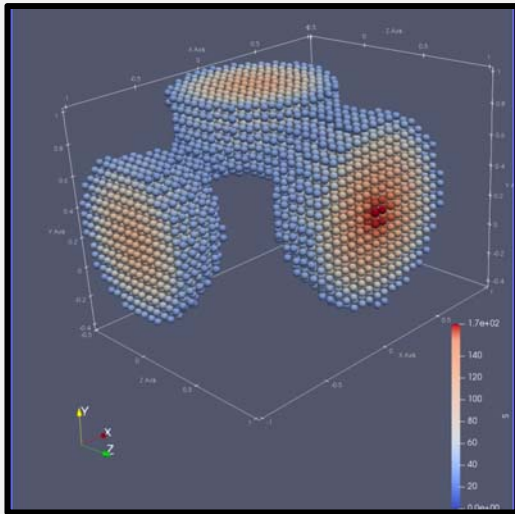


ParaView

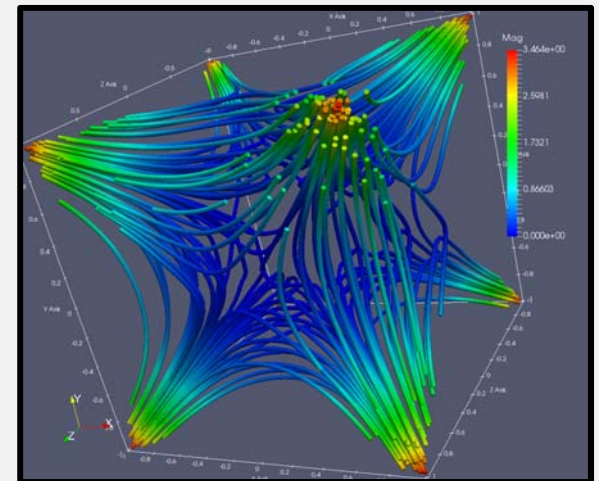
<http://cs.oregonstate.edu/~mjb/paraview>



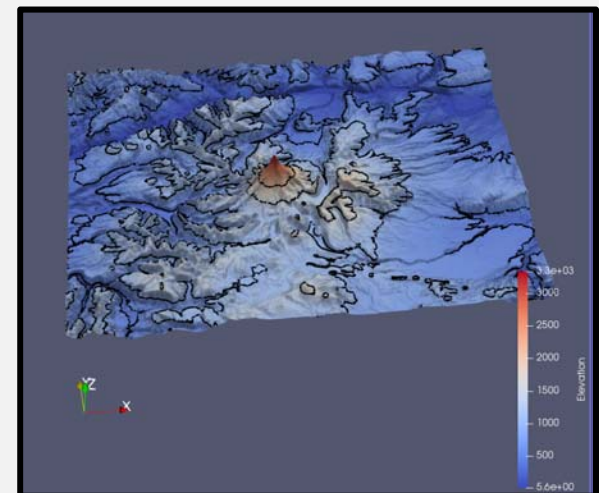
Oregon State University

Mike Bailey

mjb@cs.oregonstate.edu



This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/)



ParaView Things I Need to Do:

Notes:

Table of filters

Describe Selecting

Describe Warp By {Scalar, Vector}

Describe light sources

Any way to turn numeric data into a slider (e.g., isovalue in Contour)?

Any way to read and process a .shp file?

Any way to export triangles (.obj, .stl)?

Is there a properties menu to set the camera eye, look, and up?

Plot data on a globe

Can we write our own graphics filters (e.g., Extruded Time Volumes and faster volume rendering)?

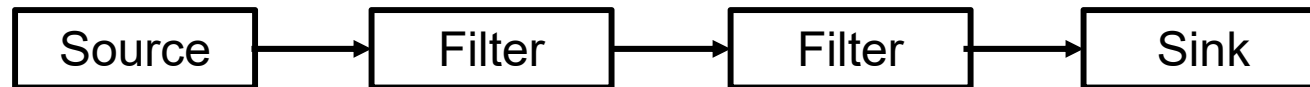


What is ParaView?

ParaView is a free interactive visualization package produced by KitWare.

It is built upon Vtk, the Visualization Toolkit.

It uses a dataflow paradigm:



In which data arrives via sources (typically files), is filtered by various numeric algorithms, and is sent to various sinks (typically the computer graphics display).

Besides the interactive interface, ParaView also has a Python scripting interface, so that you can create these dataflow networks auto-magically.

http://www.paraview.org

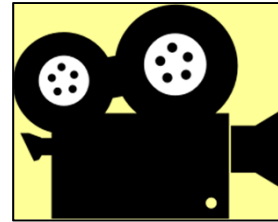
Click here to download, or grab the .zip file from Mike's ParaView web site

The screenshot shows the ParaView website homepage. The browser's address bar contains the URL <https://www.paraview.org>, which is circled in red. A red arrow points from this URL to the 'Download' link in the top navigation menu, also circled in red. The main content area features a large blue header with the text 'Visual Effects' and a 3D rendered skull. Below this, a white box contains the text 'What have you #RenderedInParaView?' and a button labeled 'Show us on Twitter'. The page is titled 'Welcome to ParaView' and contains several sections: a paragraph about ParaView as an open-source application, a paragraph about its use for large datasets, and six action items: 'Download Latest Release', 'Support and Services', 'Contact Us', 'Request a New Feature', 'The ParaView Guide', and 'ParaView Forum'. Each item includes a green icon and a brief description.

In these notes, what do these icons mean?



scalar.csv



scalar.ogv

They tell you that if you go to our notes web site:

<http://cs.oregonstate.edu/~mjb/paraview>

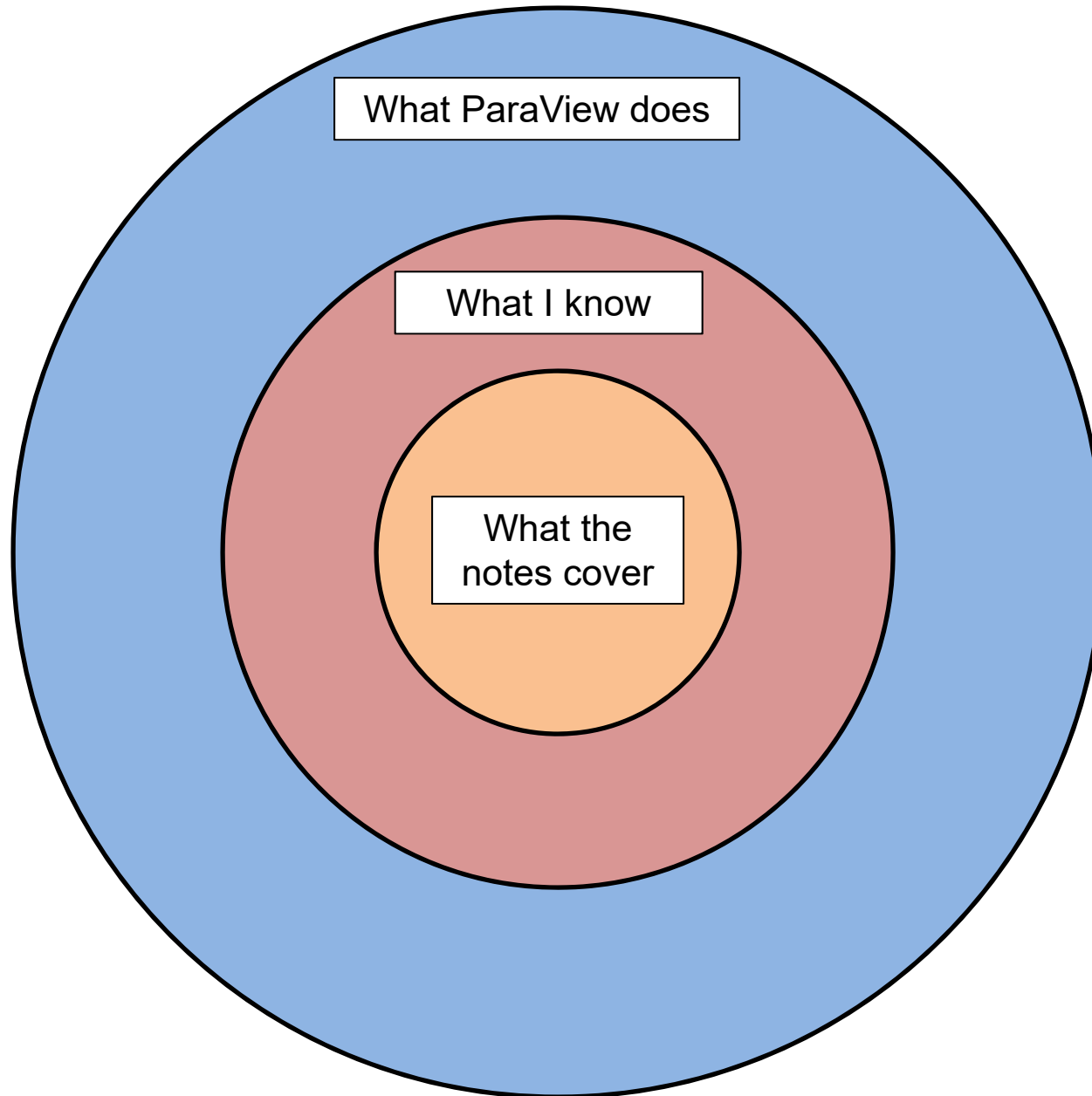
you will find pre-created ParaView input data (*.csv) and pre-created animation movie files (*.ogv).

You can read a .csv file right into ParaView so that you can experiment with these examples without having to first create them yourself.

You can play an .ogv movie file right from your browser so that you can see how these examples look without having to run ParaView at all.



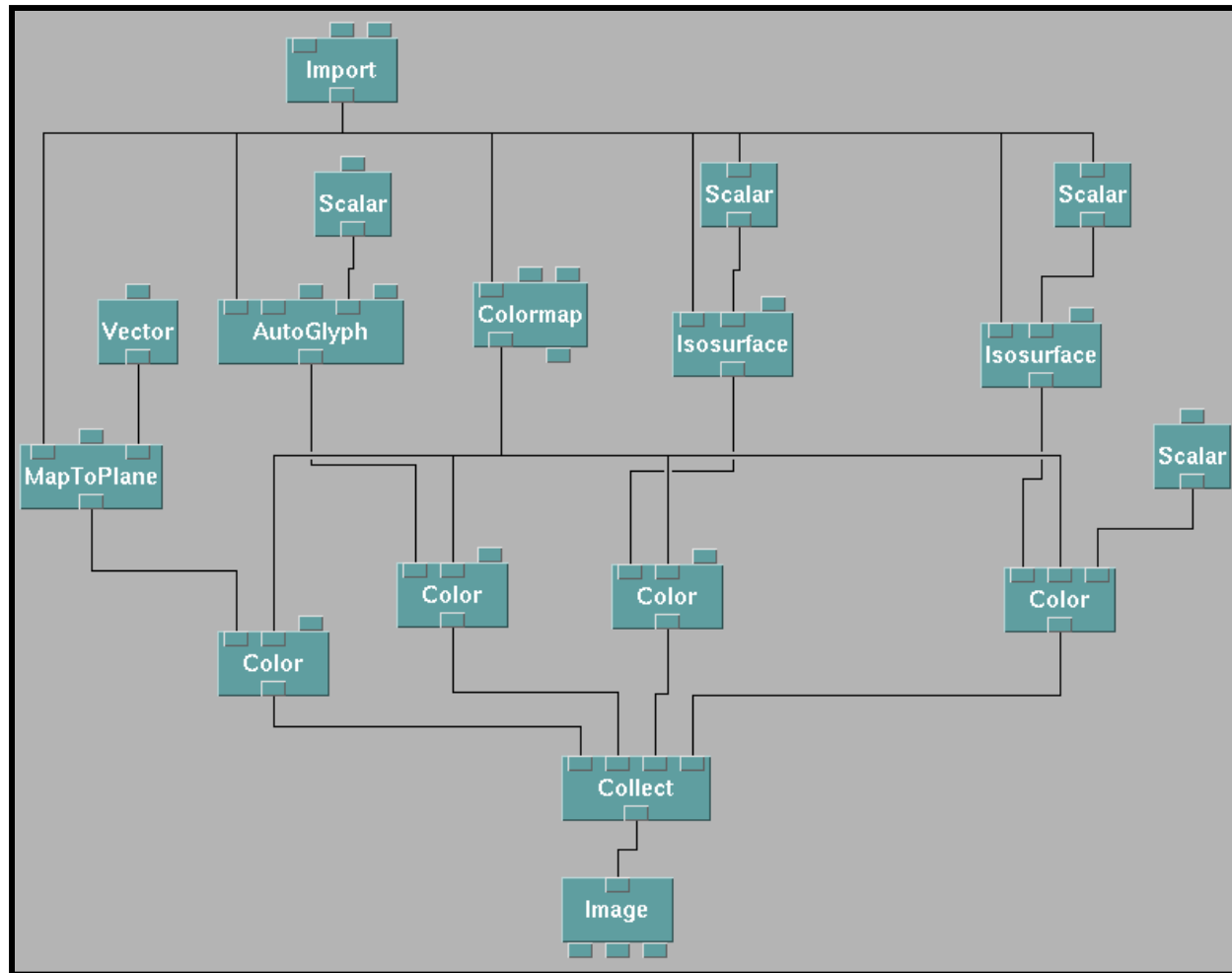
A warning about me and the Notes



Screen Layout, Color Editor, and 3D Display



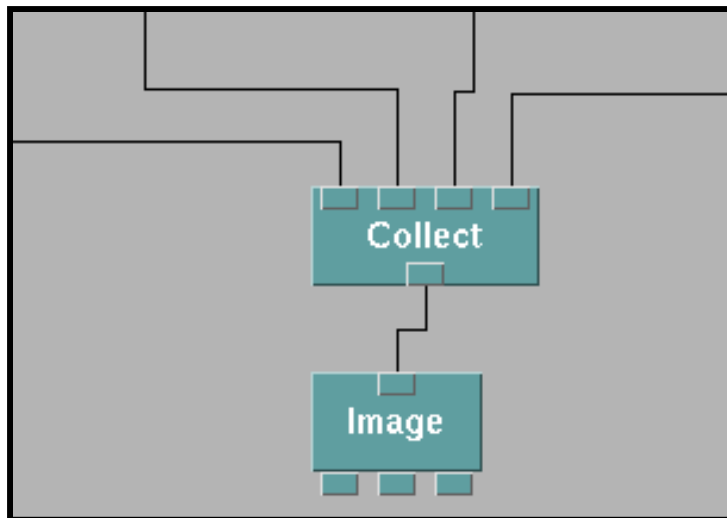
In the Beginning, there was OpenDX ...



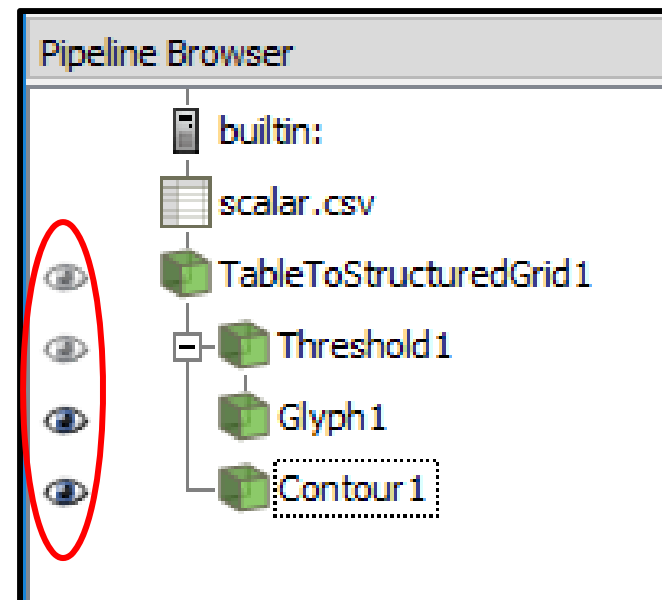
“DX” stands for “Data Explorer”. Like the name implies, it let you *explore!* But, once it became “open”, all reliable support went away. Also, it required a lot of screen area just to hold the block diagram.

Fan-In to the Full Scene

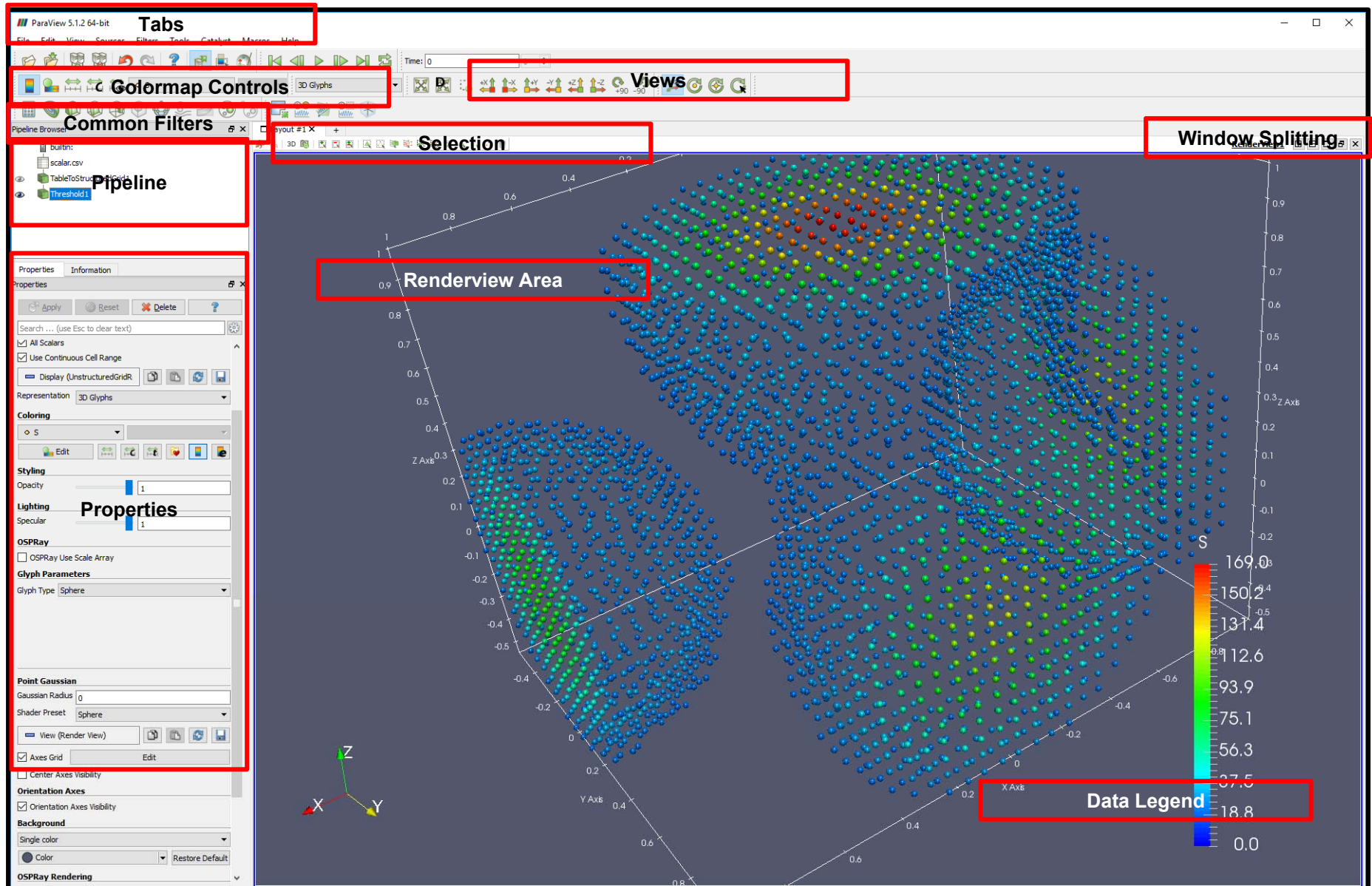
OpenDX:



ParaView:



Screen Layout



Window Icons



Toggle between 2D and 3D interaction

Adjust camera

Add, subtract, and toggle selections

Selecting cells, points, and blocks

Clear selection

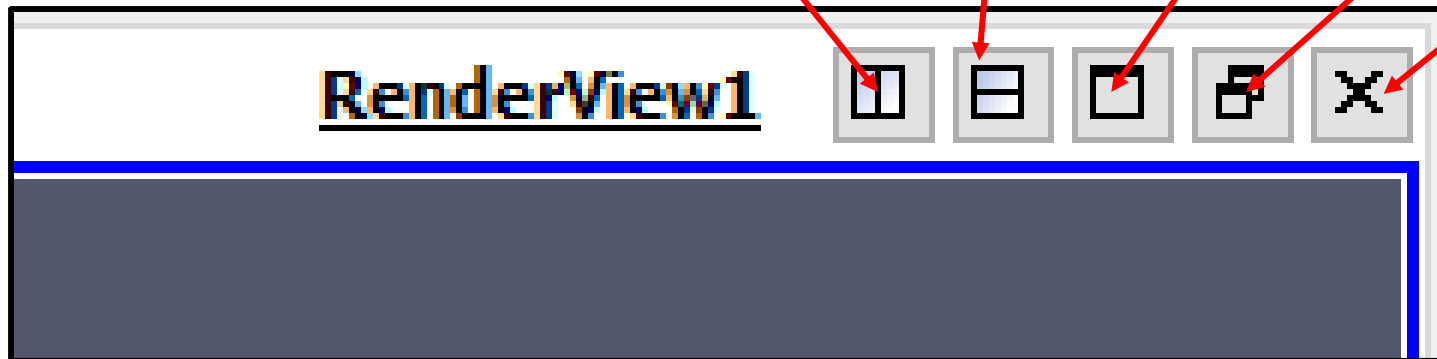
Split a window left-right

Split a window up-down

Maximize a window

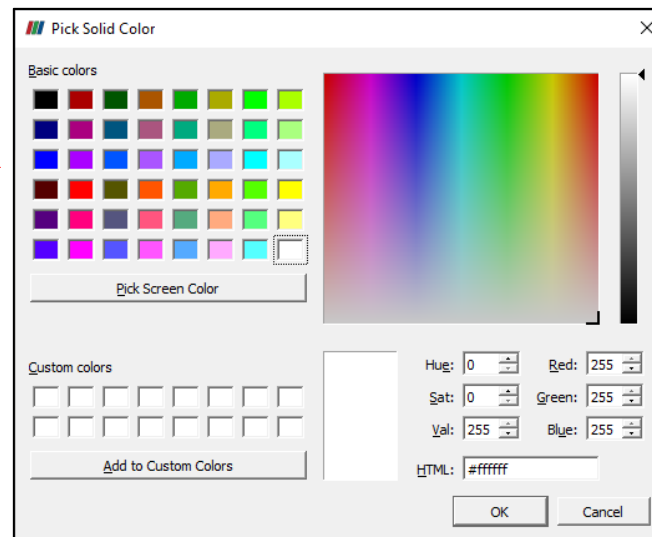
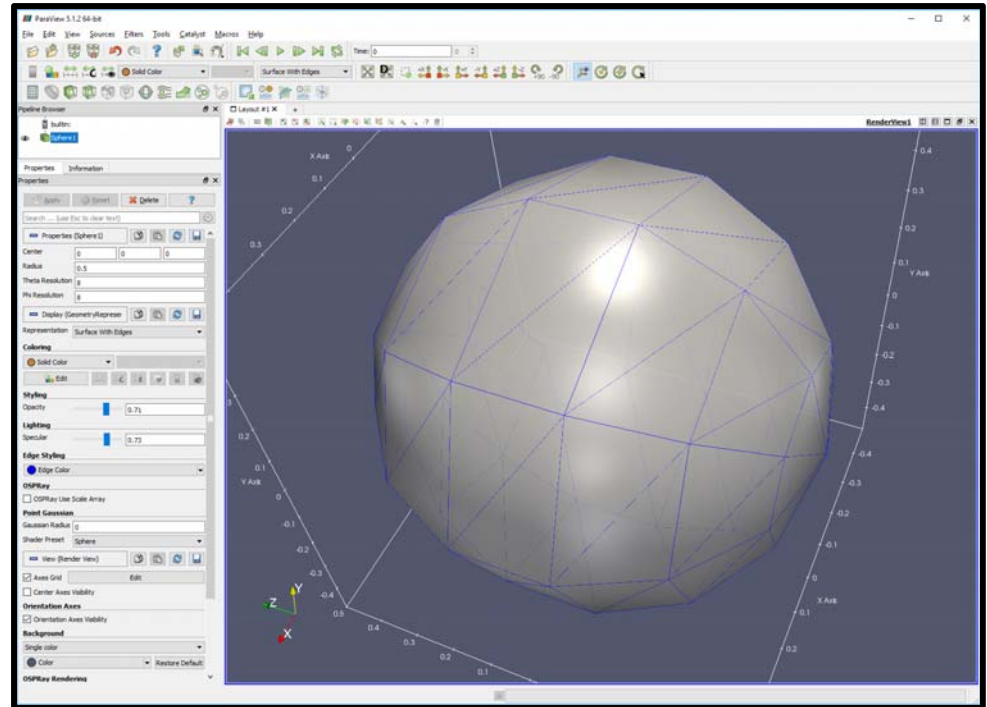
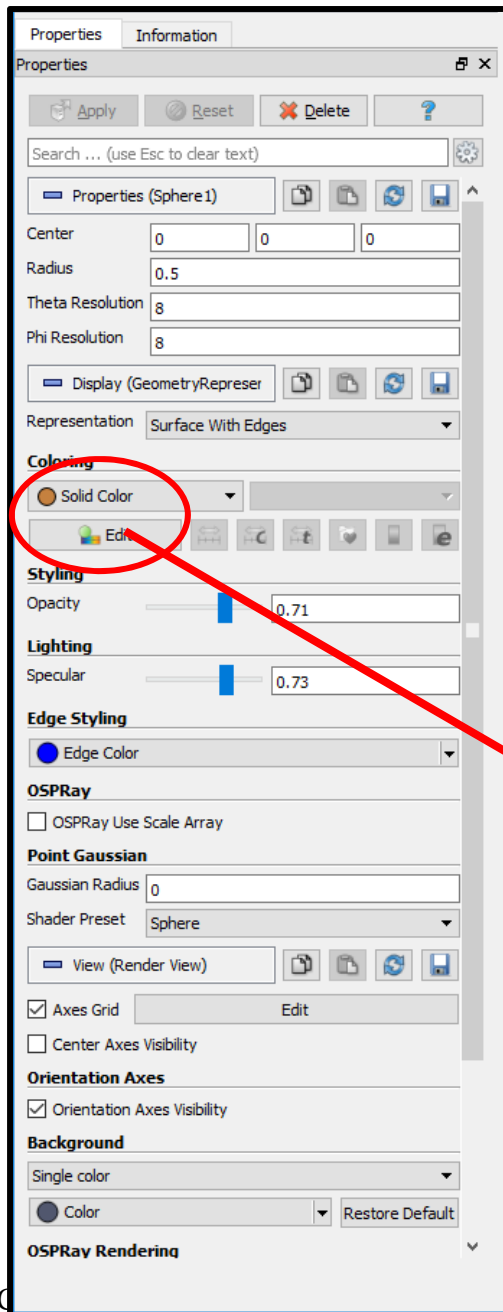
Restore after maximizing

