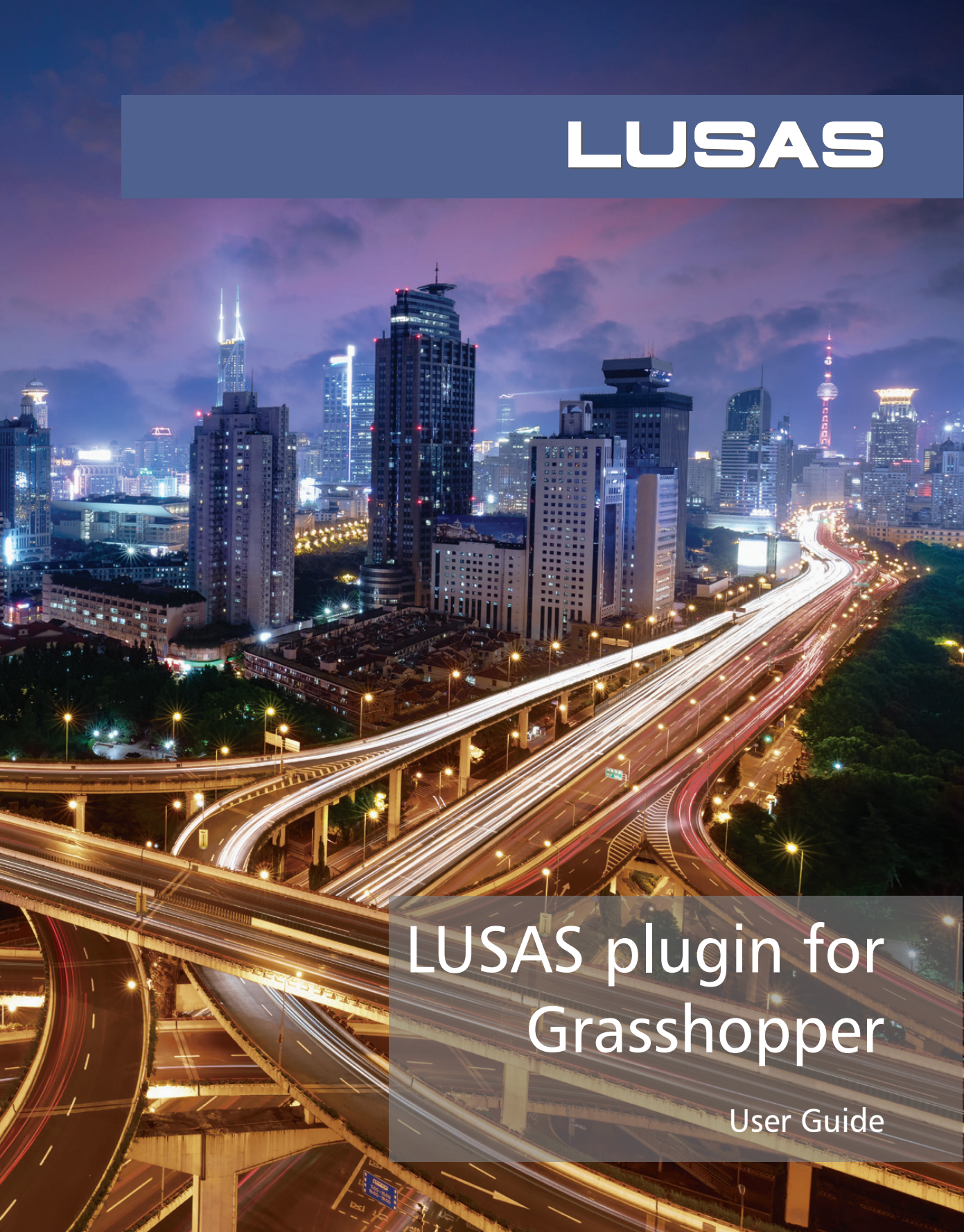
LUSAS



LUSAS plugin for Grasshopper

User Guide

LUSAS plugin for Grasshopper

Plugin Version 2.1.0 : Issue 1

LUSAS
Forge House, 66 High Street, Kingston upon Thames,
Surrey, KT1 1HN, United Kingdom

Tel: +44 (0)20 8541 1999
Fax +44 (0)20 8549 9399
Email: info@lusas.com
http://www.lusas.com

Distributors Worldwide

Table of Contents

LUSAS plugin for Grasshopper

Grasshopper

Grasshopper is a visual programming language and environment that runs within the Rhinoceros 3D computer-aided design application. By using its visual programming, you can algorithmically generate geometry by composing diagrams that link data to functions.

LUSAS plugin

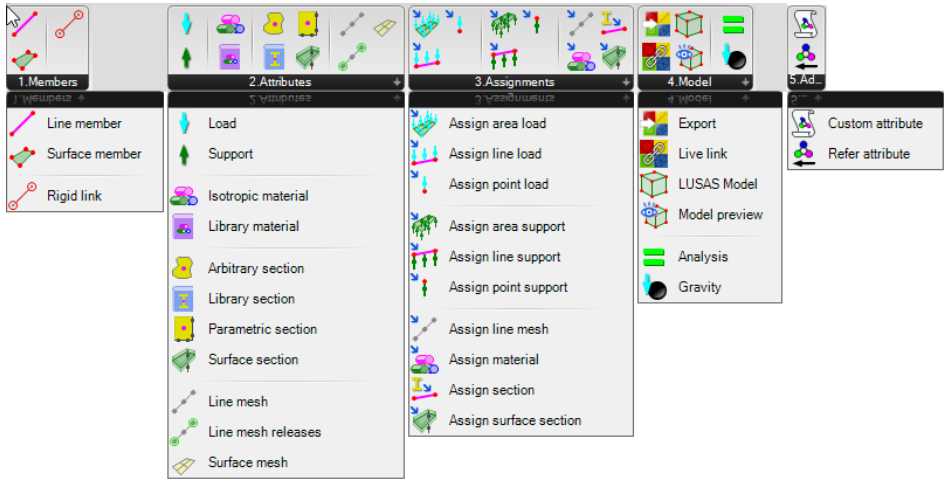
By using the LUSAS Grasshopper official plugin, parametric modelling of any structure is possible, and the models created can be output to LUSAS for further modelling, analysis and design purposes.

The plugin is available from within Rhino's package manager as well as from the food4rhino website at https://www.food4rhino.com/en. It is for use with Rhino (for Windows) version 8.13 and above, and LUSAS version 21.1 and above.

LUSAS components

The LUSAS-authored Grasshopper plugin creates a ‘LUSAS’ entry in the main menu and a ‘LUSAS’ tab in Grasshopper’s component ribbon. The LUSAS components are grouped into four named categories: Members, Attributes, Assignments, Model and Advanced.

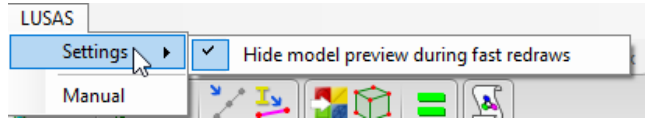
When expanded, each category shows its respective components separated logically into further subcategories. By selecting a component from this menu it can be placed on Grasshopper’s canvas to be used in a script.



LUSAS on the main menu

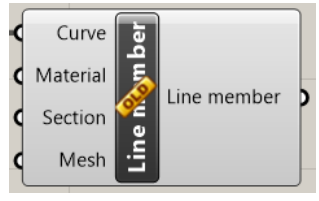
The LUSAS entry in Grasshopper's main menu holds a sub-menu of global features or settings for the plugin. Currently the available menu items are:

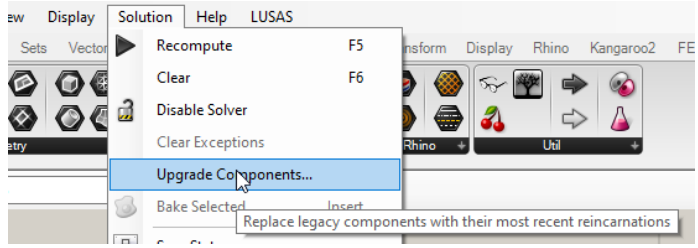
- **Settings** - Displays an option to 'Hide model preview during fast redraws', which is 'on' by default.
- **Manual** - Provides a link to the help topics / manual for the plugin.



Upgrading components

Some components have had their parameters modified since the last version (for example, 'Line/surface' member or 'Analysis' component). These components will be marked as 'OLD' in existing scripts and can be manually upgraded from within Grasshopper itself by clicking on the Solution>Upgrade Components menu item.





Model units

All numeric values on LUSAS components are assumed to be in the same units as selected on the LUSAS model component. It is important to decide on the model units before starting a LUSAS Grasshopper script, as the LUSAS model component requires that the length units in the Rhino model match with the selected units on the component.

Note that whilst the units can be changed at any time, any numeric values which have been used as input to other components (for example the values for an isotropic material or the coordinates of geometry) will not be automatically converted.

Standard workflow

The standard workflow for a Grasshopper script that creates LUSAS members, attributes and assignments and links them to a model within LUSAS is:

1. Select the desired model units at the beginning and set the same units in the Rhino model.
2. Create structural members (line or surface members), by specifying the geometry using standard Grasshopper components or referencing Rhino model geometry.
3. (Optional) Create attribute definitions (materials, sections etc) and assign them to the structural members.
4. (Optional) Create load and support attribute definitions and assign them to geometry matching a structural member's nodes, lines or surfaces.
5. Assemble all structural members and load/support assignments on the 'LUSAS model' component with the same length units as in the Rhino model.
6. (Optional) Use the 'Model preview' component to preview member sections, loads, supports and rigid links.
7. Use the 'Live link' or 'Export' component with the assembled model to link with a running instance of LUSAS or to create a model or distributable LVB script.

Geometry and Attributes in LUSAS

There are four geometric feature types in LUSAS to which attributes such as a mesh type, geometric properties, material, supports and loading can be assigned:

- **Points** - These define the vertices of the finite element model.
- **Lines** - These define members and the extent of the surfaces in the finite element model and are the equivalent of Rhino's curves.
- **Surfaces -** These define internal or external faces of a model.
- **Volumes** - These define simple solid components of a model.

Features are defined hierarchically, i.e. Points define Lines, Lines define Surfaces, Surfaces define Volumes.

Attributes are assigned on a feature basis, therefore the positions of geometric and material discontinuities, supports and loads must be carefully considered when defining the features.

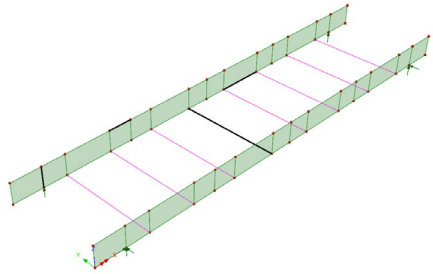
Line (Curve) Connectivity

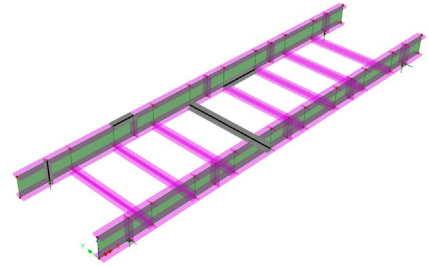
Note. With the LUSAS-Grasshopper plugin, points cannot be directly defined but are always automatically created at the ends of lines (curves) or at every surface vertex. All Grasshopper scripts must therefore make sure that lines / spline lines / arcs are split at every location where a connection is required in LUSAS. This is also the case for lines intersecting surfaces, where connections are not automatically created.

The following unfleshed and fleshed LUSAS models show the correct geometric line feature modelling of structural members that were defined using a Grasshopper script. Individual lines exist between each connection point. Example of these line extents are highlighted.

