

ParaView/VoxFE2

User guide
March 2015

Table of Contents

Overview	3
Main features of the ParaView window.....	4
The Manage Plugins dialog.....	8
Using VoxFE2	9
VoxFE2 toolbar	10
Reading images: the '.voxfe' Group file format.....	12
Options in the Properties tab	14
Properties panel for loading VoxFE files.....	15
Load model panel options	16
Adding boundary conditions	17
Nodal boundary conditions (Properties panel).....	19
Parallel force loading condition	22
Force to endpoint loading condition	23
Outputting the FE problem for solution	24
Change Input Dialog window (select model)	25
Change Input Dialog window (select BCs)	27
Right-click Browser dialog	29
Visualizing the results.....	30
VoxFE2 Strain Load Dialog window.....	32
2D and 3D plotting	33
An example session	34
Building the VoxFE2 plugin	40
To Do:	41

Overview

ParaView

ParaView is an open-source, multi-platform tool for visualization based on the Visualization Toolkit (VTK) library. In visualization problems, there is very rarely a single solution and what is needed is often more of a steering process, where we would like to back-track and try different perspectives. The approach in ParaView is to present the user with a 'pipeline' where different processing algorithms (called 'filters') can be tried and tested without affecting the source data. There are also numerous options for running ParaView, loading and saving data. Whilst those features which are of particular relevance to VoxFE are summarised below, the reader is strongly encouraged to explore ParaView first with example data sets available from [the ParaView website](#). At the time of writing, we use ParaView version 4.1, see [Main features of the ParaView window](#).

There are also a number of good video tutorials for ParaView on YouTube (see [Navigating around the model](#)). One option for extending ParaView, which VoxFE adopts, is to make use of the plugin interface. A large number of such plugins are distributed with ParaView and these provide methods for reading different file formats, extracting particular sections of the data and so on. A plugin exists in the form of an external library, which is usually loaded after the main program has been started, so that there is a very specific mode of operation. A plugin can, however, be automatically loaded as the program starts and is accessible either through the *Filters* menu of ParaView, or if provided, buttons in the menu bar.

VoxFE2

VoxFE aims to assist the assembly and analysis of finite element (FE) problems based upon a highly regular voxel mesh (usually derived from image data). From the FE perspective, the VoxFE ParaView plugin provides a number of supporting functions:

- Reading VoxFE data files (unstructured grid data)
- Selection of nodes at mesh/boundary surfaces to specify boundary conditions (nodal constraints and forces)
- Output of these data for solution by a separate solver
- Reading strain/displacement data (from the solver) to visualize the results.

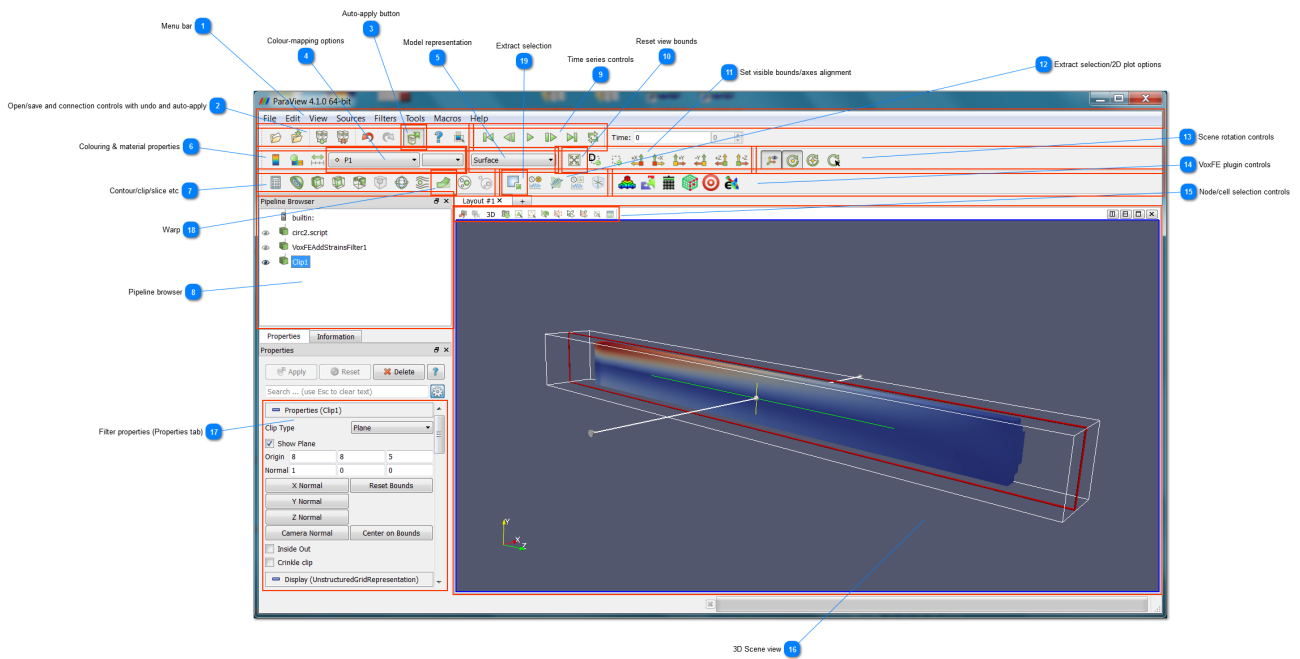
At the time of writing, an introductory video tutorial for VoxFE can be found [here](#), but note that this is already a little out of date (especially with regard to importing displacement/strain data). We expect to update YouTube posts as more features are added.

Note that the solver is currently a separate module which has been developed for unix or linux-based, multi-processor environments and access to such a resource is recommended.

Loading the VoxFE2 plugin

When compiled the plugin appears as a shared object library (linux) or a dynamic link library (MS windows) eg. libVoxFETools.so, VoxFETools.dll. The plugin can be loaded through the [Manage Plugins dialog](#) available from the Tools Menu (via *Tools > Manage Plugins > Load New*). The status of the plugin is shown in this menu tree; if successfully loaded, it should be visible as a small collection of buttons in the ParaView menu bar, see below and the Status should appear as "Loaded". We expect to distribute the plugin as a pre-compiled binary for Windows 7 or Ubuntu 14.04 (both 64-bit). Compiling from source code requires some expertise and some guidance is given in the section: [Building the VoxFE plugin](#).

Main features of the ParaView window



1 Menu bar

File Edit View Sources Filters Tools Macros Help

2 Open/save and connection controls with undo and auto-apply

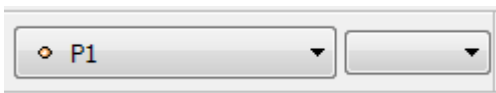


3 Auto-apply button



Apply changes to model parameters automatically

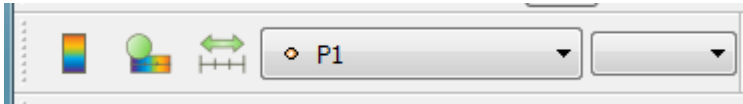
4 Colour-mapping options



5 Model representation



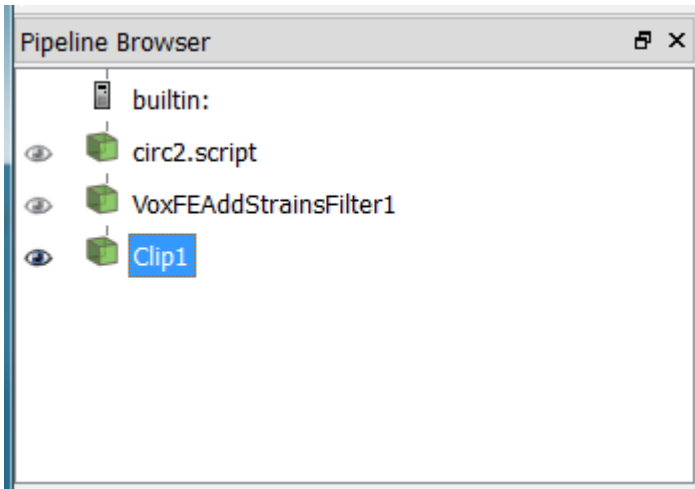
6 Colouring & material properties



7 Contour/clip/slice etc



8 Pipeline browser



9 Time series controls



10 Reset view bounds



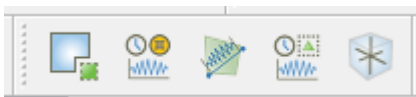
Bring all objects in the scene into view