Eliminate a window

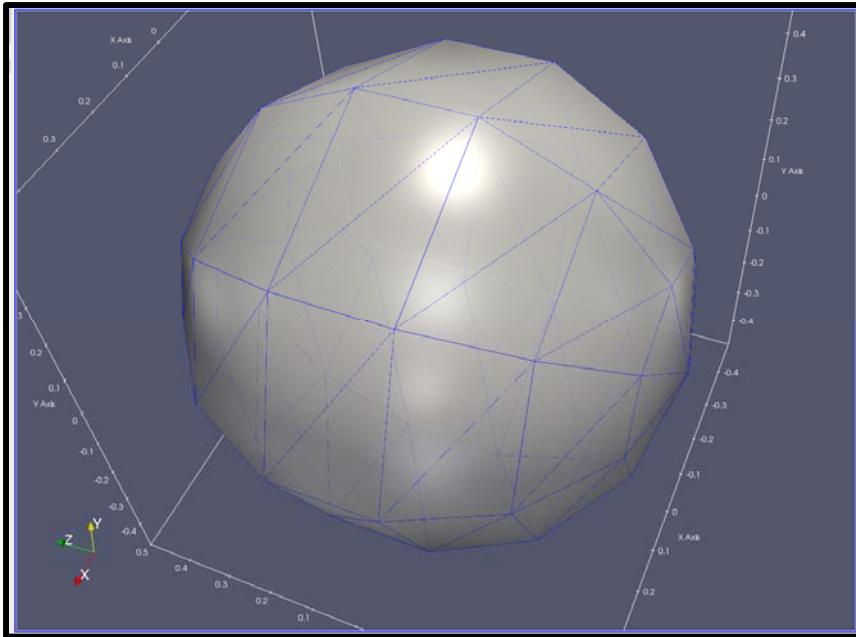


3D Scene Manipulation

Select Sources → Sphere



3D Scene Manipulation



By default, these are the 3D Scene Manipulators (plus the mouse wheel, which is also a Zoom):

(You can change these in the **Edit** → **Settings** → **Camera** menu)



3D Interaction Options

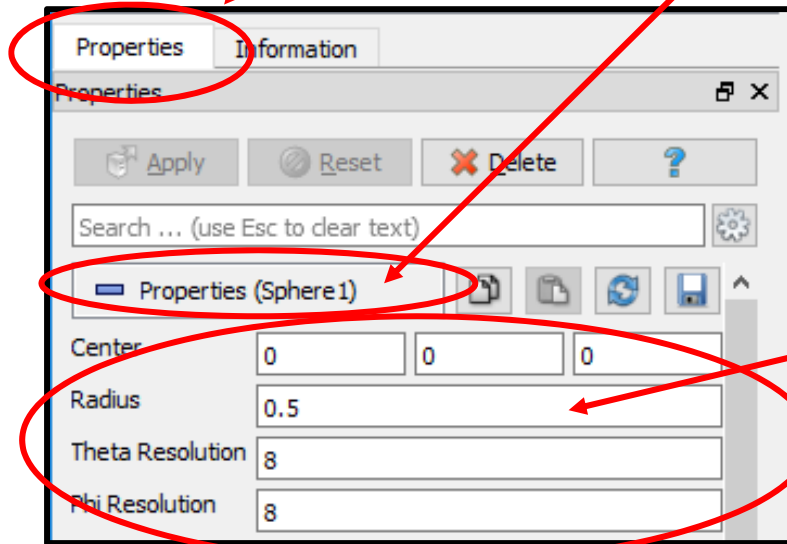
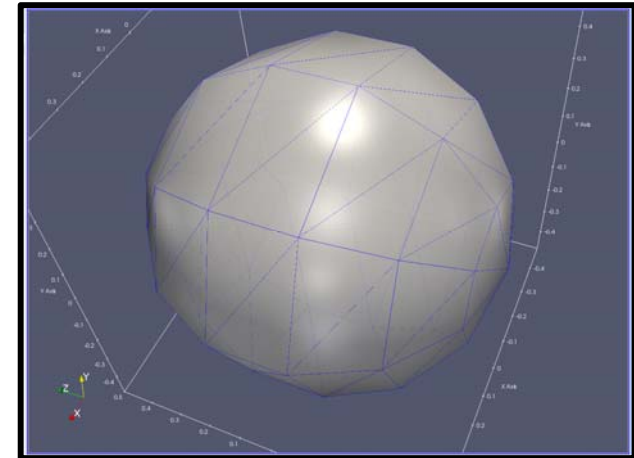
Camera3DManipulators: Select how interactions are mapped to camera movements when in 3D interaction mode.

	Left Button	Middle Button	Right Button
	Rotate	Pan	Zoom
Shift +	Roll	Rotate	Pan
Ctrl +	Zoom	Rotate	ZoomToMouse

You Can Change Sphere Properties

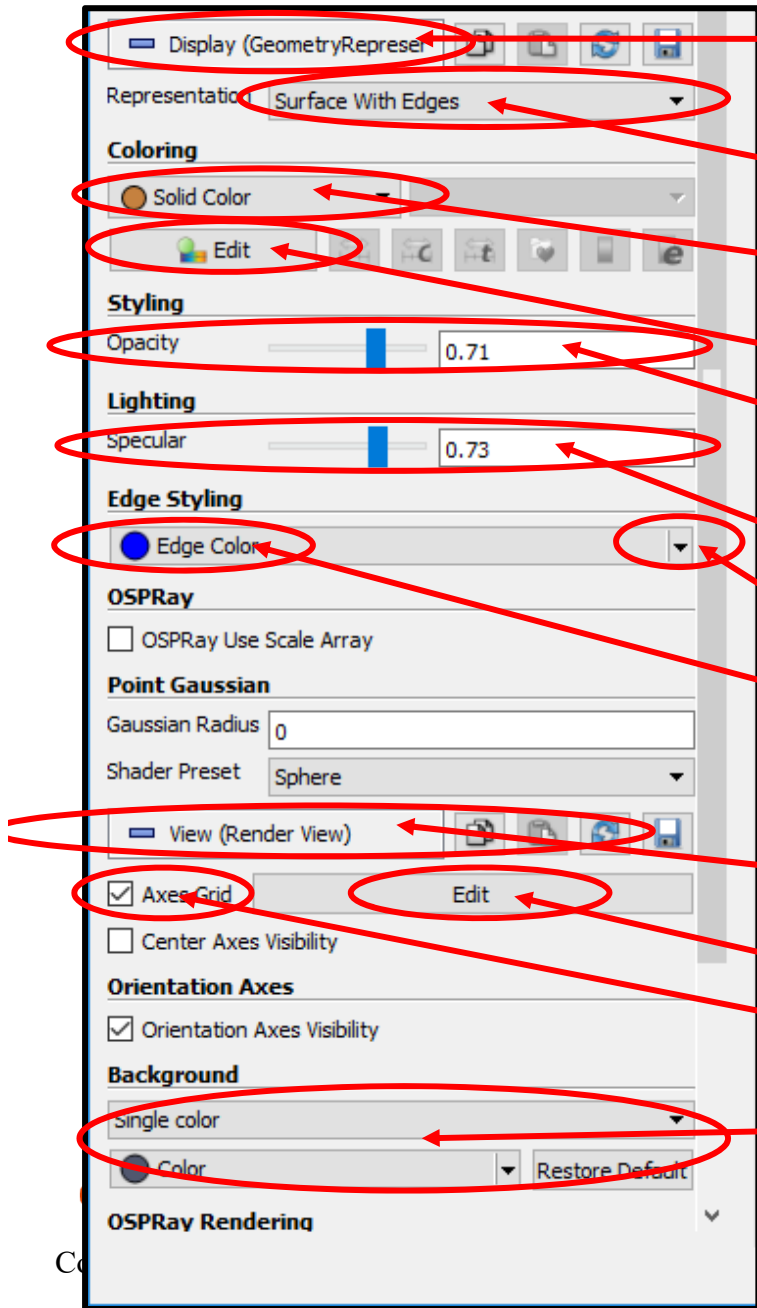
Properties Tab

Show/Hide the Geometric Properties



The Geometric Properties of the Sphere

You Can Change the Sphere's Display Properties



Show/Hide the Display Properties

How to Represent the Sphere

How to Color the Sphere

Edit the Sphere Color

Set the Sphere Opacity

Set the Sphere Specular Lighting

Bring up other Features to Color-Edit

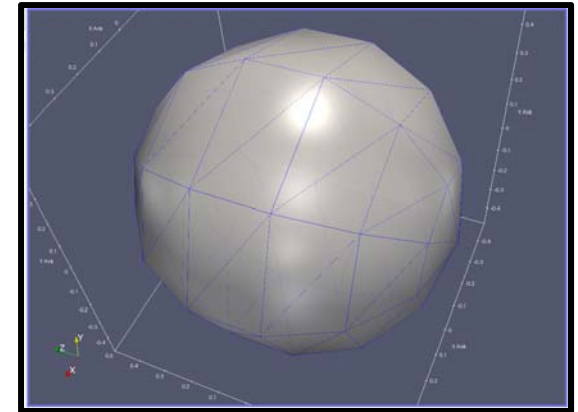
Edit the Edge Color

Show/Hide the Render View Properties

Edit the Features of the **Axes Grid**

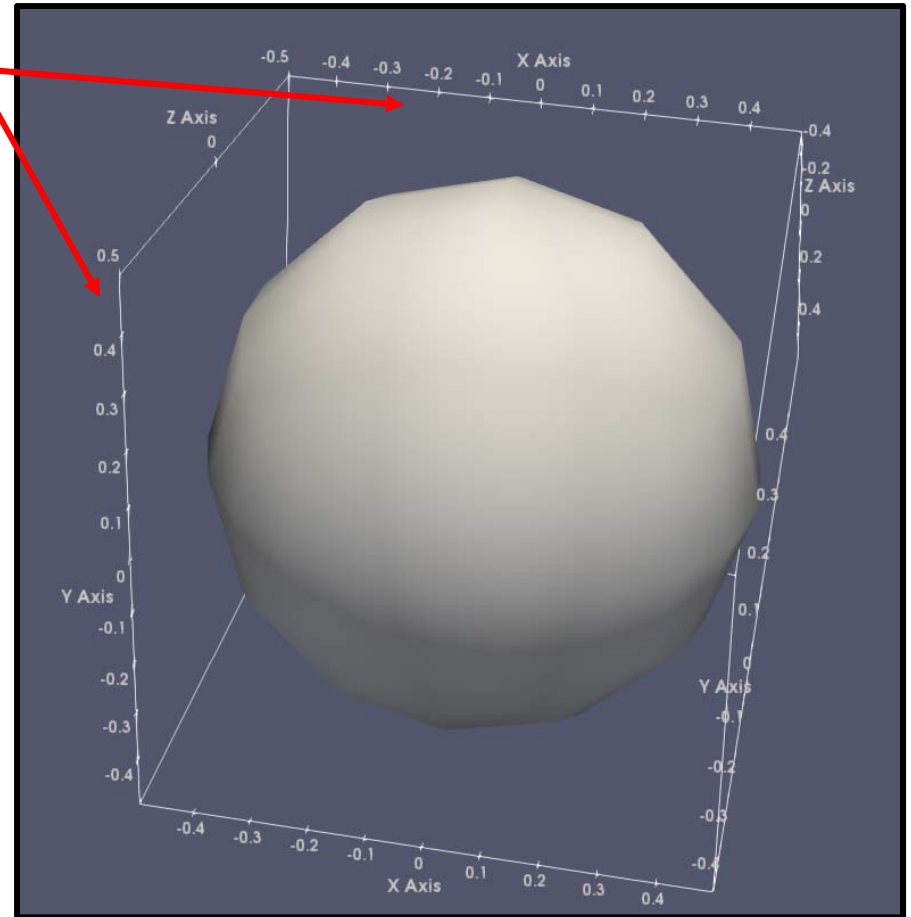
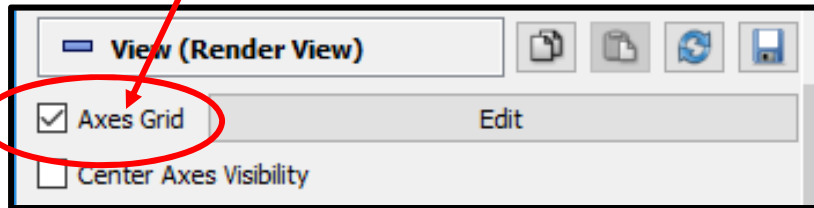
Turn on/off the **Axes Grid**

Edit the Background Style and Color

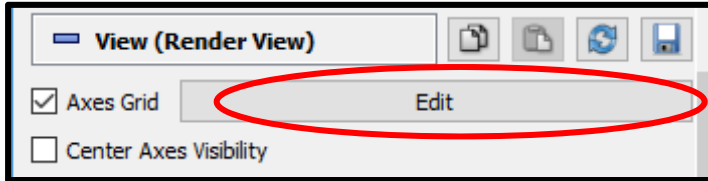


The Axes Grid

ParaView has a nice **Axes Grid** feature. Scroll way down in the Properties area to the **Render View** menu to turn it on.

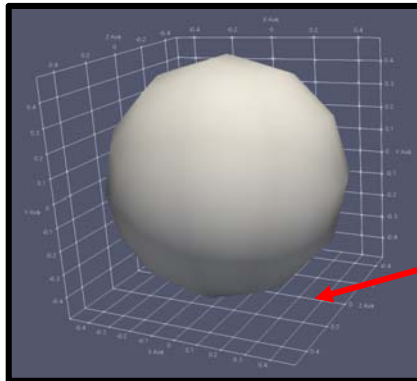


Editing the Axes Grid

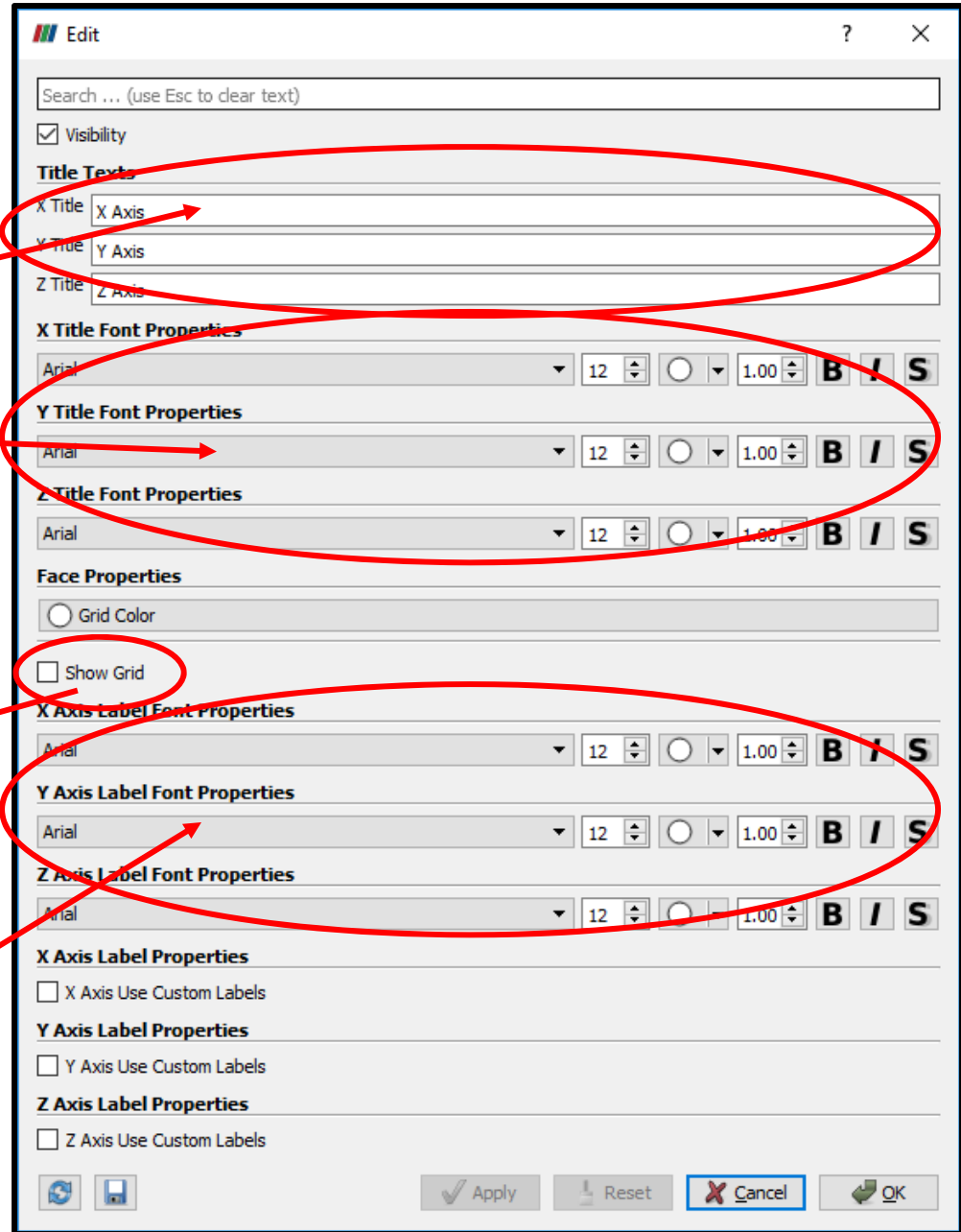


Titles for the axes

Title font styles

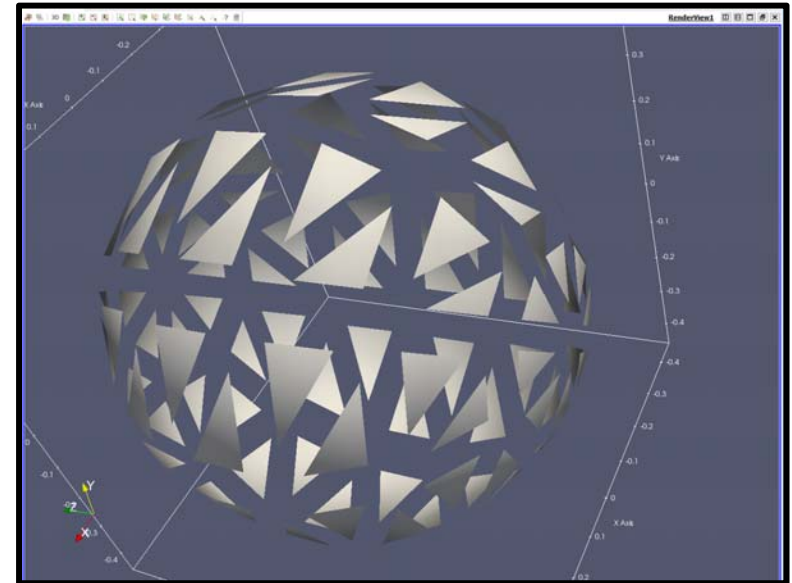
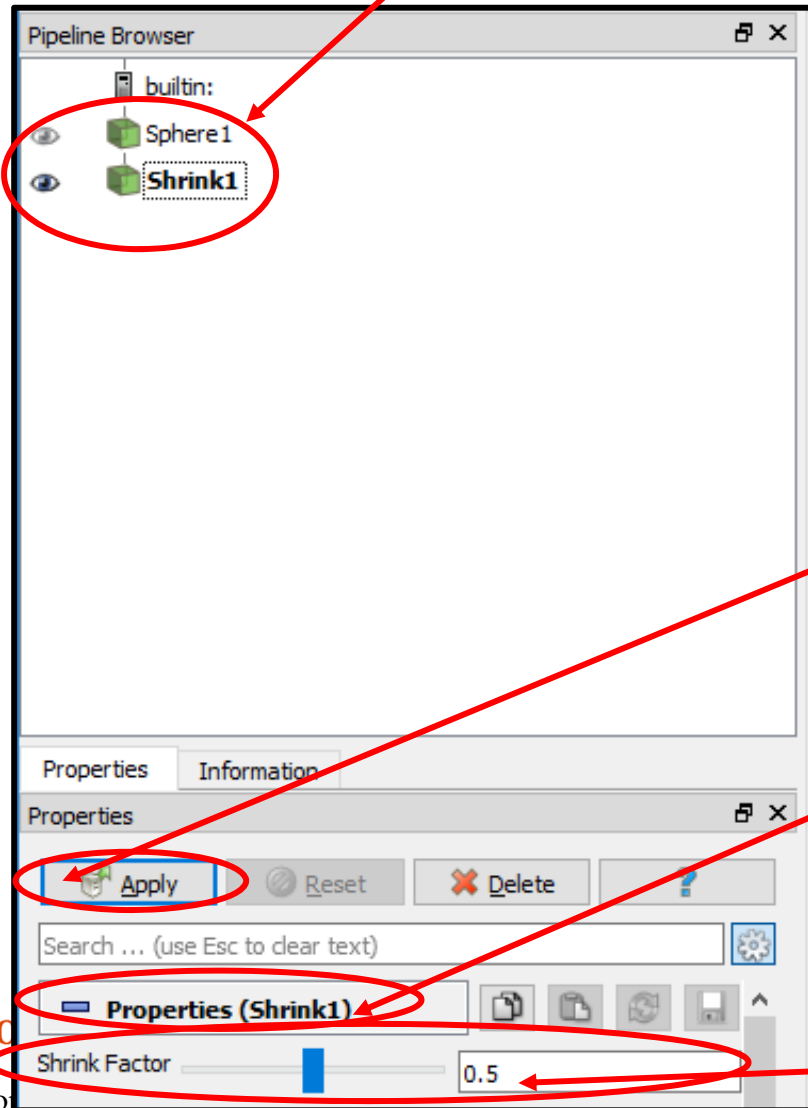


Number label font styles



Filters → Alphabetical → Shrink

Be sure the Shrink eyeballs are clicked on and the Sphere eyeballs are clicked off



Step #2: Hit **Apply**

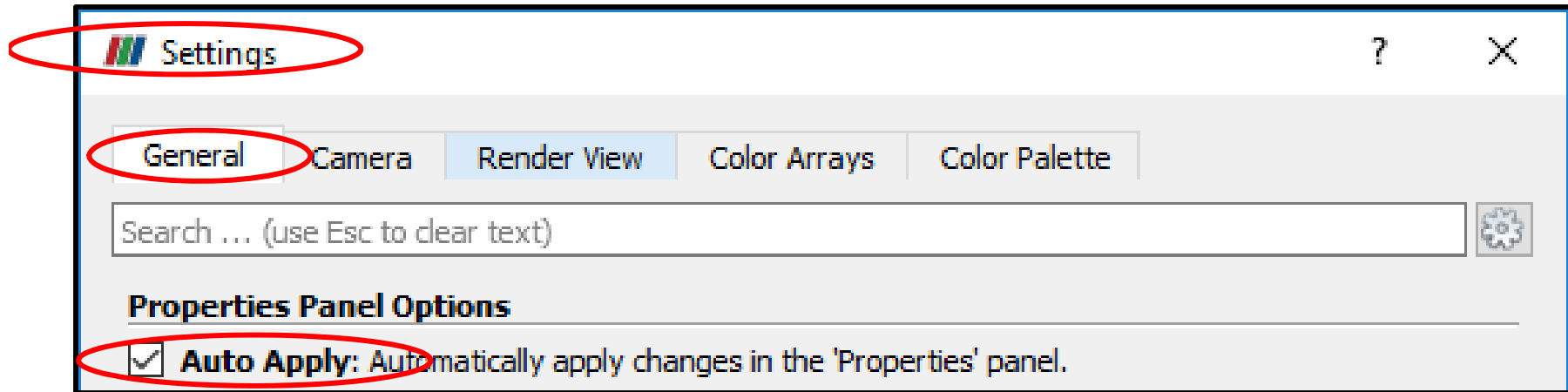
Show/Hide the Shrink Properties

Step #1: Set the Shrink Factor (1. = no shrinking, 0. = all shrinking)

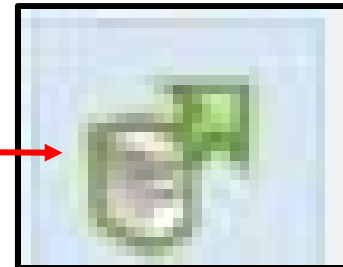
Co

Are You Getting Tired of Hitting *Apply* All the Time?