Half-through bridge

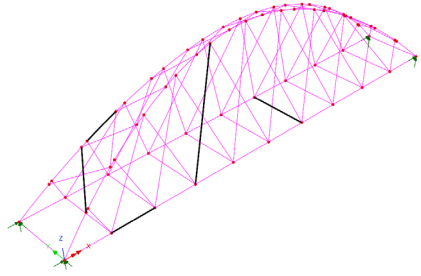
Lines represent the top and bottom flanges of the plated longitudinal beams and also the crossbeam members. Surfaces representing the webs of longitudinal beams are connected to the crossbeams at common points.

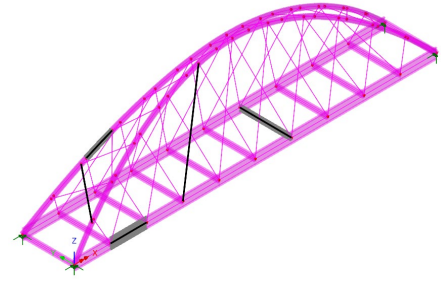




Network tied-arch bridge

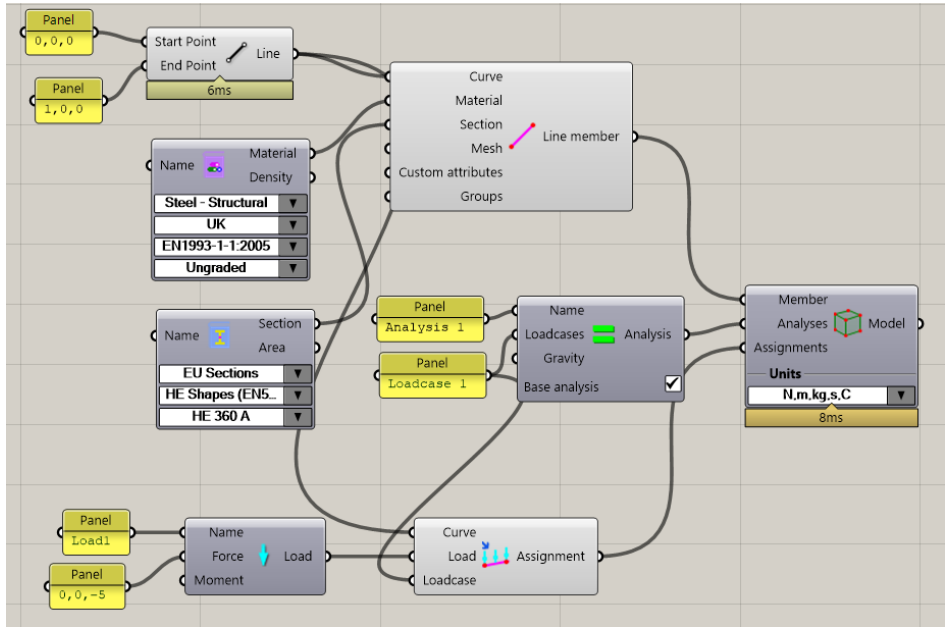
Lines represent all members and are connected to other lines with points at each connection. Note that the hangers and cross-bracing members have no mid-line connections.





Simple script

The following image shows a simple script which creates a line member with a material and a section, assigns a line load of -5kN in the Z direction and assembles the model ready to be linked to LUSAS.



Component input parameters

There is a distinction to be made between the two types of input:

1. Generic input types, which can be set with standard Grasshopper components, for example curve/surface geometry, number, text, vector etc.
2. Input types which can be set exclusively using the output of other LUSAS components, for example material, section etc.

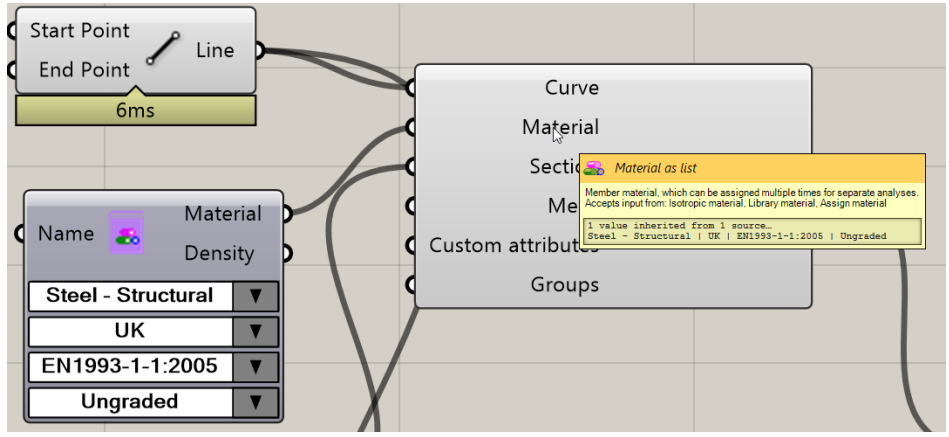
The type of a parameter can be seen by hovering the mouse cursor over the parameter's name.

To prevent confusion, any parameter inputs on LUSAS components of the 2nd type that accept only the input of other LUSAS components have an extended tooltip stating which component outputs can be used to set that parameter's value.

Example

The following example for a line member component shows the two types of different parameter inputs. The 'Curve' parameter is set from a standard Grasshopper component, and the 'Material' parameter is set from a LUSAS component.

A tooltip for the material parameter states which LUSAS components can be connected.



Structural line and surface member components

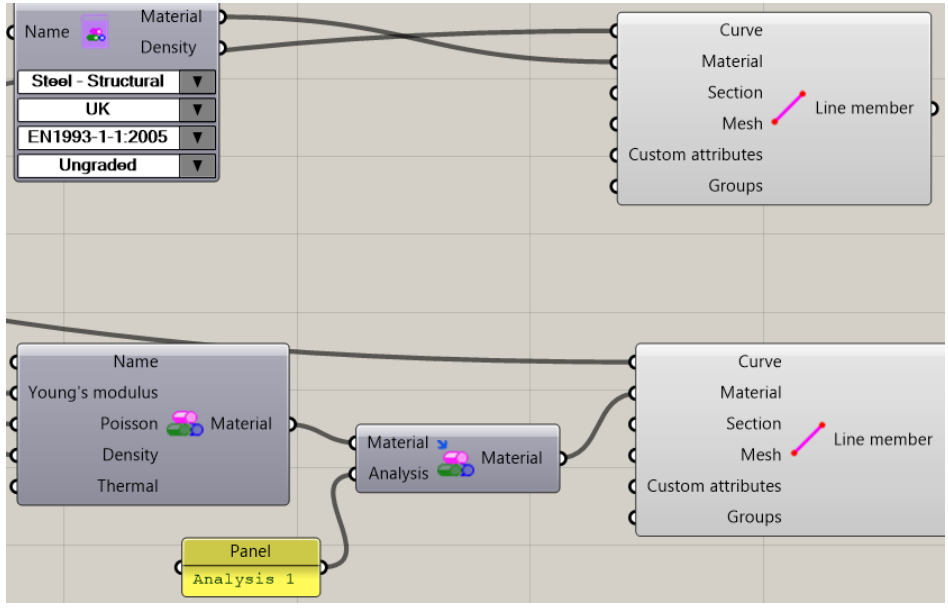
Some inputs on the structural line and surface member components (for example material, section etc) can be set in two separate ways:

1. By using the output of the attribute definition components, for example Isotropic material, Parametric section etc.
2. By using the intermediate assignment component if additional data like a specific analysis need to be provided, for example Assign material, Assign section etc.

Example

The following image shows the two possible ways of defining the material for a line member:

- by using the output of a material attribute definition component directly
- by using an intermediate assignment component to state a specific analysis.



Load / Support assignment components

Load and support assignment components require the geometry (point, line, surface) as an input. The plugin will automatically find which member the attribute should be assigned to.

Currently, loads and supports can only be assigned to points that are actual nodes of a structural member (curve endpoints, surface vertices) or curves that define a line member or are edges of a surface member. Loads or supports assigned to points or curves within the length of a line member or within a surface member are not supported.

Nurb surface compatibility

Nurb surfaces are exported to LUSAS without any internal trims (holes).

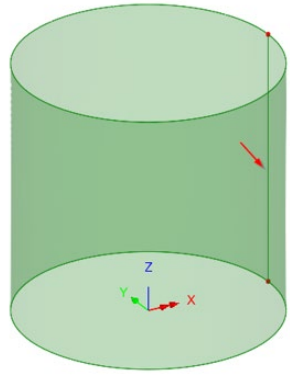
Some types of nurb surfaces are incompatible and will not be exported. Warning messages will appear on the surface member component with an explanation when:

- The surface contains a seam (e.g. cylinder).
- The surface contains a singularity point (e.g. sphere apex).

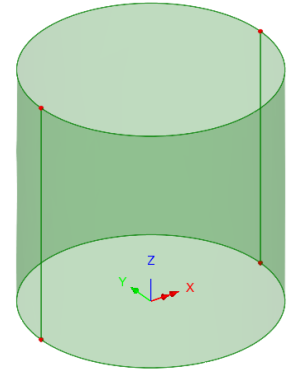
In most cases, problems with nurb surface compatibility can be overcome by subdividing the surface.

Example: Cylinder

Cylinders can be exported successfully by splitting the cylinder in half, so that there is no seam.



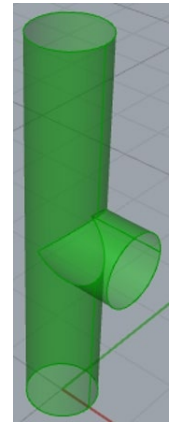
Cylinder prior to splitting with seam highlighted



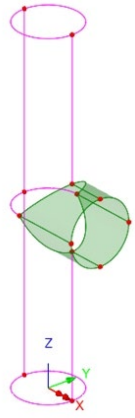
Cylinder after splitting

Example: CHS connections

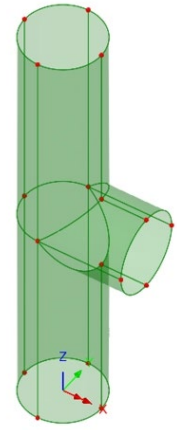
A more complex model of CHS connections can be exported successfully if an appropriate number of subdivisions of the original geometry is made.



CHS connection boundary representations (brep) exported to LUSAS



The horizontal CHS section is subdivided horizontally and vertically.

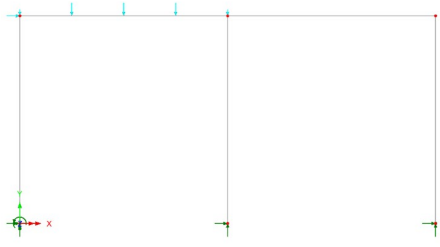


The vertical CHS section is subdivided further to eliminate the cylinder seam and make the export possible for the whole boundary representation.

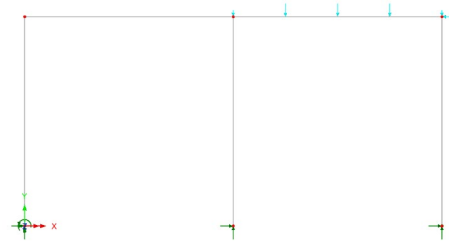
Example script

LUSAS model (for reference)

A model of a simple loaded frame is shown below. The frame consists of 3 columns and 2 beams with different sections. The leftmost column is fixed at the base while the others are pinned. There are two loadcases each containing one concentrated point load and one distributed line load.