11 Set visible bounds/axes alignment

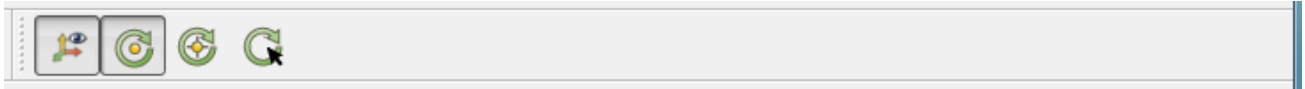


Aligning the model with axes can assist with node selection

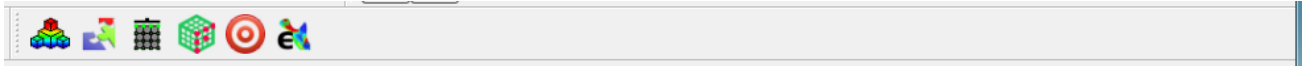
12 Extract selection/2D plot options



13 Scene rotation controls



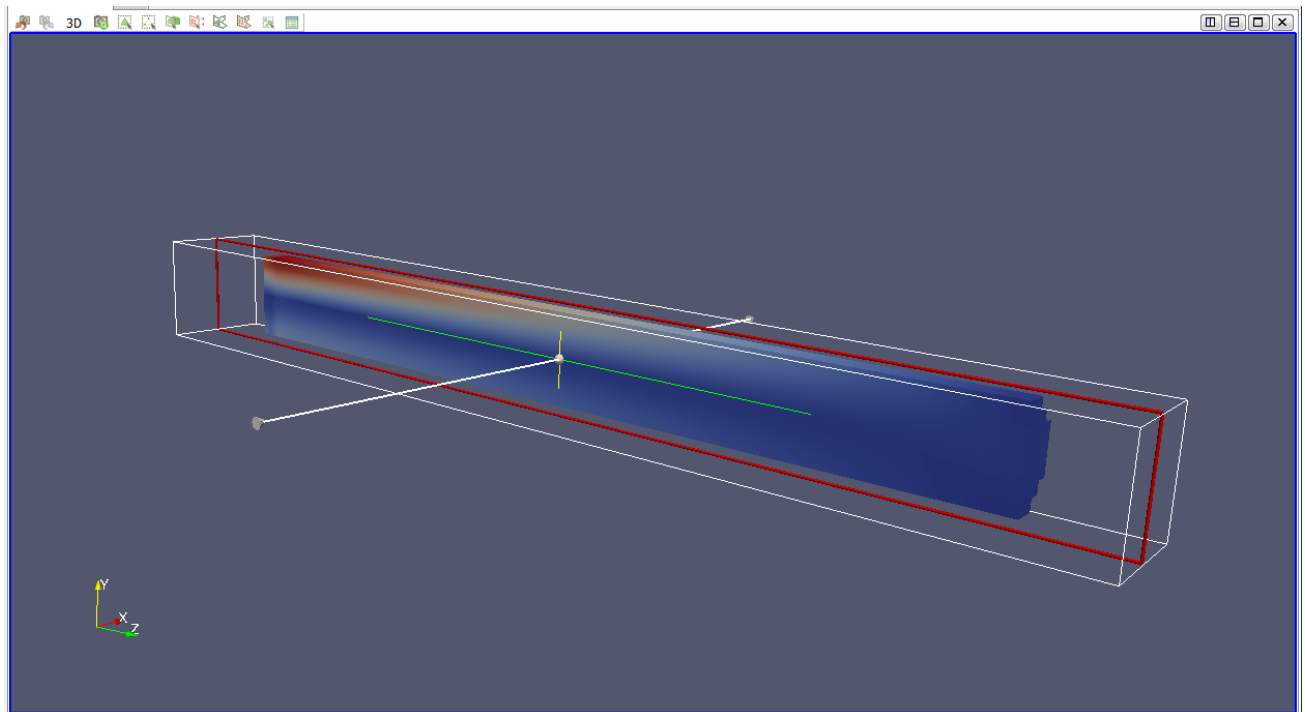
14 VoxFE plugin controls



15 Node/cell selection controls

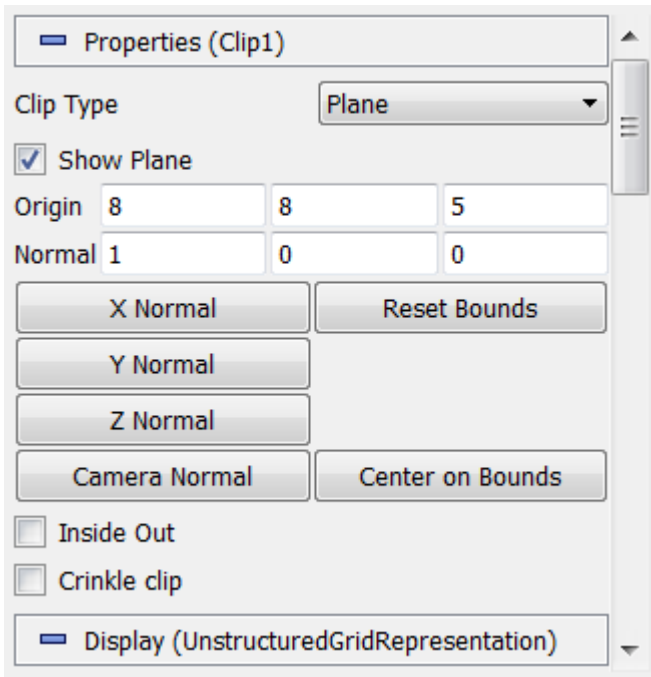


16 3D Scene view



Note the buttons along the top edge: to the left are the node/cell selection controls; to the right multiple views can be created, splitting the screen vertically or horizontally. Alternatively, new tabbed windows can be created using the tab buttons above (not shown).

17 Filter properties (Properties tab)



These panels (unique to each filter) allow the user to set relevant parameters

18

Warp



Warp By Vector (eg. to view displacements)

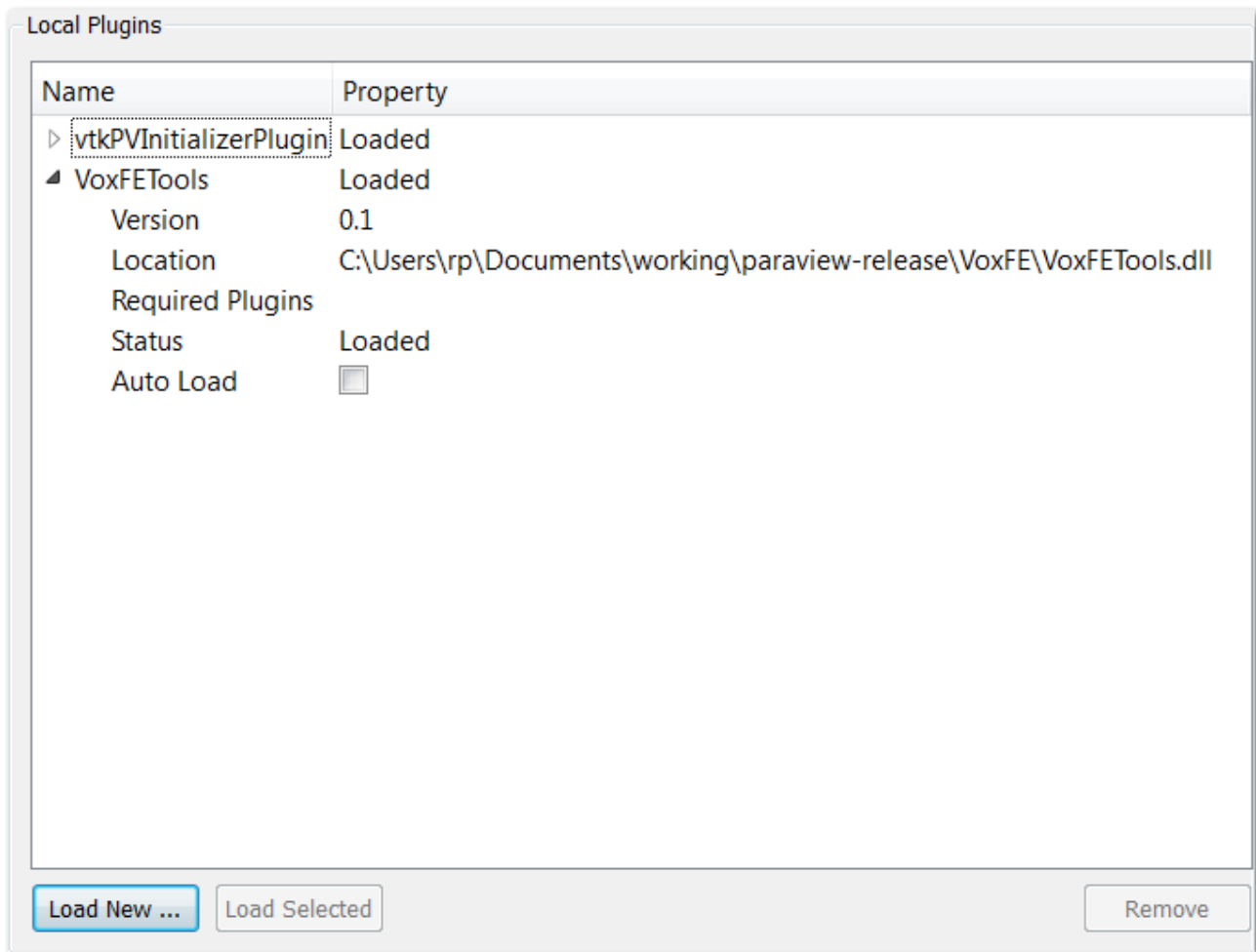
19

Extract selection



Used to create separate node selections

The Manage Plugins dialog



Using VoxFE2

Navigating around the model

A tutorial on using ParaView is available [here](#); a guide has also been released by [KitWare](#). Most of the navigation controls lie above the location of the plugin in the [ParaView GUI](#) figure and include resetting (to centre the object within the view), viewing along selected axes and selection of the centre of rotation. Several videos are now available on YouTube showing, for example, [basic usage](#), use with the [PCL plugin](#), [clipping](#) and [isosurfacing](#).

Although the connected components algorithm (see [Options in the Properties tab](#)) will remove disconnected objects, it is particularly important to check the model for other errors in image processing which may have caused the model to appear overly weak or fused. The ParaView *Clip* or *Slice* tools can help with this but present the user with only a 2D view in which weaknesses can be missed (see item #7 of [the ParaView window](#)). In this case, changing the opacity (by adjusting the slider in the *Properties panel*), or extracting blocks, may be invaluable. Blocks may be extracted via menus (*Filters > Alphabetical > Extract Block*) or by the [Extract block](#) button, shown in [the VoxFE toolbar](#) (button #2). Besides convenience, the only difference between these options is that the property "Prune Output" is set to "On" in the ParaView filter. *VoxFE* turns this parameter off, as it can generate unwarranted point selections when attempting to define boundary conditions.

The VoxFE2 toolbar

After loading the plugin, the [VoxFE toolbar](#) should appear in the main [ParaView toolbar display](#) (item #14). If there is an error, the toolbar will not load and it is likely that the correct path for the libraries has not been set, or there is a mis-match between versions. It may be advisable to try the release version of *ParaView/VoxFE2*. Also -- try to see if any of the other plugins distributed with ParaView load without issue, such as SLACTools (*VoxFE* is similar in form to SLAC).

The toolbar currently comprises six buttons, which are used, roughly, from left-to-right (along with some of the other ParaView functions). As far as ParaView is concerned, these buttons appear as separate filters and are also available in the *Filters* menu; though the user will generally find the toolbar buttons much more convenient. Just as most other filters, when invoked, the *VoxFE* buttons have default values for their parameters which are displayed in the *Properties tab* in ParaView, below the *Pipeline browser*.

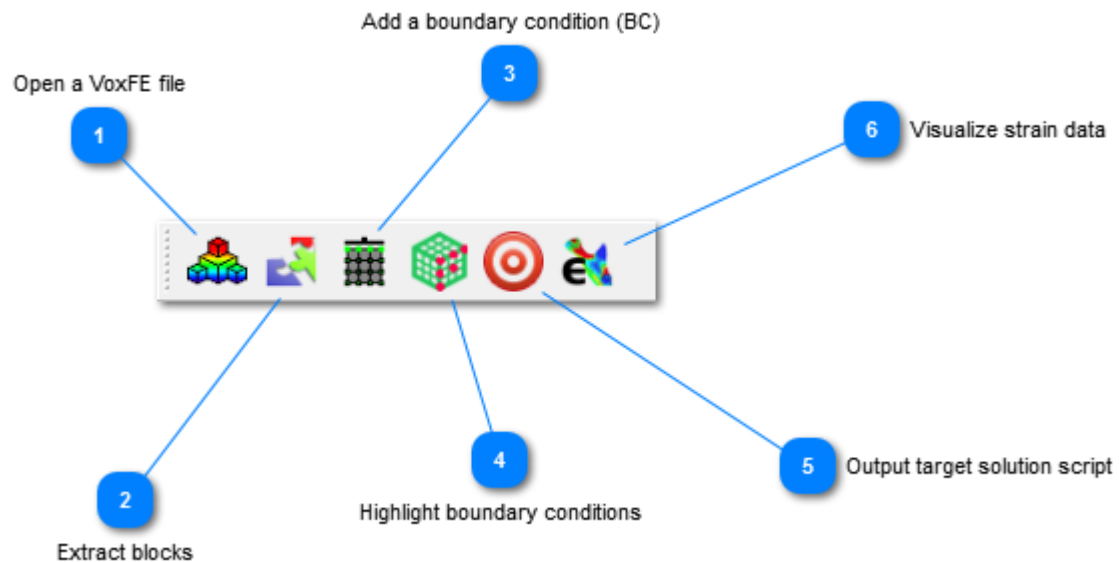
The *VoxFE* plugin buttons are described in turn below.

Loading image data

By default, the filters in ParaView need to be activated by pressing the *Apply* button (visible in the *Properties tab*). The reason is simply because when working with larger data sets, it is preferable to examine and (re-)set the the default parameters before running a potentially expensive algorithm. Before loading any data then, ensure that *Auto-apply* is disabled, via the button to the top of the menu display, see [Main features of the ParaView window](#) (button #3). (For more interactive visualization tasks, it may well be preferable to enable *Auto-apply*.) Click the button to load VoxFE data, shown in [the VoxFE toolbar](#) (button #1), and select the file browse button ('...') to navigate to your data file. Script ('.script') and Group ('.voxfe') file formats are searched for by default.