In **Edit** → **Settings** → **General**, turn on **Auto Apply**



Or, click this icon in the top row of icons



Be careful about doing this with large datasets that are slow to display.

Don't do this until after you have completed the entire **TableToStructuredGrid** operation.

Visualizing Scalar Data, I



scalar.csv

What File Formats Can ParaView Read?

AVS UCD	BYU	CML Molecule	CSV
DEM	DICOM (Single File)	DICOM (directory)	ENZO AMR Particles
EnSight Master Server	EnSight	Enzo	ExodusIIReader
FLASH AMR Particles	FacetReader	Flash	Fluent Case
Gaussian Cube	Image	JPEG Series	LSDynaReader
Legacy VTK	MFIXReader	MRC Series	Meta File Series
NetCDF CAM	NetCDF MPAS	NetCDF POP	NetCDF
Nrrd	OpenFOAMReader	PDB	PLOT3D Meta-File
PLOT3D	PLY	PNG Series	PTS
PVD	Parallel NetCDF POP	Particles	Partitioned Legacy VTK
Phasta	RTXMLPolyDataReader	Restarted Sim Exodus	Restarted Sim Spy Plot
SLAC Data	SLAC Particle Data	STL	TIFF
TIFF Series	Tecplot	Unstructured NetCDF POP	VPIC
VRML	Wavefront OBJ	WindBlade	XDMF
XML Hierarchical Box Data	XML Image Data	XML MultiBlock Data	XML Partitioned Image Data
XML Partitioned Polydata	XML Partitioned Rectilinear Grid	XML Partitioned Structured Grid	XML Partitioned Unstructured Grid
XML PolyData	XML Rectilinear Grid	XML Structured Grid	XML UniformGrid AMR
XML Unstructured Grid	XYZ	proSTAR (STARCD)	spcth history

Creating Scalar Data in a CSV File

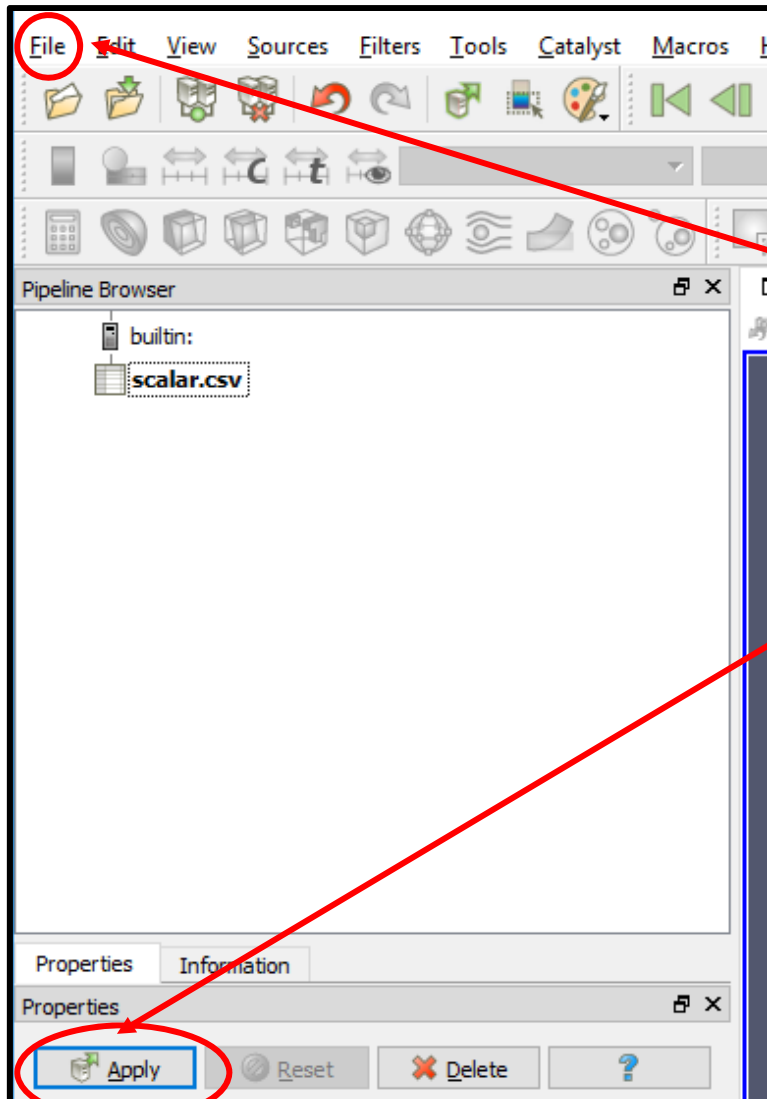
```
X32, Y32, Z32, S
-1.00, -1.00, -1.00, 0.00
-0.94, -1.00, -1.00, 0.00
-0.87, -1.00, -1.00, 0.00
-0.81, -1.00, -1.00, 0.00
-0.74, -1.00, -1.00, 0.00
-0.68, -1.00, -1.00, 0.00
-0.61, -1.00, -1.00, 0.00
-0.55, -1.00, -1.00, 0.00
-0.48, -1.00, -1.00, 0.00
-0.42, -1.00, -1.00, 0.00
-0.35, -1.00, -1.00, 0.00
-0.29, -1.00, -1.00, 0.00
-0.23, -1.00, -1.00, 0.00
-0.16, -1.00, -1.00, 0.00
-0.10, -1.00, -1.00, 0.00
-0.03, -1.00, -1.00, 0.00
```

Do a **File** → **Open** and navigate to your CSV file.
Hit the **Apply** button to actually do the read.



scalar.csv

Reading and Converting the CSV File



1. Select **File** → **Open** and navigate to scalar.csv

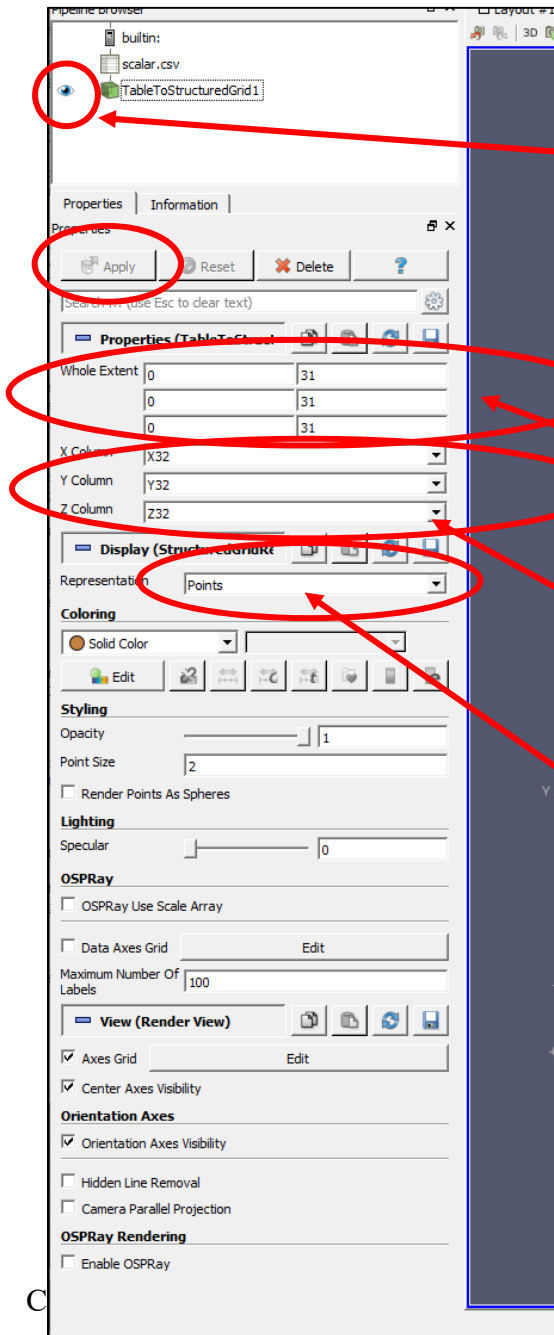
2. Then, click **Apply**

The screenshot shows a spreadsheet window titled 'SpreadSheetView'. The table has columns for 'Point ID', 'uctured Coordina', and 'Points'. The data is as follows:

Point ID	uctured Coordina	Points
0	0 0 0	-1 -1 -1 0
1	1 0 0	-0.94 -1 -1 0
2	2 0 0	-0.87 -1 -1 0
3	3 0 0	-0.81 -1 -1 0
4	4 0 0	-0.74 -1 -1 0
5	5 0 0	-0.68 -1 -1 0
6	6 0 0	-0.61 -1 -1 0
7	7 0 0	-0.55 -1 -1 0
8	8 0 0	-0.48 -1 -1 0
9	9 0 0	-0.42 -1 -1 0
10	10 0 0	-0.35 -1 -1 0
11	11 0 0	-0.29 -1 -1 0
12	12 0 0	-0.23 -1 -1 0
13	13 0 0	-0.16 -1 -1 0
14	14 0 0	-0.1 -1 -1 0
15	15 0 0	-0.03 -1 -1 0
16	16 0 0	0.03 -1 -1 0
17	17 0 0	0.1 -1 -1 0

3. This will bring up a table window to confirm that the data has been read properly. You can close it if you want.

Reading and Converting the CSV File



Turn on the “eyeballs” so that you can view this data

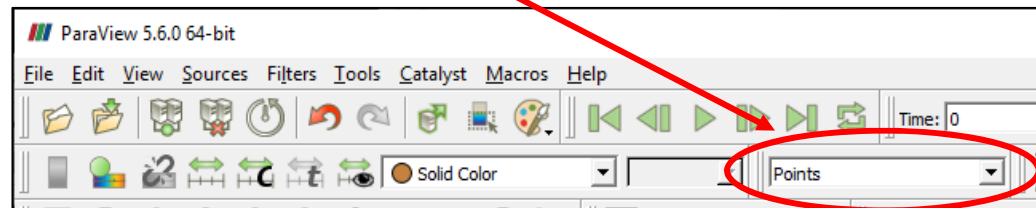
Now, go to
Filters → **Alphabetical** → **TableToStructuredGrid**

Fill in the **Whole Extent** boxes showing the first and last index in each dimension (the last index is one less than the number of points in that dimension).

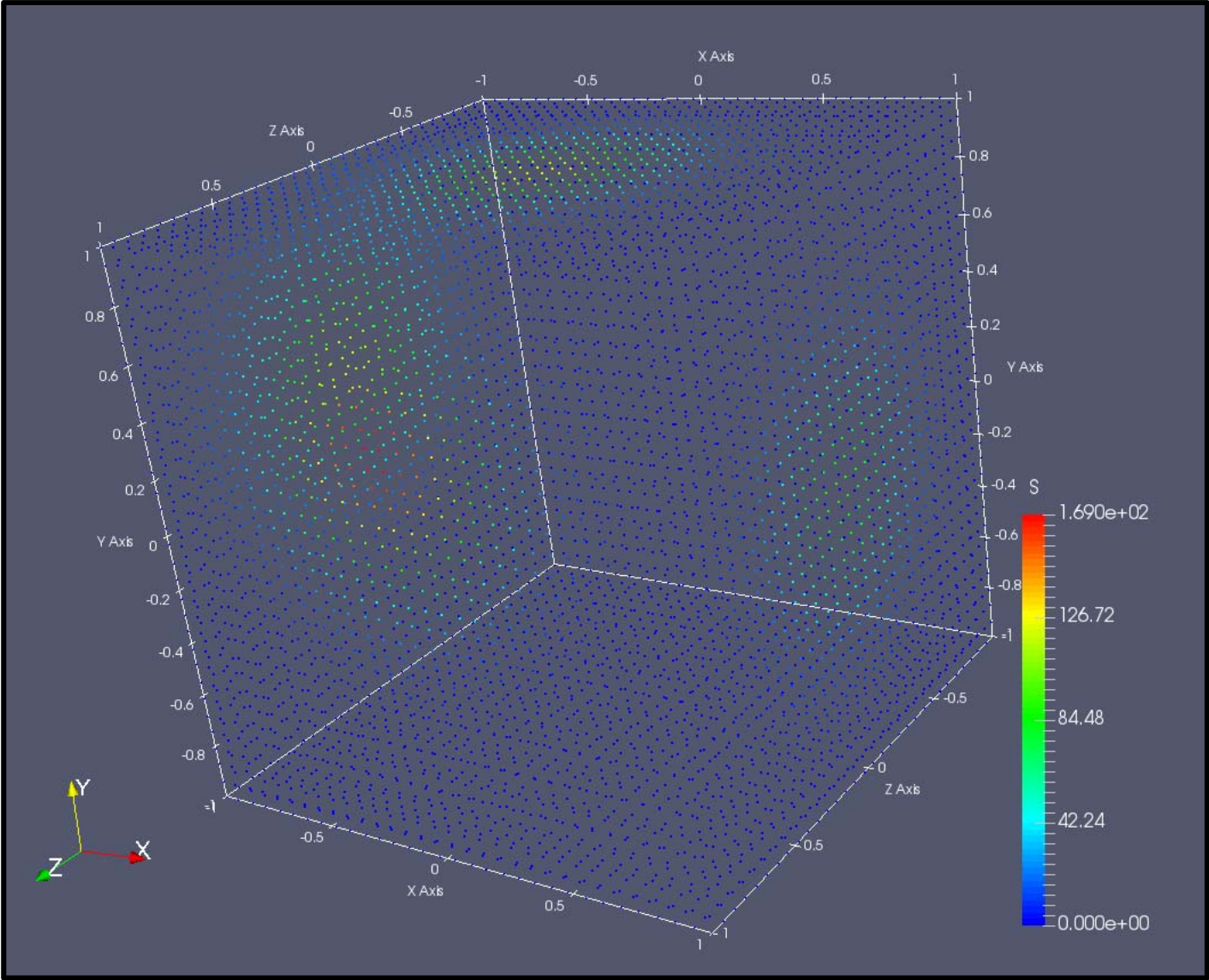
Fill in the **{X,Y,Z} Column** information so ParaView knows how to make your 3D display.

Hit the **Apply** button to actually do the conversion.

Set the **Representation** to **Points**



As Points



A Side Trip: Choosing Colors

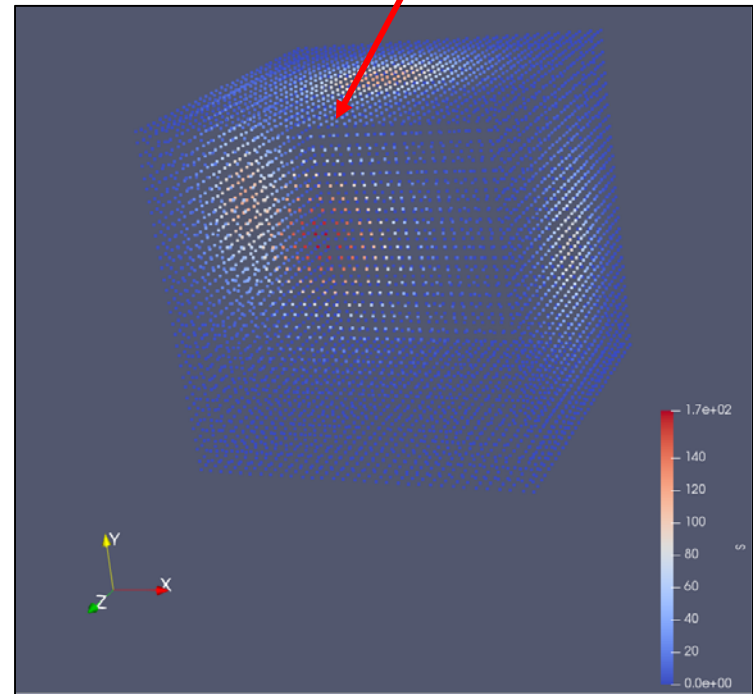
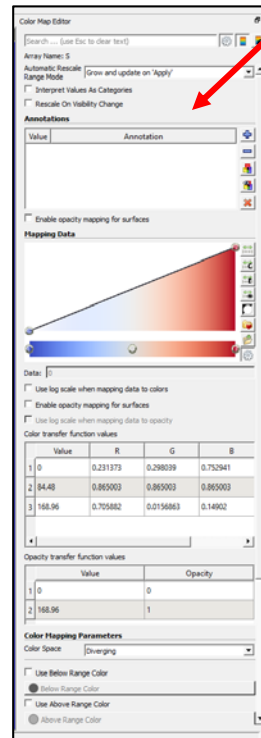
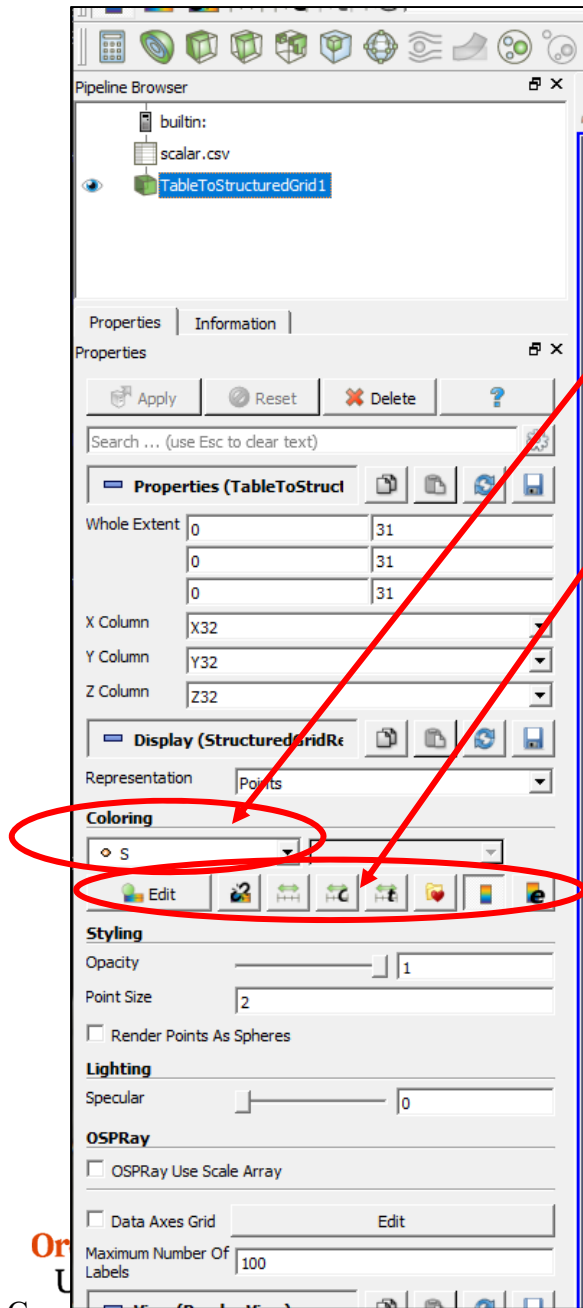


Turning on Color

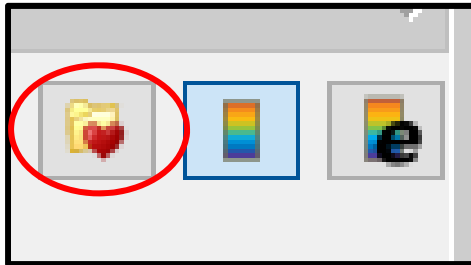
Change the coloring from **Solid Color** to **S**.

This will bring up a row of color command icons.