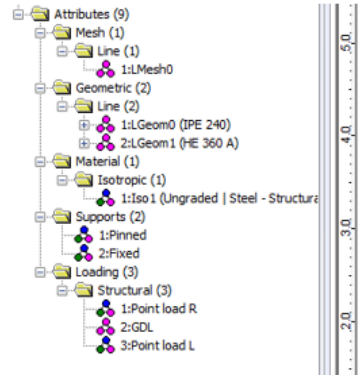


Loadcase 1



Loadcase 2

The corresponding Attributes that would be generated for this model in LUSAS are these:



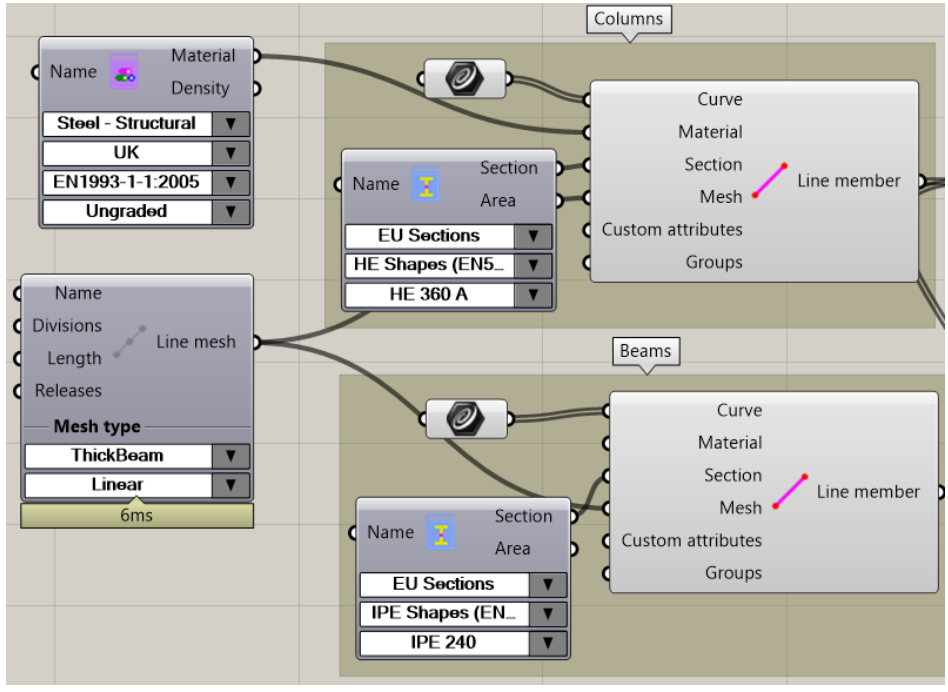
Grasshopper script

This is a step-by-step guide showing the Grasshopper script required to create the model above using LUSAS components. Note that for the sake of brevity the geometry used is always referenced from the Rhino model, but this could also be created parametrically using standard Grasshopper components.

The model consists of 5 line members (3 columns and 2 beams) which share a common mesh and material attribute but have different sections. The geometry is referenced from the Rhino model but could be also set from standard Grasshopper components that create curves.

For this example, a line mesh with default values is used along with a material and sections from the LUSAS libraries.

Defining column and beam members



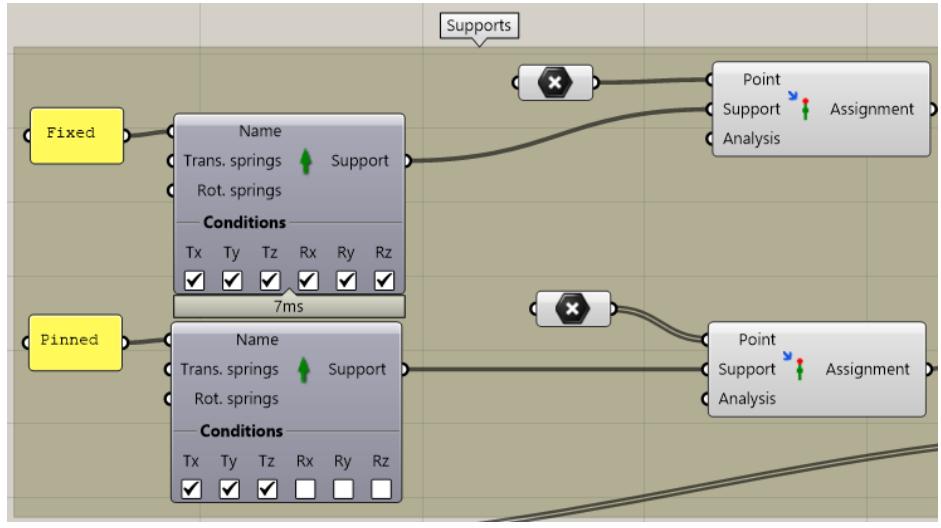
Column and beam members with their geometry, mesh, material and section

Defining supports

To define the supports, the ‘Support’ component is used to create a pinned and a fixed support which is later assigned to the base of the columns using the ‘Assign point support’ component.

For the geometry, point references from the Rhino model are used here; they also could be extracted from the column curves used for the line members.

The analysis parameter is left empty, which signifies that the supports will be assigned to the default base analysis.



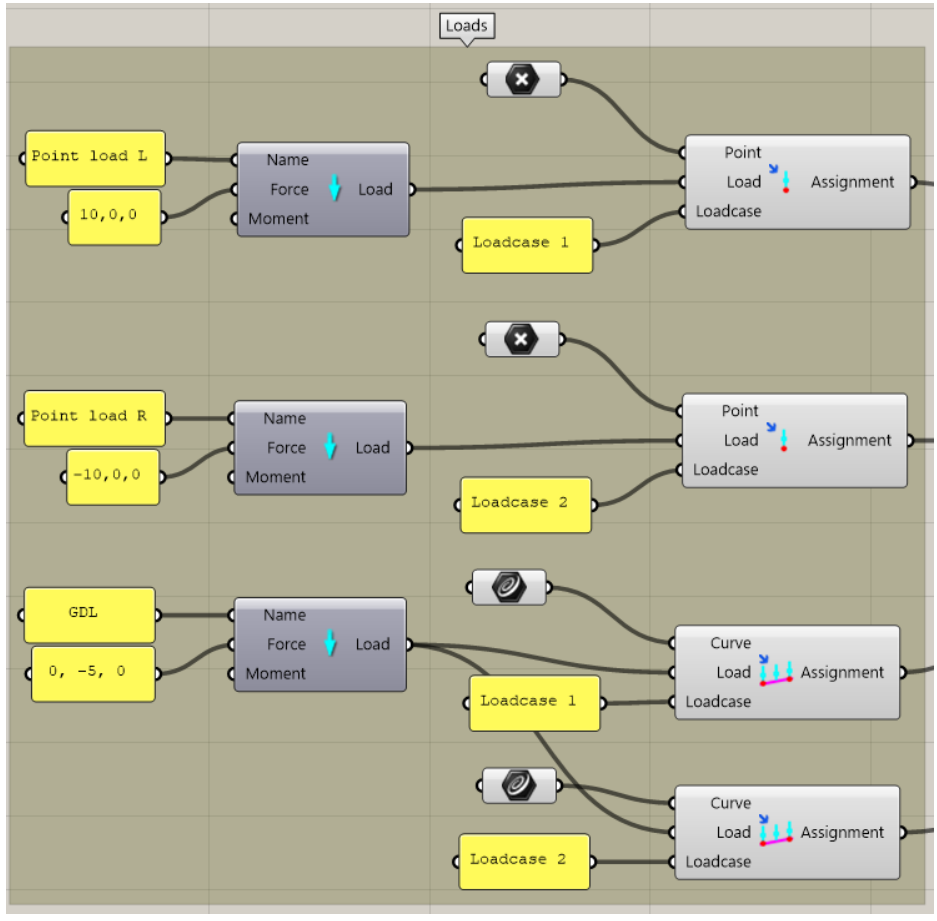
Support definitions at the column bases.

Defining loading

Loads are defined similarly using the Load component to create the 3 different load definitions: a point load of 10kN in the +X direction, a point load of 10kN in the -X direction and a distributed load of 5 in the -Z direction.

The ‘Assign point load’ component is used with referenced points from the Rhino model to assign the point loads; each point load is assigned to a different loadcase.

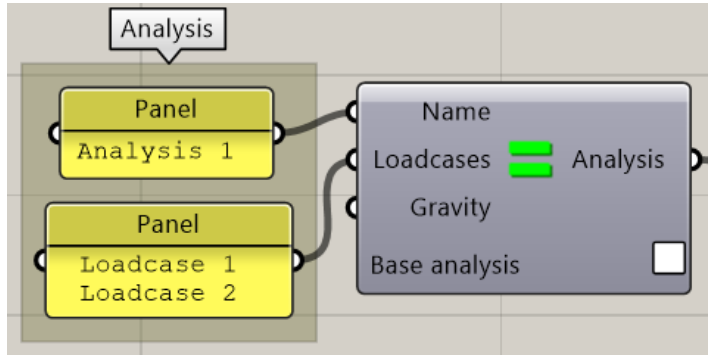
The ‘Assign line load’ component is used with referenced curves to assign the distributed load in each beam for each loadcase.



Load definitions for the beams.

Analysis

The plugin creates an analysis by default called ‘Analysis 1’ with 1 sole loadcase called ‘Loadcase 1’. For this example two loadcases are required, therefore the ‘Analysis’ component is used to define them. If only one analysis is defined in a model, it will be the base analysis by default.

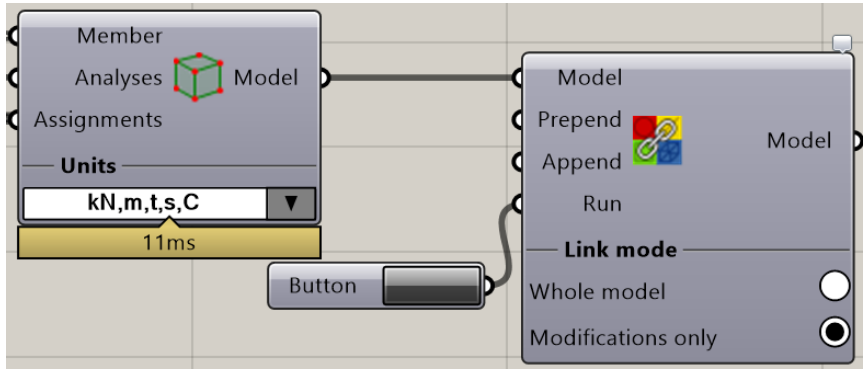


Model analysis and loadcases definition.

Assemble Model data and Link to LUSAS

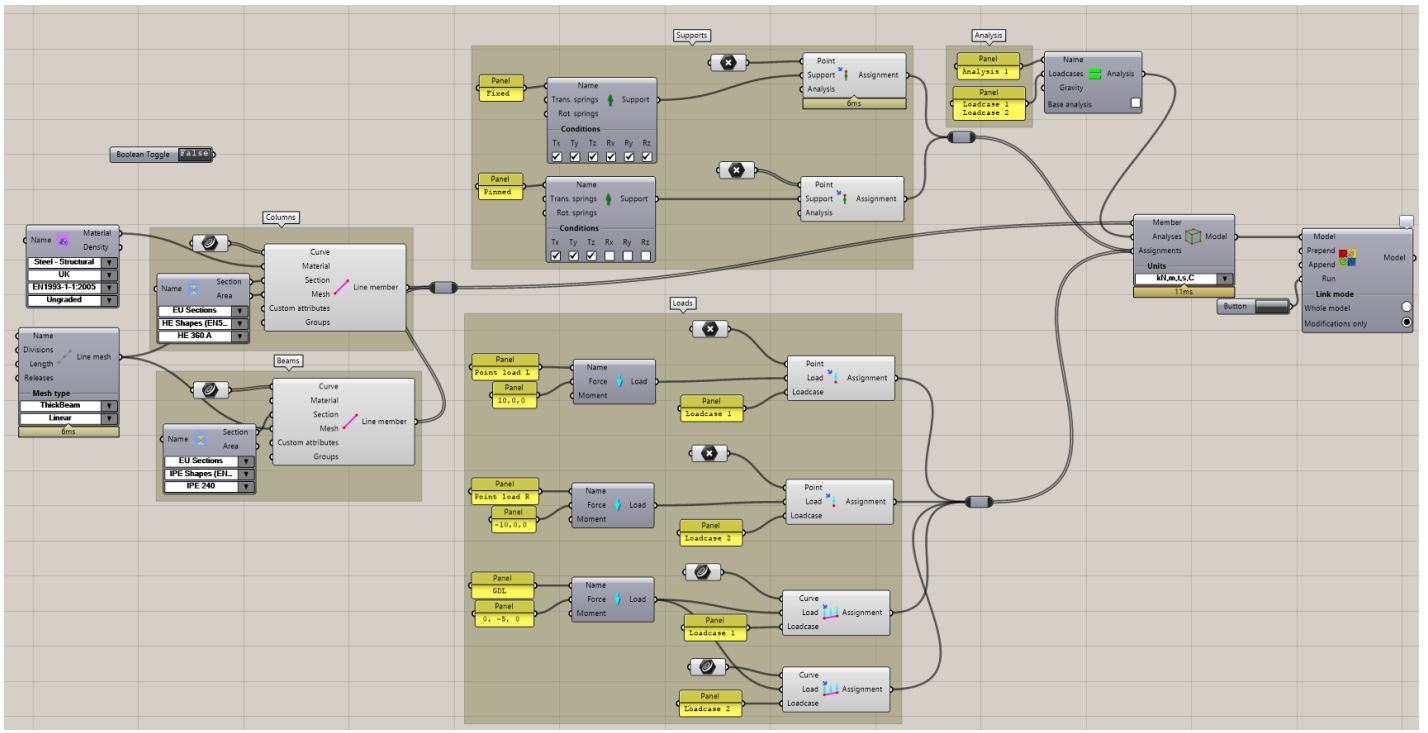
Lastly, the 'Model' component is used to assemble all model data and set the model units. The output of the previously shown components is connected to this component.