The '.script' file format was developed in previous versions of VoxFE: it is a basic text file which can be easily hand-edited and is the main input to the VoxFE solver (see solver documentation). Whilst the *VoxFE* plugin can read the model data from script files, group files are needed to work with raw image data and to allow labels to be logically grouped eg. labels 1-50 might define a number of different bone material types, which for the purposes of adding boundary conditions, can all be treated as a single 'bone' type. Creating a group ('.voxfe') file is described [below](#).

VoxFE2 toolbar



1 Open a VoxFE file



Open a file browser dialog to import voxel data from a `.voxfe` or `.script` file [[Open VoxFE file](#)].

2 Extract blocks



Separate VoxFE groups, as defined in a `.voxfe` file [[Extract blocks](#)].

3 Add a boundary condition (BC)



Add a boundary (displacement) or force loading condition [[Add BC](#)].

4 Highlight boundary conditions



Highlight the different boundary conditions applied so far (changes point size and colour) [[Highlight BC](#)].

5 Output target solution script



Write out the constraints to complete the script files needed to generate the target solution [[Output script](#)].

6 Visualize strain data





Open a file browser dialog to import strain data and colour-map to the model surface (by default) [[Import strain data](#)].

Reading images: the '.voxfe' Group file format

Basic definition

The basic format of the VoxFE Group file (with '.voxfe' extension) is:-

```
#Hashed lines indicate comments
#(any number of these can be included at the top of the file)
#Note: Group definitions are given as:
#starting-label finishing-label group-name remodelling-status
#[More hashed comments...]
LOAD_IMAGE ./block.mhd
GROUPS 3
1 99 Bone 1
100 150 Teeth 0
250 255 Dilated 0
```

where:-

- **LOAD_IMAGE** This loads a 3D image in the Insight Toolkit (ITK) Metaheader ('.mhd') file format to create an unstructured grid model.
- **GROUPS** Here, 3 groups are defined (Group 1: labels 1-99; Group 2: labels 100-150; and Group 3: labels 250-255). The essence of the group definition is that, say for 'Bone', we can define a number of different types of bone (eg. cortical, trabecular ...) with varying Young's moduli which can be treated as one unit for the purpose of display. Interior points and geometry are removed (by default) to improve rendering speed and to allow the surface points to be more easily selected when defining boundary conditions. The remodelling flag allows remodelling if set (ie. if non-zero).

The Group file can also read a series of 2D image files (bmp,png,jpg), given a definition such as:-

```
#hashed comment 1:
#hashed comment 2... etc
LOAD_IMAGE ./sq-%1d.bmp
VOXEL_SIZE 0.1
BORDER_OFFSET 10 10 10
GROUPS 0
```

The argument given to the **LOAD_IMAGE** command here is of the form of a [C-printf specifier](#), where, for name %NNNN.bmp:-

- % – indicates the beginning of the numeric identifier;
- NNNN indicates each file number;
- d – indicates a decimal integer (and will always be used here);
- 0 – if a zero is given, as for example "%03d", then we know that the numeric identifier is 3 characters wide and padded by zeros such as 001, 002, ... 998, 999;
- 1 – eg. "%1d", if the numeric identifier is not padded by spaces or zeros, then this would imply that the smallest number was 1 character wide (ie. 0, 1 ...). The specifier "sq-%1d.bmp" would match 10 possible files: sq-0.bmp, sq-1.bmp, ..., sq-9.bmp.

Working with 2D image files

If a 2D image series (bmp,png,jpg) is being read, the user should supply beginning and end numbers (see [Load model panel options](#)). If any of the requested files to match do not exist, the *VoxFE* plugin loads all the image files contained within the selected directory and creates a log file to show which files were loaded. The log file will be named after the identifier nominated in the Group file eg. “./block.mhd_v.log” or “sq-_1d.bmp_v.log”.

Important: note that VoxFE only works with 8-bit grey-scale images.

The BORDER_OFFSET command is optional, but if it appears, should be given just before the GROUPS option. This command is intended to be used with bone remodelling commands (described elsewhere). Typically one might wish to add a small border around an object to allow for sufficient growth without generating negative index voxels. A border size of zero (BORDER_OFFSET 0 0 0) is legal.

The voxel size command (VOXEL_SIZE) must also be supplied when working with 2D image files. This is because once the Group file and data have been read, the *VoxFE* plugin creates a stub ‘.script’ file, which can then be used by the solver and hence dimensions are required. Since no boundary conditions will have been set as yet, this is not a well-defined problem, but the user is encouraged to record the correct voxel size for future use.

Groups

When no groups have been defined (ie. “GROUPS 0” is given), each label encountered in the image(s) is considered to be a separate group and the remodelling flag is assumed to be “1” ie. permitted. For groups defined with remodelling flag set to “0”, then those materials will not be allowed to shrink or grow regardless of computed strain values.

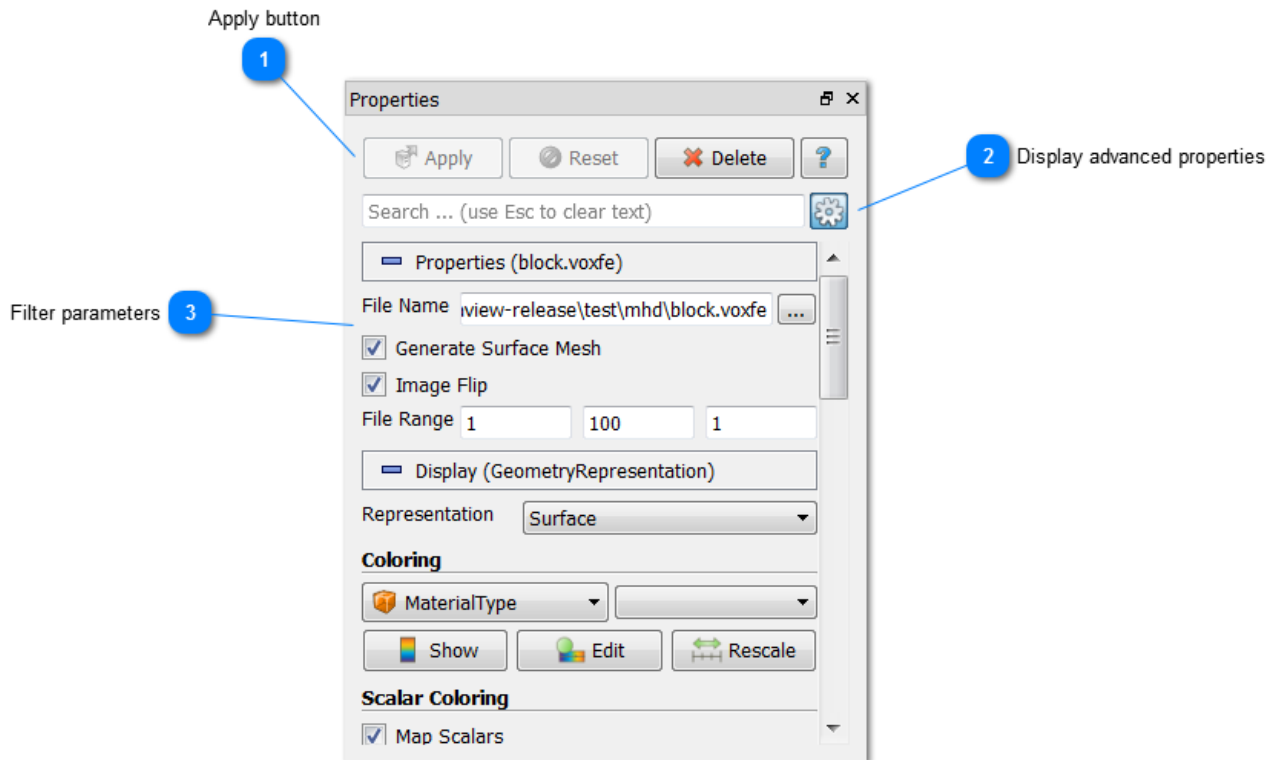
Options in the Properties tab

If you try to load a *VoxFE* file (say, "test/mhd/block.voxfe" in the release version), the *Properties panel* below the *Pipeline Browser* will switch to that shown in the figure, [panel for loading VoxFE files](#). This shows the default options used to load a test block model, which contains a number of artificially constructed but disconnected regions. Given the selected options in the *Properties tab*, the following actions occur when reading in this image data:-

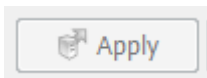
- All images are read using the functions from the ITK library: bitmap (bmp) and ITK mhd formats are currently supported (though more could be added). The resulting greyscale image is filtered by ITK's connected components algorithm to remove all but the largest object in the image ie. we assume that all smaller disconnected objects can be rejected as noise.
- The remaining object will be composed of separate named groups according to the supplied group definitions. These are assembled in ParaView into a 'multi-block' object, which can be inspected by switching to the Information tab. [The *Extract Block* filter can also be applied to this object (available via the *VoxFE* plugin toolbar) if these meshes need to be separated visually.]

The panel options are discussed in [Load model panel options](#).

Properties panel for loading VoxFE files



1 Apply button



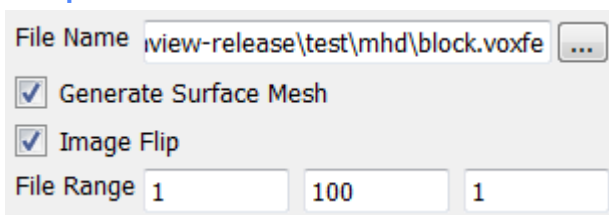
Select this after setting desired parameters

2 Display advanced properties



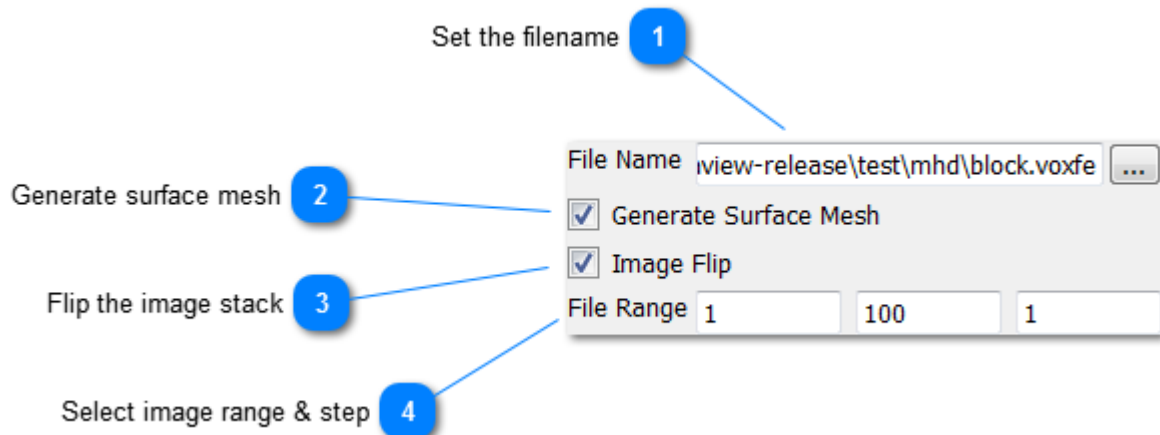
Toggle on/off (this has no effect currently for this filter)

3 Filter parameters



See text

Load model panel options



1 Set the filename

view-release\test\mhd\block.voxfe ...