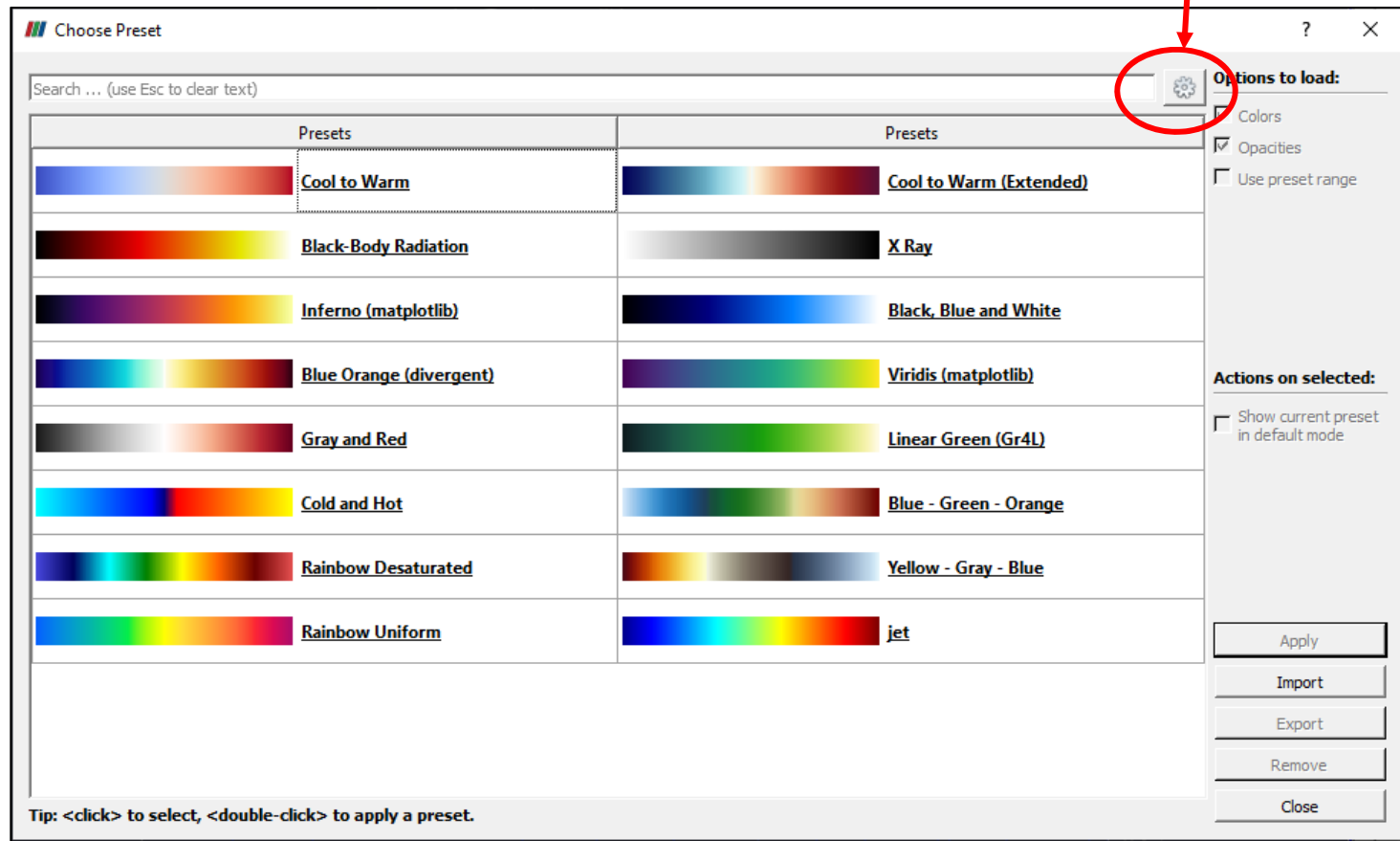
This will also change your display to this and will bring up the color map editor.



















Choose Among Standard Color Transfer Functions



Turn on the “More Detail Gear” and scroll down – there are lots more

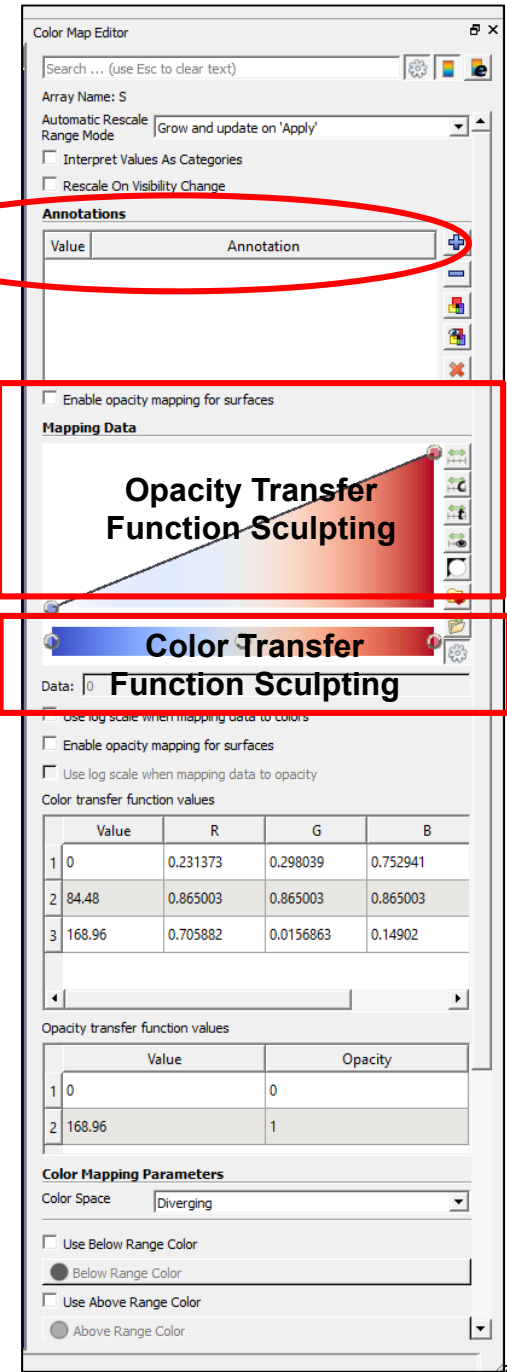
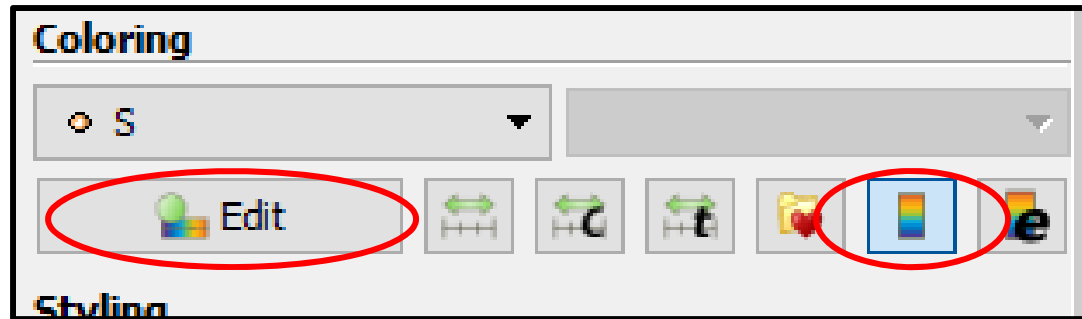


The screenshot shows a 'Choose Preset' dialog box with a search bar and two columns of color transfer function presets. A gear icon in the top right corner is circled in red, with a red arrow pointing to it from the text box above. Another red arrow points downwards from the text box. The 'Options to load' panel on the right has checkboxes for 'Colors', 'Opacities', and 'Use preset range'. The 'Actions on selected:' panel has checkboxes for 'Show current preset in default mode'. Buttons for 'Apply', 'Import', 'Export', 'Remove', and 'Close' are at the bottom right. A tip at the bottom left reads: 'Tip: <click> to select, <double-click> to apply a preset.'

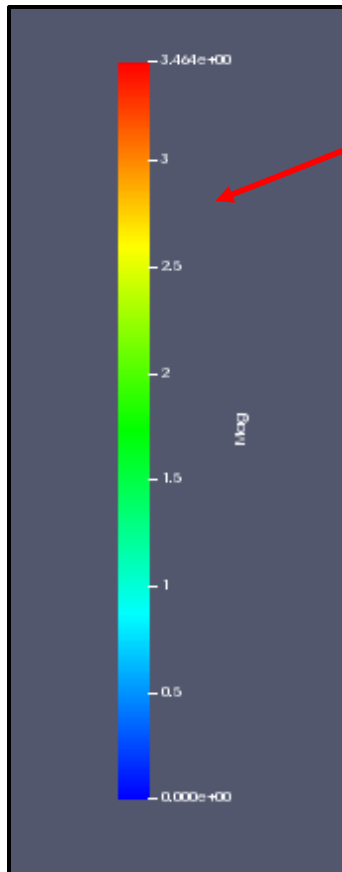
Presets	Presets
 <u>Cool to Warm</u>	 <u>Cool to Warm (Extended)</u>
 <u>Black-Body Radiation</u>	 <u>X Ray</u>
 <u>Inferno (matplotlib)</u>	 <u>Black, Blue and White</u>
 <u>Blue Orange (divergent)</u>	 <u>Viridis (matplotlib)</u>
 <u>Gray and Red</u>	 <u>Linear Green (Gr4L)</u>
 <u>Cold and Hot</u>	 <u>Blue - Green - Orange</u>
 <u>Rainbow Desaturated</u>	 <u>Yellow - Gray - Blue</u>
 <u>Rainbow Uniform</u>	 <u>jet</u>

Color Map Editor

Add annotations to specific values on the legend



Changing the Legend

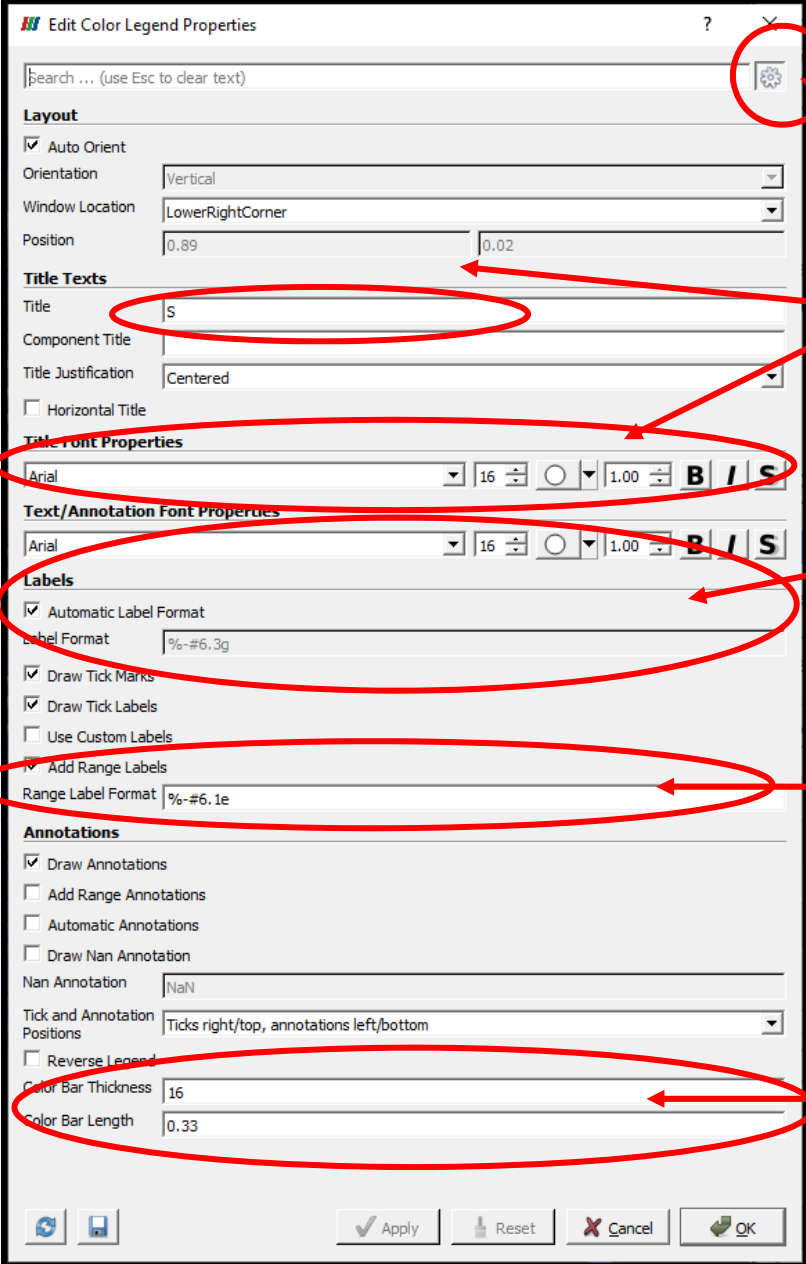


The default legend is barely readable.
You can do better.

Start by clicking here.



Changing the Legend

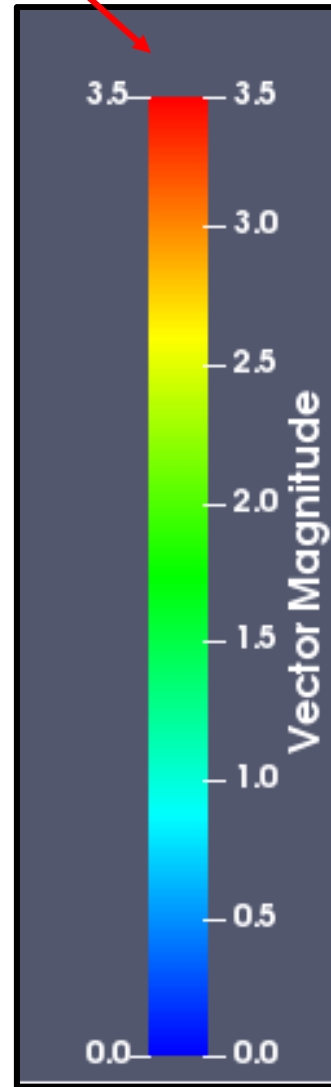
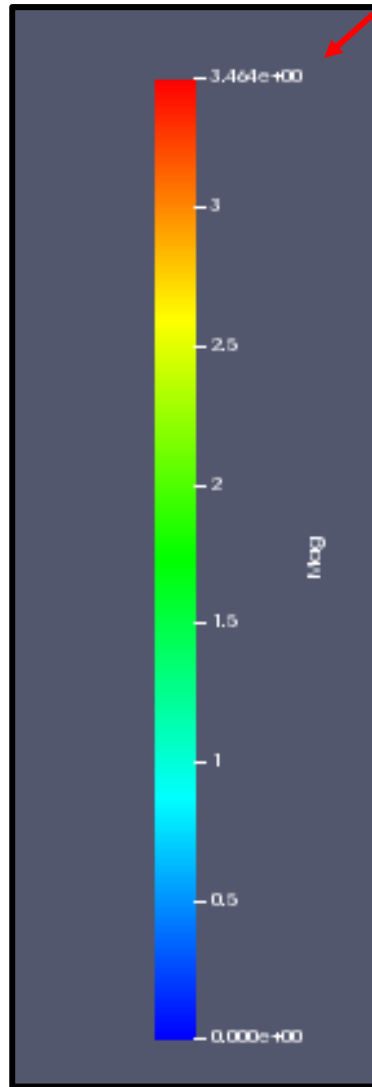


The image shows a screenshot of the 'Edit Color Legend Properties' dialog box in ParaView. The dialog is divided into several sections: Layout, Title Texts, Title Font Properties, Text/Annotation Font Properties, Labels, and Annotations. Red circles and arrows highlight specific settings:

- Settings Gear:** A red circle around the gear icon in the top right corner, with an arrow pointing to the text: "Click here to bring up all of the options. (This is a good idea on *all* ParaView dialog boxes.)"
- Title Texts:** A red circle around the 'Title' text field containing the letter 'S', with an arrow pointing to the text: "Legend title and font".
- Title Font Properties:** A red circle around the font settings (Arial, size 16, bold, italic, strikethrough), with an arrow pointing to the text: "Tick mark font and number format ('printf-style')".
- Labels:** A red circle around the 'Range Label Format' field containing the format string '%-#6.1e', with an arrow pointing to the text: "Range numbers at the end of the legend".
- Color Bar:** A red circle around the 'Color Bar Thickness' and 'Color Bar Length' fields (values 16 and 0.33), with an arrow pointing to the text: "Color bar".

At the bottom of the dialog, there are buttons for 'Apply', 'Reset', 'Cancel', and 'OK'.

From this, to this

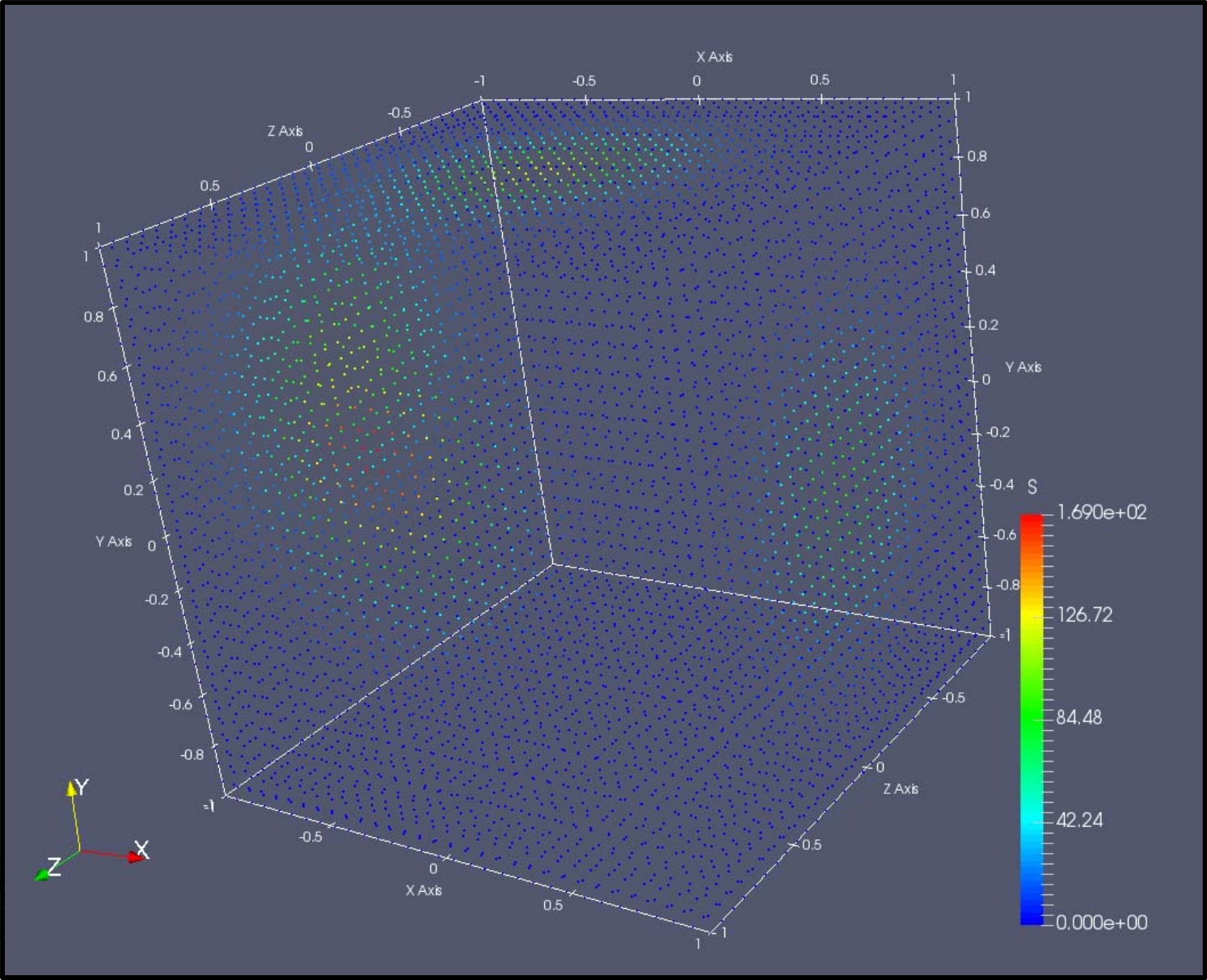


Visualizing Scalar Data, II



scalar.csv

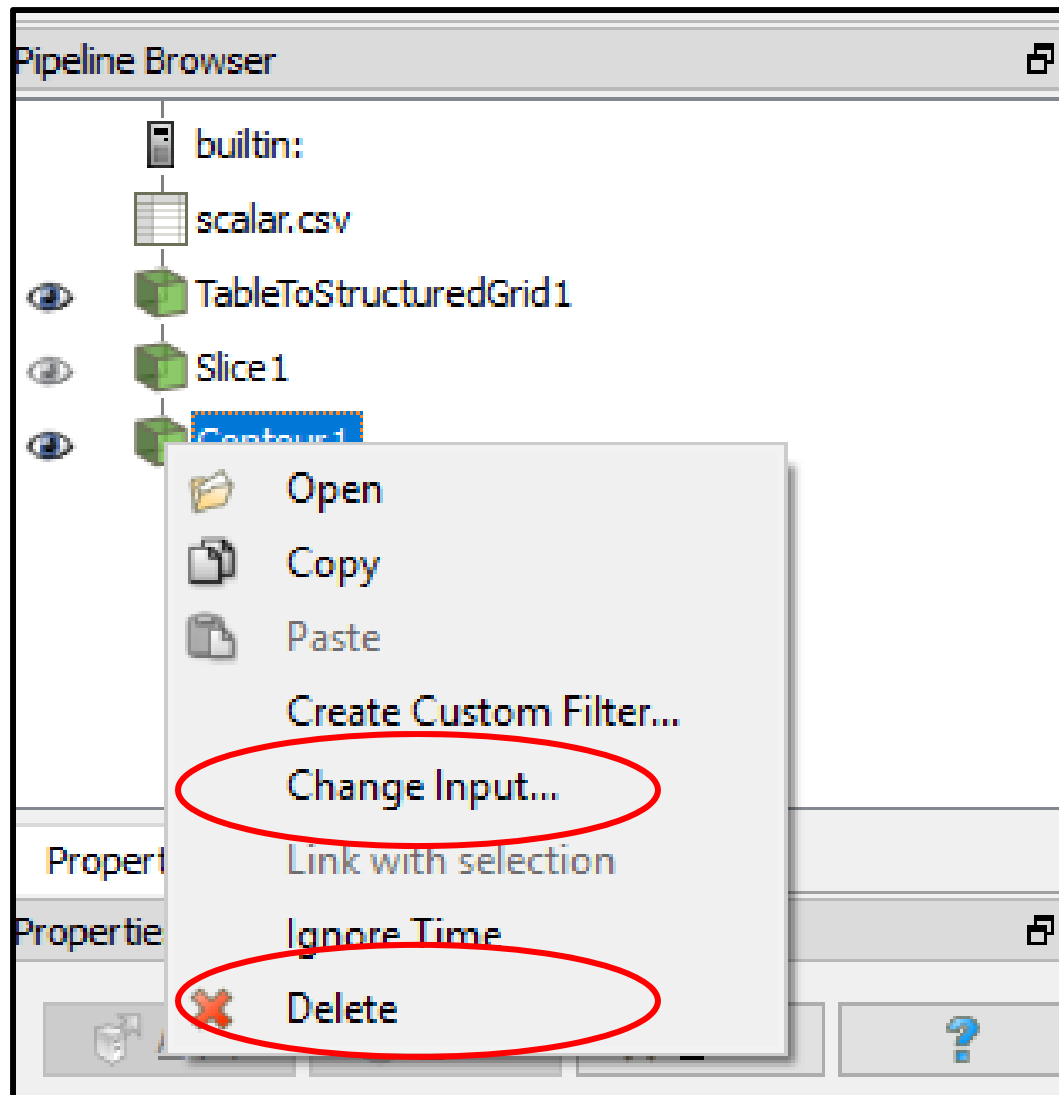
As Points



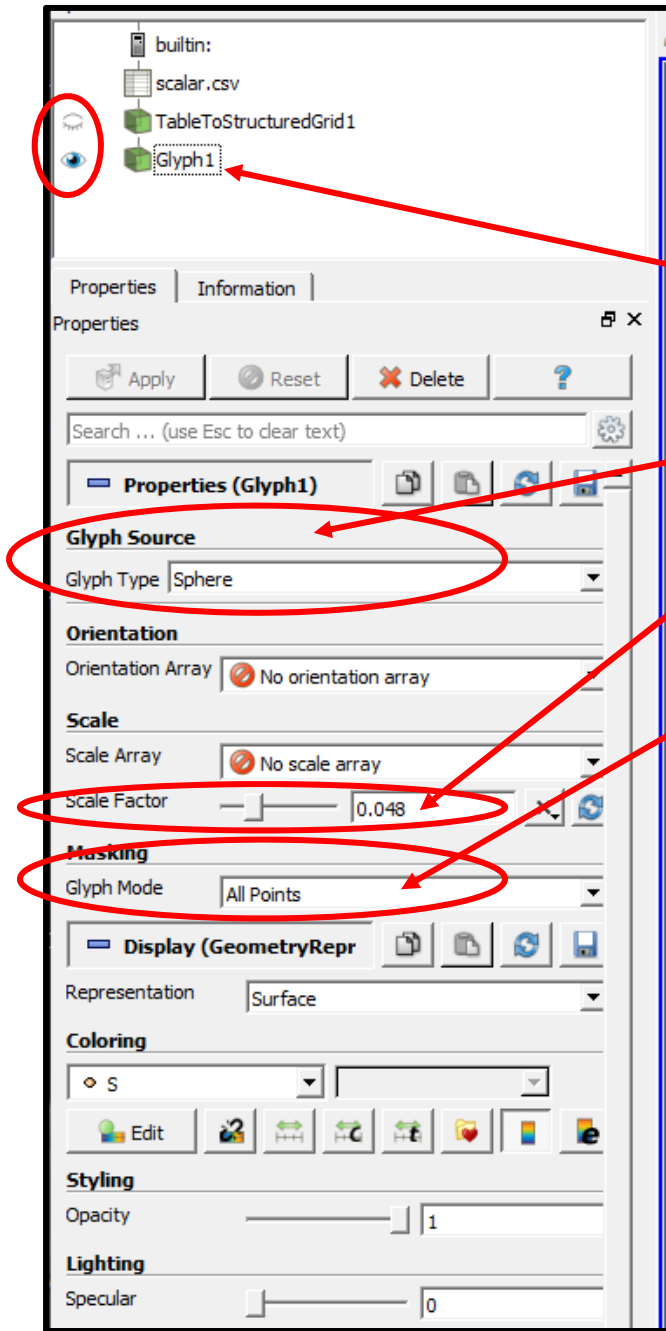
Pipeline Element and Filter Observations

- Whatever pipeline element you have most-recently clicked on, that's what Properties you will see.
- Whatever pipeline element you have most-recently clicked on, that will be the parent of the next Filter you select. The parent's output will become the Filter's input.
- Be careful of Filter order. In general, Filters are not commutative or associative.
- For data-size reasons, it is helpful if any datasize reduction Filters are included early in the pipeline.
- As far as I can tell, you can't inject a filter in the middle of a pipeline. You can re-parent it. You can delete it and pipeline elements around it and start over. But, adding a new Filter between two existing pipeline elements creates a tee from the parent, not a new pipeline.
- Whatever "eyeballs" you have clicked on, that's what pipeline elements' visual representations you will see in the display.
- Turn on the **TableToStructuredGrid** "eyeballs" and set the Representation to **Outline**. That keeps ParaView displaying the data as 3D-fullsize, regardless of what downstream pipeline elements do.