The model output is then connected to the 'Live link' component which can link the specified model data to a running instance of LUSAS.



Model assembly and linking to LUSAS.

The complete script containing all components along with their required connections is shown in the following image.**Error! Reference source not found.**



Live link

The current plugin version supports continuous updates between the LUSAS model in the Grasshopper script and a running instance of LUSAS, thus greatly enhancing the workflow between Grasshopper and LUSAS. A one-way connection is established with Grasshopper as the source and LUSAS as the target, whereby any modifications to the components in Grasshopper can be transferred and previewed in LUSAS itself.

To establish a live link and update the model in LUSAS with the modifications in Grasshopper in real time, the ‘Live link’ component is used. Details about the linking process and the proper use of LUSAS components to warrant problem-free updates between Grasshopper and LUSAS are given in the following subchapters.

Connection to LUSAS

In the current version, the live link will work only if one single instance of LUSAS is open and running with an active model file. As a result, the ‘Live link’ component will present two options if no instance of LUSAS is running at the start of a live link process:

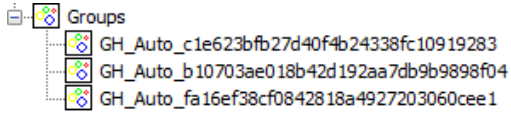
- New model: start LUSAS automatically with an empty model.
- Open existing model: browse for a LUSAS model file, start LUSAS automatically and open the selected model.

Note that in the current plugin version, only the latest LUSAS version can be automatically started. If a specific LUSAS version is required, it is advised to start it manually, create/open a model and then run the live link process.

Process overview

Due to the nature of Grasshopper components, all of their data are reevaluated with every component solution, making it impossible to map every individual object in Grasshopper with a model object in LUSAS. As a workaround, the plugin creates groups in LUSAS for every ‘Line/Surface member’ or ‘Rigid link’ component in the Grasshopper script that contain all LUSAS features created by each component.

These groups are automatically named and should not be modified, renamed or deleted by users.



Depending on the selected link mode, the plugin determines which objects to update on the linked LUSAS model. In LUSAS, the features selected for update are first deleted on a group-by-group basis. In the next step, the link operation creates the updated

feature groups with their attributes and assignments. Modifications in the Grasshopper model therefore cause an entire feature group (or even multiple groups) to be updated in LUSAS.

For every link operation, an undo action is recorded in LUSAS so that the model can be reverted to the exact state before the last update easily.

Link modes

There are two link modes available:

- 'Whole model' which updates the whole model with each link operation,
- 'Modifications only' which updates only the modified objects with each link operation.

The 'Whole model' link mode is recommended as a "reset" for the link between LUSAS and Grasshopper. As it updates the whole model with each link operation, it ensures that every feature and attribute is up to date. For smaller models, this mode might not be noticeably slower than the 'Modifications only' mode.

The 'Modifications only' mode avoids updating every feature and attribute in LUSAS by determining only the modified objects in Grasshopper. This mode can make the link operations noticeably faster for larger models. However, it depends on the automatically created feature groups and other information in the LUSAS model that in extreme cases may be disrupted. The 'Whole model' mode can be used in cases of unexpected behaviour.

Structural member updates

Structural members in Grasshopper are created with the 'Line/Surface member' component and correspond to LUSAS features (i.e. geometry) with their attributes. The link operation maps each component to a group in LUSAS and updates all features in that group to accommodate modifications to the component in Grasshopper.

If the 'Whole model' link mode is selected, all feature groups will be updated. In case the 'Modifications only' mode is selected, the plugin will determine the modified components and attempt to update only their respective groups in LUSAS, making the process faster and more efficient. However, it is possible that structural members of one component are related to members of another component, for example a line member that also forms the edge of a surface member. In that case, all related groups will be updated in LUSAS.

Attribute updates

Attributes are created in LUSAS for structural member parameters such as material, section, mesh as well as for any assigned loads and supports. Attribute name

parameters are in general optional for the dedicated LUSAS attribute components, they are however mandatory for the link operation.

Attributes are identified in the LUSAS model by their name and type. Modifications to attribute parameters (for example, material values) will get updated in LUSAS, as long as the attribute name is not modified. If an attribute name is modified, a new attribute with the new name will be created and the previous attribute will not be deleted. If an attribute component is removed from the script entirely, it will also not be deleted from the LUSAS model.

It is possible that a renamed attribute may clash with attributes already existing in the LUSAS model. The ‘Live link’ component will perform this check and report an error for any attributes that cannot be created due to clashes with existing attributes.

There are also considerations about the load and support attributes. These are defined in Grasshopper using the ‘Load’ and ‘Support’ components respectively and can be used as input to multiple assignment components. It is however not obvious that a load assigned to a point will create a different attribute in LUSAS than a load assigned to a line or a surface. Likewise, a support with springs will create different attributes in LUSAS when assigned to points, lines and surfaces. In such cases, separate ‘Load’ and ‘Support’ components must be used with different attribute names.

Analysis and loadcase updates

Analyses and loadcases in LUSAS contain information about most attribute assignments. As such, care is taken by the plugin to ensure that no assignment information is lost, thereby posing constrictions to the automatic update of analyses and loadcases on every link operation.

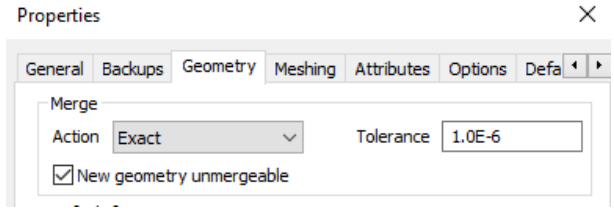
Analyses are created with their loadcases in LUSAS for every ‘Analysis’ component in the Grasshopper script. If an analysis name is modified in Grasshopper, the plugin attempts to rename the analysis in LUSAS if there is no other analysis with the same name. If an analysis is removed from Grasshopper entirely, it is not deleted in LUSAS.

Loadcases are only identified by their name and parent analysis and as such, they will only be created under their respective analysis as defined in the ‘Analysis’ component if there are no loadcases already present with the same name in the LUSAS model. Renaming or deleting loadcases will have no effect on the loadcases in LUSAS.

Interactions with user-defined features

Features (i.e. geometry) that are automatically created in LUSAS via the live link process are always created as mergeable geometry. This means that they will automatically be merged with any user-defined mergeable features in the model according to the geometry merging rules defined in the ‘Model properties’ dialog of LUSAS.

By specifying the merge ‘Action’ and ‘Tolerance’ settings, it is possible to fine tune the merging behaviour to the desired effect. This makes it possible to have user-defined geometry seamlessly integrate with geometry linked from Grasshopper.



Performance tips

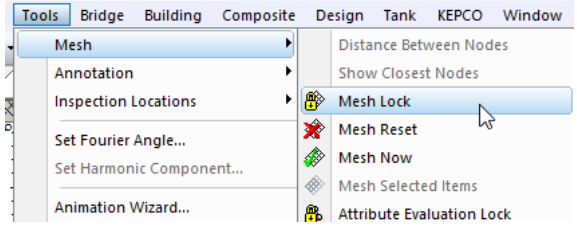
Model preview

A preview of member cross-sections, loads and supports in Rhino’s viewports is shown when the ‘Model preview’ component is used. For large models, the preview objects might impact performance. Some actions that might help:

- Turning off the preview of specific assignment on the ‘Model preview’ component such as ‘Sections’, ‘Loads’, ‘Supports’.
- Enabling the ‘Hide model preview during fast redraws’ option in the main menu under LUSAS > Settings. When this is active, cross-sections, loads and supports will not be previewed when Rhino’s viewport is rapidly redrawing (for example, when rotating or panning).

Live link

Link operations that result in modification of features in LUSAS will require the mesh to be updated. This behaviour is on by default and may take considerable time to complete. It is possible to speed up the link operations by locking the mesh in LUSAS.



LUSAS example Grasshopper scripts for download

Example LUSAS Grasshopper scripts may be downloaded from the LUSAS website at the following address:

https://www.lusas.com/products/external_plugins/grasshopper/index.html

The following examples are currently provided:

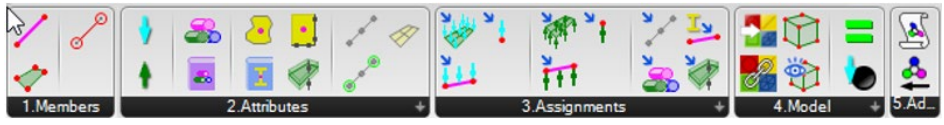
- Cooling tower
- Curved plate girder bridge
- Geodesic roof
- Mobius strip
- Network tied-arch bridge
- Spherical tank
- Twin girder bridge with vehicle load optimisation
- Twisting tower
- U-frame bridge

Assignment of loading

Note that the assignment of gravity loading or other applied loading is required either within the Grasshopper script or within LUSAS in order to allow solving of the model to take place.

LUSAS component reference

LUSAS component groups



These comprise:

- Members
- Attributes
- Assignments
- Model
- Advanced

Members

Structural members (LUSAS line and surface features with attributes) are called line and surface members respectively in this reference guide.

Line member(s)

Line members have a ‘Curve’ input (one or multiple), which is a data type native to Rhino. This curve is converted to a curve type compatible with LUSAS.

The ‘Mesh’ input accepts the output from both the ‘Line mesh’ and the ‘Assign line mesh’ components. In the case of the former, a beta angle of 0 degrees will be assumed.

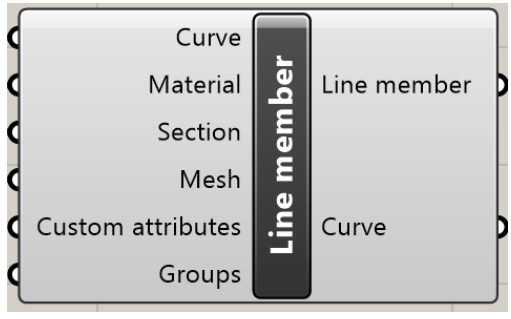
The ‘Material’ and ‘Section’ inputs accept the output of their respective attribute definition components either directly or through their respective assignment component.

The ‘Custom attributes’ input accepts the output of the advanced ‘Refer/Custom attribute’ components to assign them to all members created by the component.

The ‘Groups input accepts a list of group names to add the members to.

The component also outputs the member geometry (the same as the input curves) so that it can be fed to other components as an input (for example, ‘Load/support assignment’ components.

