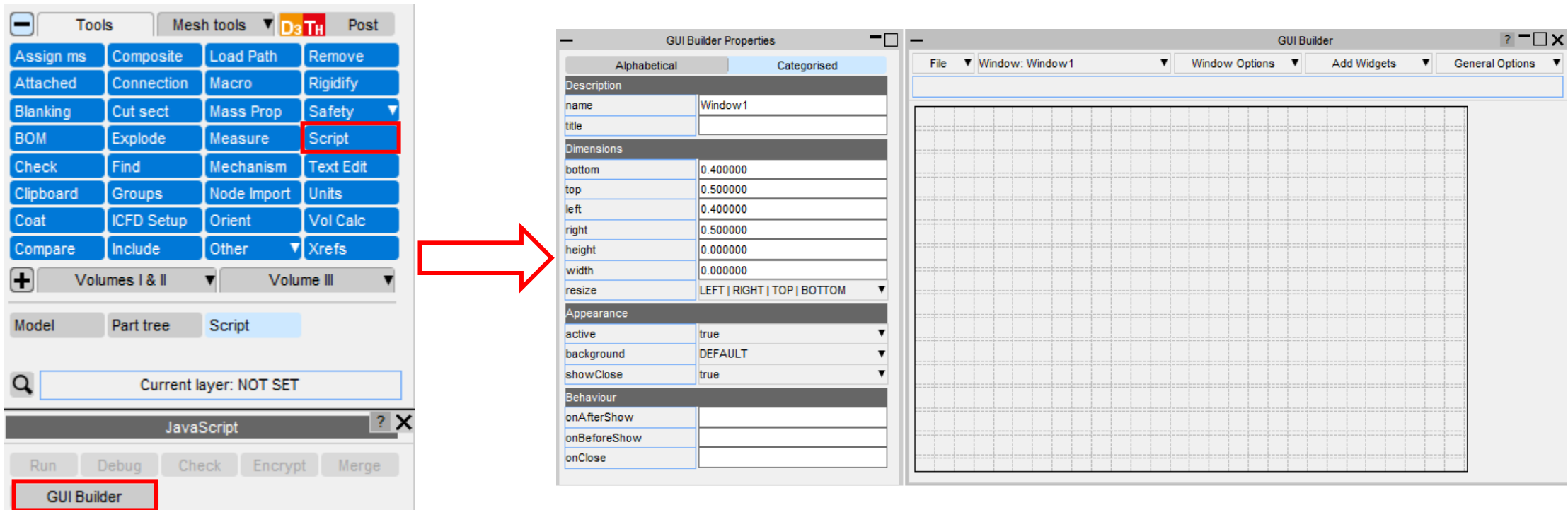
## 2. Read the file in your script





# JavaScript GUI Builder

To open the GUI Builder in PRIMER go to *Script* → *GUI Builder*



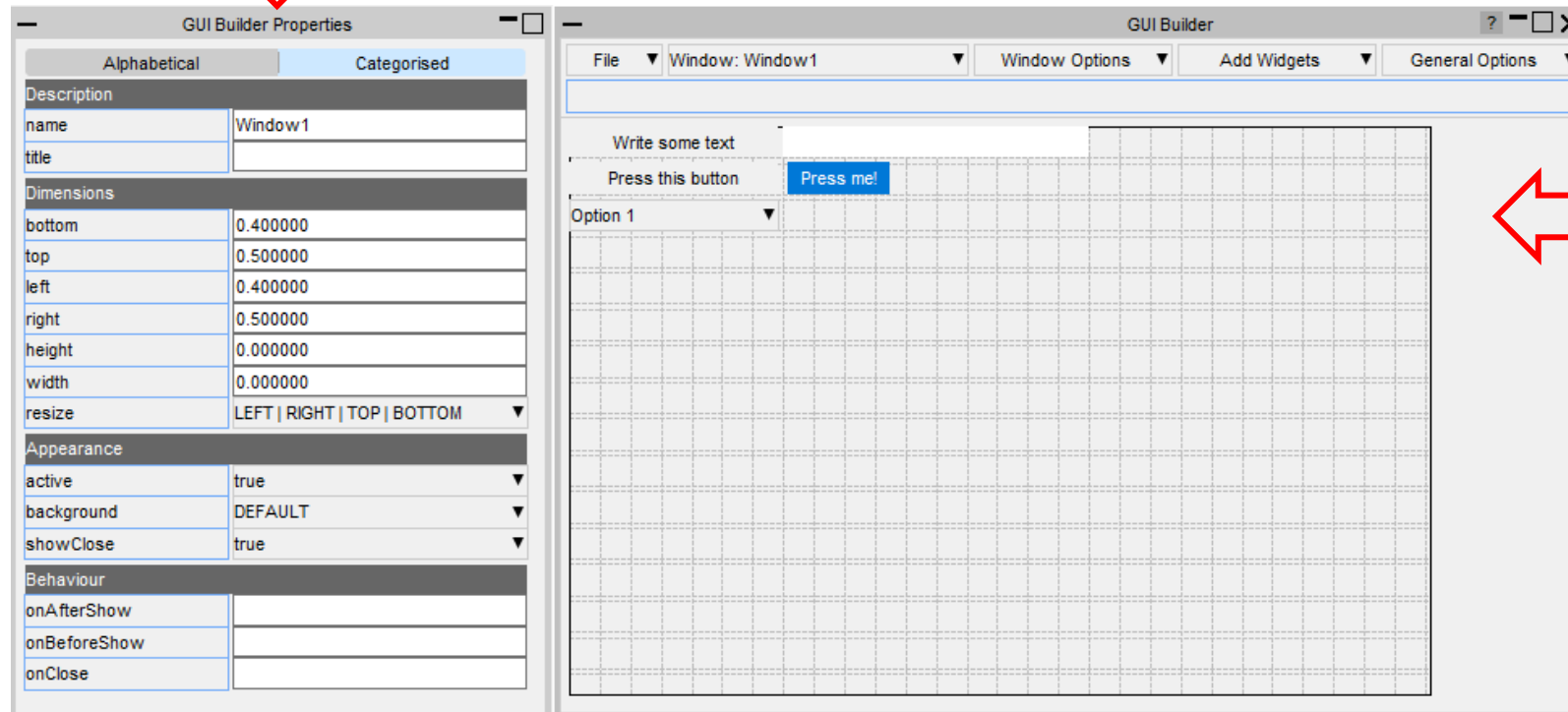
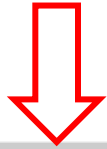
# JavaScript GUI Builder

How to use the GUI Builder to build a GUI

# JavaScript GUI Builder

## Properties Window

The properties of widgets and windows are set here.



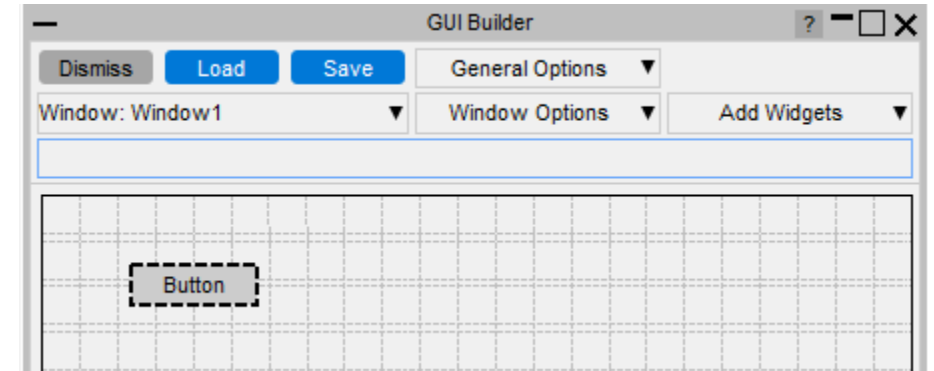
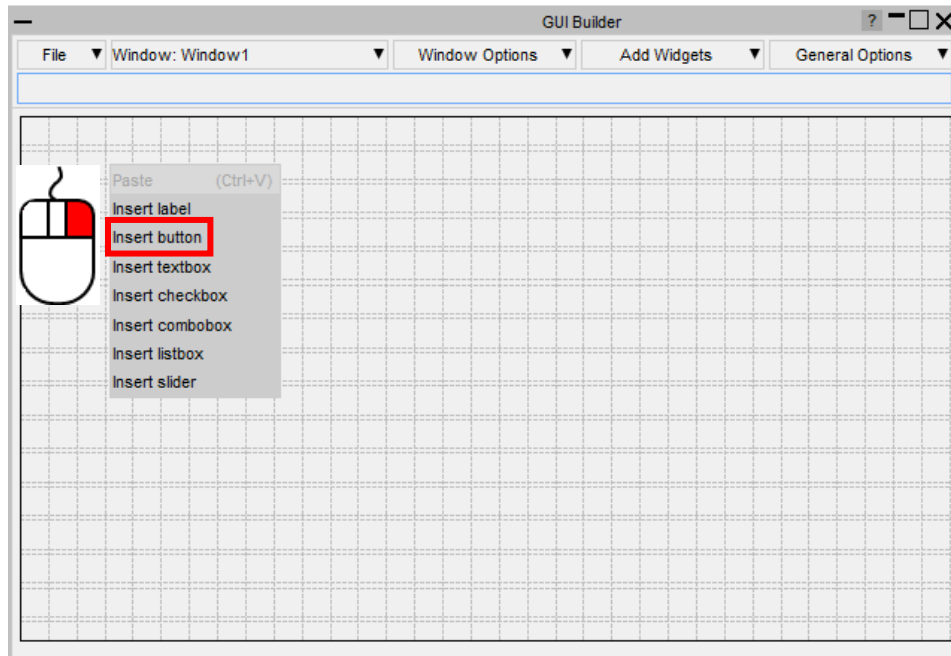
## Design Window