This property specifies the file name for the VoxFE reader; the `...' button opens a *File browser* dialog box to select a model file

2 Generate surface mesh

☒ Generate Surface Mesh

Remove internal geometry from each defined grouped material: this option (on by default) removes interior geometry from the groups identified in the image. To examine the results of analysis, however, eg. after solution of the FE problem, it may be more useful to retain the interior geometry. Deselecting this option will completely reload the model, but will allow the user to investigate interior values eg. of strain (see Section [Visualizing the results](#))

3 Flip the image stack

☒ Image Flip

This option reverses the order of the images that are loaded along the stack of slices. This makes more sense in the case of a 2D image series (bmp,png, jpg) which happens to have been numbered from top-to-bottom order

4 Select image range & step

1 100 1

Set the number of the first, last and increment for image files to be read in as part of the current image. Again, this option only makes sense in the case of an image series, should you only wish to load a small section of the image slices. (Presently, this selection has no effect upon data loaded from mhd image files or pre-existing `.script' files.) The first two boxes represent the starting and finishing images desired; the third box defines a step value, which may be useful to avoid intermediate image slices, so that cubic voxels can be obtained

Adding boundary conditions

Adding a boundary condition (BC) is at least a 3-step process:-

1. Nodes must be selected, such that all points in the selection will share the same BC;
2. The selected points must be "extracted" to form a separate identity;
3. Additional data defining the BC must added.
4. (Optional) Highlight the BCs and adjust the glyphs to check.
5. (Optional) Save the BC data.

Whilst it may be a common error to over-constrain FE models, the plugin does not attempt to evaluate whether too many or insufficient constraints have been set. Conceptually, the user must ensure that nodal constraints are sufficient to avoid translation or rotation of the model if placed under some loading condition. **Note: BC data are not automatically saved with the model -- use 'Save state' to do this if you wish (see below).**

(1) Selection

The buttons for selection of nodes, cells or blocks are situated above the 3D view area (item #15 of [the ParaView window](#)). When the interior geometry has been removed, it is usually easiest to choose to select points with a simple rectangle. For more complex geometry, polygon selection may be more convenient. Note that point selections can be combined by pressing the Ctrl key at the same time as making selections, but that there is no similar native method to remove points from a selection. A complex point selection, however, can be reduced by clipping the extracted data, using ParaView's *Clip* filter.

(2) Extraction

Once points have been selected, pressing the Extract Selection button (denoted as #19 in [the ParaView window](#)), creates a separate entity in the ParaView pipeline. This is necessary in order to create a permanent structure within the ParaView pipeline which does not modify the original data.

Note that, ordinarily, the extraction filter separates points if the selection region happens to overlap more than one group/block (points are effectively assigned to each respective block). For this reason, VoxFE adds the *MergeBlock* filter when a BC filter is added. This can normally be ignored by the user except under the condition that you wish to delete the BC filter, when it will also be advisable to delete the associated *MergeBlock* filter as well (after deleting the BC filter).

The default behaviour for this function is to switch off the visibility of the original mesh data; visibility can be restored by selecting the eye symbol in the *Pipeline browser*, or by defining the BC, as below.

(3) Adding BC data

The VoxFE plugin provides a button to add BCs, visible in [the VoxFE toolbar](#) as button #3. Ensure that the extracted selection of points is highlighted in the Pipeline Browser and that *Auto-apply* is off. Click the *Add BC* button — a new filter is created (*VoxFEBoundaryConditionN*). The plugin assumes that the initial state of any added BC is "none" ie. no condition is imposed. This allows some flexibility later if we wish to remove BCs temporarily and allows the condition to be further defined before accepting. Currently, 5 BC types are allowed:-

1. None
2. Nodal constraints
3. Force (parallel)

4. Force (to endpoint)
5. No remodelling

Note that the selected BC dictates the form of the property display panel, so that only the required properties are requested. These are shown in the figures captioned [below](#). Nodes are considered to be constrained along an axis if the respective check box is selected. For parallel force conditions, a direction vector and magnitude must be specified. For 'to point' force conditions, the vector end-point and magnitude must be specified. Both force BCs have an additional drop-down box (*Distribution*) to specify whether the load is to be added identically to each data point (ie on each *Node*), or divided equally across all points in the BC (*Area*).

The "No remodelling" option is not a true FE boundary condition as such, but is used with remodelling options to protect areas near other BCs from being remodelled.

Glyphs are small icons added to the data to indicate whether the BC represents what was intended. In VoxFE, we add 2D triangles to signify that nodes are constrained along a particular axis and a single (2D) arrow to show the direction of any added forces. The glyphs, when created, are placed at the centroid of the data and may be hidden initially. See the [example session](#) to see how to manipulate the glyphs for better visibility.

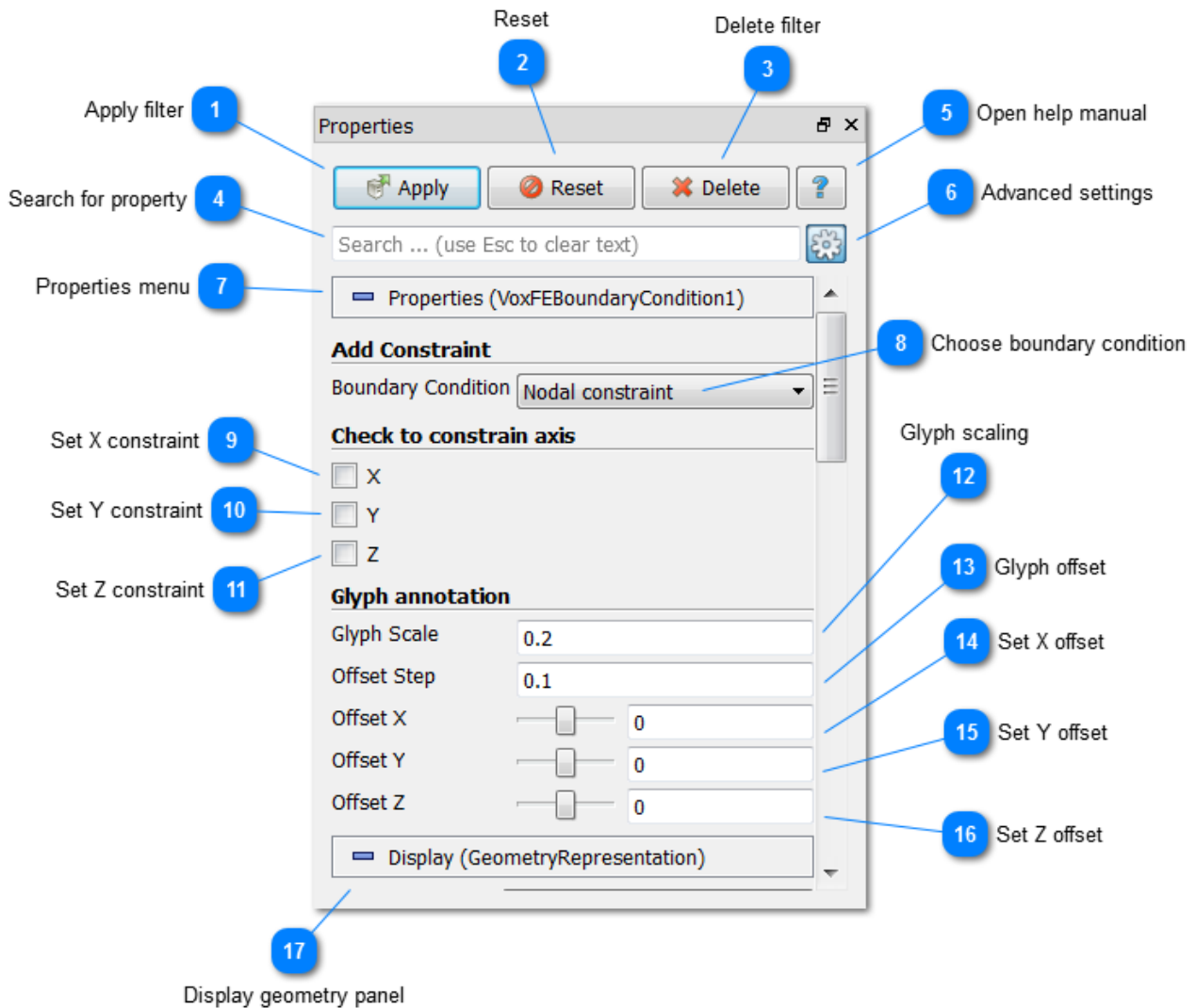
(4) Highlighting boundary conditions

After setting one or more boundary conditions, clicking this button (item #4 of the [VoxFE toolbar](#)) adds a larger point size and pseudo-randomly assigned colour definitions to highlight defined BC. The user may further emphasize point-sizes etc. by editing the *Coloring* and *Styling* options of the *Properties tab* for each individual BC.

(5) Saving the BC data to the model

By using *File > Save State...* the entire model can be reloaded later at any time from the saved control file. Note that in using the reverse (*File > Load State...*) the entire session is recreated, rebuilding boundary conditions etc as before, so that this may be a relatively lengthy operation.

Nodal boundary conditions (Properties panel)

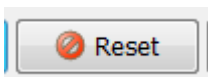


1 Apply filter



Apply the filter with the current settings

2 Reset



Reset to default settings

3 Delete filter



Delete the filter

4 Search for property

Search ... (use Esc to clear text)

Search for properties by name (eg. here for 'x')

5 Open help manual



6 Advanced settings



Toggle advanced properties on/off (note: this should be on to show the glyph offset sliders, below)

7 Properties menu

Properties (VoxFEBoundaryCondition1)

Toggle *Properties menu* on/off

8 Choose boundary condition

Nodal constraint

This property indicates which boundary condition will be set for the selected nodes

9 Set X constraint

☐ X

10 Set Y constraint

☐ Y

11 Set Z constraint

☐ Z

12 Glyph scaling

0.2

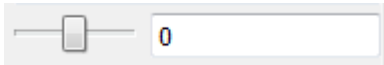
Specify the scaling of the glyph for this boundary condition

13 Glyph offset

0.1

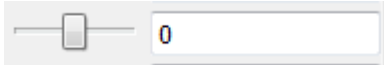
Specify the offset step of the glyph for use with slider, to nudge the glyph into view (a good initial guess would be roughly the mesh size)

14

Set X offsetA slider control with a horizontal track and a central knob. To the right of the track is a text input field containing the number 0.

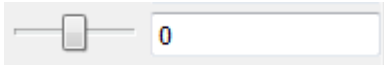
Specify the X offset of the glyph in conjunction with the step size, to nudge the glyph into view

15

Set Y offsetA slider control with a horizontal track and a central knob. To the right of the track is a text input field containing the number 0.

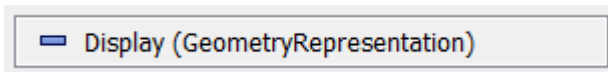
Specify the Y offset of the glyph in conjunction with the step size, to nudge the glyph into view

16

Set Z offsetA slider control with a horizontal track and a central knob. To the right of the track is a text input field containing the number 0.