Inputs	Outputs
Curve(s) – *native Rhino type*	Line member(s)
(Material) – *output of the (Assign) Material component*	Curve(s) – *native Rhino type*
(Section) – *output of the (Assign) Section component*	
(Mesh) – *output of the (Assign) Line mesh component*	
(Custom attributes) – *output of the Refer/Custom attribute component*	
(Groups) – *multiple texts*	



The line member component.

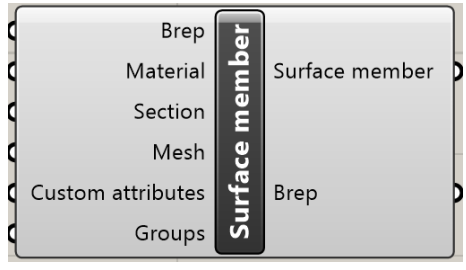
Surface member(s)

Surface members have a boundary representation (brep) input (one or multiple), which is a data type native to Rhino potentially consisting of many surfaces. Each surface will be converted to either a flat or a NURBS surface in LUSAS depending on the geometry in Rhino.

The component also outputs the member geometry (the same as the input breps) so that it can be fed to other components as an input (for example, ‘Load/support assignment’ components.

The component will validate if the surface can be exported and show a warning message otherwise. Some nurb surfaces cannot be directly exported to LUSAS and might require subdivisions (read more in the Nurb surface compatibility section).

Inputs	Outputs
Brep(s) – *native Rhino type* (Material) – *output of the (Assign) Material component* (Section) – *output of the (Assign) Surface section component* (Mesh) – *output of the (Assign) Surface mesh component* (Custom attributes) – *output of the Refer/Custom attribute component* (Groups) – *multiple texts*	Surface member(s) Brep(s) – *native Rhino type*

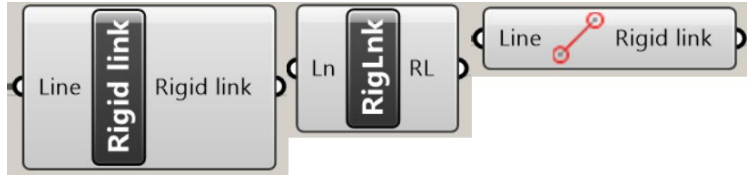


The surface member component.

Rigid link

Rigid links are defined with native Rhino Lines which will connect their end points with a rigid joint in the resulting LUSAS model. The component's output must be connected to the 'Members' input of the 'Model' component.

Inputs	Outputs
Line(s) – *native Rhino type*	Rigid link(s)



The rigid link component in all possible display modes.

Attributes

Attributes have their own definition component and will be assigned to members with their respective assignment components where necessary. Attributes that require no additional assignment data in LUSAS (or in which the assignment data will be assumed) can be connected directly from the attribute definition component to the corresponding member.

Line mesh

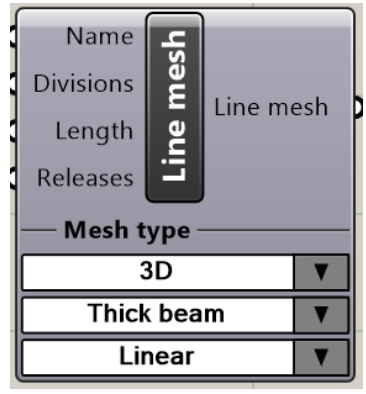
The line mesh component allows for the definition of line mesh attributes. It features dropdown lists for selecting the line mesh type and order for a specified analysis category. The available line mesh types are a subset of the available mesh types in LUSAS.

The 'Divisions' input optionally specifies the number of mesh divisions. If left undefined, the default spacing option in LUSAS is assumed.

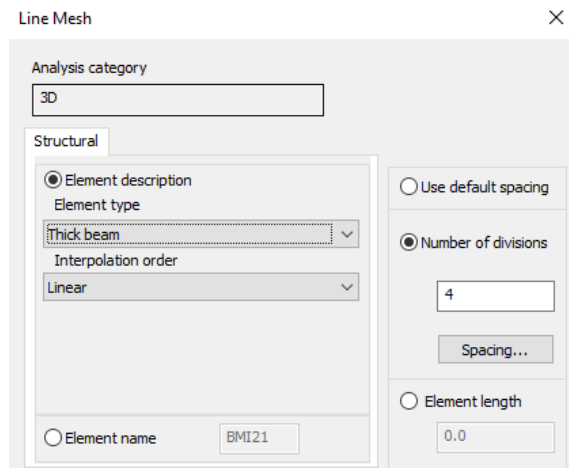
The 'Length' input optionally defines a fixed element length for the mesh. The component only allows one of the two inputs to be specified at any time and reports an error otherwise.

The 'Releases' input can optionally be used to specify releases for the line mesh, by taking the output of the 'Releases' component as input. Note that not all line mesh types support end releases. This is validated by the component and an error message is shown if necessary.

Inputs	Outputs
(Name) – *single line text*	Line mesh
(Divisions) – *numeric, > 0, default: 4*	
(Length) – *numeric, ≥ 0, default: 0*	
(Releases) – *output of the Line mesh releases component*	



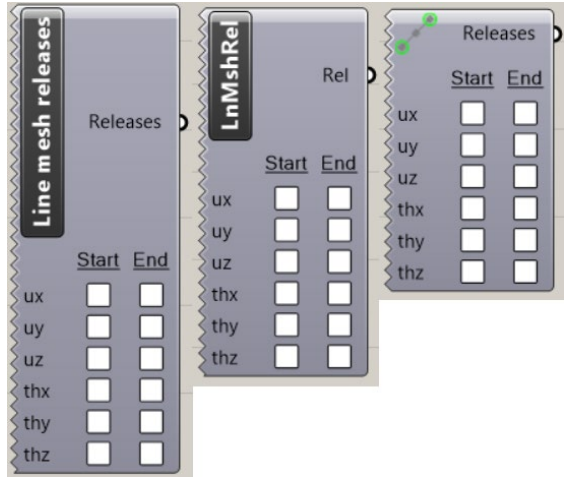
The line mesh component.



Line mesh dialog in LUSAS.

Line mesh releases

The releases component allows the definition of releases for the ‘Line mesh’ component. By checking any of the custom checkbox controls, the respective degree of freedom for the start/end of the line mesh can be released.



The line mesh releases component in all possible display modes.

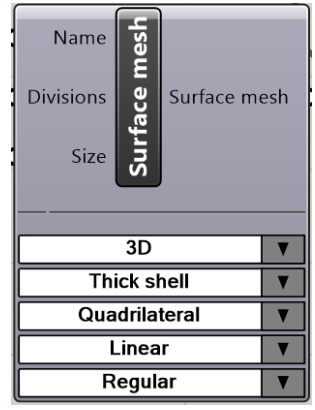
Surface mesh

The surface mesh component allows for the definition of surface mesh attributes. It features dropdown lists for selecting the surface mesh type, shape and order for each analysis category and one dropdown list for selecting between regular, regular (allow irregular) and irregular mesh. The available surface mesh types are a subset of the mesh types available in LUSAS.

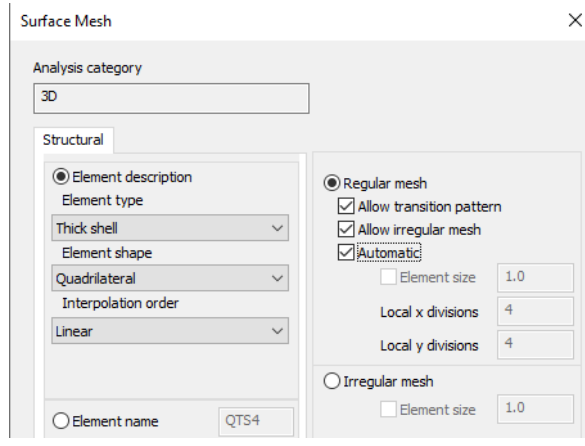
The ‘Divisions’ input optionally specifies the number of mesh divisions in both directions. The ‘Size’ input optionally defines a fixed element size for the mesh. If both inputs are left undefined, the ‘Automatic’ option will be set on the Surface mesh dialog within LUSAS. The component only allows one of the two inputs to be specified at any time and reports an error otherwise.

If an irregular mesh is specified, only the ‘Size’ input is valid, and an error is shown if any input is connected to the ‘Divisions’ input.

Inputs	Outputs
(Name) – *single line text*	Surface mesh
(Divisions) – *numeric, > 0, default: 4*	
(Size) – *numeric, ≥ 0, default: 0*	



The surface mesh component.

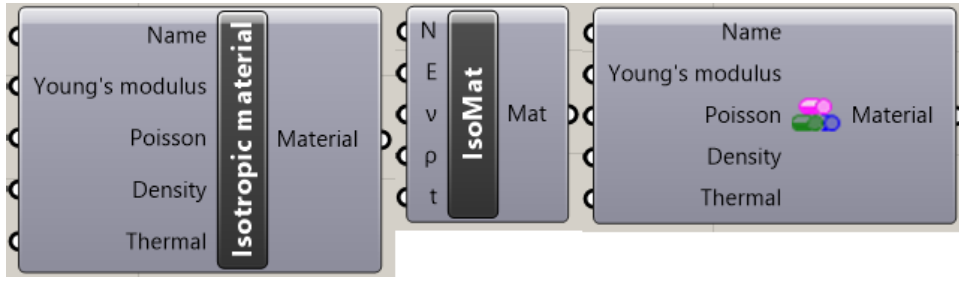


Surface mesh dialog in LUSAS.

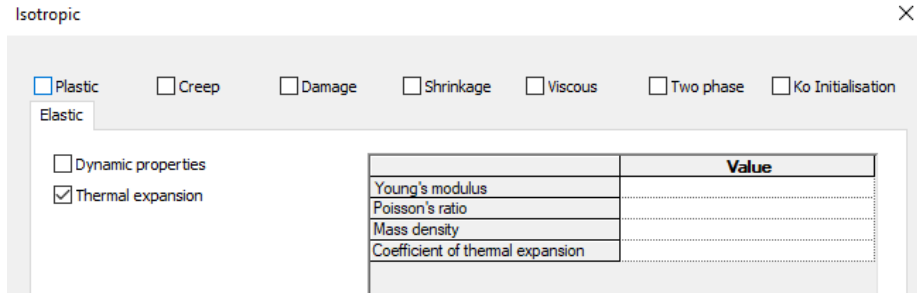
Isotropic material

Isotropic material inputs match the inputs in Modeller's Isotropic dialog.

Inputs	Outputs
(Name) – *single line text*	Isotropic material
Young's modulus – *numeric, > 0*	
Poisson ratio – *numeric, > 0*	
Density – *numeric, > 0*	
(Thermal expansion coefficient) – *numeric, > 0, default: 0*	



The isotropic material component in all possible display modes.



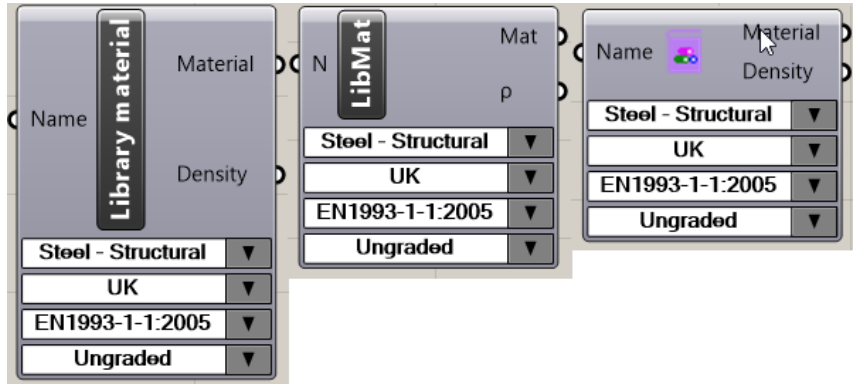
Isotropic material dialog in Modeller.