Widgets are added, positioned and resized here.



# JavaScript GUI Builder

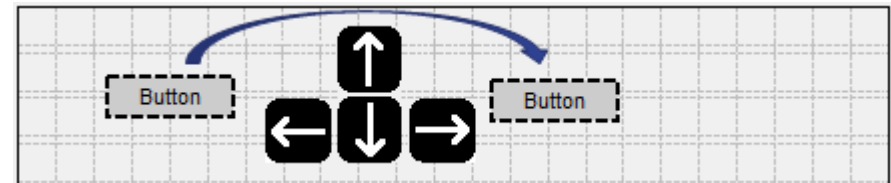
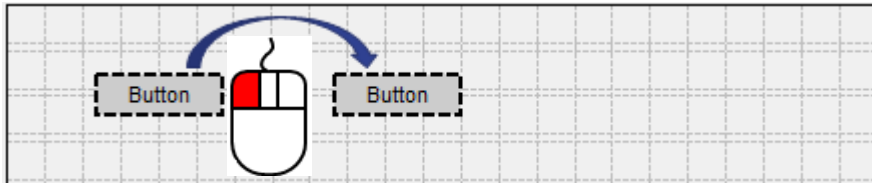
Widgets can be added by right-clicking on the design window and selecting the widget type to add.



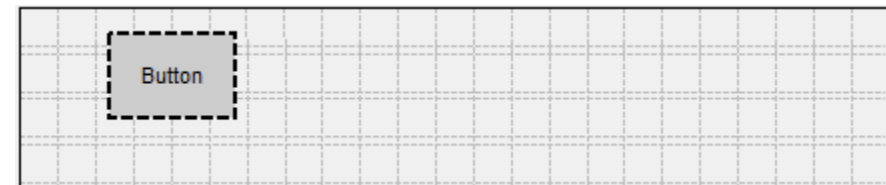
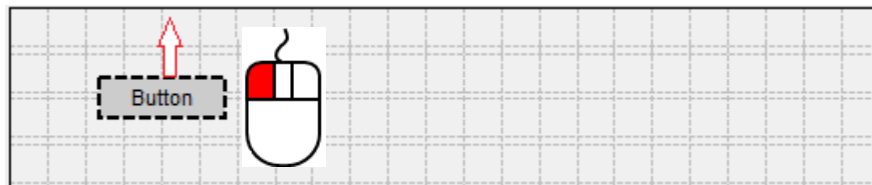
The widget will be added with default properties and highlighted with dashed lines to indicate that it's the current widget.

# JavaScript GUI Builder

Widgets can be moved by left-clicking on them and dragging, or by using arrow keys.



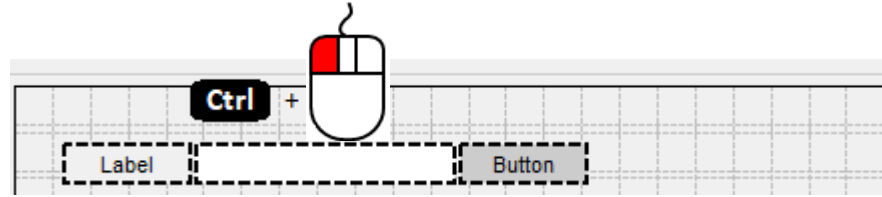
They can be resized by left-clicking on their border and dragging



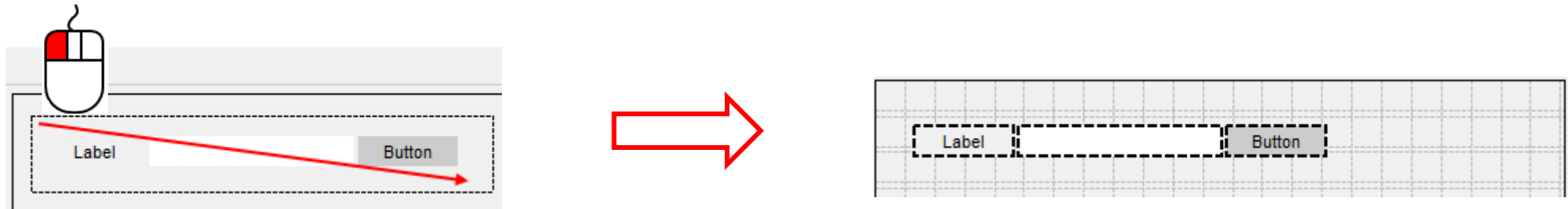
# JavaScript GUI Builder

---

Multiple widgets can be selected by holding the Ctrl or Shift keys and left-clicking.

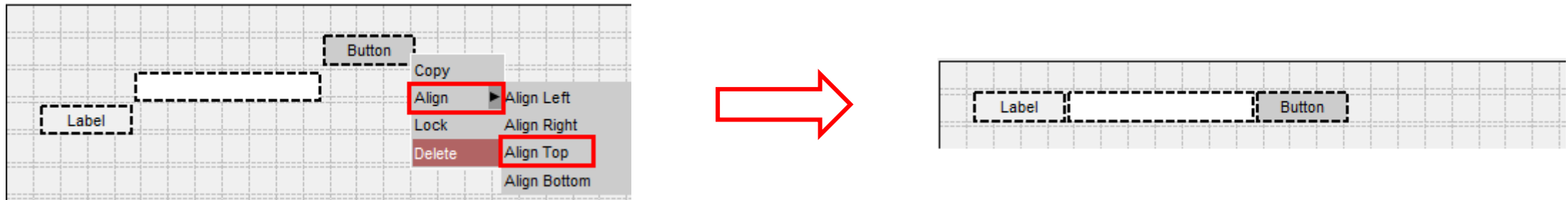


Alternatively a box can be dragged around the widgets you want to select.



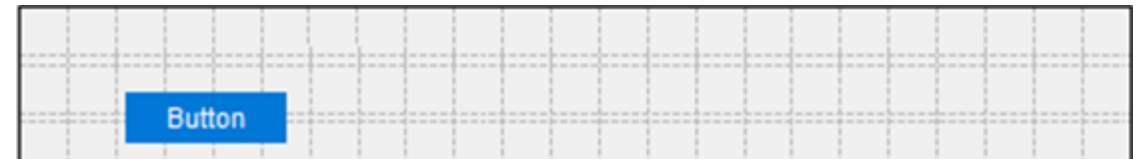
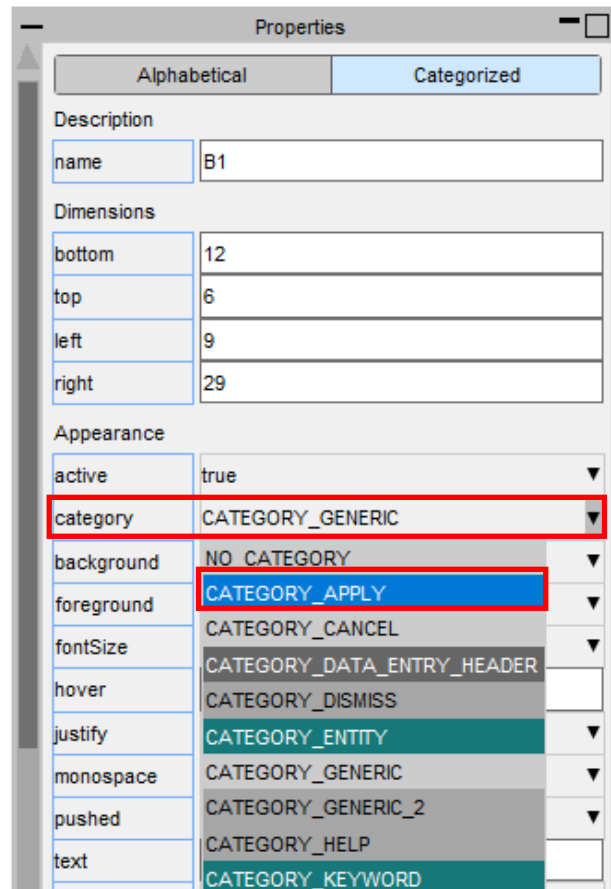
# JavaScript GUI Builder

When multiple widgets are selected the borders can be aligned by right-clicking on the widget you want to align the other widgets to, and then selecting how you want them to be aligned:



# JavaScript GUI Builder

The properties of a widget can be modified in the properties window, e.g. change the category to CATEGORY\_APPLY:



The appearance of the widget will update in the design window.