Right-clicking on a Pipeline Element



As a Glyph Cloud

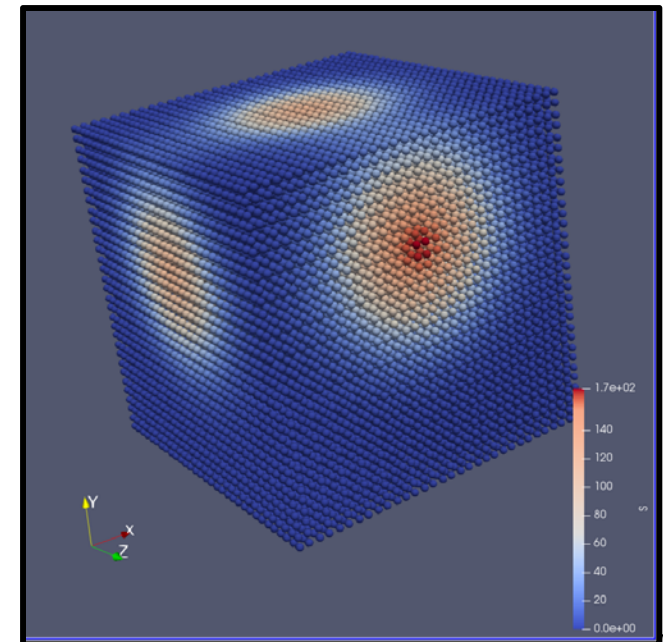


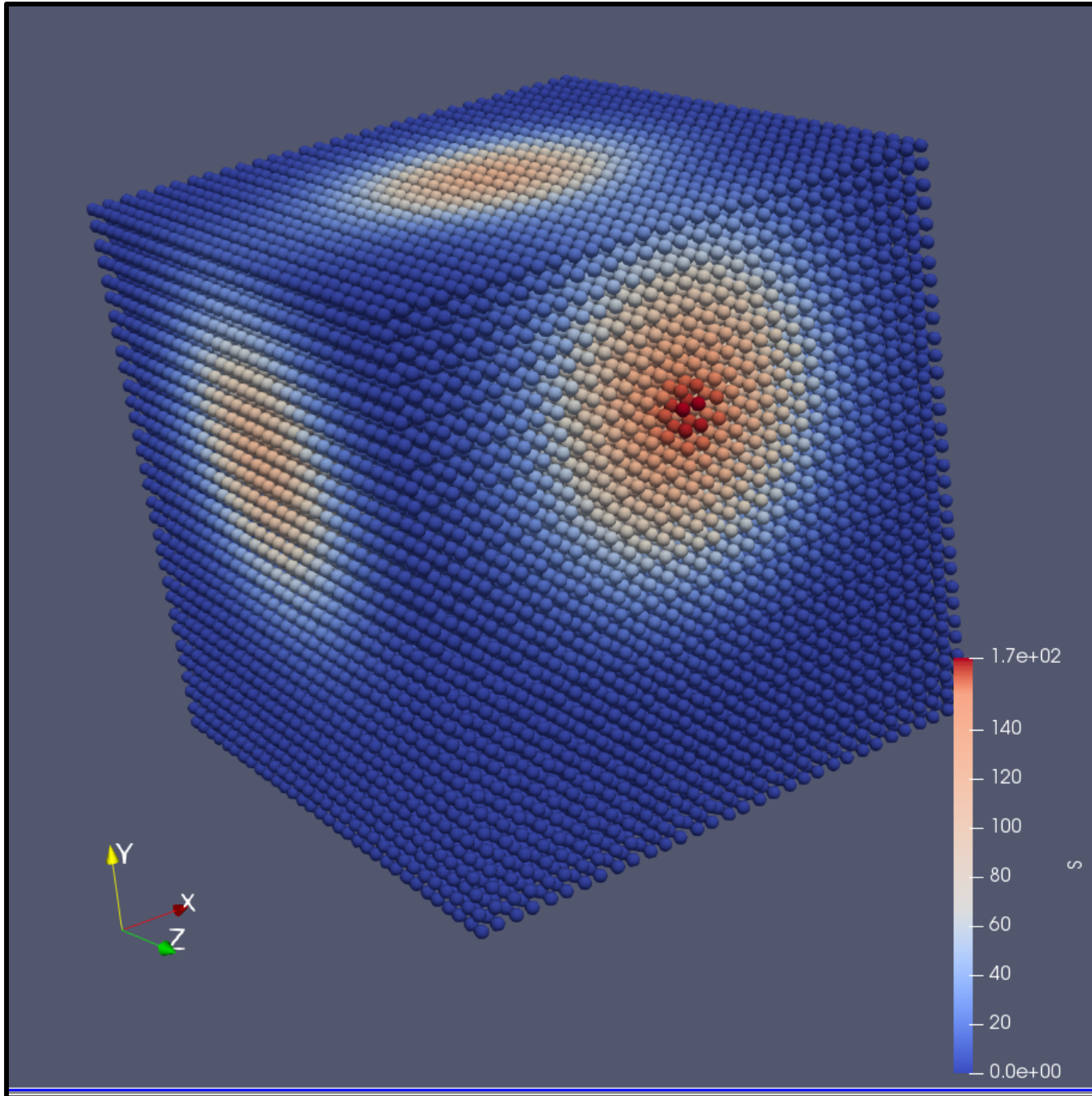
Filters → **Alphabetical** → **Glyph** adds the glyph cloud to the pipeline. Hide the **TableToStructuredGrid** (click off the eyeball) and un-hide the **Glyph**.

Set the **Glyph Type**

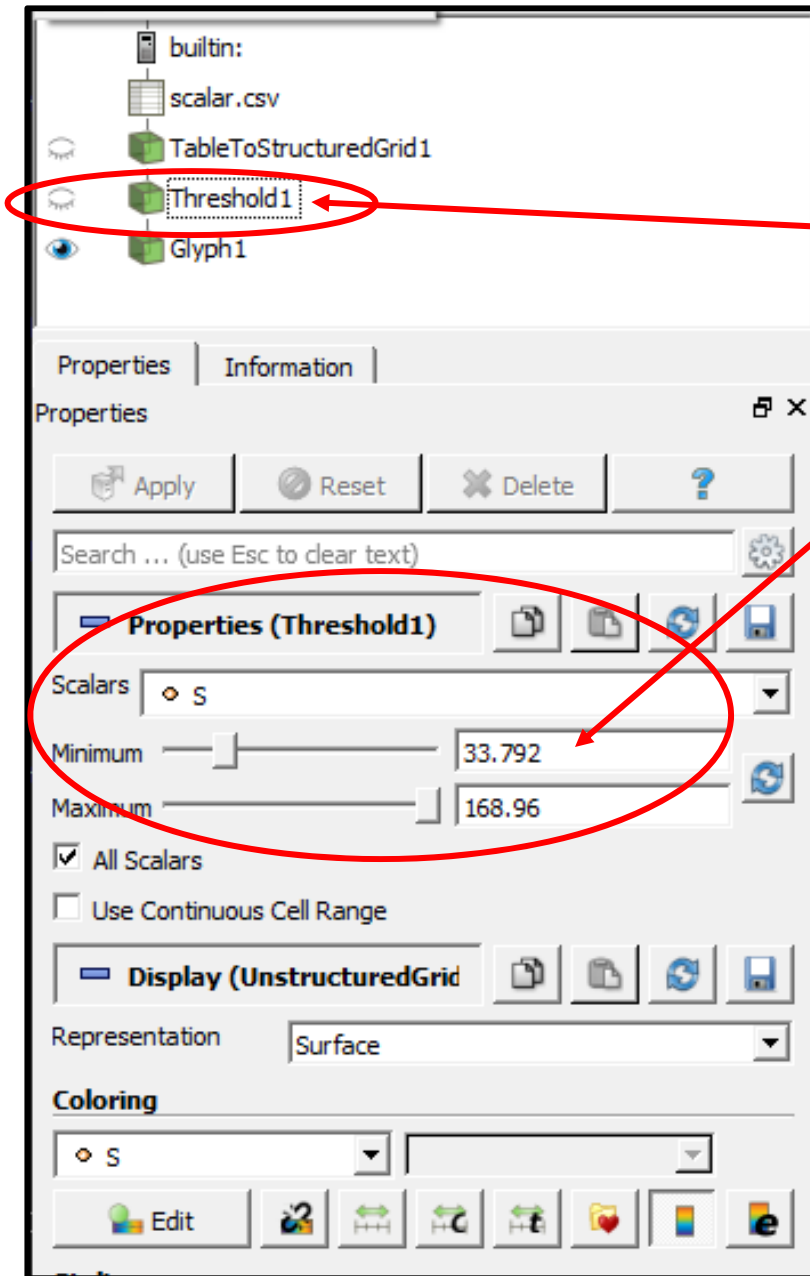
Play with the **Scale Factor**

Play with the **Glyph Mode**





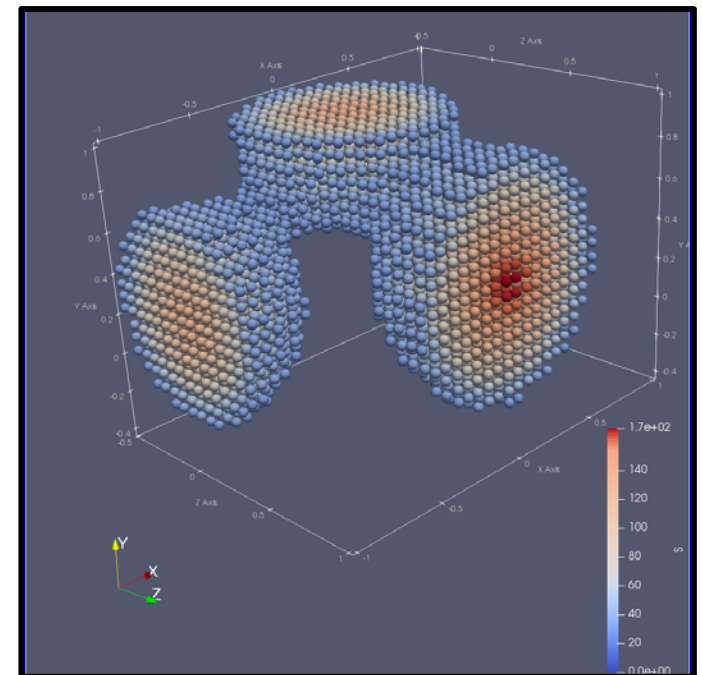
As a Threshold Glyph Cloud

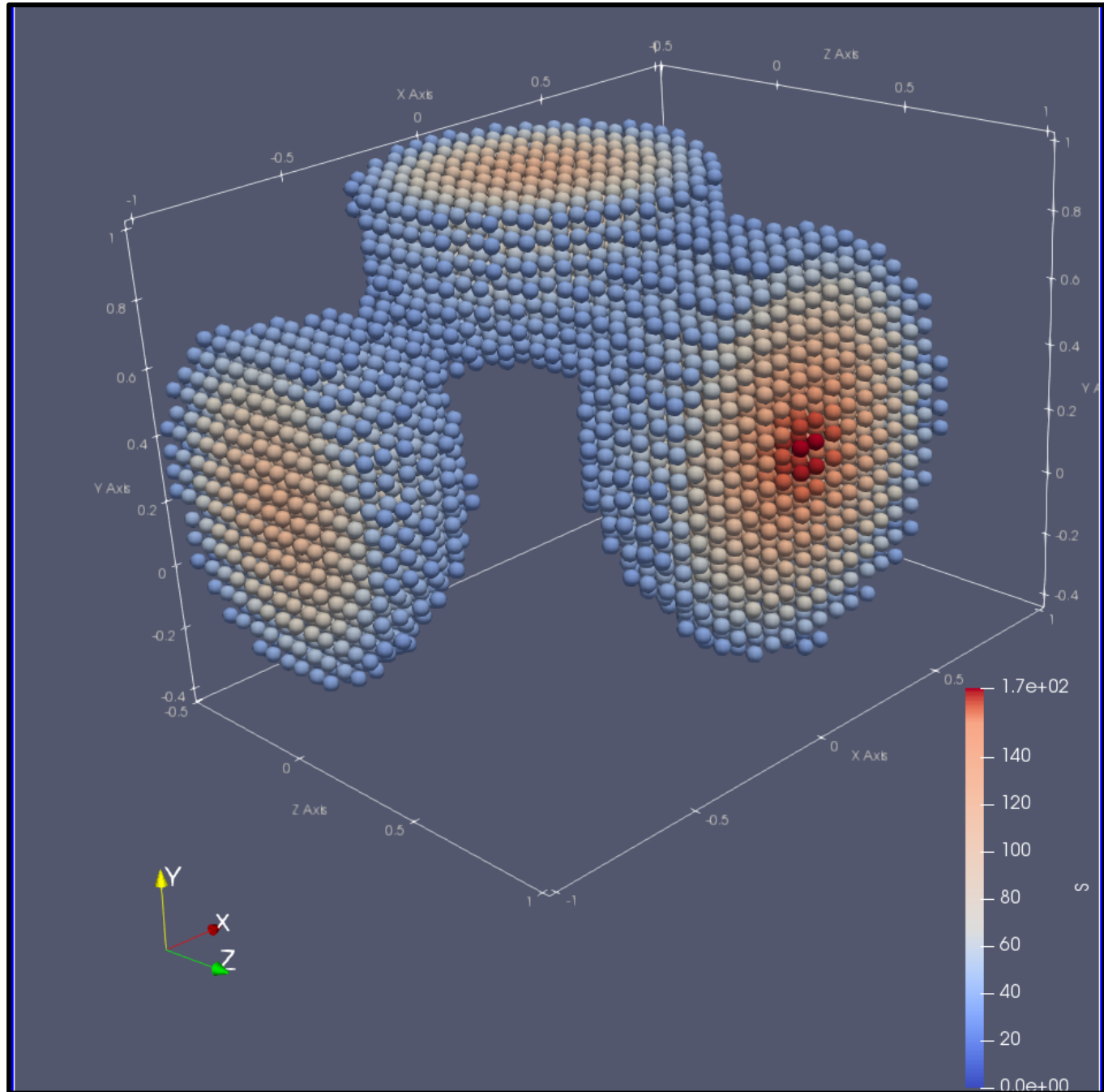


Click on **TableToStructuredGrid**, then select **Filters** → **Alphabetical** → **Threshold**.

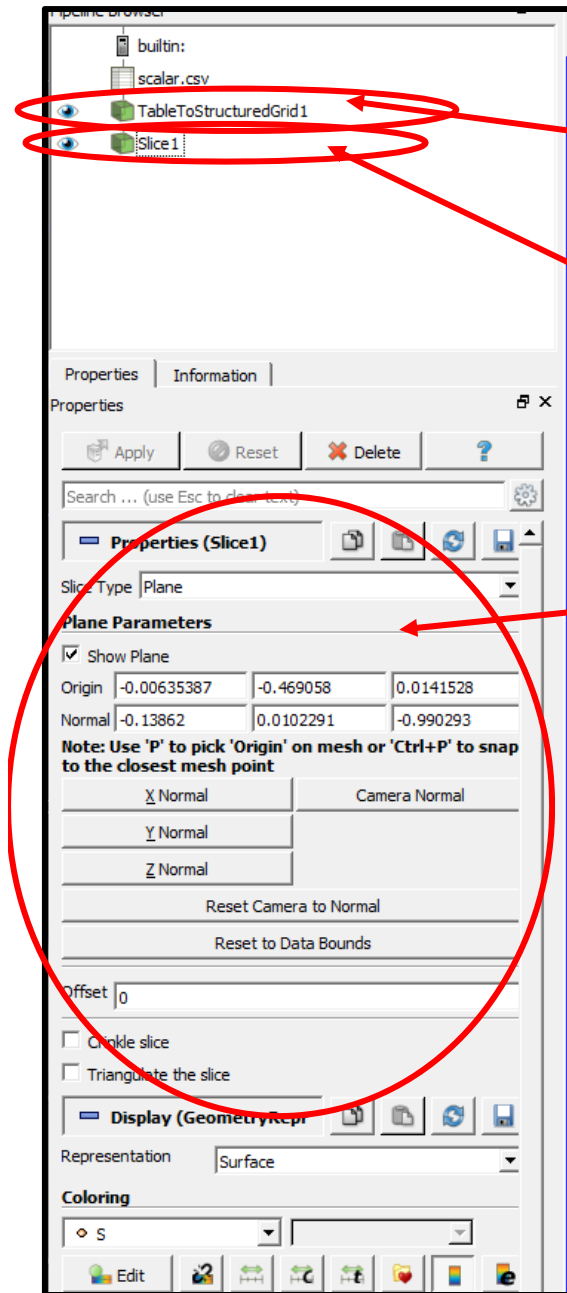
Hide the **TableToStructuredGrid** and the **Glyph**, then un-hide the **Threshold**.

Set the **Minimum** and **Maximum**. (Be sure to click on **Apply**.)





As a Colored Cutting Plane

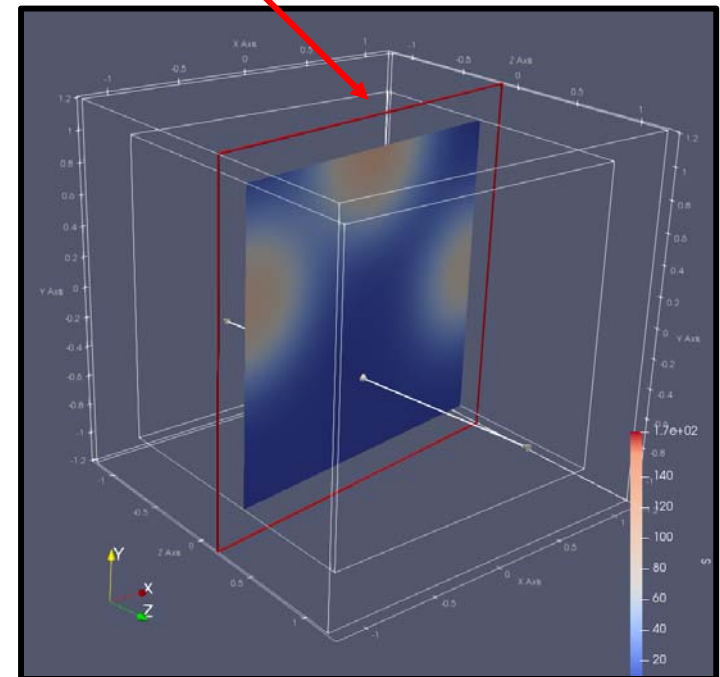


ParaView trick – turn on the **TableToStructuredGrid** display and set the Representation to **Outline**. That keeps ParaView from displaying the plane as 2D-only

Click on **TableToStructuredGrid**, then select **Filters** → **Alphabetical** → **Slice**

Click on the red lines to move the plane.

Click in here to change slice parameters.

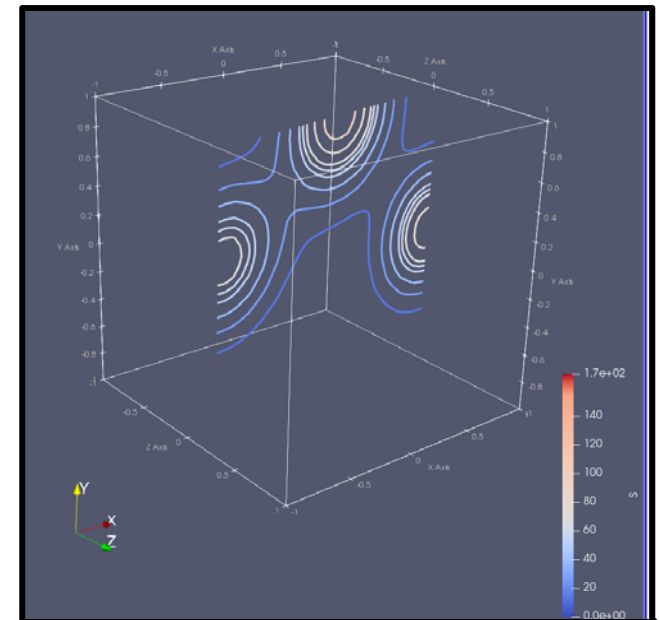
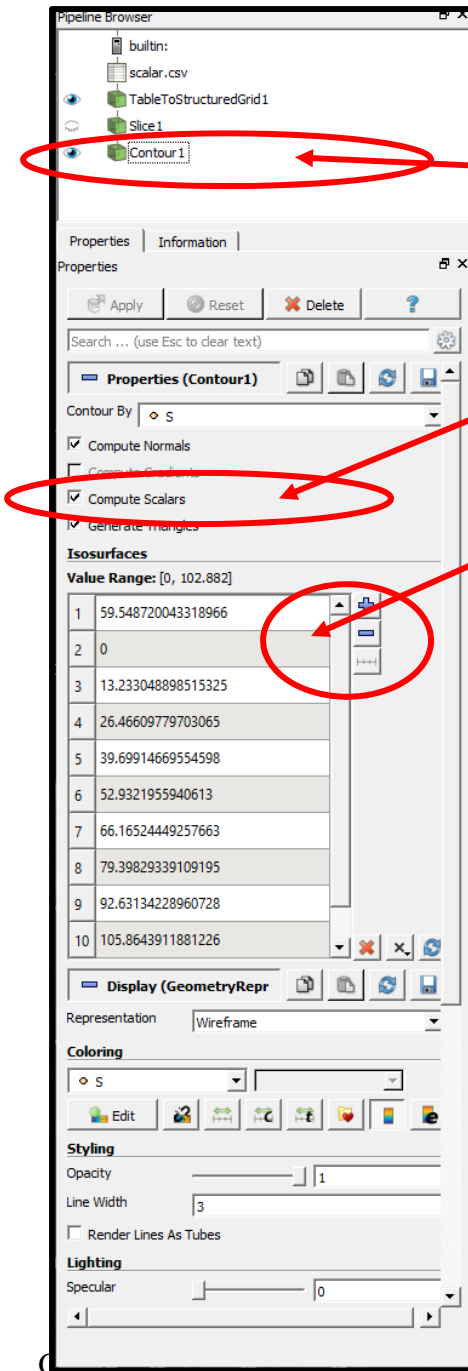


As Contours

Add a **Contours** filter

Click on Compute Scalars

Create contour values (see next slide)