Library material

With the library material component you can select from the available materials in the LUSAS material library using the custom dropdown menus. Note that the plugin will first look for a valid LUSAS installation, in lieu of which it will fall back to the locally installed library files that come with the plugin installation.

The component outputs the material density (converted to model units) in case it can be used for other calculations.

Inputs	Outputs
(Name) – *single line text*	Isotropic material Density – *numeric*



The library material component in all possible display modes.

Parametric section

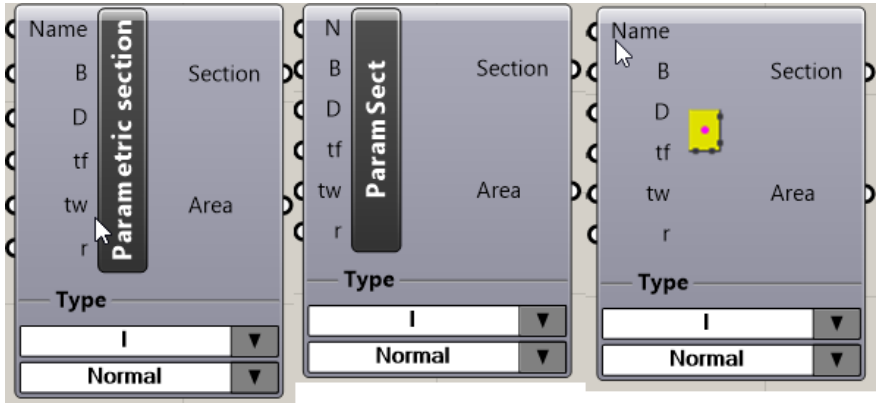
The parametric section component features two custom dropdown menus where you can select the type and subtype of the parametric section. When the selection changes, the component inputs apart from the 'Name' input are updated to reflect the selected type and subtype, while any already connected inputs will be preserved only for parameters that match in name and order with the previous selection. For example, in the scenario where the selection is changed from rectangular section (B, D) to rectangular hollow section (B, D, t, ri, to), the inputs to B, D parameters will be preserved.

The component also shows descriptive tooltips for each parameter and supports optional parameters, for example for the I section the r (radius) parameter.

Although most of the parametric sections have an available preview shape, the component will show an informative message if a preview is not possible.

The component also outputs the section area (converted to model units) in case it can be used for other calculations.

Inputs	Outputs
(Name) – *single line text*	Parametric section
Required parameters e.g. B, D – *numeric, > 0*	Area – *numeric*
(Optional parameters e.g. r) – *numeric, > 0, default: 0*	



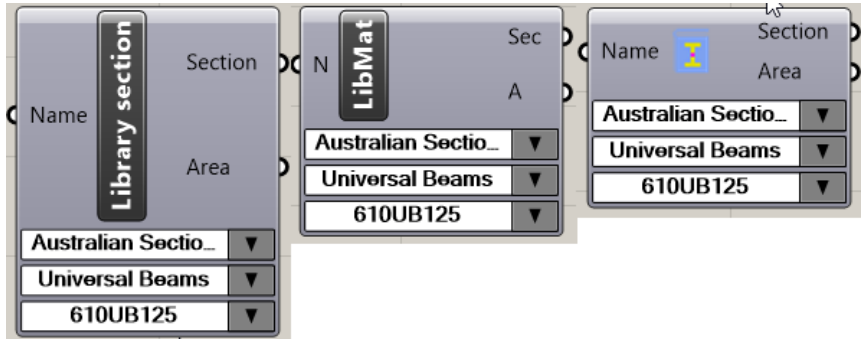
The parametric section component in all possible display modes; I section.

Library section

With the library section component you can select from the available sections in the LUSAS section library using the custom dropdown lists. Note that the plugin will first look for a valid LUSAS installation, in lieu of which it will fall back to the locally installed library files that come with the plugin installation.

For any section type that does not have an available preview shape, the component will show an informative message.

Inputs	Outputs
(Name) – *single line text*	Library section
	Area – *numeric*



The library section component in all possible display modes.

Arbitrary section

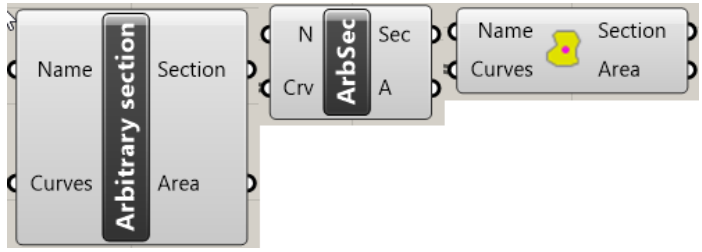
The arbitrary section component accepts a list of curves as an input, which define the shape of an arbitrary section. These curves need to form one or more closed loops in the XY plane, which will be trimmed in Modeller to create the section shape with any internal holes.

The component validates that the curves are planar in the XY plane and that they form closed loops. Otherwise, an error is reported, and no output is provided.

Section preview is by default available for any arbitrary section.

The component also outputs the section area (converted to model units) in case it can be used for other calculations.

Inputs	Outputs
(Name) – *single line text*	Arbitrary section
Curves – *native Rhino type*	Area – *numeric*

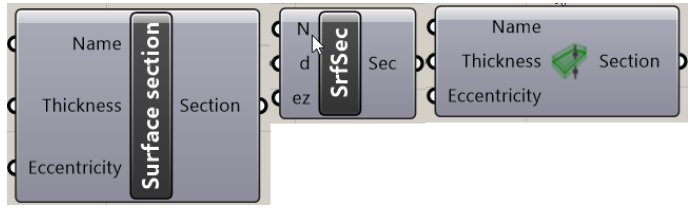


The arbitrary section component in all possible display modes.

Surface section

The surface section component is the equivalent of the surface geometric attribute in LUSAS. It allows for the definition of a surface's thickness and eccentricity.

Inputs	Outputs
(Name) – *single line text*	Parametric section
Thickness – *numeric, > 0*	
(Eccentricity) – *numeric, default: 0*	



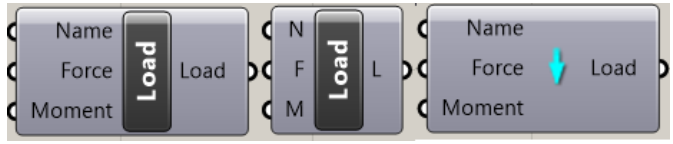
The surface section component in all possible display modes.

Load

The 'Load' component allows the definition of load attributes, by specifying a force and a moment 3D vector for the load values. The moment load values will be ignored if the load is assigned to a surface member.

Inputs	Outputs
(Name) – *single line text*	Load
(Force) – *3D numeric vector, default: (0,0,0)*	

(Moment) – *3D numeric vector, default: (0,0,0)*

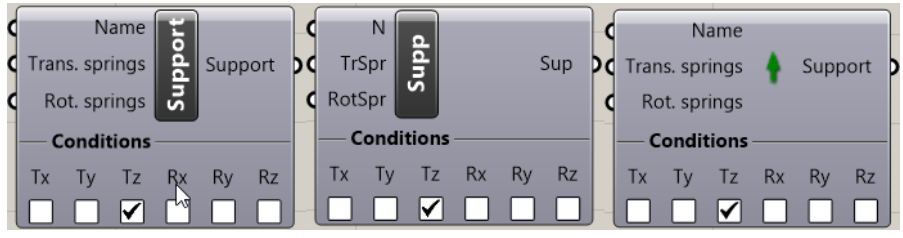


The load component in all possible display modes.

Support

The support component allows for the definition of a support attribute. Select the fixed degrees of freedom by enabling the respective checkbox controls. The 'Translation/Rotation springs' inputs each accept a 3D vector for the spring values. Any non-zero spring value will override the 'Conditions' controls and will force the respective degree of freedom to use the specified spring value.

Inputs	Outputs
(Name) – *single line text*	Support
Translation springs – *3D vector, numeric, > 0, default: (0,0,0)*	
Rotation springs – *3D vector, numeric, > 0, default: (0,0,0)*	



The support component in all possible display modes.

Assignments

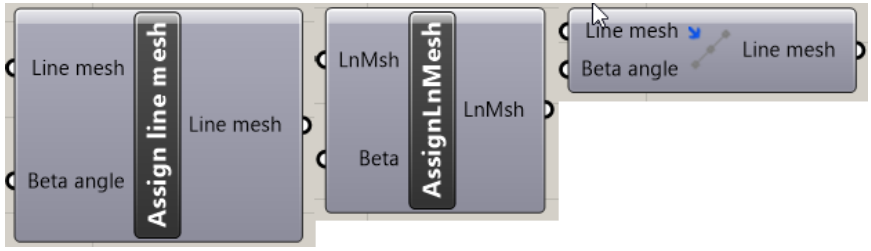
The assignment components sit between the attribute definition components and the inputs of member of model definition.

Assign line mesh

The line mesh assignment component accepts a line mesh and a beta angle as inputs, allowing you to customise the beta angle for specific members (in degrees). Use of this component is optional; the output of the line mesh component itself can be directly connected to a line member and assume a beta angle of 0.

This component will also check if the definition of a beta angle is accepted by LUSAS for the specific line mesh type. For example the following line mesh types do not accept a beta angle and will produce an error on the component: none, bar.

Inputs	Outputs
Line mesh – *output of the Line mesh component* Beta angle – *numeric*	Line mesh

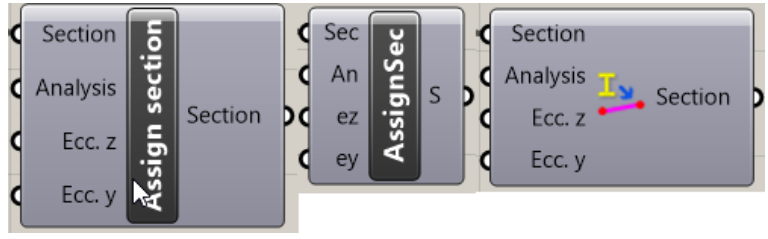


The assign line mesh component in all possible display modes.

Assign section / material

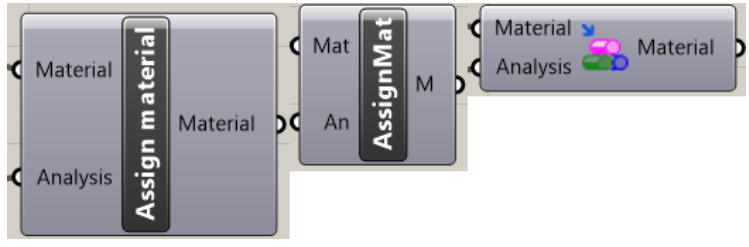
The section and material assignment components allow for specifying a specific analysis and or optional eccentricity values in the case of a section. Their use is optional, i.e. the output of the section or material components can be directly assigned to members assuming the base analysis of the model.

Inputs	Outputs
Section – *output of any Section component*	Section
Analysis – *single line text*	
(Ecc. z) – *numeric, default: 0*	
(Ecc. y) – *numeric, default: 0*	



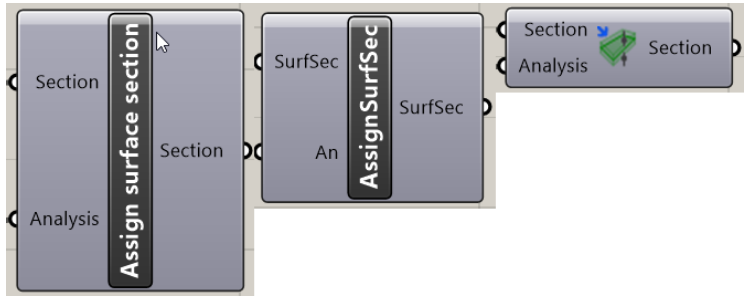
The assign section component in all possible display modes.

Inputs	Outputs
Material – *output of any Material component*	Material
Analysis – *single line text*	



The assign material component in all possible display modes.