Specify the Z offset of the glyph in conjunction with the step size, to nudge the glyph into view

17

Display geometry panelA toggle control consisting of a small square button with a minus sign icon, followed by the text "Display (GeometryRepresentation)".

Toggle representation property on/off

Parallel force loading condition

Select boundary condition

1

Add Constraint

Boundary Condition **Force (parallel)**

Enter vector data

Force Vector 0 0 0

Magnitude 0

Distribution **Node**

Glyph annotation

Glyph Scale 0.2

2 Set the force vector

3 Magnitude

4 Distribution

5

Client area

1 Select boundary condition

Force (parallel)

Selecting this option applies a parallel force to each node

2 Set the force vector

0 0 0

This property specifies the X,Y,Z force vector components to apply at each point

3 Magnitude

0

Specify the total force magnitude

4 Distribution

Node

Set whether the total force should be allocated on each node or distributed over all nodes

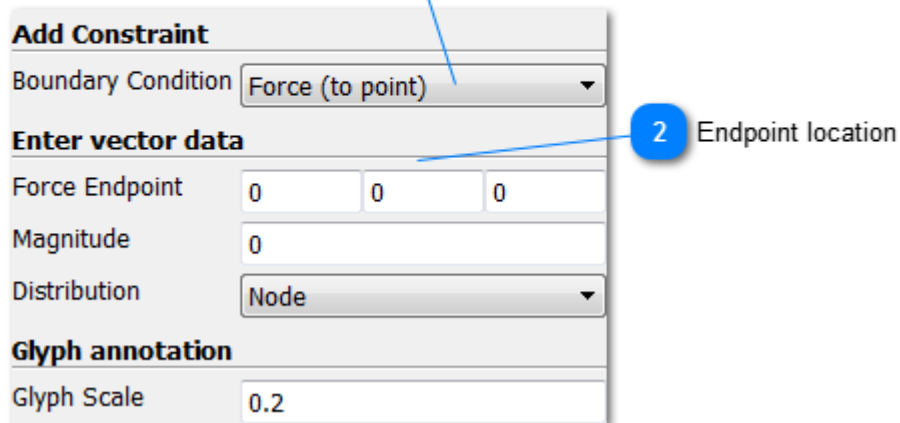
5 Client area

0.2

Specify the scaling of the glyph for this boundary condition

Force to endpoint loading condition

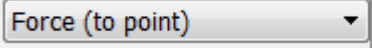
Select force to endpoint



The screenshot shows the 'Add Constraint' dialog box. A blue circle with the number '1' points to the 'Boundary Condition' dropdown menu, which is set to 'Force (to point)'. Another blue circle with the number '2' points to the 'Force Endpoint' input fields, which are set to '0', '0', and '0'. The text 'Endpoint location' is placed next to circle 2. The dialog box has sections for 'Add Constraint', 'Enter vector data', and 'Glyph annotation'.

Add Constraint			
Boundary Condition	Force (to point)		
Enter vector data			
Force Endpoint	0	0	0
Magnitude	0		
Distribution	Node		
Glyph annotation			
Glyph Scale	0.2		

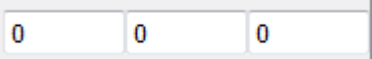
1 Select force to endpoint



A close-up of the 'Boundary Condition' dropdown menu, showing the selected option 'Force (to point)'.

If this option is selected, vector forces are computed from each node to the endpoint

2 Endpoint location



A close-up of the 'Force Endpoint' input fields, showing three boxes each containing the value '0'.

This property specifies the target endpoint for which to compute each force vector

Outputting the FE problem for solution

The 'target solution' button (item #5 of the [VoxFE toolbar](#)) invokes a *Change Input Dialog*, as this filter requires multiple inputs. These are sub-divided into VTK "ports" which need to be allocated as:-

Port 0:

Input Model: this needs to be directed towards the input file (Reader) filter eg. "block.voxfe"
eg. see [Change Input Dialog window \(select model\)](#)

Port 1:

Boundary Conditions: one or more boundary conditions must be selected as inputs to this port (using Ctrl + left-mouse-click)
eg. see [Change Input Dialog window \(select BCs\)](#)

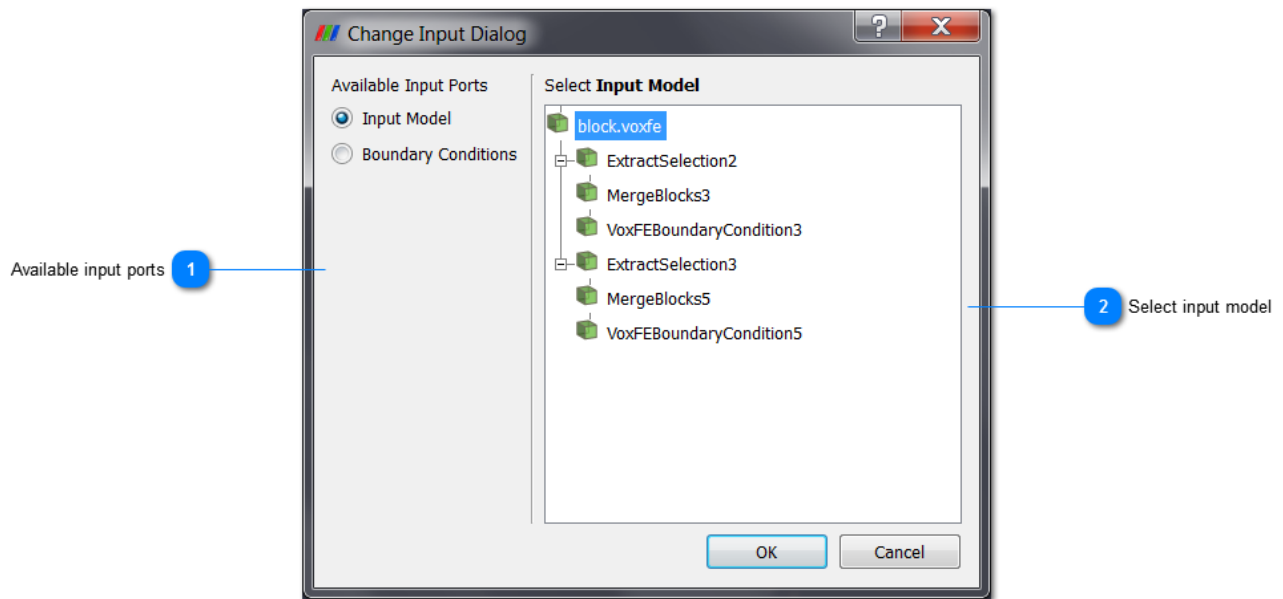
It is advisable to double-check that these have been set correctly before finally clicking "OK". Note that, other than the appearance of the *VoxFEOutputScript* filters in the *Pipeline browser*, there are no further outputs to this filter. A file, "constraints.txt", should be created in the directory of the original data model, which can be used by the target script.

Bug: Updating the Change Input dialog

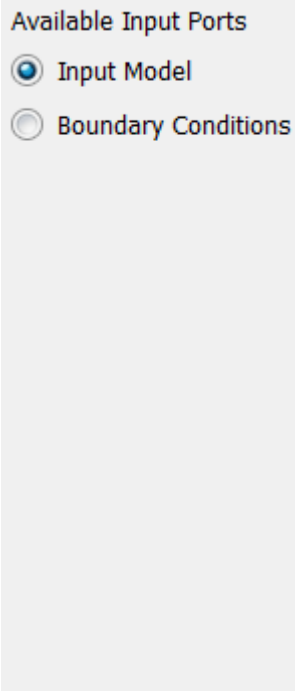
You may notice that the *Change Input Dialog* can be invoked by right-clicking on filters in the *Pipeline browser* (see [Right-click Browser dialog](#)). This can be useful if the wrong object is active (ie. selected) when you apply another filter and this generates unexpected results. However, in this case, invoking the *Dialog* a second time only adds to the existing port inputs, most likely generating an error.

A simple workaround exists: use the same right-click dialog to delete existing *VoxFEOutputScript* objects and the 'target solution' button of the [VoxFE toolbar](#) to re-generate from scratch.

Change Input Dialog window (select model)

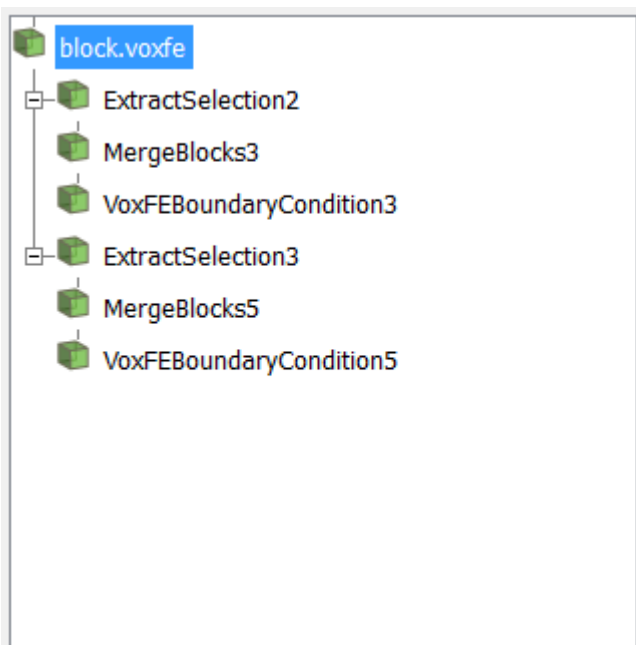


1 Available input ports

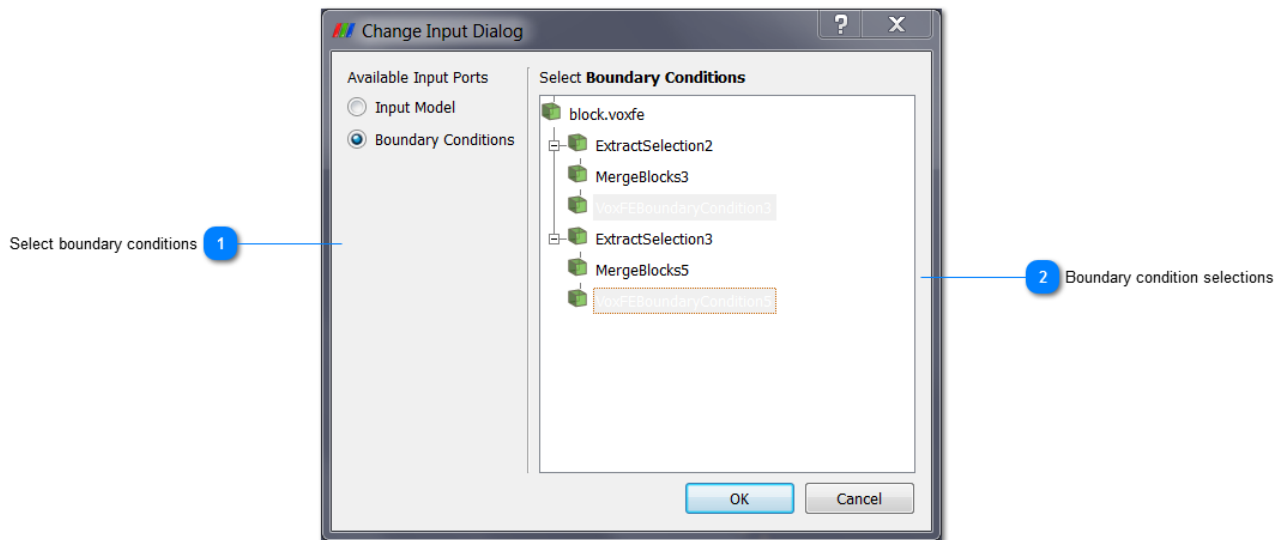


2 Select input model

2

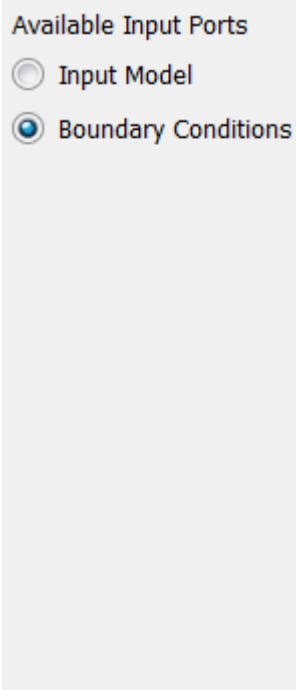


Change Input Dialog window (select BCs)