As Contours

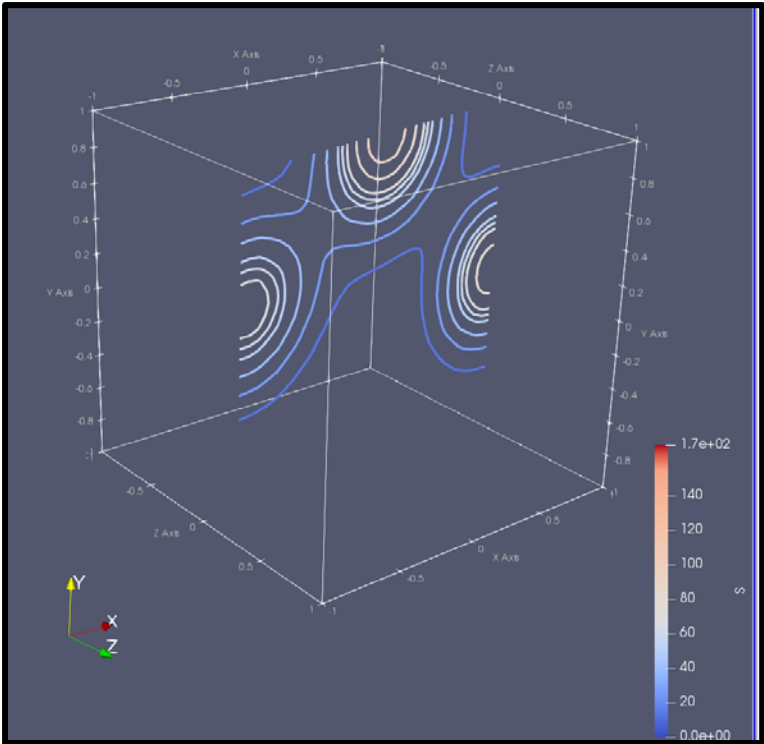
Value Range: [0, 153.575]

1	30
2	0
3	18.773333333333333

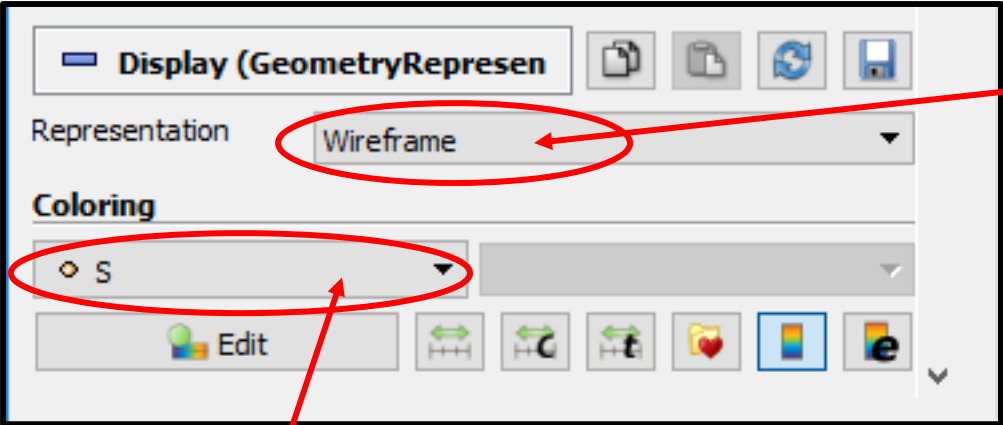
Add a contour isovalue

Delete a contour isovalue

Add a range of contour isovalues



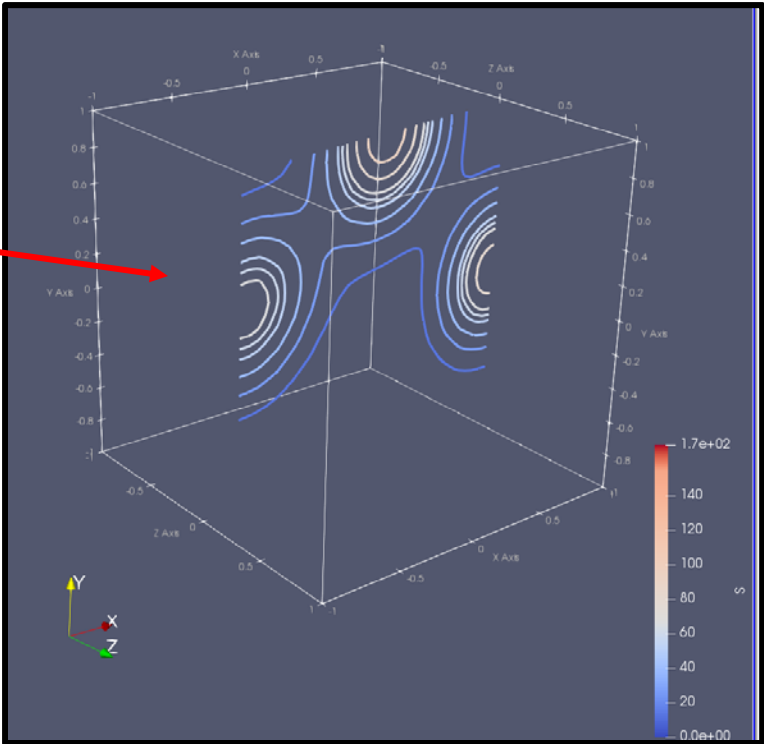
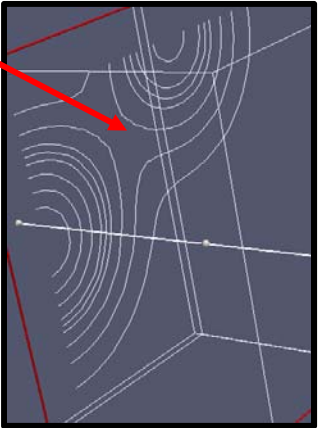
As Contours



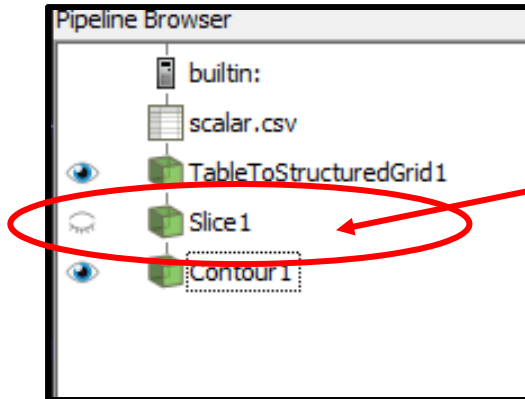
This needs to be **Wireframe** to get contour lines

Coloring by S will give you colored contour lines.

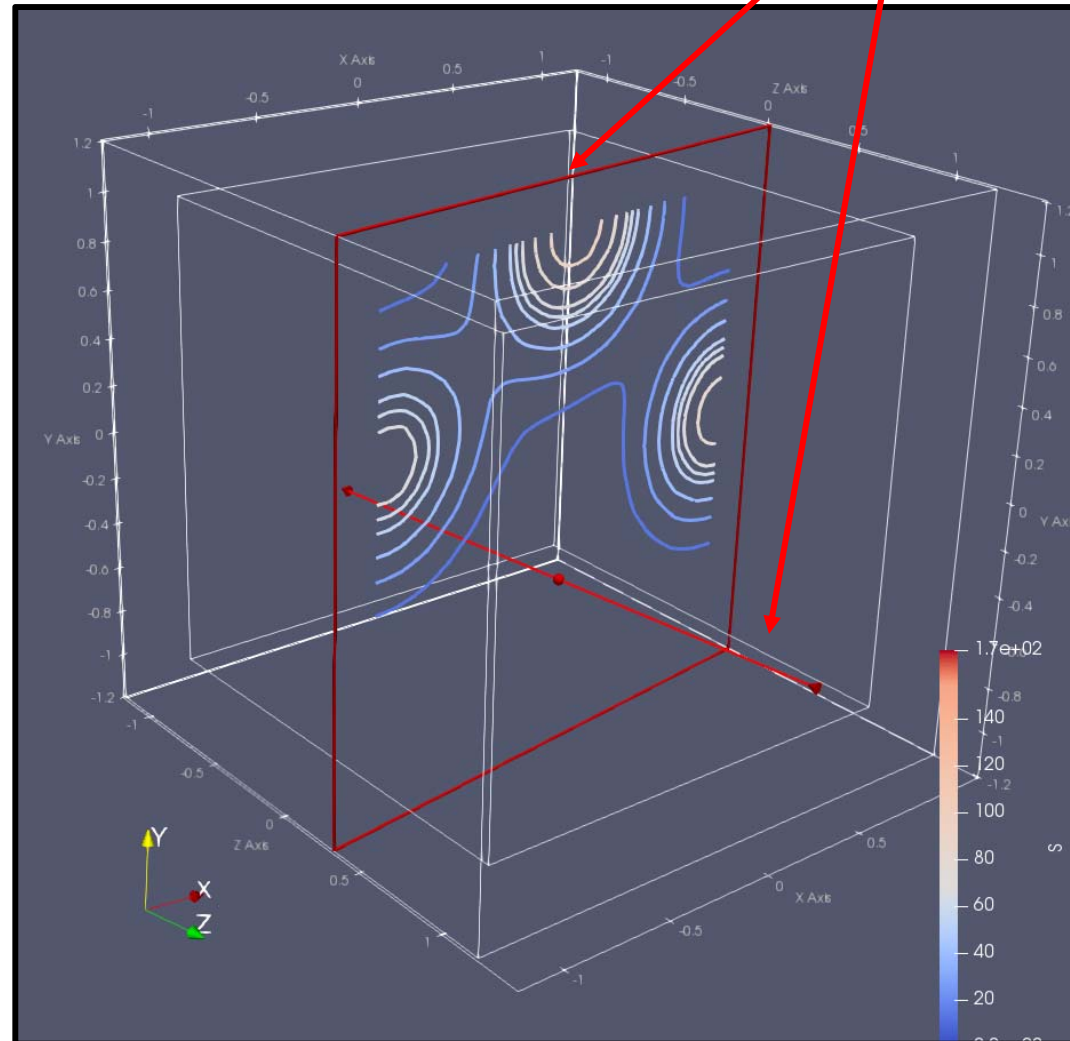
Coloring by Solid Color will give you a single color.



As Contours

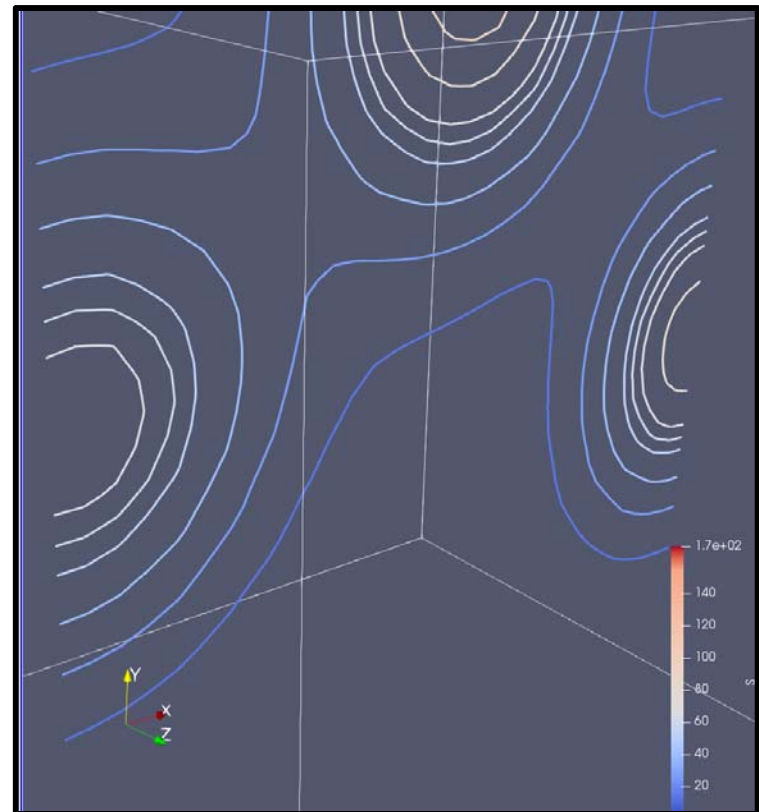
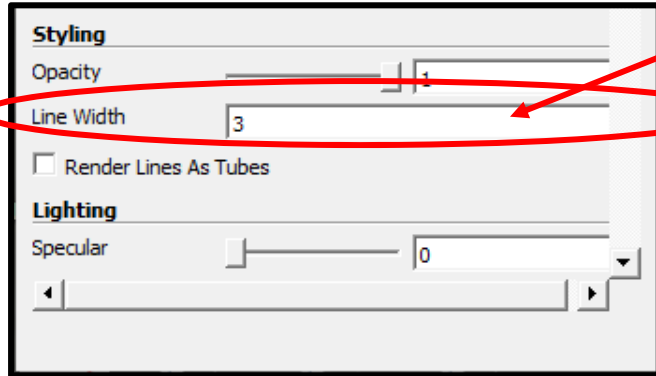


Clicking on the Slice filter will bring up these slice handles so that you can move and re-orient the slice plane



As Contours

Adjusting the **Line Width**



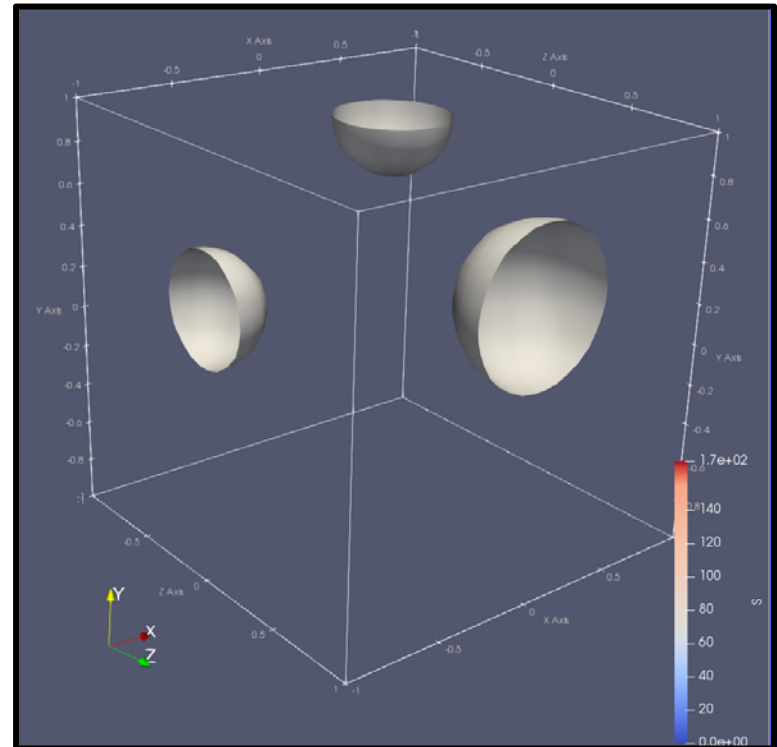
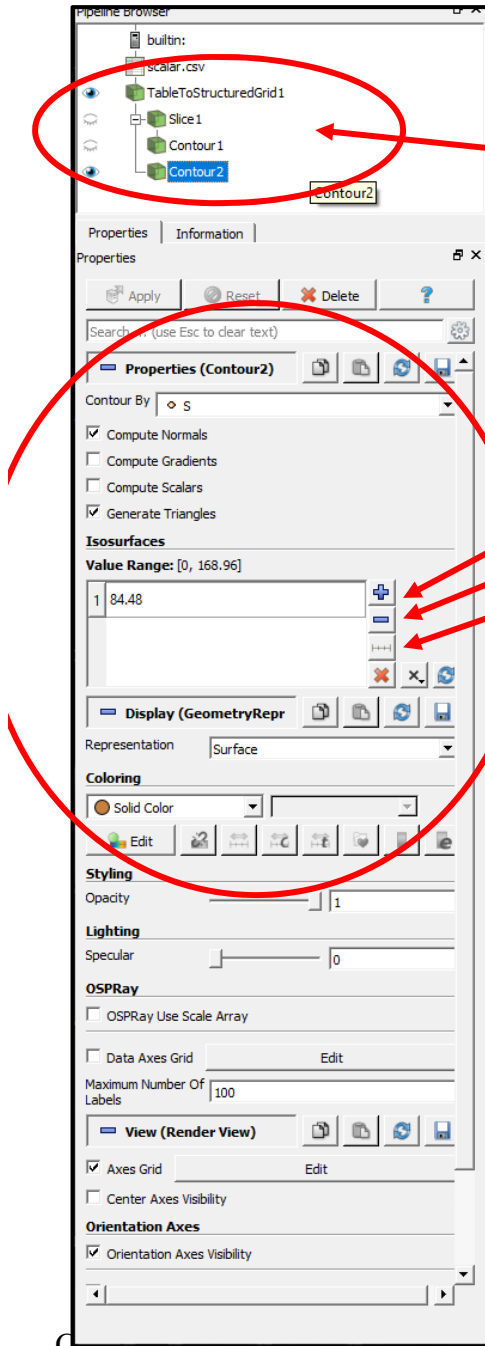
As Isosurfaces

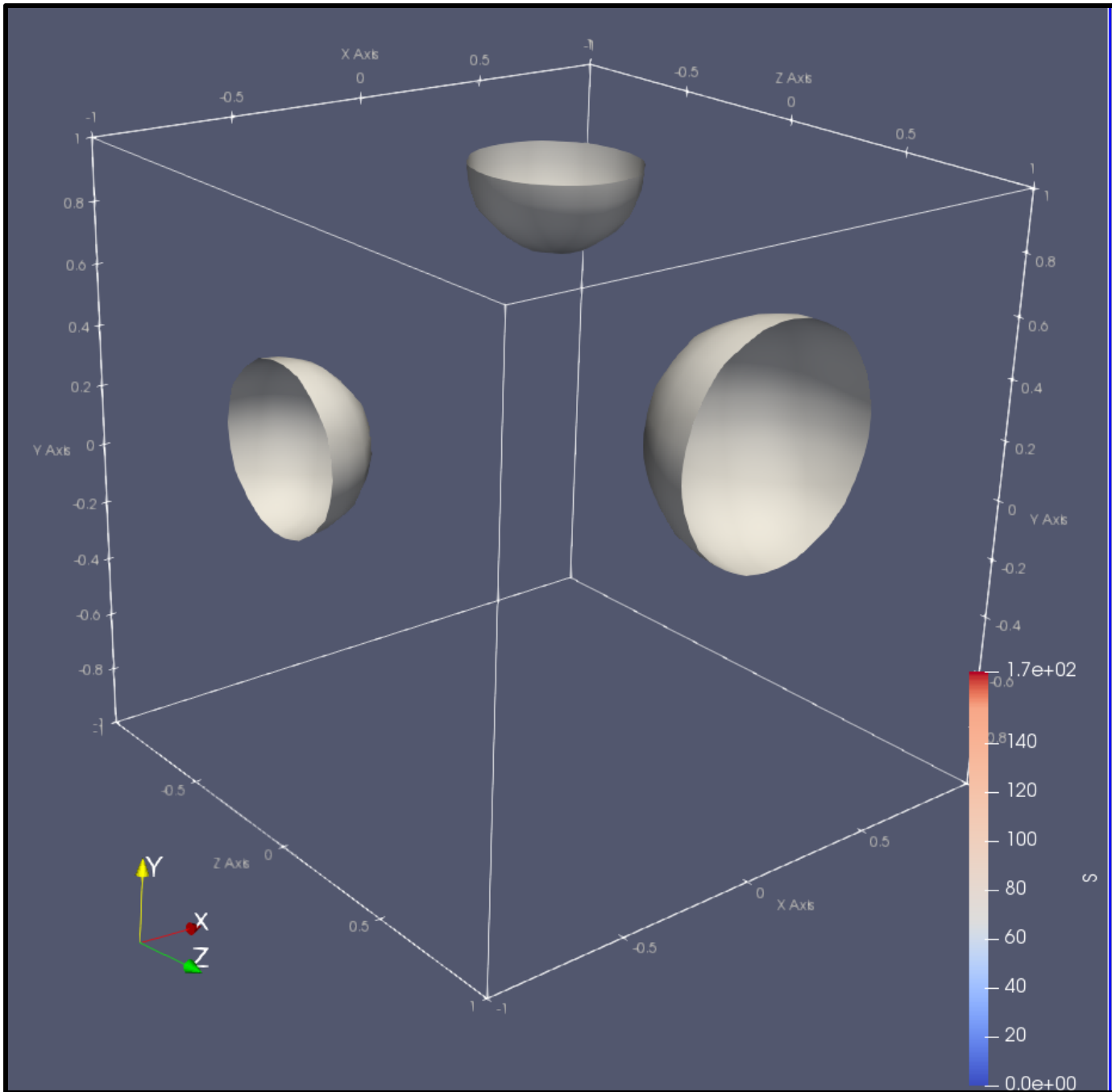
Note – This instance of **Contour** is parented from **TableToStructuredGrid**, not **Slice**