If multiple widgets are selected the property will be applied to all the selected widgets.

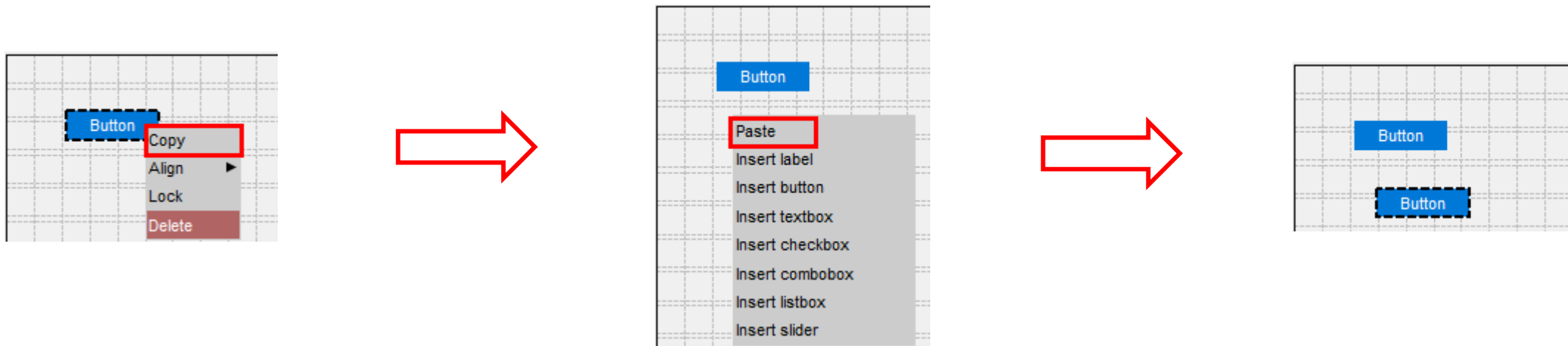


# JavaScript GUI Builder

---

You can copy and paste widgets by right-clicking on them and selecting 'Copy' and then right-clicking on the window and selecting 'Paste'. The new widget will have all the same properties as the copied widget.

Alternatively you can use the shortcuts Ctrl-C and Ctrl-V.

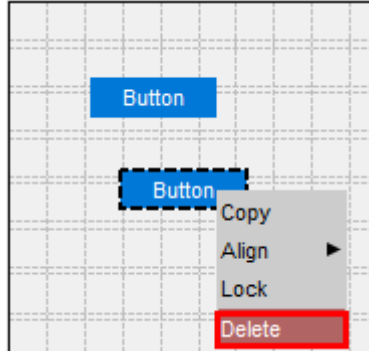


# JavaScript GUI Builder

---

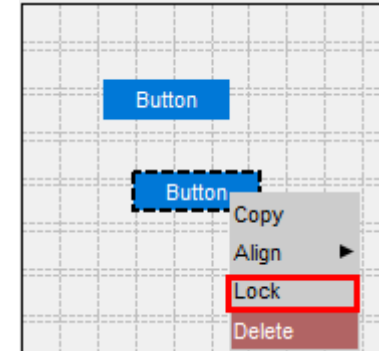
To delete a widget, right-click on it and select 'Delete'.

Alternatively you can press the Delete shortcut key.



To lock the position of a widget so it can't be repositioned or resized, right-click on it and select 'Lock'.

To unlock it again, right-click on it and select 'Unlock'.

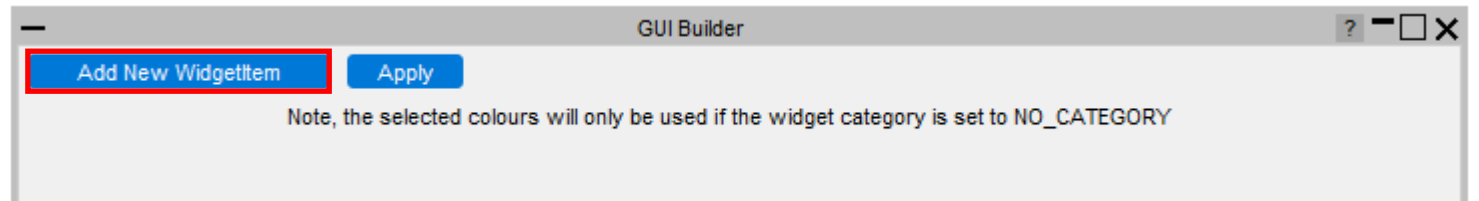
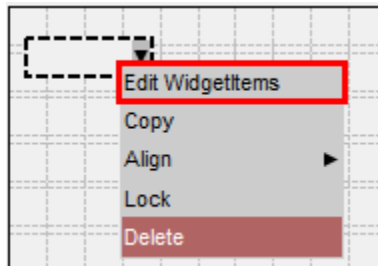


# JavaScript GUI Builder

---

To add WidgetItems to a Combobox or Listbox, right-click on it and select 'Edit WidgetItems'.

This will update the design window where you can add WidgetItems by pressing the 'Add New WidgetItem' button.

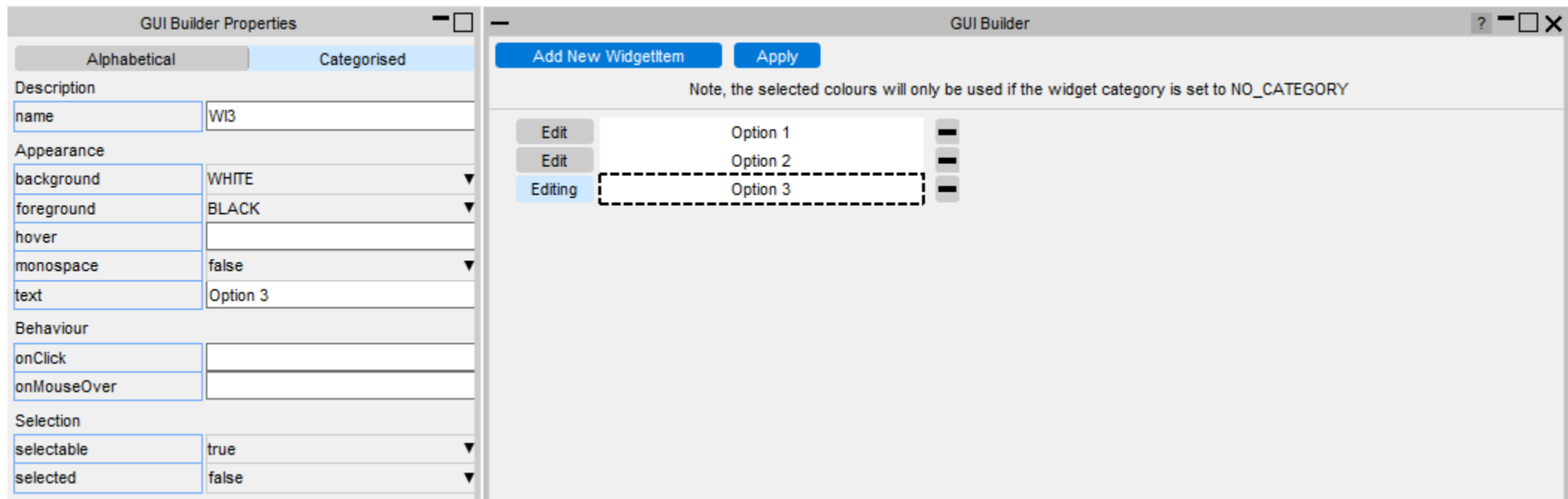


# JavaScript GUI Builder

The appearance of the current WidgetItem can be modified in the same way as Widgets by clicking on the WidgetItem and updating its properties.