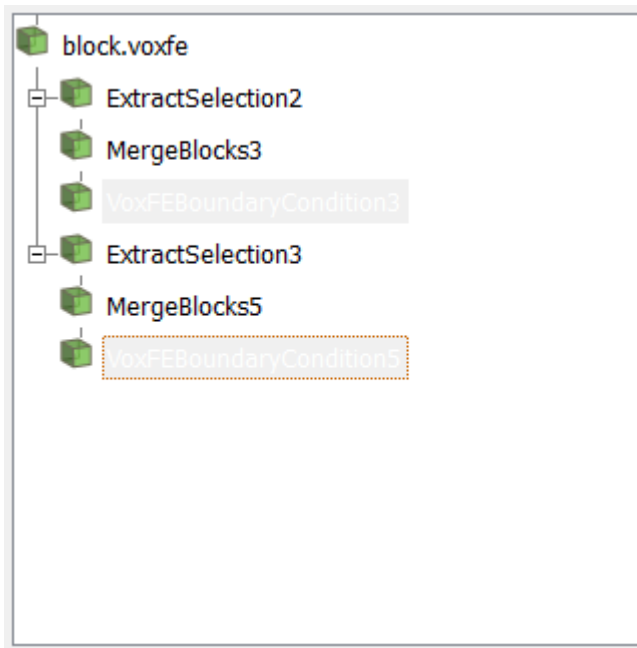
1

Select boundary conditions



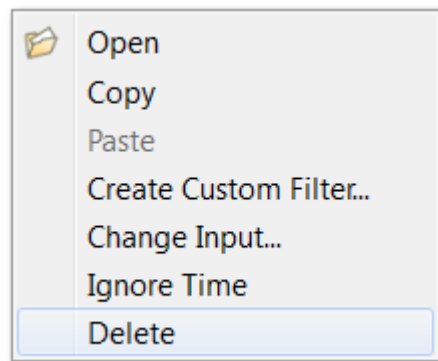
2

Boundary condition selections



Use Ctrl + click (left mouse button) to make multiple selections to input to this port

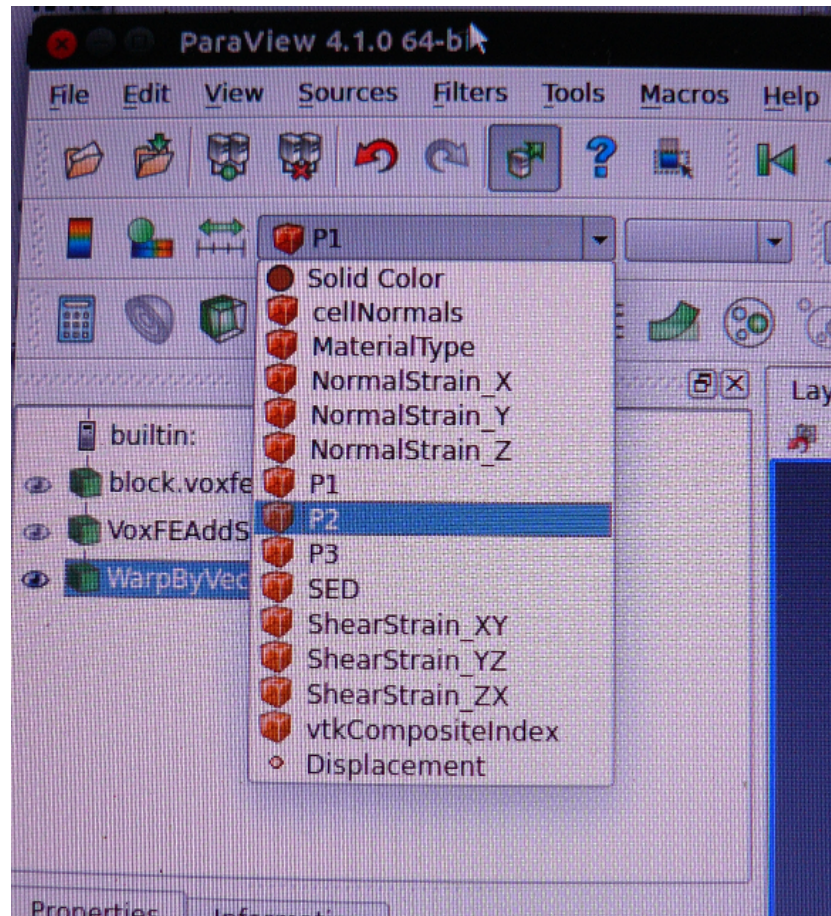
Right-click Browser dialog



Visualizing the results

Load and select the input model to make it active. Clicking button #6 of the [VoxFE toolbar](#), invokes a file browser dialog which can be used to navigate (by the `...` button) towards a file of displacement data, generated by the solver.

After loading the [displacement data file](#), there is no immediate change until you vary the selection in the *Coloring* drop-down box (button #4 of the [ParaView window](#)). A number of new colour-mapping options should be available:-



Screen view of colour-map options

- Displacement (X,Y,Z)
- NormalStrain (X,Y,Z)
- PrincipalStrain 1 (P1)
- PrincipalStrain 2 (P2)
- PrincipalStrain 3 (P3)
- ShearStrain (XY, YZ, ZX)
- Strain Energy Density (SED)
-

Notes

1.

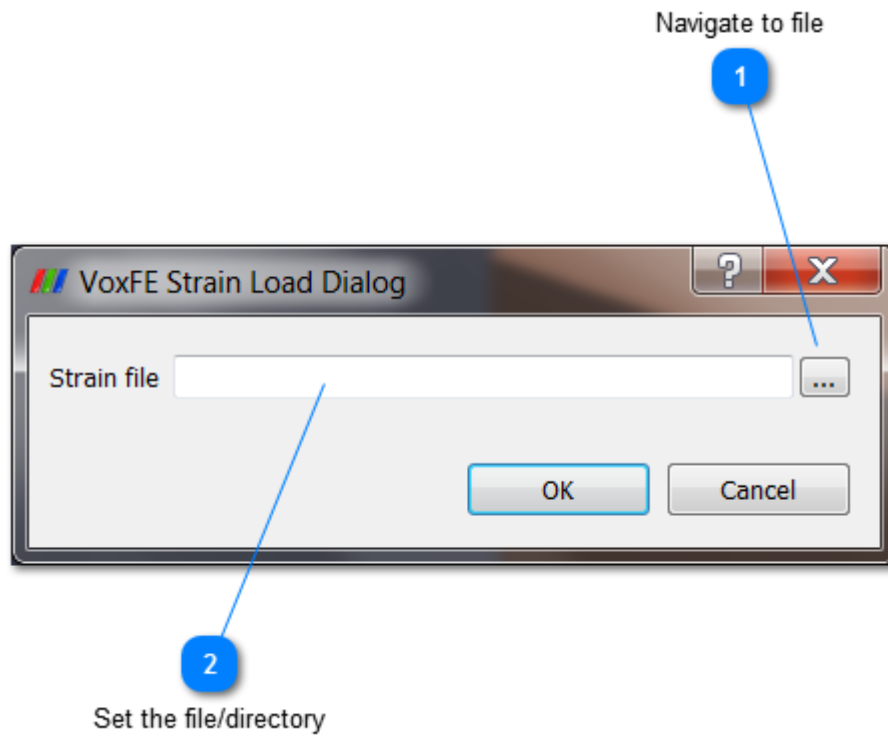
Other than displacement, these are treated as scalar quantities only; for displacement, it is possible to display colour maps of a component, the magnitude and to apply the *Warp* filter (button #18 of [the ParaView window](#)) to show the deformed mesh (see the [example session](#)).

2. To view strain within the model, deselect "Generate Surface Mesh" (button #2 of [the model panel options](#)), and apply *Clip* or *Slice* filters. (**If only the *Displacement* option is visible, then it is likely that "Generate Surface Mesh" remains checked, or perhaps, that the action has not been not *Applied*.**)

Plotting 2D data

The figure [2D and 3D plotting](#) shows the results of plotting "P1" in the 3D window (clipped) versus 2D plots of "P1" and "P3". Creating 2D line and probe plots is described [here](#) and in the [example session](#).