Inputs	Outputs
Section – *output of the Surface section component*	Surface section
Analysis – *single line text*	



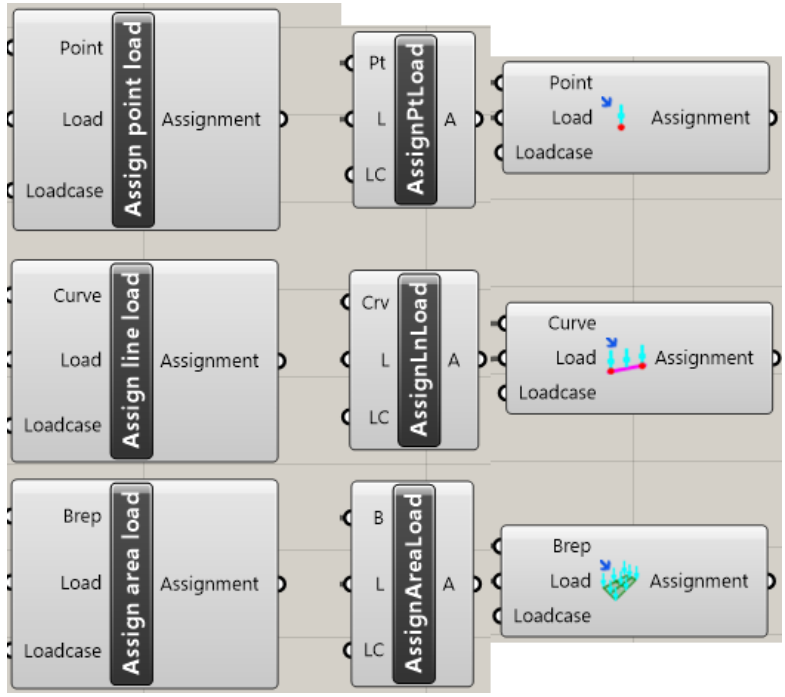
The assign surface section component in all possible display modes.

Assign point/line/area load

Loads are assigned by specifying the underlying geometry (points, curves or breps) which must match that of a corresponding line/surface member. This is necessary for point loads in order to know which point of a line member to assign them to, and also for line loads assigned to the edges of surface members.

Load assignment components also have the ‘Loadcase’ input, where you must specify a valid loadcase name, as is defined via an ‘Analysis’ component connected to the ‘Model’ component.

Inputs	Outputs
Point(s)/Curve(s)/Brep(s) – *native Rhino types* Load – *output of the Load component* Loadcase – *single line text*	Point/Line/Area load assignment



All assign load components in all possible display modes.

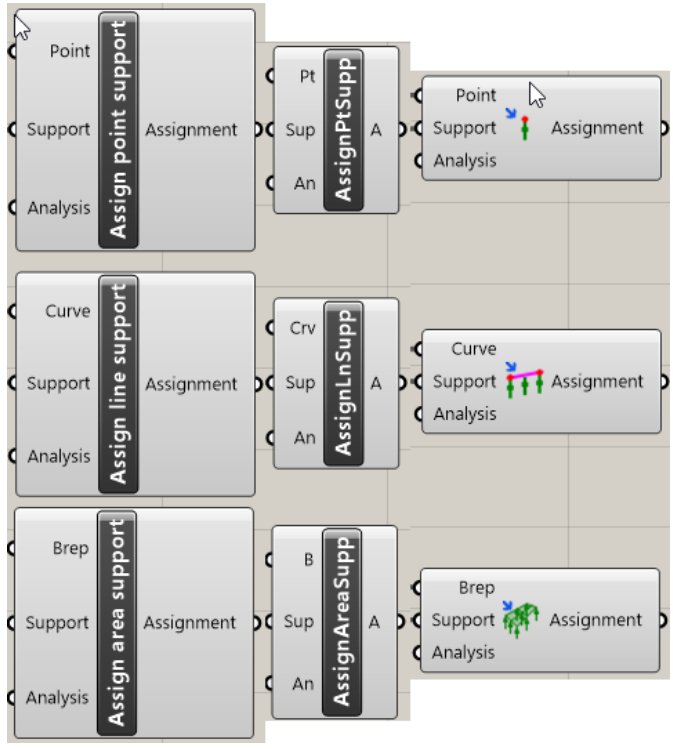
Assign point/line/area support

Supports are assigned by specifying the underlying geometry (points, curves or breps) which must match that of a corresponding line/surface member. This is necessary for point supports in order to know which point of a line member to assign them to, and also for line supports assigned to the edges of surface members. Line supports on line members do not require a curve specified.

Support assignment components also have the ‘Analysis’ input, where an analysis name can be optionally specified, as is defined via an ‘Analysis’ component connected to the ‘Model’ component. If lieu of data in this parameter, the base analysis of the model will be assumed.

Inputs	Outputs
Point(s)/Curve(s)/Brep(s) – *native Rhino types* Support – *output of the Support component*	Point/Line/Area support assignment

(Analysis) – *single line text*



All assign support components in all possible display modes.

Model components

Export

The Export component can be used to export the model to a LUSAS model file (mdl) or to a LUSAS Visual Basic script (lvb) for loading into LUSAS at a later date.

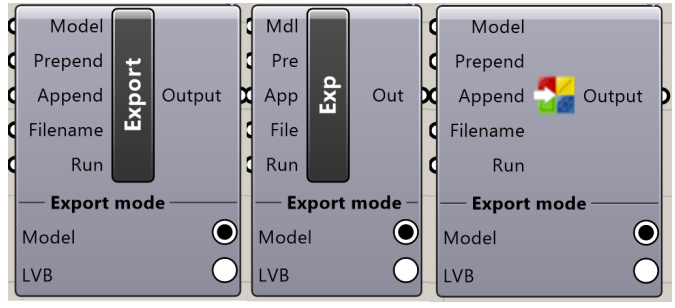
The ‘Filename’ input parameter can be optionally used to specify the path and filename for the exported file. In lieu of data in this parameter, a dialog is shown so a filename can be selected.

The ‘Run’ parameter is used to turn this component on/off, enabling full control over its operation.

The ‘Output’ parameter will contain the LVB script output for the current model and will be populated even if the ‘Run’ parameter is not activated. This can be connected to a ‘Panel’ in order to inspect the LVB script. Due to major performance issues of panels with text content that is too long, the output will be trimmed after 60000 characters.

Note that the model export option requires a background instance of LUSAS.

Inputs	Outputs
Model – *output of the LUSAS model component*	Output – *multiple line text*
(Prepend) – *multiple line text*	
(Append) – *multiple line text*	
(Filename) – *text*	
(Run) – *boolean*	



The export component in all possible display modes.

Live link

The Live link component enables live updates to an open model in a running instance of LUSAS. The model in LUSAS is updated with the modifications in the Grasshopper script components with every link operation.

Live link requires that a single instance of LUSAS is running with an open model. It is possible for you to start LUSAS manually before issuing a link operation. The component can also start a LUSAS instance to create the model the first time or open an existing model to link to.

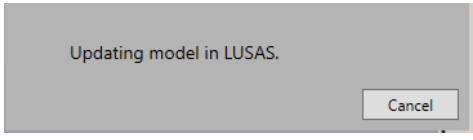
As model updates might be a costly operation, the ‘Run’ parameter is used to turn the component on/off so you have full control of when the model updates will be issued. This is especially useful when a lot of modifications are queued in succession (such as sliding a value slider).

Vanilla Grasshopper components such as ‘Button’ or ‘Boolean toggle’ are some examples of ways to control the link operation. Note that by using the ‘Button’ component any error messages on the Live link component will not be available.

Two link modes are available:

- **Whole model** – the whole model will be updated in LUSAS with each link operation.
- **Modifications only** – only the modified parts of the model in the Grasshopper script will be updated in LUSAS. This might reduce the time required to link substantially for large models.

During the linking process no interaction with Grasshopper is possible, and a progress window is shown to inform you of the current status until the operation is complete. The ‘Cancel’ button can be used to forcefully disable the component from further link operations even if the ‘Run’ parameter is set to true. This works as a failsafe in case multiple link operations have been queued (for example by a wide value slider swing). To reset the disabled state, the ‘Run’ parameter needs to be toggled.

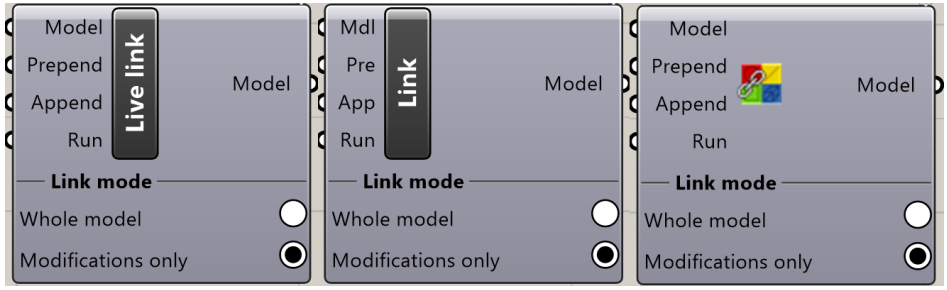


It is possible to prepend or append additional text to the final script by connecting text to the designated input parameters.

During the live update, if the model units and/or analysis category is different between the ‘LUSAS model’ component in the Grasshopper script and the model in LUSAS, a dialog displayed for you for confirmation that these properties will be updated in LUSAS. By selecting ‘Cancel’ on this dialog, the link process is cancelled.

Note that the ‘Live link’ component will be forcefully disabled when a Grasshopper document is opened, to prevent any accidental link operation if the ‘Run’ parameter was left in a ‘True’ state when the document was saved.

Inputs	**Outputs**
Model – *output of the LUSAS model component*	Model
(Prepend) – *multiple line text*	
(Append) – *multiple line text*	
(Run) – *boolean*	



The live link component in all possible display modes.

LUSAS Model

The LUSAS model component assembles all of the model information: line/surface members, analyses and their loadcases/loads, load/support assignments and the model units.

The ‘Members’ input accepts both line and surface member inputs. If no members are supplied, the component reports an error as a model without members does not make any sense.

The ‘Analyses’ input accepts the output of one or more analysis components. If no data is supplied, the model is created with 1 analysis (‘Analysis 1’) and 1 loadcase (‘Loadcase 1’) by default. The loadcase names must be unique among all analyses. The base analysis is selected as follows:

- If any analysis component has the ‘Base analysis’ checkbox enabled, it will be marked as base.
- If multiple analysis components have the ‘Base analysis’ checkbox enabled, the first one in order will be assumed with a warning message.
- If no analysis components have the ‘Base analysis’ checkbox enabled, the first one in order will be assumed.

The ‘Assignments’ input accepts the output of the load and support assignment components.

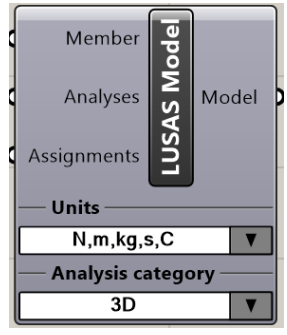
The component also has two dropdown lists:

- Selection of the desired model units, which must be the same in Rhino and LUSAS. An error message is shown on the component if the units do not match, and no output is produced.

- Selection of the desired analysis category, which must be compatible with the 'Line/surface mesh' attributes used for members in the model. An error is shown otherwise and no output is produced.

Note that in LUSAS attributes of the same type must have unique names. In case duplicate names are found, a unique name will be created automatically by appending a dash (-) and a number to the original attribute name (does not apply when using 'Live link' to link with a live model, in which case all names must be unique).

Inputs	Outputs
Member(s) – *output of any Member component* (Analyses) – *output of the Analysis component, default: 'Analysis 1'* (Assignments) – *output of any Assign load/support component*	LUSAS Model



The LUSAS model component.