To delete a WidgetItem, click on the '-' on the right hand side.

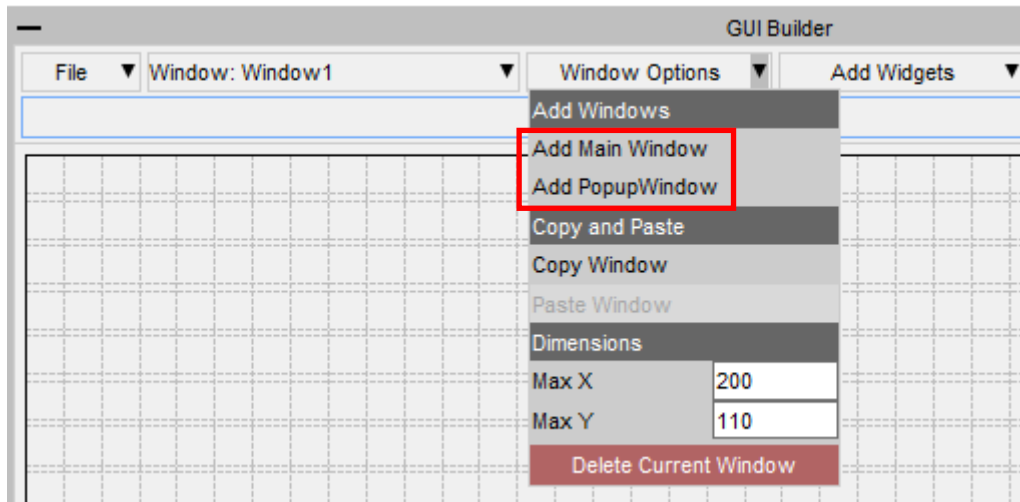
Once you have finished, press 'Apply' to return to the normal design window.



# JavaScript GUI Builder

Additional windows can be created by clicking on the Window Options dropdown menu.

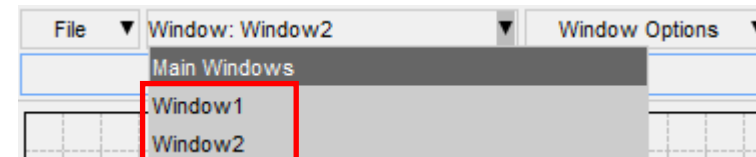
You can add either a Main Window or PopupWindow.



The name of the current window is displayed in the Window selection dropdown menu.




To change to a different window, select it from the dropdown menu.




# JavaScript GUI Builder

PopupWindows can be linked to widgets by setting the popupWindow property.

Dimensions	
bottom	14
top	8
left	8
right	28
Popups	
popupDirection	BOTTOM
popupSymbol	true
popupWindow	<no popup>
Timer	PopupWindow1
timerDelay	PopupWindow2
timerRepeat	<no popup>
Misc	
macroTag	



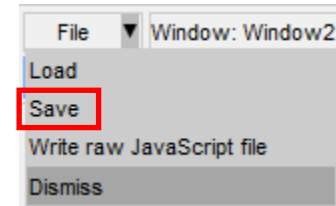
Dimensions	
bottom	14
top	8
left	8
right	28
Popups	
popupDirection	BOTTOM
popupSymbol	true
popupWindow	PopupWindow1
Timer	
timerDelay	1
timerRepeat	false
Misc	
macroTag	



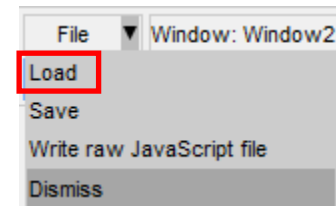
To remove a PopupWindow linked to a widget, set the popupWindow to <no popup>.

# JavaScript GUI Builder

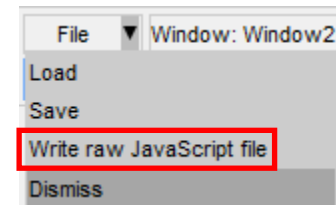
The GUI can be saved to file by pressing the 'Save' button and then selecting a file. The saved file is a JavaScript file containing the window and widget definitions in a JSON string, and a call to `Window.BuildGUIFromString()` which builds the GUI when the script is run. Further details are given in the next few slides.



It can be reloaded by pressing the 'Load' button and selecting the file to load.



The GUI can also be saved as a raw JavaScript file, with the calls to create and position the windows and widgets, explicitly defined, rather than using `Window.BuildGUIFromString()`. This cannot be loaded back into the GUI Builder, however it may be useful for creating GUI's to run in versions prior to v18 that don't have the `Window.BuildGUIFromString()` function.



# JavaScript GUI Builder

How to use the GUI in a script



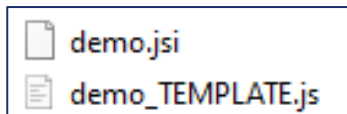
# JavaScript GUI Builder

---

The GUI is saved to a JavaScript file, containing the GUI definition in a JSON string and a call to `Window.BuildGUIFromString()`. It is saved with the extension `.jsi` to indicate that it should be included from another file. You should not need to edit this file.

A `*.js` file is also written to demonstrate how to include the `*.jsi` file and display the GUI. This can be used as a template to follow and modify.

It is written to the same folder as the `*.jsi` file and named `'<jsi_filename>_TEMPLATE.js'`, e.g. if the `*.jsi` file is called **`'demo.jsi'`**, the `*.js` file will be saved as **`'demo_TEMPLATE.js'`**.



The following slides explain what is in the file and how you can reference the Windows, Widgets and WidgetItems in the script.

# JavaScript GUI Builder

---

To read the GUI in a script you need to include the file using the Use() function.

This will create a global variable ('gui' by default) containing all the GUI objects. The name of the variable can be changed in the GUI builder menu under General Options.

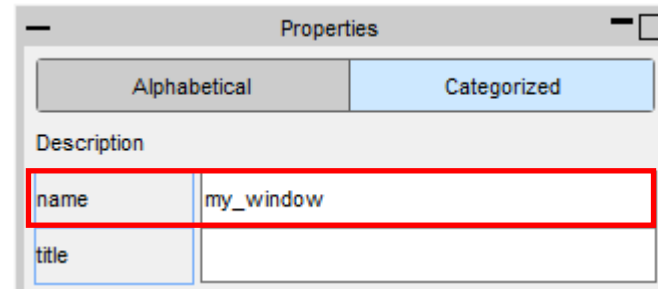
For example, to build the GUI saved in C:\my\_gui.jsi:

```
Use("C:\\my_gui.jsi");
```

# JavaScript GUI Builder

---

The GUI Window objects are stored as properties on the global object. The name of the property is whatever was defined in the properties window in the GUI builder:



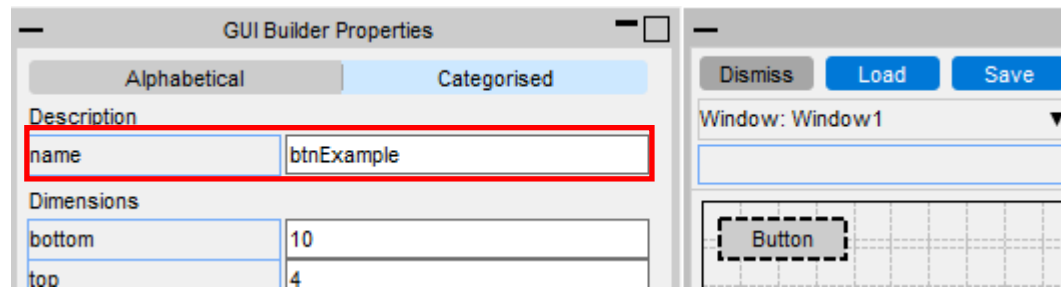
To display the Window called 'my\_window' use the Show() method:

```
if (gui) gui.my_window.Show();
```

# JavaScript GUI Builder

---

Similarly, each Widget object is a property of the Window object. The name of the Widget property is whatever was defined in the properties window in the GUI builder:



For example if the window is called 'my\_window' and the widget is called 'btnExample', the Widget object can be accessed and modified with:

```
var btn = gui.my_window.btnExample;  
  
btn.text = "Test";
```

# JavaScript GUI Builder

---

WidgetItem objects are a property of the Widget.