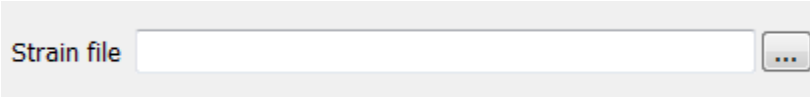
VoxFE2 Strain Load Dialog window



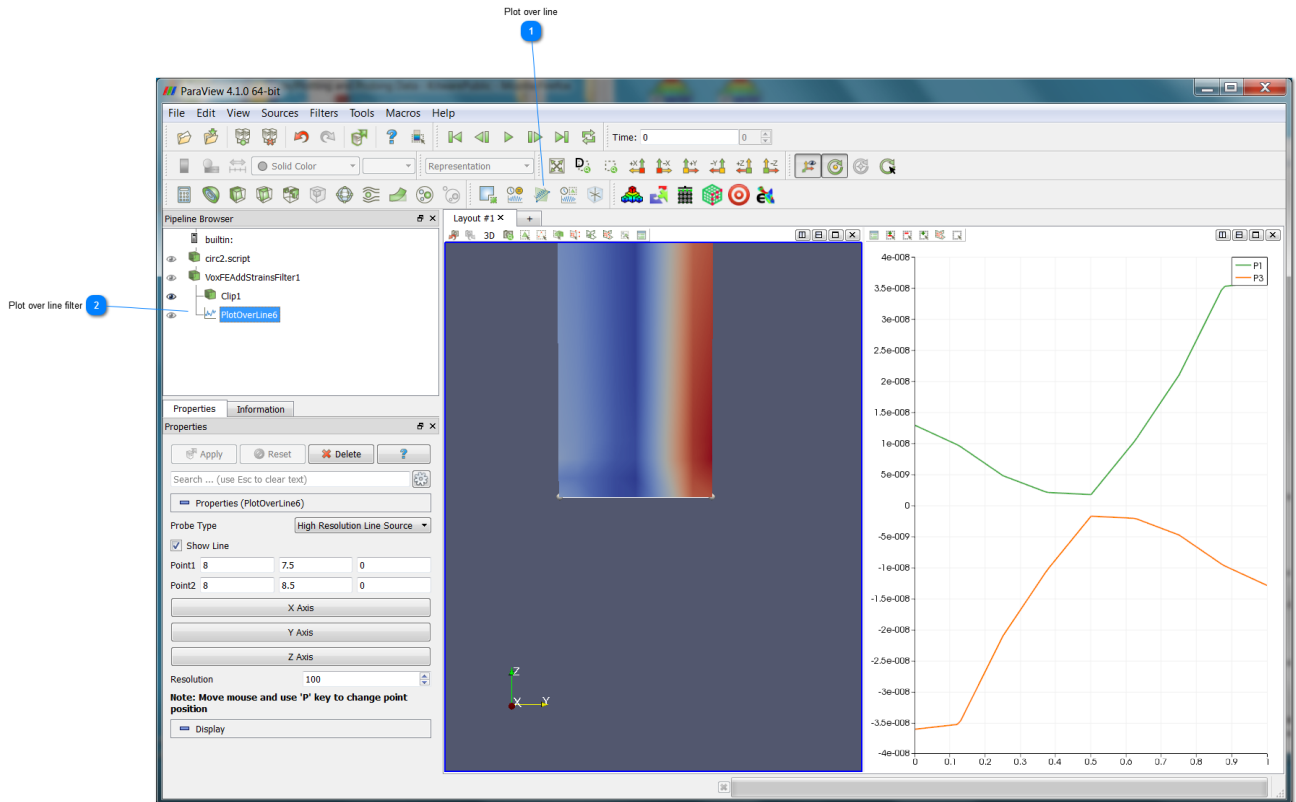
1 Navigate to file



2 Set the file/directory



2D and 3D plotting

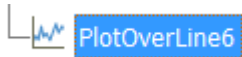


1 Plot over line



Creating a 2D plot window directs the plot command

2 Plot over line filter



2D plot control

An example session

Video tutor

This example is covered in a [YouTube video](#). As VoxFE is still being developed, the video is already a little out of date (especially with regard to some of the more advanced controls), but in conjunction with the text below should provide a useful introduction. Please note there are a number of other videos already available on YouTube giving a more basic introduction to ParaView (see links in [Navigating around the model](#)).

The 10:1 cylinder problem

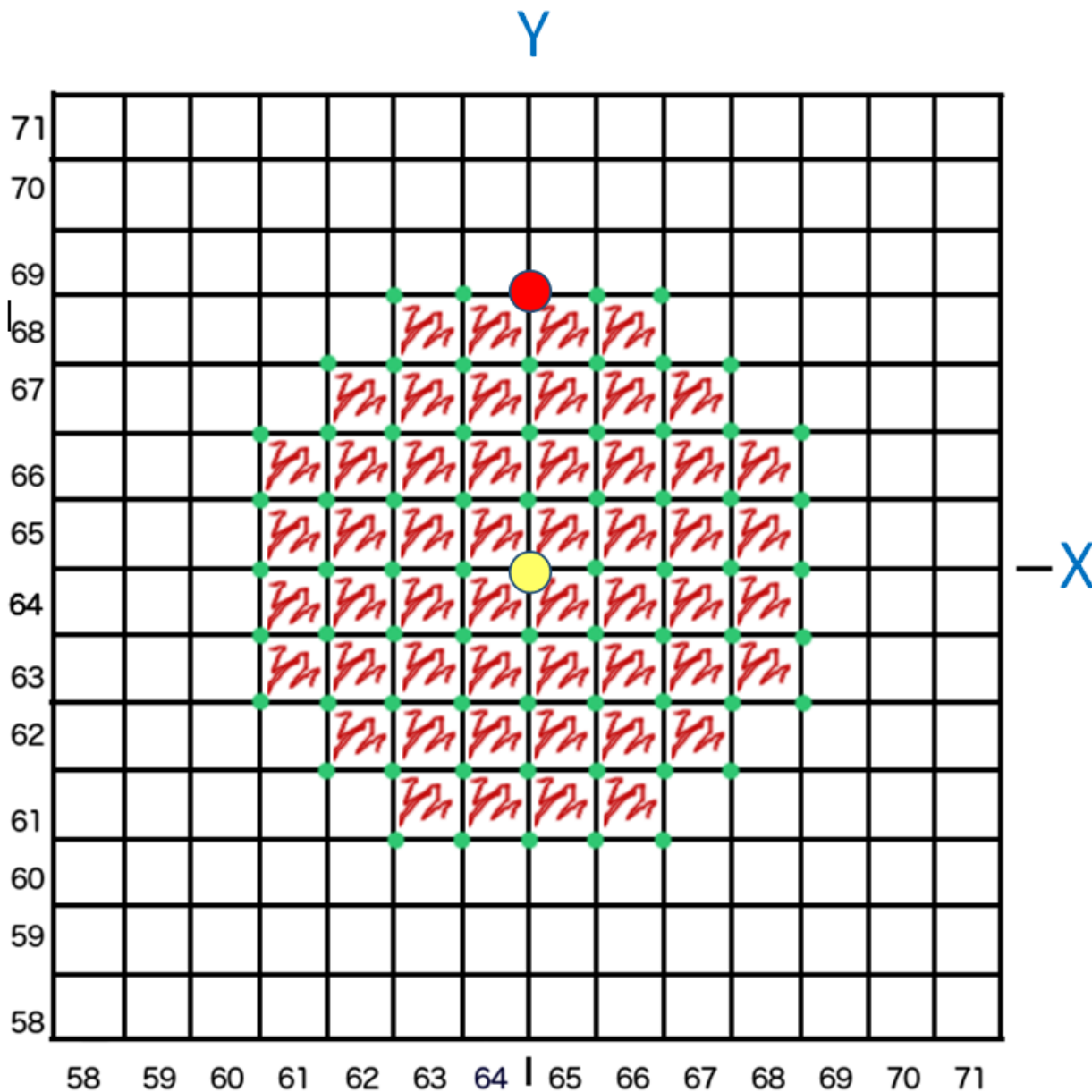
For this problem, we assume we have a small cylindrical plug, 10mm long and 1mm wide, made of a single material. The plug is constrained at one end and loaded at the other across the face of the bar, which can be checked against a theoretical model. The trickiest part is in setting the boundary constraints. We wish to constrain all the nodes on the slice ($z=0$) in the following way:-

Assuming this cross section is in the XY plane, then:

- Constrain all the green nodes in the Z direction only
- Constrain the yellow node in X, Y and Z
- Constrain the red node in X and Z

The central yellow node stops the cross section from translating in X, Y and Z.

The red node stops the rod rotating about the z-axis, but allows the cross-section to contract (due to the Poisson's effect).



At the other end of the cylinder ($z=80$) we wish to apply an axial force of 10N equally over all nodes.

1. Starting the Session : loading the model

- Turn *Auto-apply* off
- Start ParaView and load the plugin (as described in the [Overview](#)).
- Use the [Open VoxFE file](#) dialog box (button #1 of the [VoxFE toolbar](#)) to navigate to the file "circ2.script" in the directory test/std-10.1-model/d000008.
- Click *Apply*.

2. Selecting the nodal boundary conditions

Use *Set View direction to +Z* (in panel #11 of [the ParaView window](#)) to view the base of the cylinder, and zoom as needed.

For the green points:

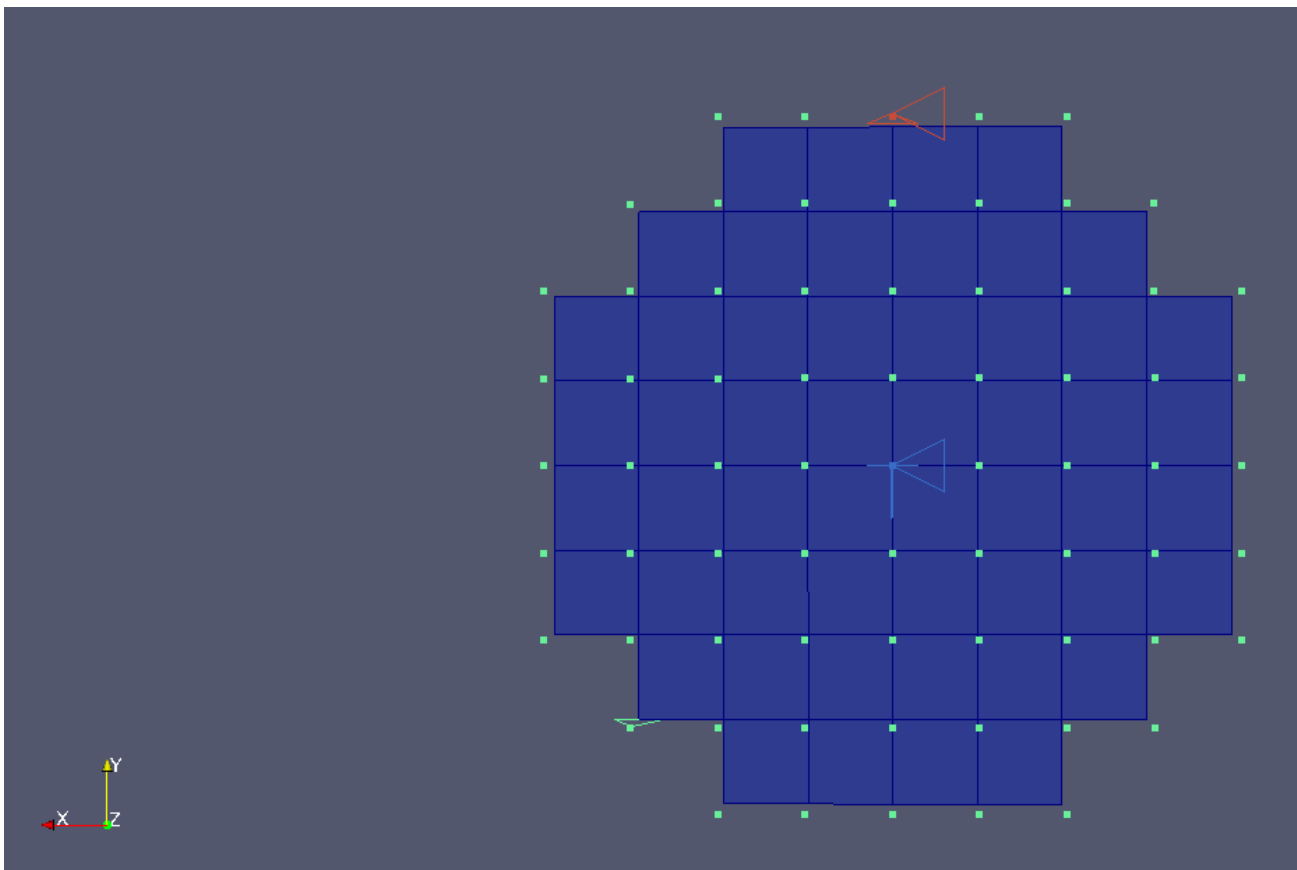
- Set the Representation type to surface with edges, to make it easier to locate the points
- (1) Click the *Select points On* selection tool
- Repeat ⁽¹⁾ with Ctrl+click selections to get all the green points
- (2) Press the *Extract Selection* and *Apply* (assuming auto-apply is off)
- Click the [Add BC](#) VoxFE button.
- Select the "Nodal constraint" and check the "Z" button. Click apply.
- [Optional] Click the [Highlight BC](#) button.
- If the glyph sliders are not shown, click the "Advanced Property settings" button and move the glyph to some convenient place on the side.