Model preview

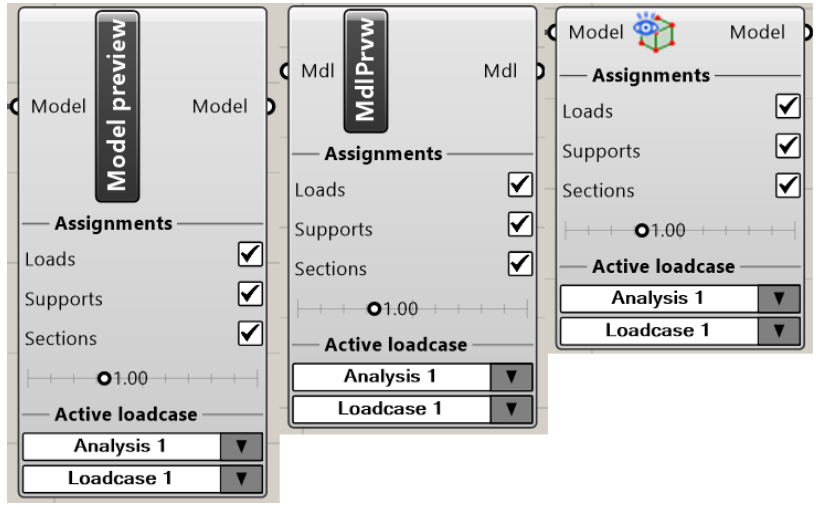
The model preview component is responsible for the preview of member sections as well as any load and support assignments in Rhino's viewport. This component must be used with an existing 'LUSAS Model' as input in order for any preview to take place. It outputs the 'Model' again so that it can be used between the 'Model' and 'Live link' components, or standalone.

If features settings for the preview in the form of custom controls:

- The 'Loads'/'Supports'/'Sections' checkboxes can be used to toggle the respective previews altogether.

- The value slider can be used to change the scale of the load/support preview props in the viewport.
- The active analysis/loadcase dropdown lists can be used to select the analysis/loadcase from which to show sections/loads/supports.

Inputs	Outputs
Model – *output of the LUSAS model component*	Model



The model preview component in all possible display modes.

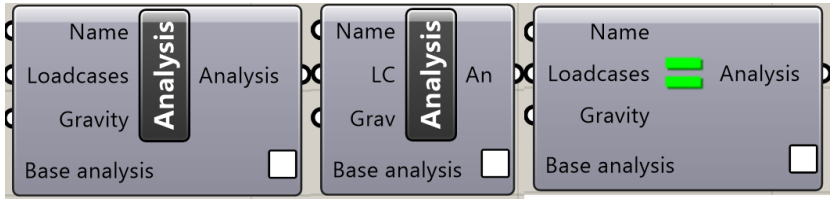
Analysis

The analysis component allows the definition of analyses for the LUSAS model. Each analysis has a 'Name' and a 'Loadcase(s)' input, where you specify the analysis name and any loadcase names to be associated with the analysis. The 'Loadcase' input accepts a list of unique loadcase names (either list of texts or multiple lines of text), in the order they will appear in the model. These loadcases must be referred to by name in the load assignment components.

The 'Gravity' input enables the definition of gravity loads for the entire analysis or for specific loadcases using the Gravity component.

The 'Base analysis' checkbox sets this analysis as the base analysis in the model.

Inputs	Outputs
Name – *single line text*	Analysis
Loadcase(s) – *single/multiple line text*	
(Gravity) – *output of the Gravity component*	

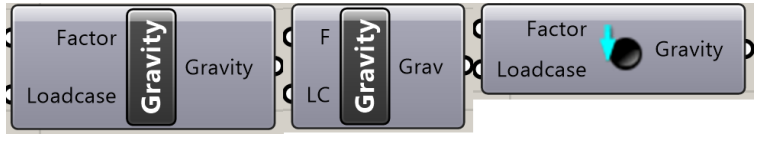


The analysis component in all possible display modes.

Gravity

The gravity component enables the definition of gravity loads for specific loadcases with the specified load factor. It is possible to use data matching to specify different load factors for each loadcase. The output is used as an input to the analysis component.

Inputs	Outputs
(Factor) – *number, default 1.0*	Gravity
(Loadcase) – *multiple texts*	



The gravity component in all possible display modes.

Advanced components

Custom attribute

The custom attribute component is an advanced component enabling the definition of any LUSAS attribute by specifying the LUSAS Programmable Interface (LPI)

commands required. With this component, attributes that do not yet have their dedicated component can be created.

The ‘DefineLPI’ parameter is used for the attribute definition while the ‘AssignLPI’ parameter is used for the attribute assignment.

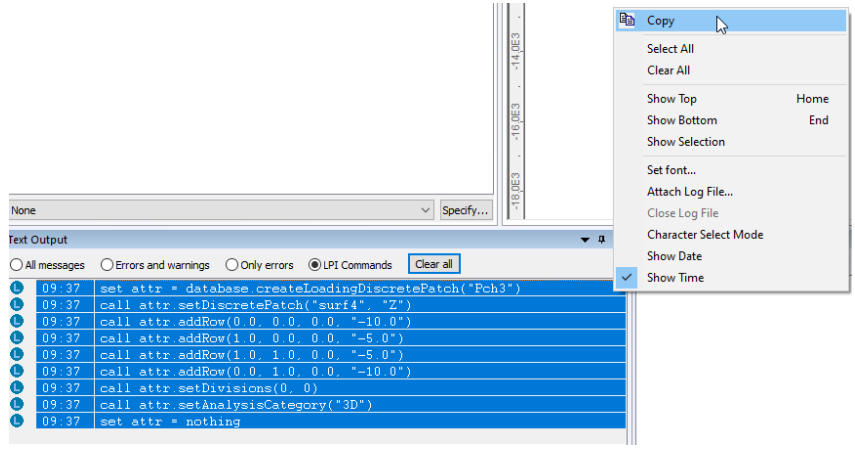
Definition LPI

To obtain the LPI commands for the definition of an attribute, you can record a script using the **File> Script> Start recording** menu item in LUSAS or by switching to LPI Commands in the text output window. After manually creating the attribute in LUSAS, the LPI commands can be copied and used as the input to the ‘DefineLPI’ parameter.

For example, the image below shows the commands that should be copied for the definition of a discrete patch load, starting from the command that creates the attribute (‘`set attr = database.create…`’), including all commands which specify the attribute values up to the ‘`set attr = nothing`’ command (optional).

Including commands from below or above these will probably lead to an issue with the model. The attribute name will be exactly as stated in the LPI.

Note that for now, it is mandatory to use ‘attr’ as the variable name (just like LUSAS) otherwise the component will report an error.



Example of copying the LPI commands directly from the text output window.

Assignment LPI

The assignment LPI commands are optional as most of the attributes in LUSAS use the same commands for their assignment to features. The following commands assign a support attribute to the selected features:

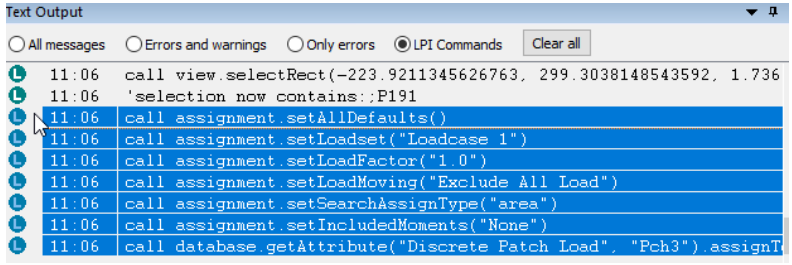
```
call assignment.setAllDefaults()
call assignment.setLoadset("Loadcase 1")
```

```
call database.getAttribute("Structural Support", "Pinned").assignTo(selection,
assignment)
```

For attributes that require special commands for their assignment the same process as with the definition LPI can be followed, this time by manually assigning the attribute to the selected features in LUSAS. The LPI commands can then be copied and used as the input to the 'AssignLPI' parameter.

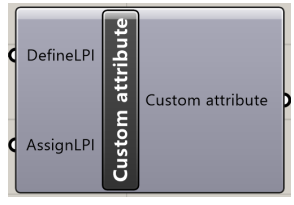
For example, the image below shows the commands that should be copied for the assignment of a discrete patch load, starting from the command '`call assignment.setAllDefaults()`', including all commands which specify the assignment options up to the '`call database.getAttribute(…).assignTo(…, assignment)`' command (optional). The plugin will modify or append the `assignTo` command to correspond with the structural members to which it is assigned in Grasshopper.

Including commands from below or above these will probably lead to an issue with the model.



Example of the LPI commands for the assignment of a discrete patch load.

Inputs	Outputs
DefineLPI – *multiline text* (AssignLPI) – *multiline text*	Custom attribute



The custom attribute component.

Refer attribute

The refer attribute component can be used to assign attributes that are defined elsewhere to structural members in the Grasshopper script. Such attributes could, for example, be defined in the live LUSAS model manually or with custom LPI commands as a prepended script on the Export/Live link components. As such, attributes that do not yet have their dedicated component can be assigned to structural members in the Grasshopper script.

The ‘Name’ and ‘Type’ parameters are required to reference the attribute in the LUSAS model. The type parameter is not directly shown in LUSAS, it can however be determined in a few different ways:

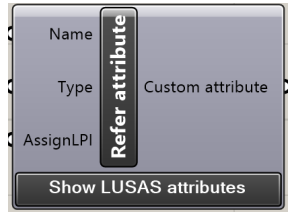
- By using the ‘Show LUSAS attributes’ button on the component (see below).
- By looking at the topmost category the attribute appears in the ‘Attributes’ tab in LUSAS (e.g. Mesh, Geometric, Material, Support, Loading etc).
- By looking at the LPI Commands text output and manually renaming or assigning the attribute in the model. A command such as ‘`call database.getAttribute("Loading", "Load1").setName("Load2")`’ will be shown which indicates that the type of the attribute is ‘Loading’.

Some of the most often used types are: ‘Mesh’, ‘Geometric’, ‘Material’, ‘Support’, ‘Loading’, ‘Local coordinate’, ‘Design’.

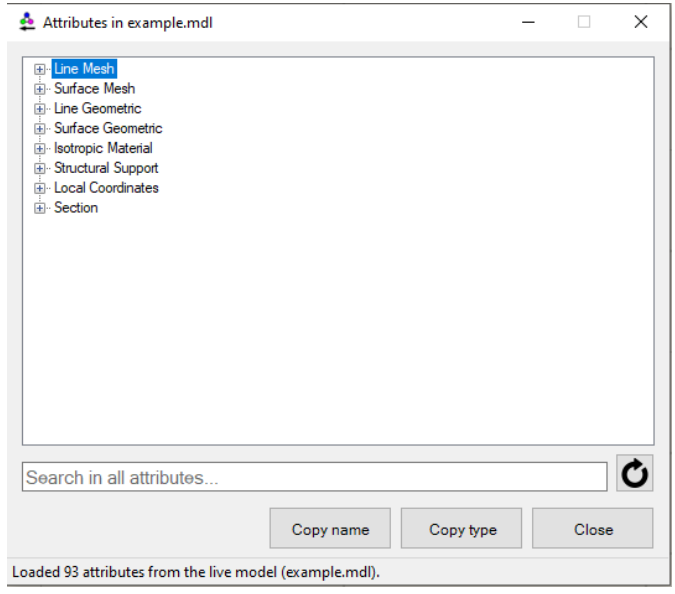
For the ‘AssignLPI’ parameter, refer to the description in the Custom attribute section.

The ‘Show LUSAS attributes’ button can be used to connect to a running instance of LUSAS and show a list of all attributes in the model to facilitate the process of finding the correct type for an existing attribute.

Inputs	Outputs
Name – *text*	Custom attribute
Type – *text*	
(AssignLPI) – *multiline text*	



The refer attribute component.



The ‘Show LUSAS attributes’ dialog.

LUSAS

LUSAS, Forge House, 66 High Street, Kingston upon Thames, Surrey, KT1 1HN, UK
Tel: +44 (0)20 8541 1999 | Fax: +44 (0)20 8549 9399 | info@lusas.com | www.lusas.com