For, example if the window is called 'my\_window', the widget the widget item is on is called 'cbxExample' and the widget item is called 'wi1', it can be accessed and modified with:

```
var wi = gui.my_window.cbxExample.wi1;
```

# JavaScript GUI Builder

---

Callback functions (onClick, onChange, etc.) can be assigned to the window and widgets in the properties window, by adding the name of a function to call.

For example to set the onClick property of a widget so it calls a function called 'pressed':

Functions	
onClick	pressed
onPopup	
onTimer	

This function then needs to be defined in your script:

```
Use("C:\\test.jsi");  
  
if (gui) gui.my_window.Show();  
  
function pressed()  
{  
    Message("You clicked me!");  
}
```

# JavaScript Engine Upgrade

# JavaScript engine upgrade

---

- For PRIMER 18.0 the JavaScript engine used in PRIMER has been significantly upgraded.
- In PRIMER 17.0 and earlier the engine only supported [ECMAScript 5](#) features.
- In PRIMER 18.0 the engine now supports [ECMAScript 6](#) (ES6) and many newer features.
  - The engine we use is [Spidermonkey](#) provided by Mozilla from the Firefox web browser.
  - For PRIMER 18.0 we are now using the current 'Extended Support Release' version (ESR78)
  - Future releases will continue to use the latest ESR version available.



# JavaScript engine upgrade

---

- The primary reason for upgrading is to give access to newer JavaScript features
- In some cases newer JavaScript code people obtained/learned from books and/or the web and tried to use in PRIMER did not work in PRIMER 17.0 as we only supported ECMAScript 5.
- Upgrading the engine allows the latest ECMAScript 6 (ES6) language features to be used.
  - Which ES6 (and newer) features are supported by the engine can be viewed at <http://kangax.github.io/compat-table/es6/#firefox78>
- Additional benefits to upgrading as well as ES6 support are outlined on the following slides.

# JavaScript engine upgrade – ES6 features

---

- Upgrading the JavaScript engine gives access to lots of significant new ES6 (and newer) language features such as
  - [class](#) keyword
  - Block scope with [let/const](#)
  - [Promises](#)
  - [Arrow functions](#)
  - [Default parameters](#), [rest parameters](#) and [spread syntax](#)
  - [Set](#) and [Map](#)
  - [Iterators](#) and [generators](#)
  - [Symbol](#)

And many more

- Further resources are available online or via reference material, e.g. JavaScript: The Definitive Guide. A few examples follow.

# JavaScript engine upgrade – example ES6 features

---

- class keyword
  - ES6 makes it much easier to create classes using the new class keyword and syntax

## ES 5

```
function Circle(radius)
{
    this.r = radius;
}

Circle.prototype.area = function()
{
    return Math.PI * this.r * this.r;
}

var c = new Circle(5);
Message("Area of circle with radius " +
        c.r + " is " + c.area() )
```

## ES 6

```
class Circle
{
    constructor(radius)
    {
        this.r = radius;
    }

    area()
    {
        return Math.PI * this.r * this.r;
    }
}

var c = new Circle(5);
Message("Area of circle with radius " +
        c.r + " is " + c.area() )
```

# JavaScript engine upgrade – example ES6 features

---

- let statement
  - The let statement in ES6 allows you to create variables with block scope (variables declared with var have scope for the containing function which can be a source of bugs)
  - Accessing variables defined with let before they are initialised is an error (helps trap bugs)

## ES 5

```
function test()
{
    Message(x);      // undefined

    var x = 1;
    {
        var x = 2;   // same variable!

        Message(x); // 2
    }

    Message(x);      // 2
}
```

## ES 6

```
function test()
{
    Message(x);      // Error

    let x = 1;
    {
        let x = 2;   // different variable

        Message(x); // 2
    }

    Message(x);      // 1
}
```

# JavaScript engine upgrade – example ES6 features

---

- Spread operator
  - The spread operator expands an array into the list of values in the array. It can be useful when array values are needed in a function.

## ES 5

```
// create a node
var coords = [ 1, 2, 3];
var n = new Node(model, nid,
                  coords[0], coords[1], coords[2]);

// draw a rectangle on a widget
var pt1 = [0, 0];
var pt2 = [100, 100];
widget.Rectangle(Widget.RED, true,
                  pt1[0], pt1[1],
                  pt2[0], pt2[1]);
```

## ES 6

```
// create a node
var coords = [ 1, 2, 3];
var n = new Node(model, nid,
                  ...coords);

// draw a rectangle on a widget
var pt1 = [0, 0];
var pt2 = [100, 100];
widget.Rectangle(Widget.RED, true,
                  ...pt1,
                  ...pt2);
```

# JavaScript engine upgrade – other benefits

---

- Memory consumption
  - JavaScript uses 'garbage collection' to manage any memory that needs to be used for a script.
  - Every object, array or string you use needs to store a small amount of data to be able to do this.
  - This storage in PRIMER 18.0 is approximately 2/3 of the size in PRIMER 17.0.

With the default memory size of 25Mb

- PRIMER 17.0 could create ~350,000 objects.
- PRIMER 18.0 can now create ~500,000 objects



# JavaScript engine upgrade – other benefits

---

- Speed
  - Scripts which do a lot of mathematical operations will be faster (~ x3.5 speed increase in our tests).
  - String manipulation in scripts is faster (~ x3 speed increase in our tests).
  - Regular expressions in scripts are faster (~ x2.5 speed increase in our tests).
  - Several other features may see some speed increase from these and other improvements.

# JavaScript engine upgrade – other benefits

---

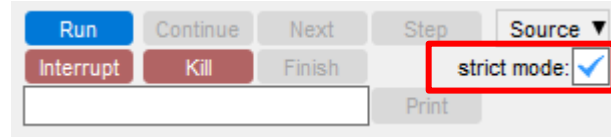
- Debugger
  - The implementation of the debugger has also changed with the new engine.
  - Stepping through code using 'Step' and 'Next' is now significantly faster compared to PRIMER 17.0, especially for scripts with many lines and/or functions.
  - In PRIMER 17.0 try and catch did not work properly in the debugger. For example, the following script would always fail with an exception in the debugger instead of 'catching' it. This now works correctly in PRIMER 18.0.

```
var o = {};  
try  
{  
    o.UndefinedMethod();  
}  
catch(err)  
{  
    Message(err);  
}
```



# JavaScript engine upgrade – other benefits

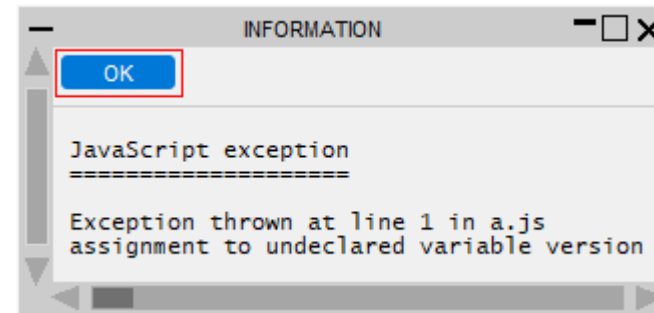
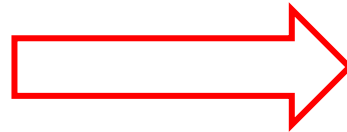
- Debugger



- Strict mode

- By default the debugger now works in 'strict mode' (see [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict\\_mode](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode) for details) as this helps to find potential errors.
    - This has changed since PRIMER 17.0. In PRIMER 17.0 the 'strict mode' checkbox actually added some more checks to the debugger. It did not enforce 'strict mode'. This has been corrected for PRIMER 18.0.

```
version = 18.0;  
Message(version);
```



- This is equivalent to running the script

```
"use strict";  
  
version = 18.0;  
Message(version);
```

- This behaviour can be turned off if required using the checkbox.

# JavaScript engine upgrade – other benefits

- Better checking
  - The new engine has better checking. For example, in the following code an error will be given when compiling that some code is unreachable (as there are { } missing so the return is not part of the if block and is always evaluated).