Add a contour isovalue

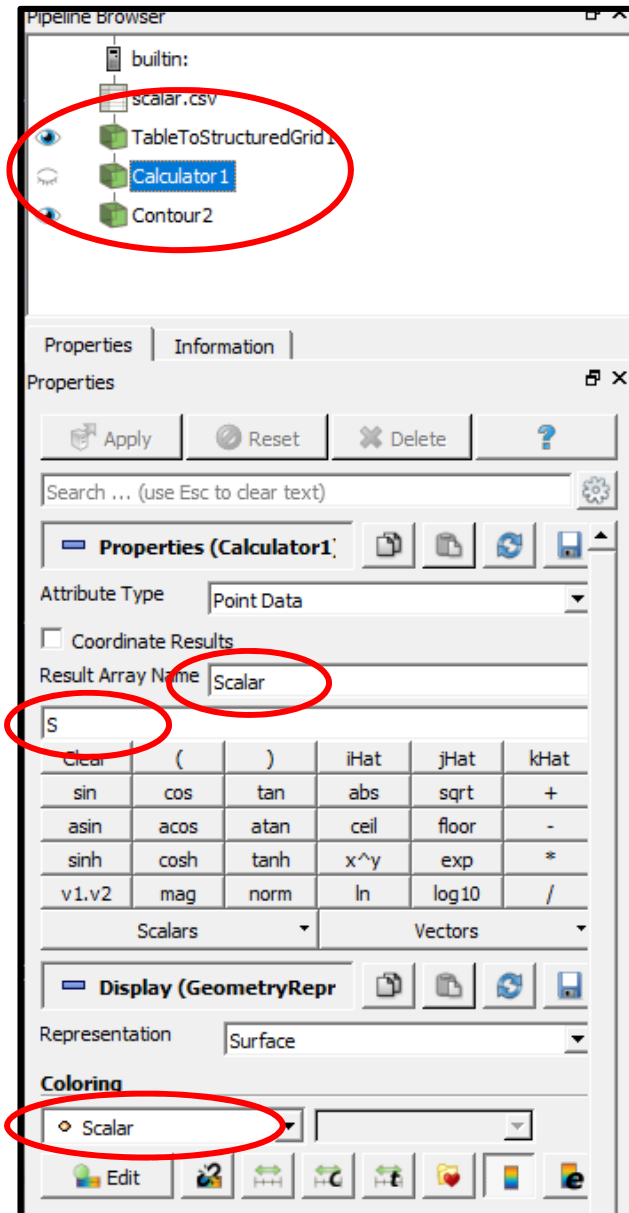
Delete a contour isovalue

Add a range of contour isovalues

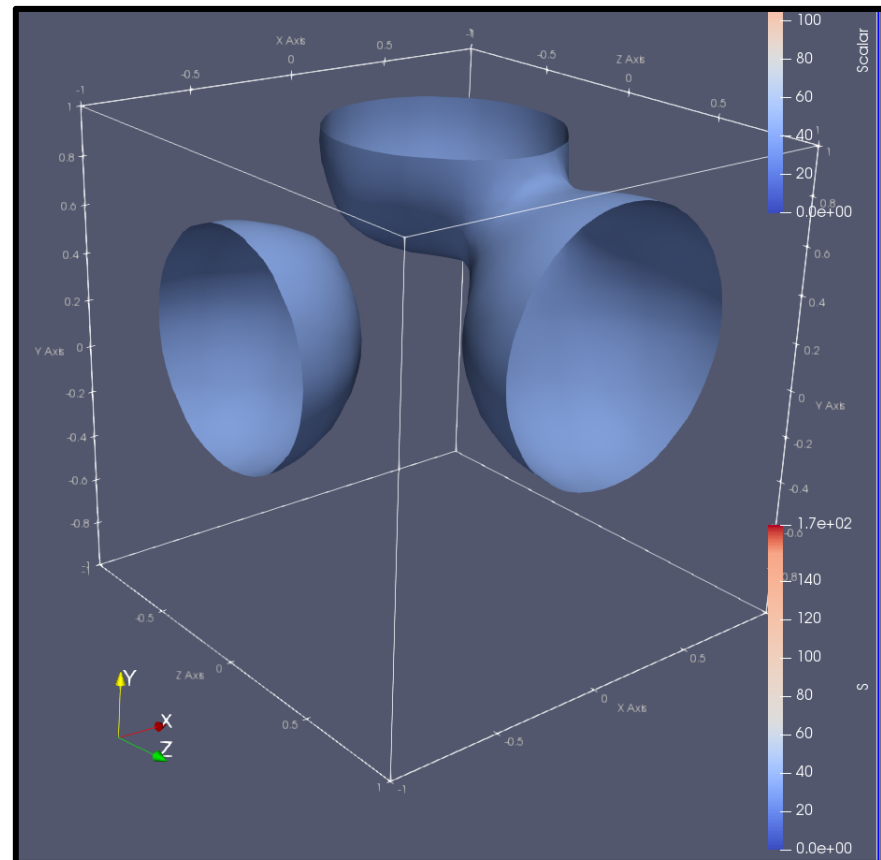


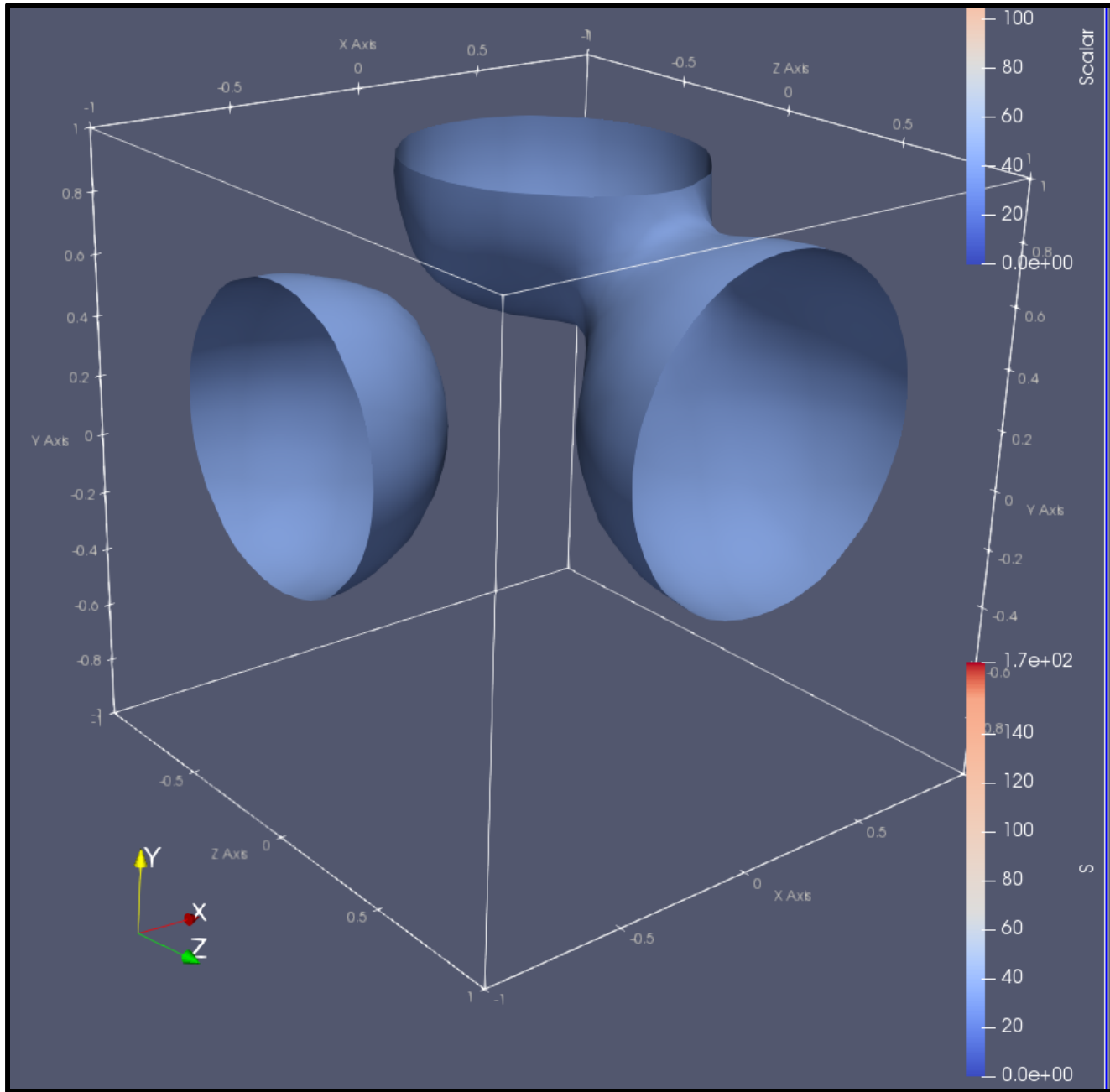


Using the Calculator to Duplicate S to be Able to Color by Scalar Value

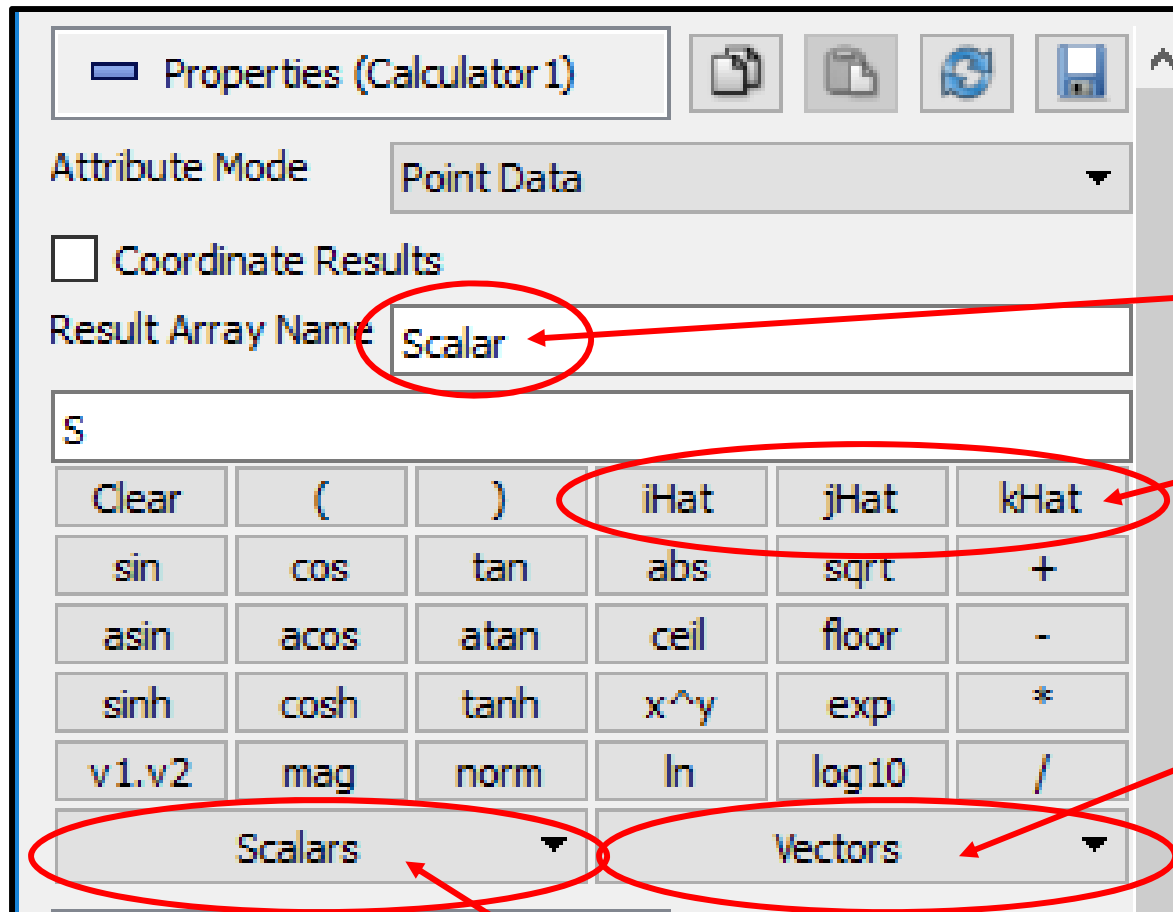


Computer Graphics





The Calculator



Name of the output field

Coordinate unit vectors

A list of the current vector variables in the dataset

A list of the current scalar variables in the dataset

As a Volume

builtin:
scalar.csv
TableToStructuredGrid1

Properties Information

Properties

Apply Reset Delete ?

Search ... (use Esc to clear text)

Properties (TableToStr)

Whole Extent

0	31
0	31
0	31

X Column X32
Y Column Y32
Z Column Z32

Display (StructuredGrid)

Representation Volume

Coloring

S

Styling

Opacity 1

Lighting

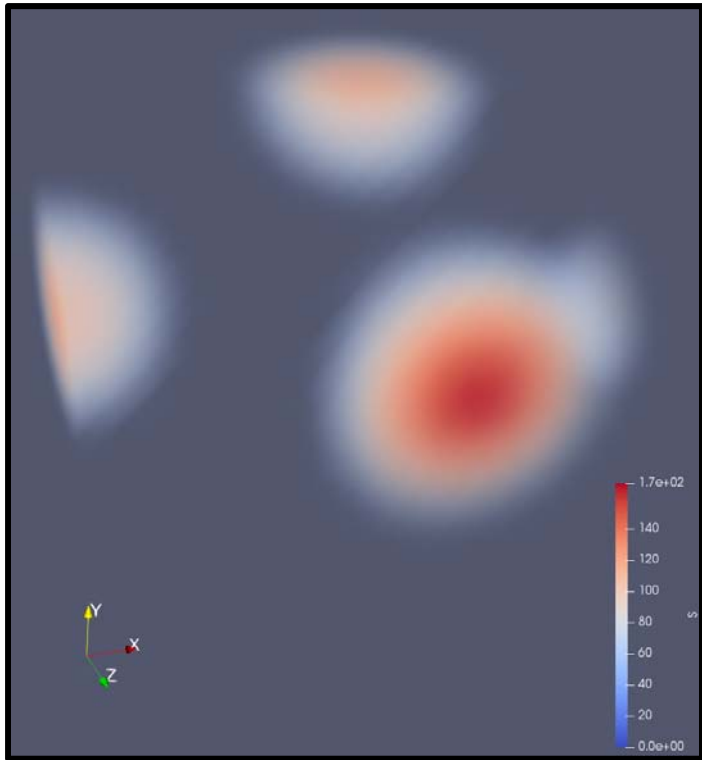
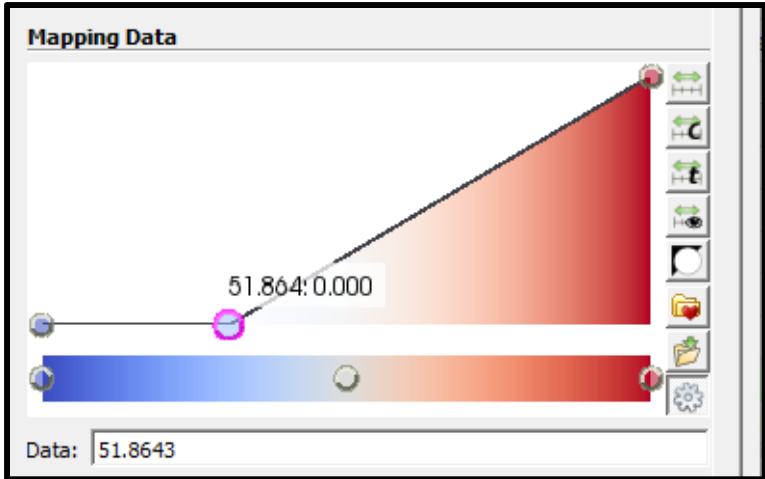
Specular 0

OSPRay

OSPRay Use Scale Array

Data Axes Grid Edit

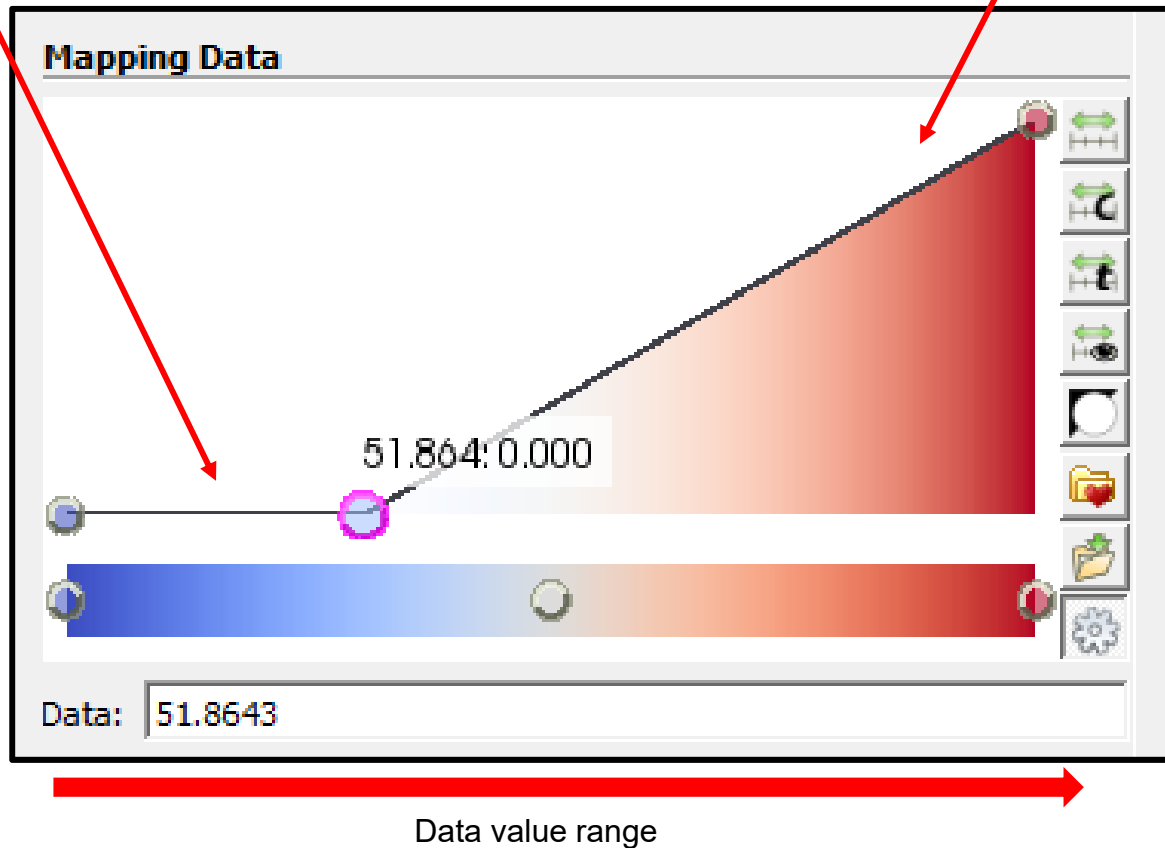
Maximum Number Of Labels 100



Sculpting the Alpha Transfer Function

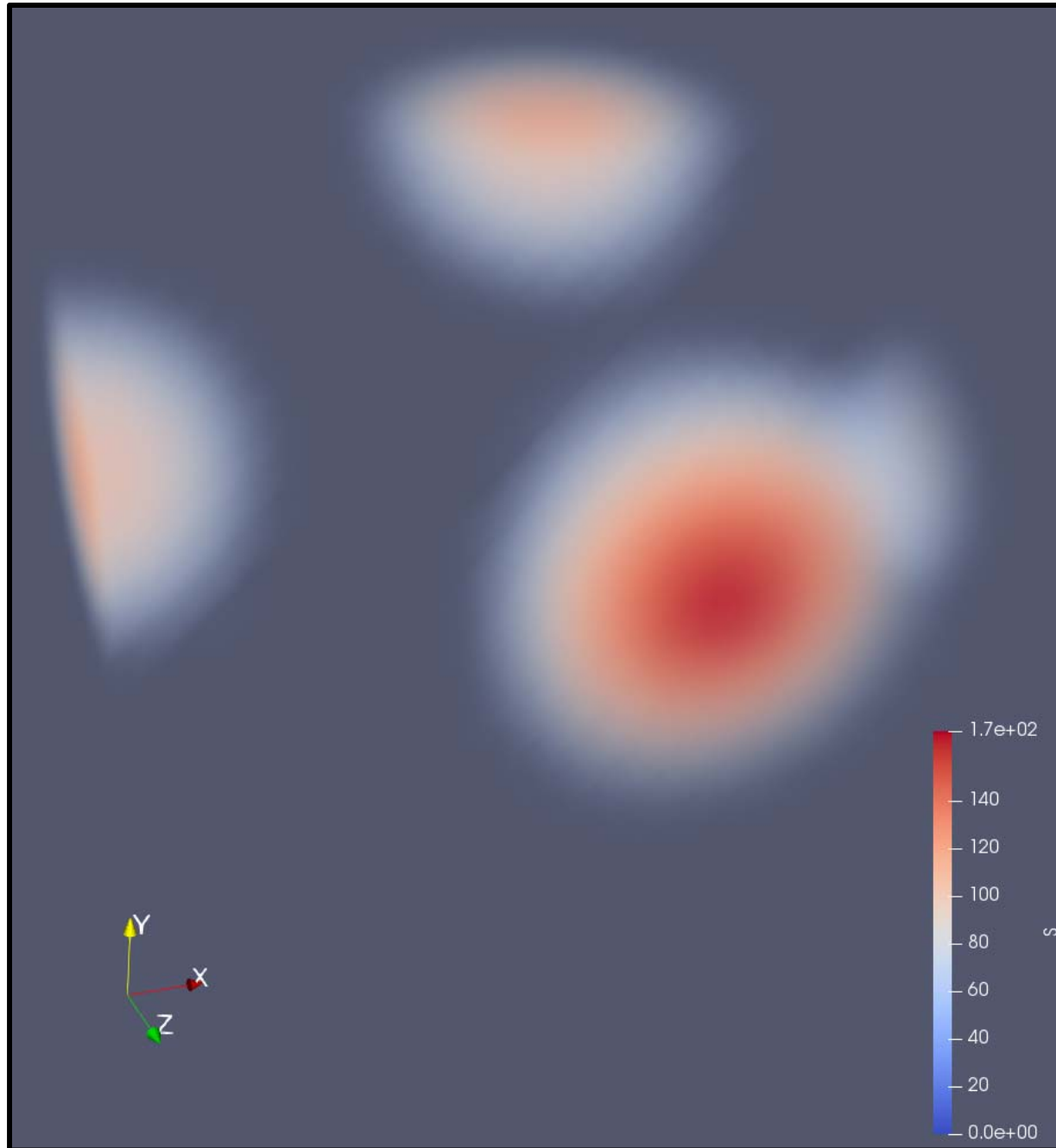
Hover over the black line and **left-click** to add a new sculpting point there

Hover over a point and hit the **Delete** key or **Middle Mouse Button** to delete a point

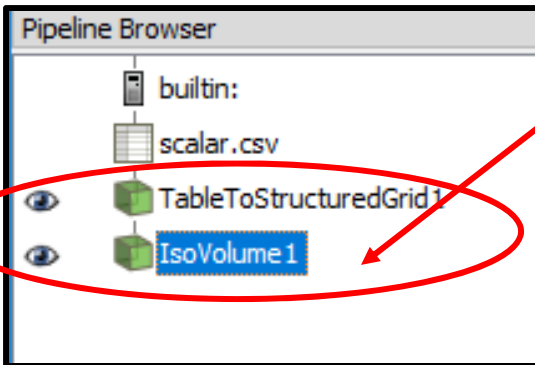


Alpha=1.
(opaque)

Alpha=0.
(transparent)



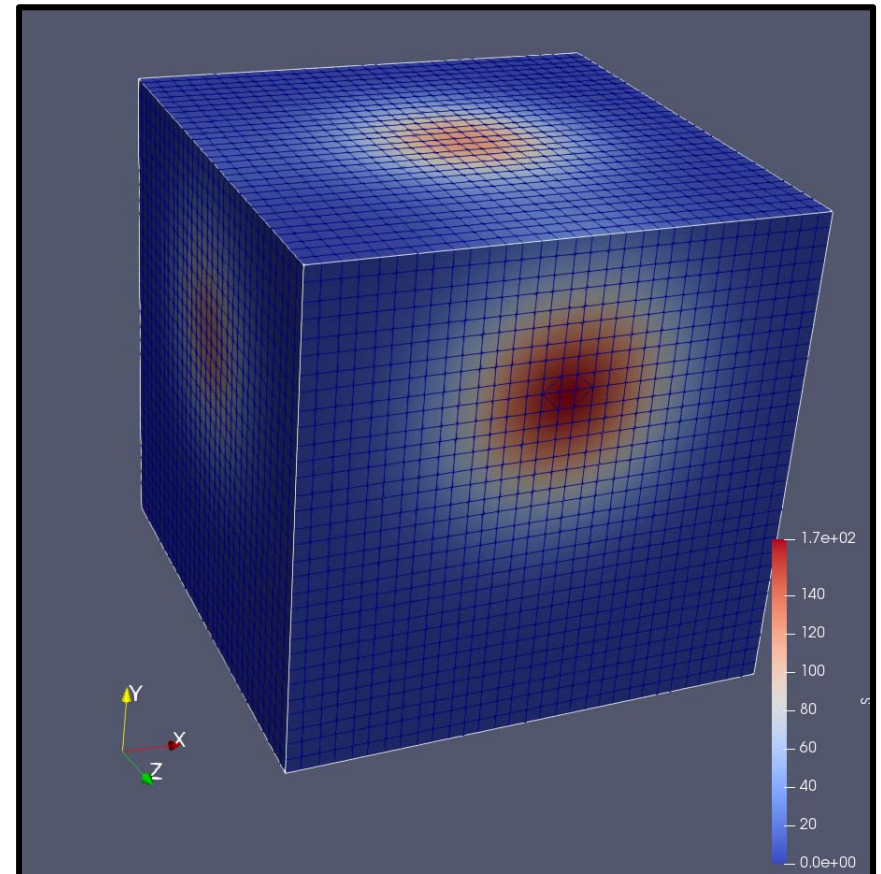
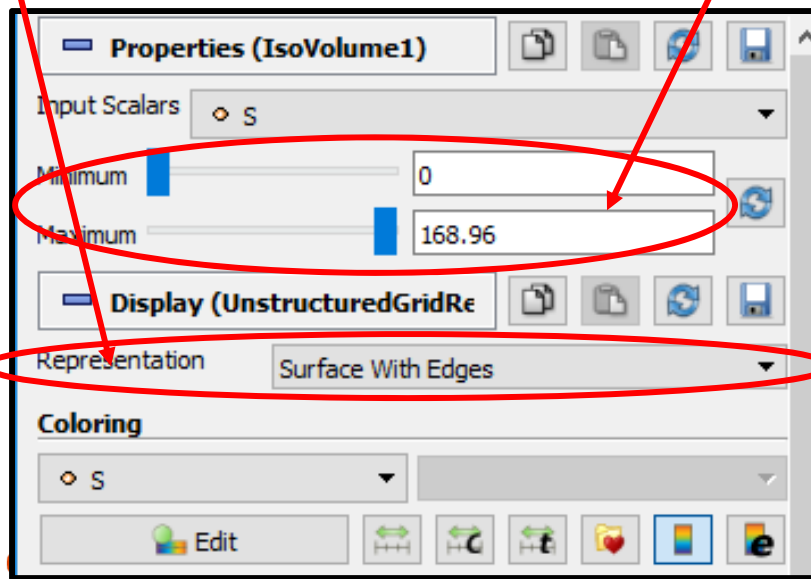
IsoVolumes



Start with this

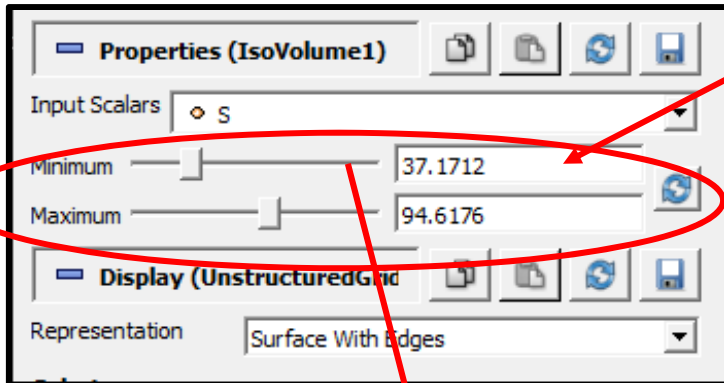
The **IsoVolume** properties start out at “allow all values” to pass through. We’re going to change this.