For the yellow point:

- Repeat ⁽¹⁾ and ⁽²⁾ to extract the central point
- Click the [Add BC](#) button.
- Select the "Nodal constraint" and check the "X", "Y" and "Z" buttons. Click *Apply*.
- Click the [Highlight BC](#) button.

For the red point:

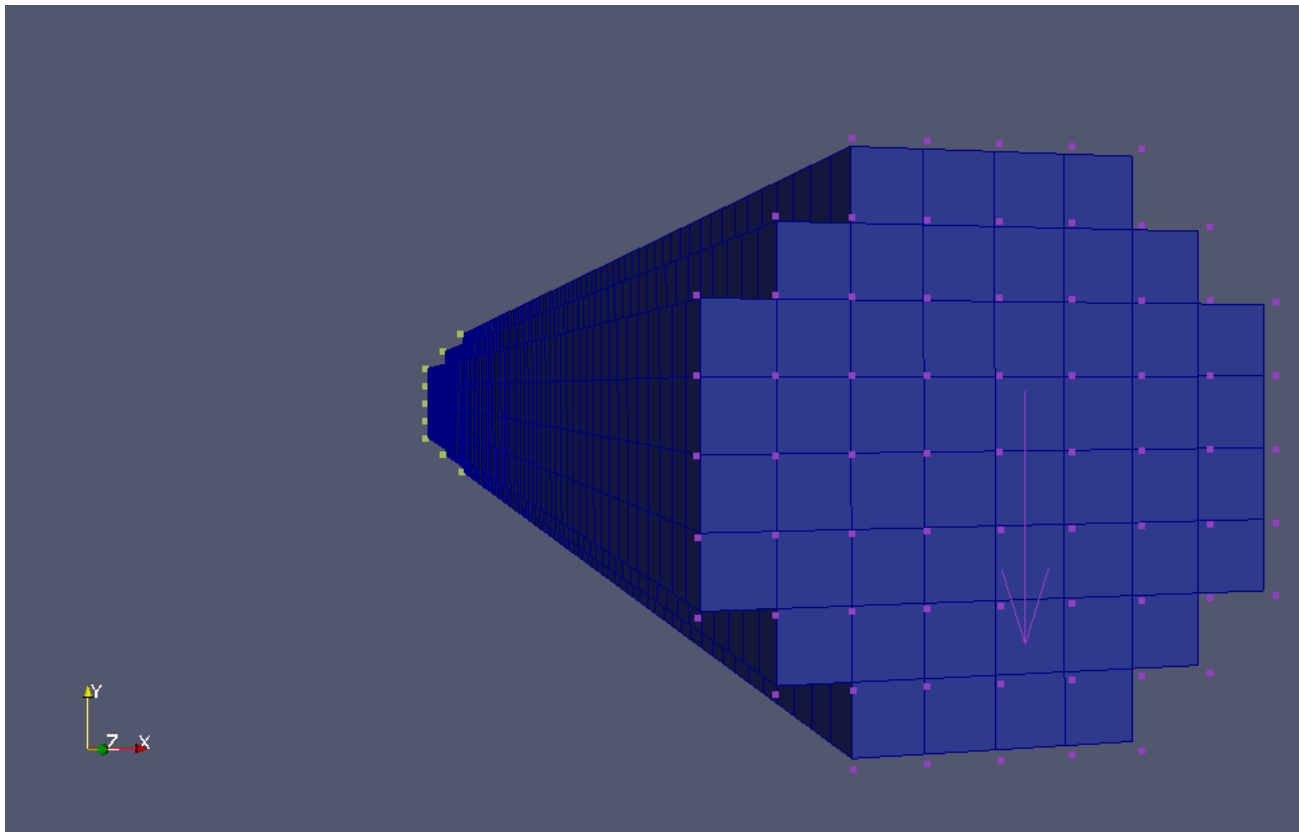
- Repeat ⁽¹⁾ and ⁽²⁾ to extract the top-most centre point
- Click the [Add BC](#) button.
- Select the "Nodal constraint" and check the "X" and "Z" buttons. Click *Apply*.
- Click the [Highlight BC](#) button.



After setting the nodal boundary conditions

3. Setting the force loading conditions

- 1) For the loading force:
 - a) Use *Set View direction to -Z* to view the top of the cylinder, and zoom as needed.
 - b) Repeat (1) and (2) to extract all the end points.
 - c) Click the **Add BC** button.
- 2) In the *Properties panel*, select "Force (parallel)" and enter:-
 - a) Force vector: 0,-1,0
 - b) Magnitude: 10
 - c) Distribution: Area
 - d) Click the **Highlight BC** button and adjust the glyph/arrow as needed.



After setting the force loading condition

4. Outputting the solver script

- Select the original model in the *Pipeline browser*.
- Click the **Output script** button.
- Select/check that "Input model" on the left is matched with "circ2.script" on the right.
- Select "Boundary Conditions" on the left and use Ctrl+click to select the four *VoxFEBoundaryCondition* elements on the right.
- Click "OK" and *Apply*.

Solve:

Check circ2.script has the correct material parameters and can open the new constraints file.

Solve eg. ./PARA_BMU circ2.script

5. Loading/viewing strain data

- For greater visibility, switch off the boundary conditions from view in the *Pipeline browser*.
- Use the **Import strain data** button open the dialog to load the strain file "displacement.txt".
- Click "OK", but before selecting "Apply" select the *Parabmu Solver output* option.
- Select the model in the *Pipeline browser* to make it active.
- Untick "Generate Surface Mesh" in the *Properties panel* (and click *Apply*).
- Select "P1" in the *Coloring* drop-down box.
- Select the *Clip* filter.
- Click "X Normal" and *Apply*.
- [Optional: Select *Auto-apply* and adjust the clip face along the chosen normal -- this is where auto-applying a filter really shines.]

To see the impact of the displacement:

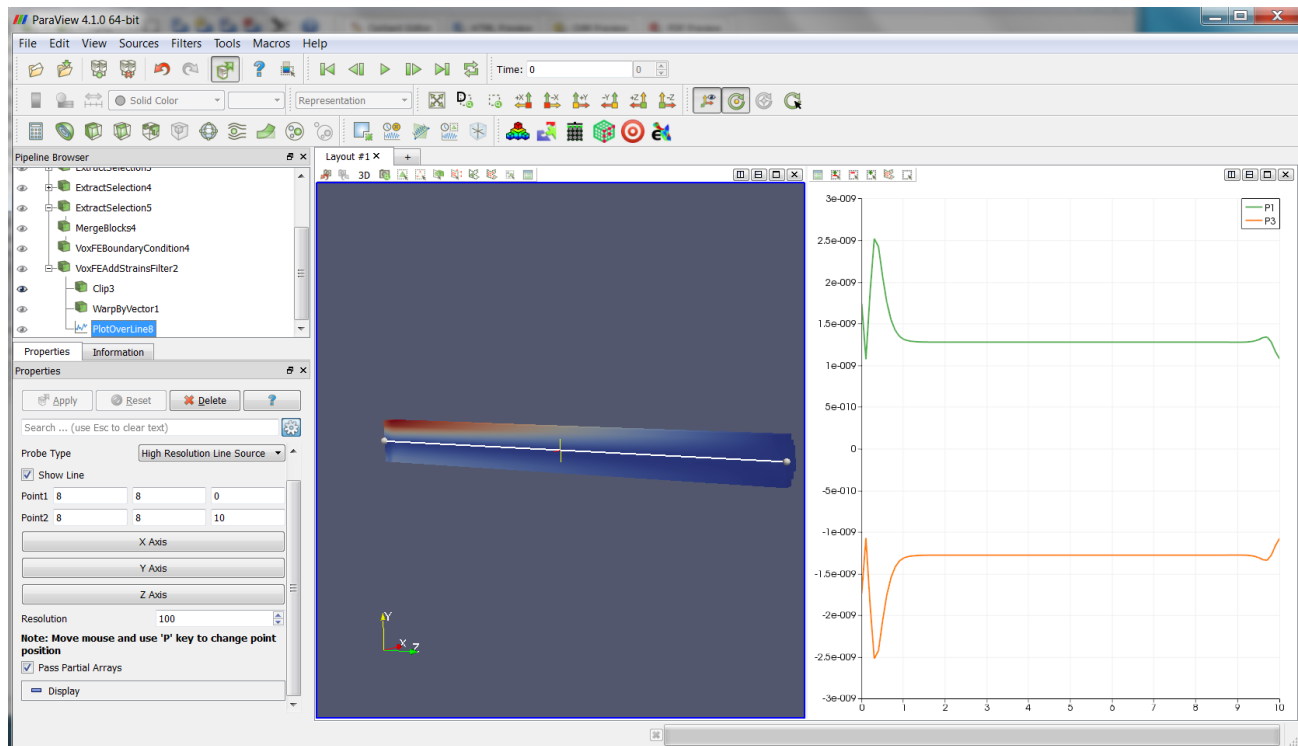
- Turn off the *Clip* filter (using the eye symbol) or delete it altogether (by right-clicking and selecting "Delete").
- Select the *VoxFEAddStrainsFilter* and turn it on using the eye symbol.
- Select *Warp by Vector* and click *Apply*.
- You should notice that "Displacement" was selected automatically by the *Warp* filter (since that is the only vector data we have imported).
- You will not likely see any warping, since the effect is so small -- try setting the scale factor (eg. to 100000) to exaggerate.

To see 2D plots eg. of strain data:

- Switch off the *Warp* and *Clip* filters and select the *VoxFEAddStrainsFilter* (turn it on if not visible).
- Use the buttons above the 3D scene view (see item #16 of [the ParaView window](#)) and click *Split Horizontal*. (Alternatively, open a new tabbed window by clicking the "+" tab button.)
- Select *Line Chart View* in the buttons of the new window.
- Whilst the 2D-plot view window is active (marked by a blue border), select *Plot Over Line* (in panel #12 of [the ParaView window](#)).
- The resulting plot looks very odd but contains all the strain data and our chosen "Line" still needs to be defined.
- Select the *Plot Over Line* filter in the *Pipeline browser*, and scroll down the *Properties panel* to the *Display* section.
- Deselect all the imported data types except "P1" and "P3".
- In the *Properties* section (above *Display*), select "Z Axis".

[Note: if you don't have any data in your 2D plot window, it may be because you didn't deselect "Generate Surface Mesh" to rebuild the internal geometry -- see above.]

If you switch the *Clip* filter back on and select the *Plot Over Line* filter, the plot window should now look something like:-



With a 2D plot window

Building the VoxFE2 plugin

The ParaView source code bundle is almost self-contained except for its dependence upon the GUI toolkit Qt, for which version 4.8.x is currently needed. A binary version can be installed, but it is advisable to build Qt, so that you can ensure that the same compiler/linker is being used and you can test Qt demos (etc) if something does not go to plan. The CMake build system is also needed to generate project/make files, but it is perfectly fine to install binary version of this tool.

Requirements:-

1. ParaView v.4.1 (or above) from [here](#).
2. Qt 4.8.x from [here](#).
3. Optional: ITK v. 4.4.x from [here](#). Later versions may work.
4. VoxFE plugin source from [here](#).

Configuration is described in a separate html document available with the source code.

-

To Do:

1. Advanced options for loading images
2. Materials and constraint file formats
3. Usefulness of Extract block
4. Glyph controls
5. Using solver?