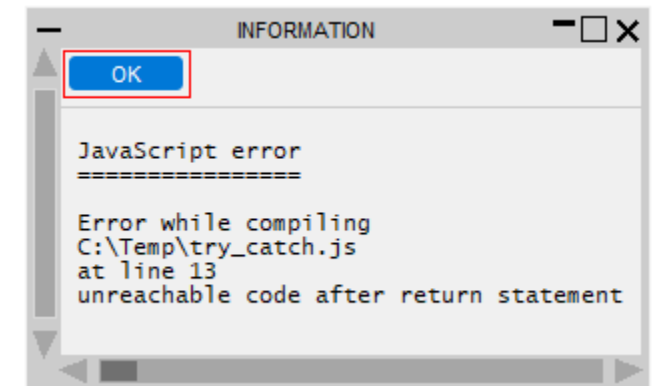
```
var vector = [ 1.0, 0.5, -0.2 ];
var length = vectorLength(vector);

function vectorLength(v)
{
    var l = 0;

    if ( !(v instanceof Array) )
        ErrorMessage("vectorLength not called with array");
        return null;

    for (var i=0; i<v.length; i++)
    {
        l += v[i]*v[i];
    }

    return Math.sqrt(l);
}
```



# JavaScript engine upgrade – important changes

---

- Garbage collection memory



- In PRIMER 17.0 the garbage collection memory was 'private' to each script.  
If multiple scripts were run at the same time each script would have 25Mb of private memory allocated.
- In PRIMER 18.0 the garbage collection memory is now shared across scripts (due to implementation differences in the new engine).

If multiple scripts are run at the same time

- The memory is allocated when the first script is run.
  - The memory is then shared/used for all the scripts running concurrently.
  - When the last script finishes the memory is returned.
- If you run scripts concurrently then the memory required may need to be increased.

# JavaScript engine upgrade – important changes

---

- ES6 Modules have not been implemented yet.
  - Upgrading the JavaScript engine has enabled ES6 (and newer) features to be used.
  - Modules are one ES6 feature that require significant changes in our software to implement and we are still resolving these.
  - For PRIMER 18.0 we want users to benefit from all the other ES6 features so have released the new engine without module support instead of waiting until we resolve this.
  - Support for ES6 modules will be added in a future release.

# JavaScript engine upgrade – important changes

---

- Set class

- ES6 introduced the [Set](#) class for collections of values.

However for many years in PRIMER we have used the Set class to support the \*SET keyword.

- By default the Set class will continue to be used to support \*SET as changing this could potentially break many existing scripts.
  - To use the ECMAScript Set class instead, set the preference `primer*set_class: ECMAScript`.
  - To enable both classes to co-exist, the class to support the \*SET keyword has been renamed to SetK and Set is an alias to SetK unless the above preference is used.

```
var s = new Set(model, 100, Set.Node);  
s instanceof Set           // false  
s instanceof SetK          // true
```

# JavaScript engine upgrade – important changes

---

- `hasOwnProperty()` bug in PRIMER 17.0 and earlier.
  - The JavaScript engine from PRIMER 17.0 (and earlier) contained a bug which meant that for the classes we define, object properties that were inherited from the object prototype appeared to be own properties of the object.
    - For example a Window object inherits properties title, left, right, top, bottom etc. from its prototype.
    - In PRIMER 17.0 this bug makes these properties appear to be an own property of the window as well as the prototype.
    - If you relied on this feature (unlikely) you will have to modify your code.

```
var w = new Window("Test", 0.8, 1.0, 0.5, 0.6);  
w.dog = "Bark";
```

```
Message(w.hasOwnProperty('title'));           // false. w does not have own property title. true in 17.0 (bug)  
Message(w.hasOwnProperty('dog'));             // true. w does have own property dog
```

```
Message(w.__proto__.hasOwnProperty('title')); // true. title is inherited from prototype  
Message(w.__proto__.hasOwnProperty('dog'));   // false. dog is not inherited from prototype
```

# JavaScript engine upgrade – important changes

---

- Script encoding differences

- In PRIMER 17.0 the default encoding used for scripts by the engine was 'Latin-1'.
- However on Windows this was actually implemented using the default encoding, which for many countries is Windows-1252.
- In PRIMER 18.0 the upgraded engine now compiles scripts as UTF-8 instead by default.
  - For scripts that just use ASCII (English) characters this will make no difference.
  - However the Windows-1252 encoding also contains special characters such as special quotes , apostrophes, en-dash and em-dash characters ( ‘ ’ “ ” – — ) and these are incompatible with UTF-8.
  - If your script contains these characters it will no longer compile as 'Latin-1' (and it would also not have run on Linux in PRIMER 17.0 and earlier). Either remove the characters or save the script in a different encoding using your editor.
- Setting a specific encoding for a script such as Shift-JIS or UTF-8 is unaffected.



# JavaScript engine upgrade – important changes

---

- Extra checking \*may\* occasionally mean old scripts that ran in PRIMER 17.0 no longer compile in PRIMER 18.0.
  - As the updated engine has better checking (such as the check for unreachable code mentioned earlier) in some rare cases it may mean that a script which worked in PRIMER 17.0 will fail to compile in PRIMER 18.0 until the error is fixed.



# JavaScript engine upgrade – important changes

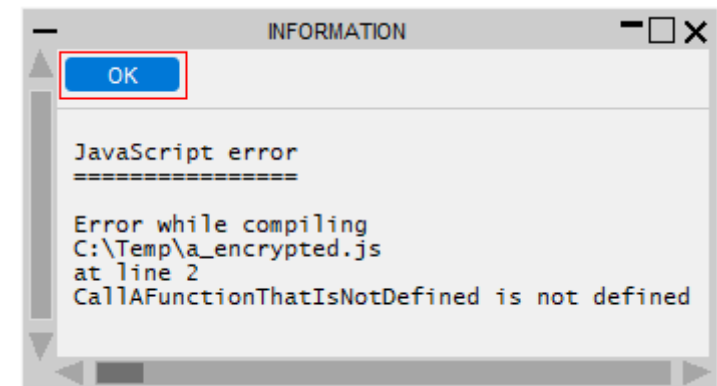
---

- Error messages have been enabled for encrypted scripts
  - In PRIMER 17.0 if a script was encrypted no error messages would be given when compiling/running.
  - For example if the following script was encrypted

```
Message("Starting...");  
CallAFunctionThatIsNotDefined();  
Message("Done.");
```

no error message would be given when the script tried to run the undefined function. This could make it very hard to determine the cause of a 'released' script failing.

- As the upgraded engine has better checking and there may be some rare cases when scripts don't run we have now changed this for PRIMER 18.0 so error messages will now be given for encrypted scripts.



# JavaScript API

# JavaScript API

---

- Various JavaScript API enhancements include:

- New SensorDefine class is added.
- `_MORTAR` keyword option can now be set for appropriate types for the Contact class.
- `*CONTROL_MPP_DECOMPOSITION_FLAG_STRESS_STRAIN_CURVE` is now accessible via JavaScript.
- The shell beta property has been changed.

In previous versions the `_BETA` option for the shell would be set if the beta property was set to a non-zero value. However this did not allow you to create an `*ELEMENT_SHELL_BETA` card with a zero beta angle from JavaScript.

In version 18 the beta property is now null if `_BETA` is not set.

- `*CONSTRAINED_LINEAR_GLOBAL` and `*CONSTRAINED_LINEAR_LOCAL` now accessible via JavaScript.
- `EVDUMP` on `*CONTROL_IMPLICIT_EIGENVALUE` now supported.

# JavaScript API

---

- Cross references are now made between the following entities and properties when created via JavaScript:
  - \*PART\_COMPOSITE and MID
  - \*BOUNDARY\_PRESCRIBED\_FINAL\_GEOMETRY and NID
  - \*INITIAL\_VELOCITY and IRIGID
  - \*LOAD\_NODE and M3
  - \*CONSTRAINED\_NODAL\_RIGID\_BODY and PNODE
  - \*SECTION\_SHELL and ELFORM, EDGSET
  - \*SECTION\_SOLID and ELFORM
  - \*SENSOR\_SWITCH and SWIT
  - \*ELEMENT\_(T)SHELL\_COMPOSITE and MID

# Checkpoint Files

Controlling Read/Write of Checkpoint Files

# Checkpoint Files

---

The following preferences will control read/write of checkpoint files.

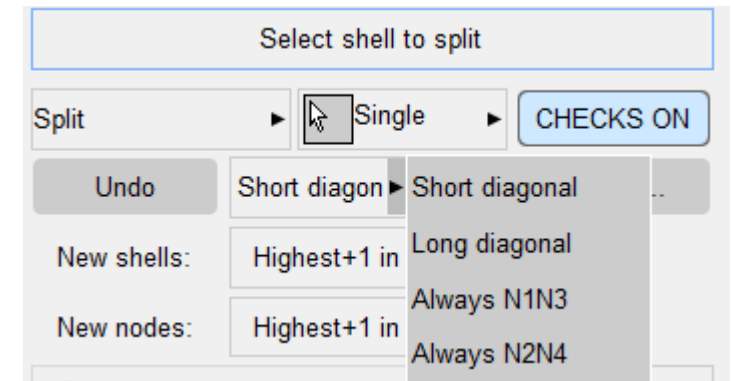
PRIMER also has command-line options with the same names.

- **primer\*write\_checkpoint\_files**: Enable/disable the recording of checkpoint files upon PRIMER startup. Valid values are TRUE/FALSE (default is FALSE).
- **primer\*show\_checkpoint\_files**: Enable/disable the reading of checkpoint files upon PRIMER startup. Valid values are TRUE/FALSE (default is FALSE).
  - If writing of checkpoint files is disabled, reading will also be automatically disabled.
- **primer\*checkpoint\_dir**: Specify the folder path to write checkpoint files to and read checkpoint files from.

# Miscellaneous

# Quad Shell Splitting

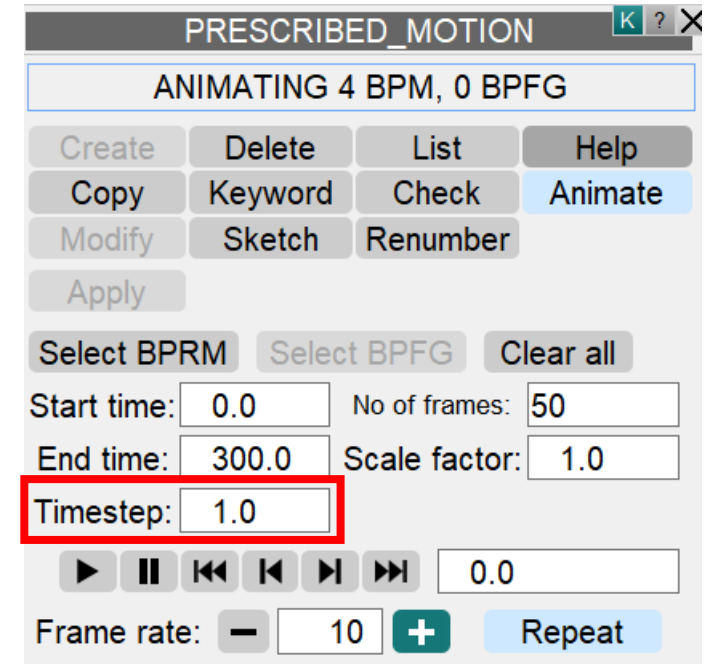
- There are new options to control how quad shells are split.
- Previously when using the Split shell option quad shells were always split along the short diagonal to preserve mesh quality.
- Sometimes this is not desirable so new options have been introduced:
  - Short diagonal (default, as previous)
  - Long diagonal
  - Always N1N3 – split is made along the node 1 to node 3 diagonal as defined by the shell topology
  - Always N2N4 – split is made along the node 2 to node 4 diagonal as defined by the shell topology





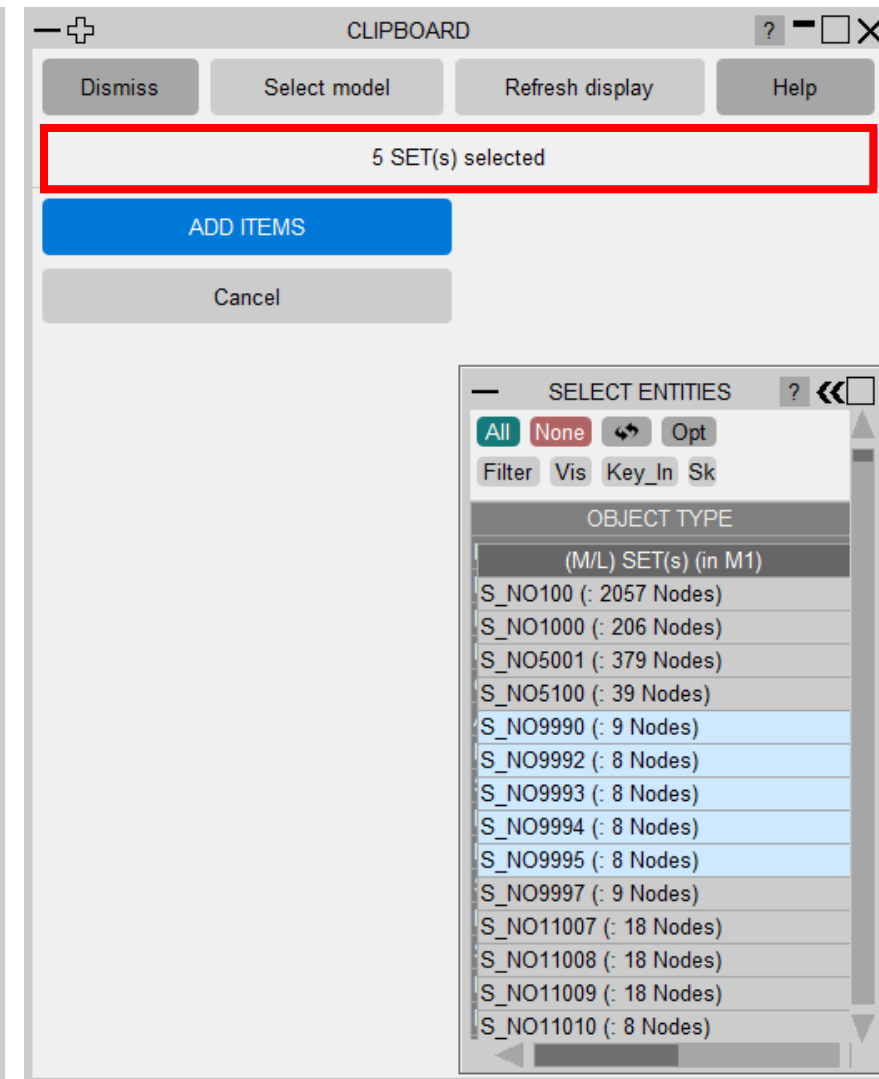
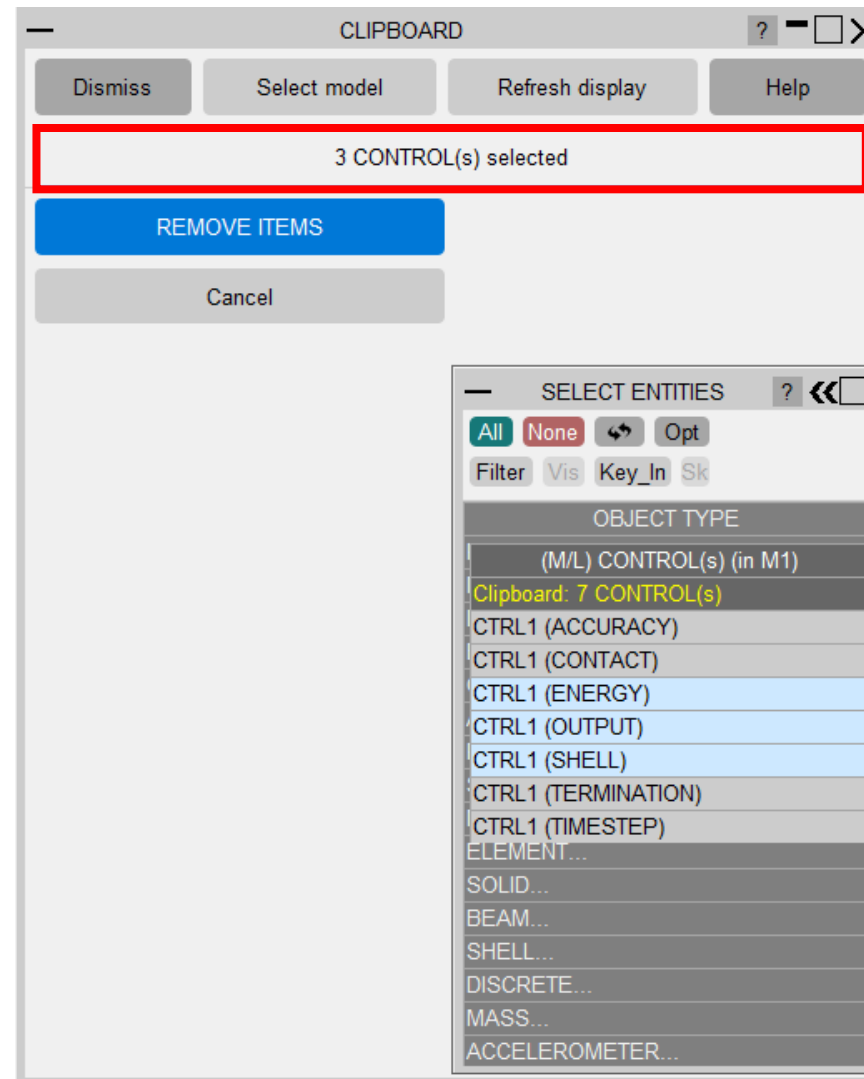
# Boundary Prescribed Motion Animation

- PRIMER's Boundary Prescribed Motion animation function computes the analysis timestep as a part of its operation.
- The timestep used by an implicit model can, sometimes, be difficult to compute in an exact manner.
- You may now specify a timestep of their choice that will override that computed by this function.



# Clipboard Item Count

- PRIMER's Clipboard header box in the object selection menu now shows the number of items selected to be added or deleted.



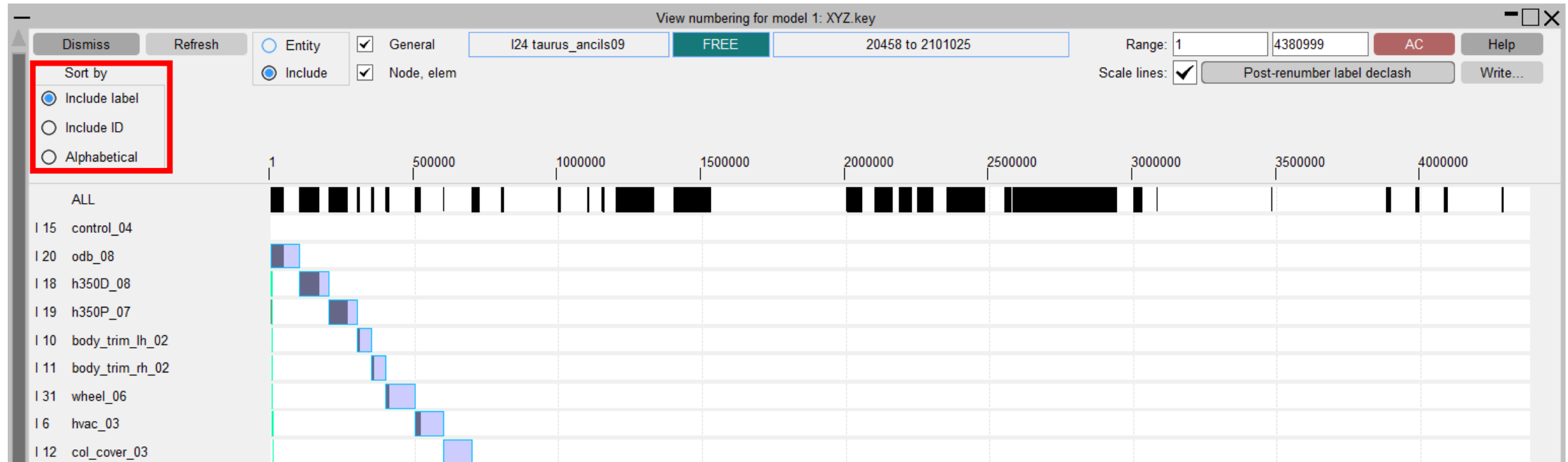
# Keyboard Shortcut Additions

---

- **Shift + Left Arrow:** Highlights from the current cursor position to the left by one character
- **Shift + Right Arrow:** Highlights from the current cursor position to the right by one character
- **Shift + Up Arrow:** Highlights from the current cursor position to the left-most character in the string (0 or prefix)
- **Shift + Down Arrow:** Highlights from the current cursor position to the right-most character (length of the string)
- **Ctrl + Left Arrow:** Jumps the cursor from the current cursor position to the left-most character of the word
- **Ctrl + Right Arrow:** Jumps the cursor from the current cursor position to the right-most character of the word
- **Ctrl + Shift + Left Arrow:** Highlight the rest of the word to the left of the cursor position up to the breaking character
- **Ctrl + Shift + Right Arrow:** Highlight the rest of the word to the right of the cursor position up to the breaking character
- **Double Click Left Mouse Button:** Highlights the whole word
- **Triple Click Left Mouse Button or Ctrl + A:** Highlights the whole line

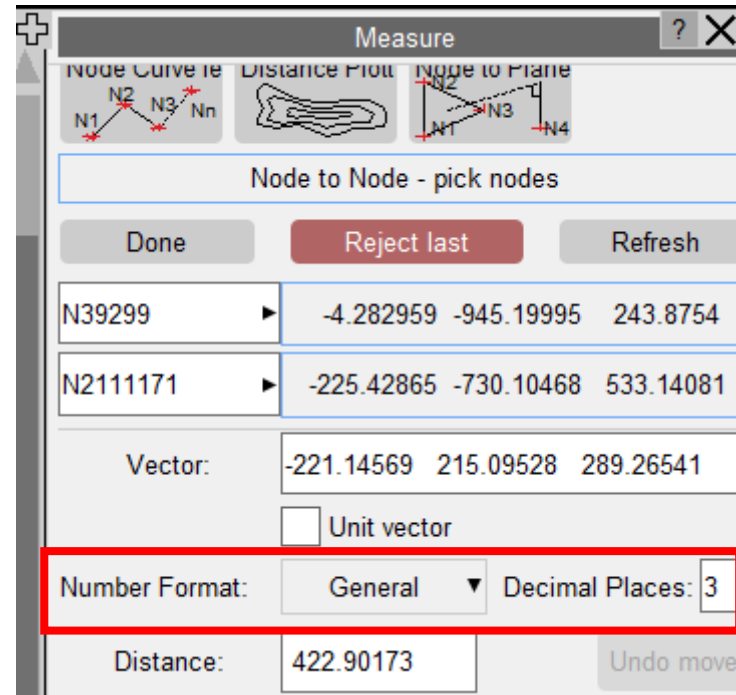
# Sort functionality in Renumber->Visualise panel

- You can now sort include files by
  - Include label (label's range in case of range provided or else minimum label present in that include file);
  - Include ID;
  - Alphabetical.



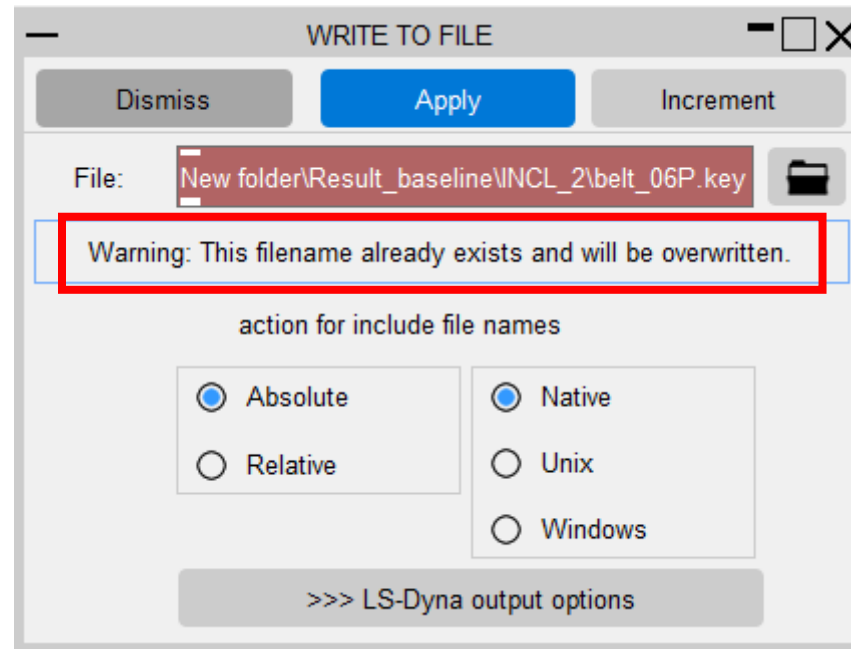
# Measure Panel Number Format

- Added option for number format in Measure panel.



# Include File Write

- Added a warning when writing out include if file exists.



# Contact Information

---



[www.arup.com/dyna](http://www.arup.com/dyna)

For more information please contact us:

## UK

T: +44 121 213 3399  
[dyna.support@arup.com](mailto:dyna.support@arup.com)

## China

T: +86 21 3118 8875  
[china.support@arup.com](mailto:china.support@arup.com)

## India

T: +91 40 69019797 / 98  
[india.support@arup.com](mailto:india.support@arup.com)

## USA West

T: +1 415 940 0959  
[us.support@arup.com](mailto:us.support@arup.com)

or your local Oasys distributor