I chose the **Surfaces with Edges** representation so you can see the cells. You’ll see why in a moment.

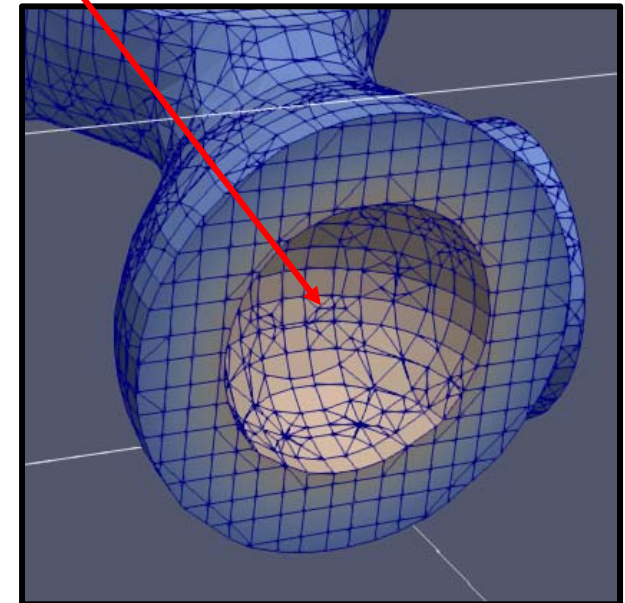
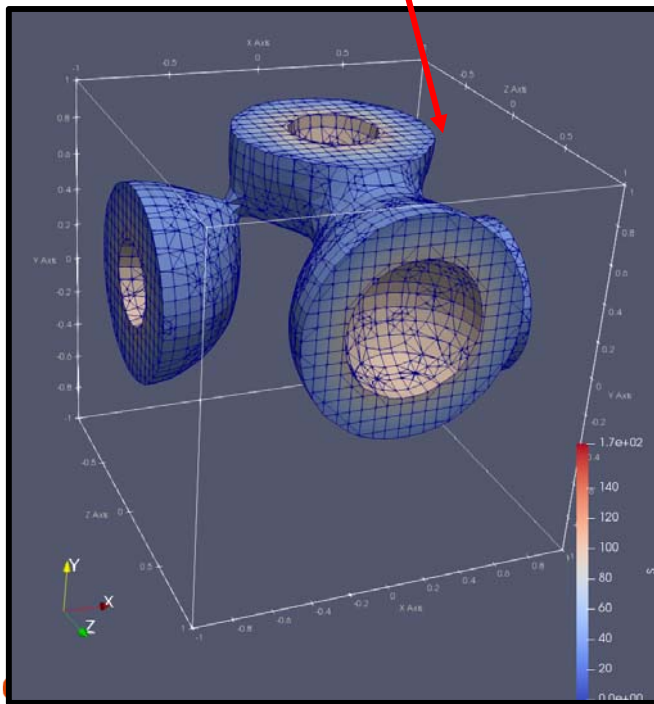


IsoVolumes

Now adjust the Minimum and Maximum to something else.

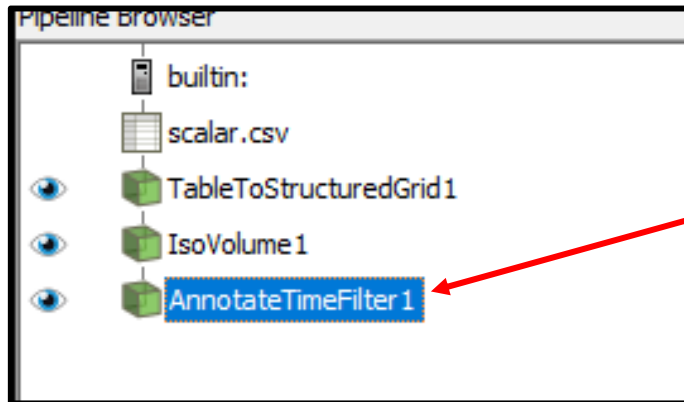


Note that the **IsoVolume** filter turned your nice, efficient structured grid into an unstructured grid. This can balloon the size of the data that is being operated on.



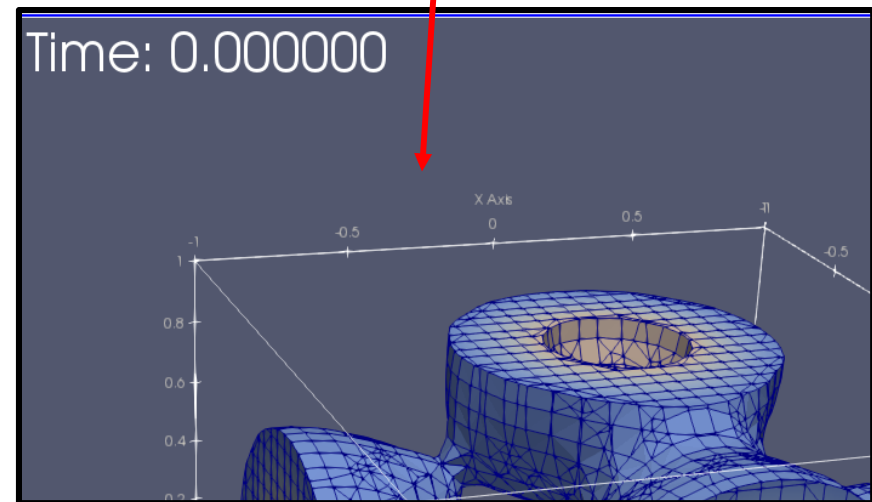
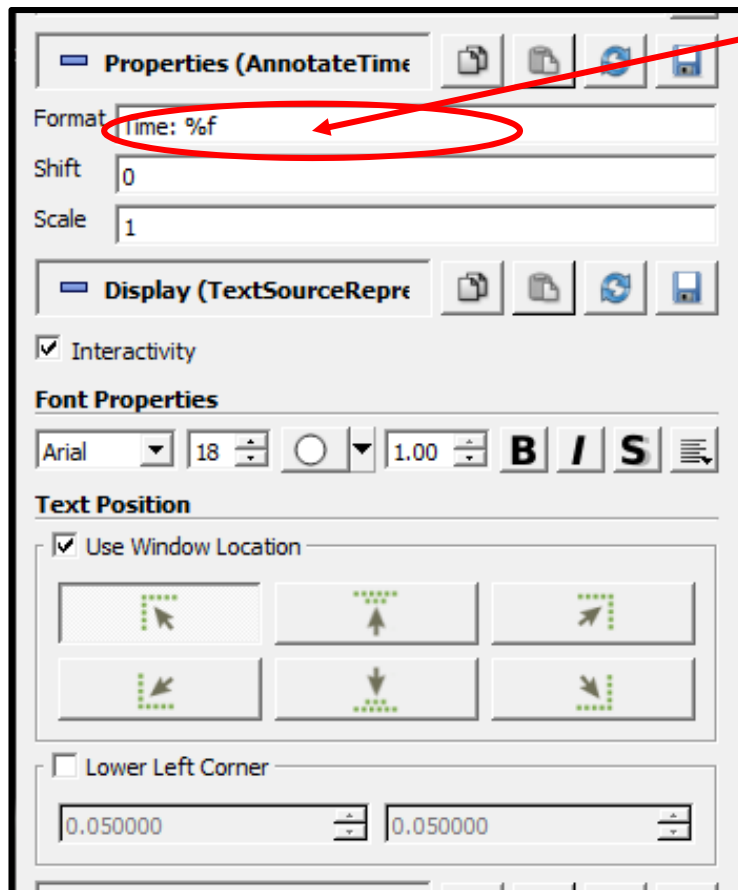
Annotating

Adding Titles

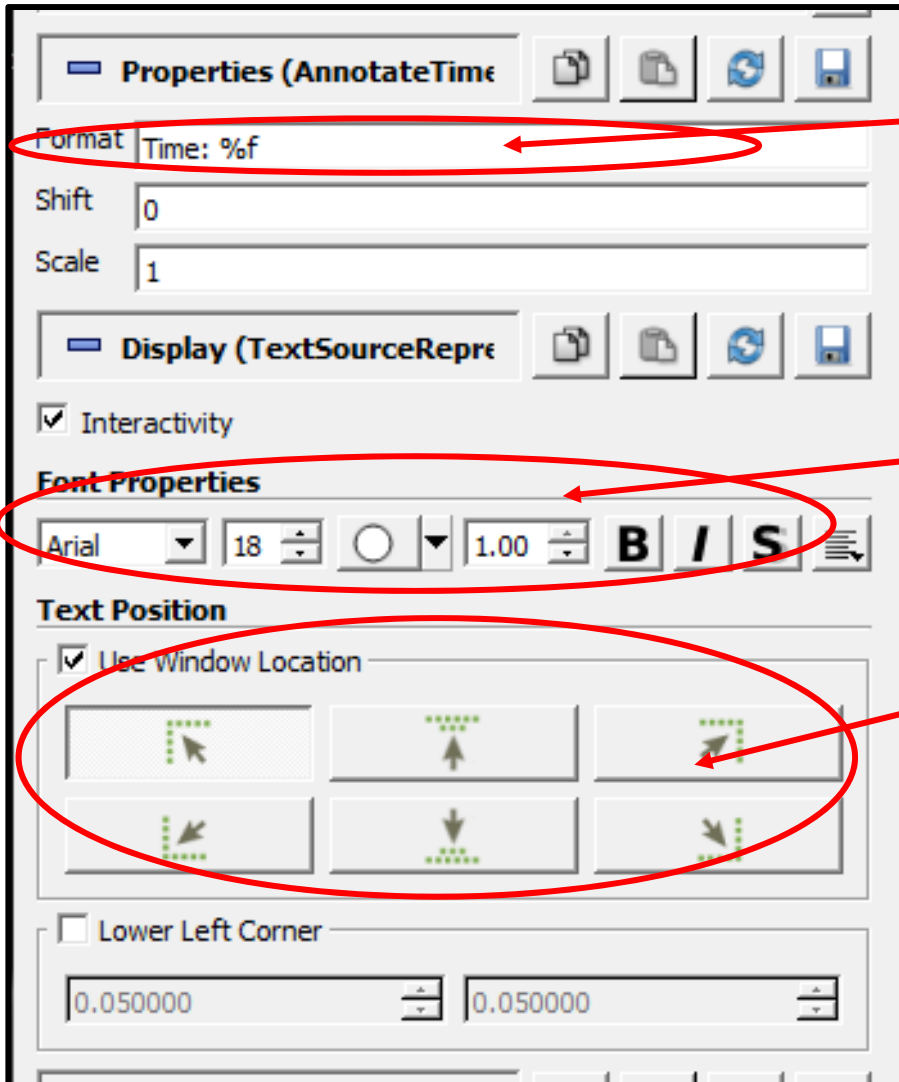


Add an **Annotate Time Filter** to the pipeline

The default annotation looks like this. We will change that.



Adding Titles

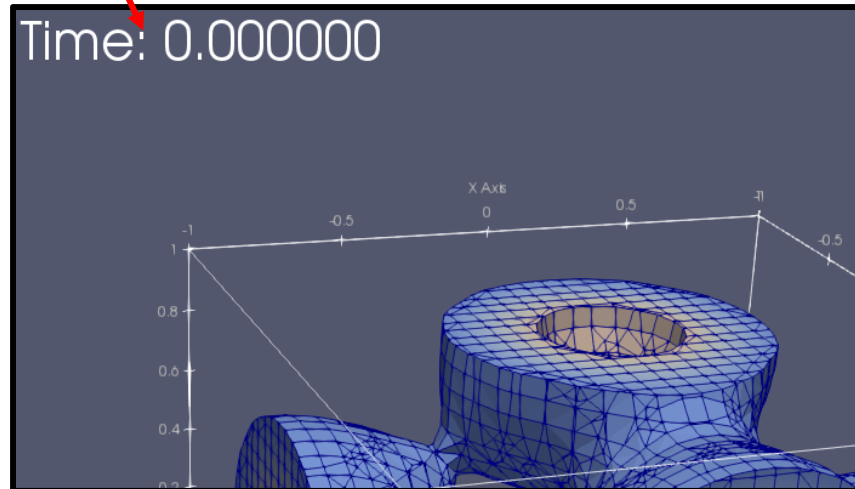


The label to use (the printf-notation is to format the Time – get rid of this if you just want a title)

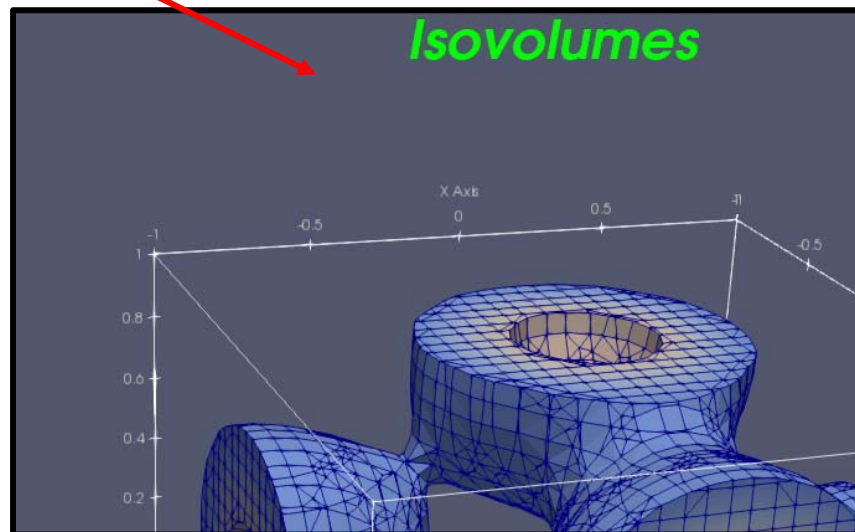
The font, size, color, opacity, style, and justification to use

The position for the title

From this:



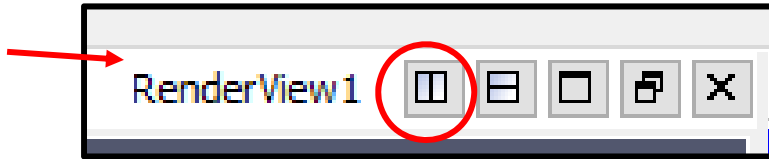
to this:



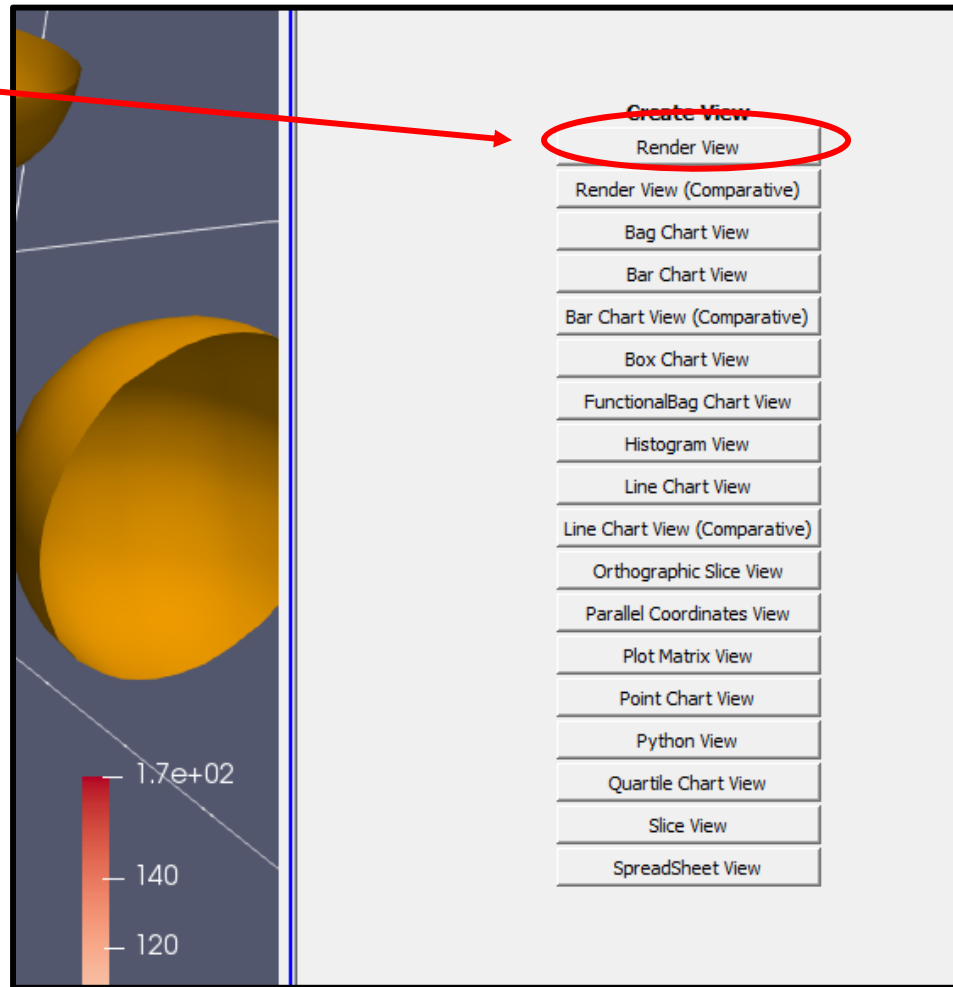
Multiple Views

Multiple Views

Step #1: Split the Window

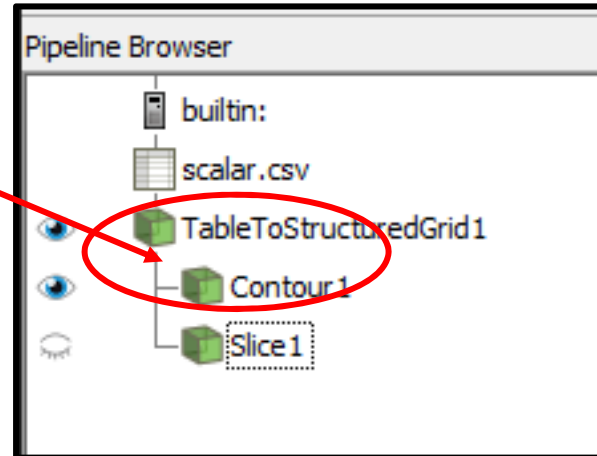


Step #2: Click on **Render View**

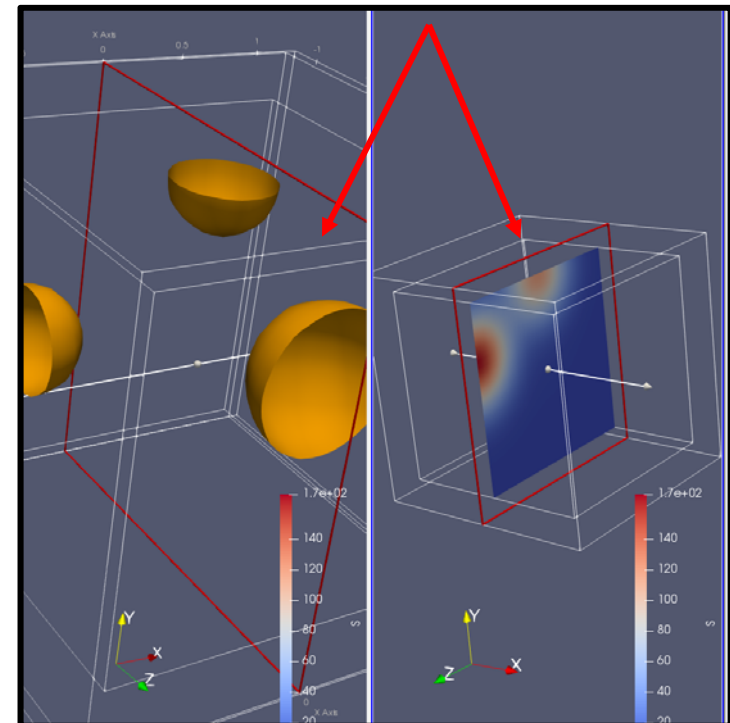


Multiple Views

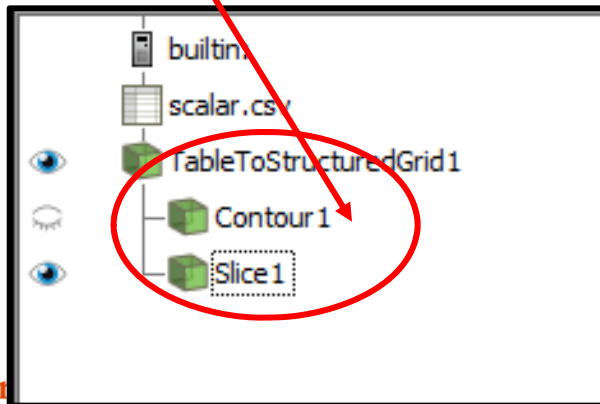
Step #3: Click in one Window and setup one visualization



... and, you get this – with each Window being allowed its own viewing transformation



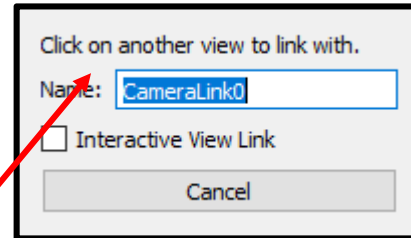
Step #4: Click in the other Window and setup a separate visualization (stay aware of how the visualizations are parented!)



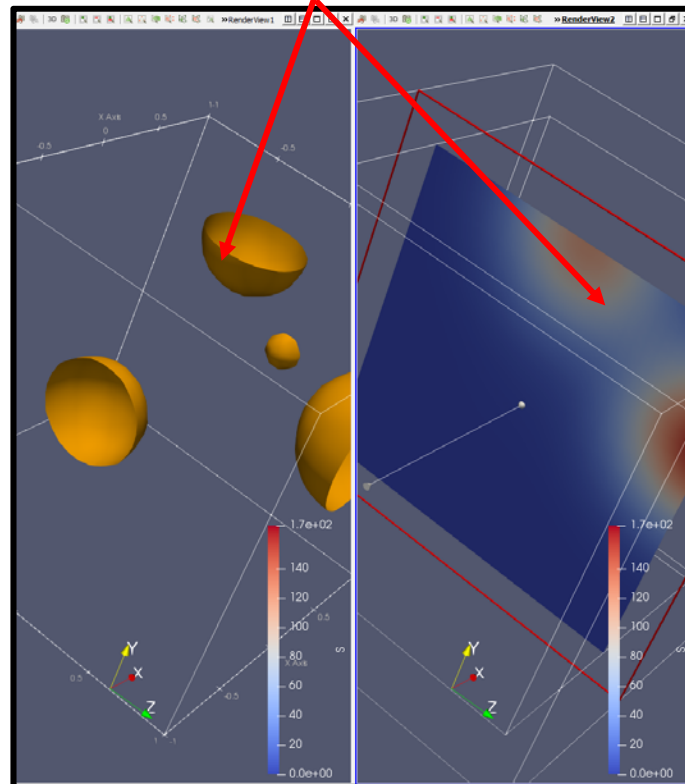
Multiple Views with Linked Viewing Transformations

Step #5: Right-click in one of the Windows and select **Link Camera...**

Step #6: You get this dialog box – now click in the other Window that you want to be linked with

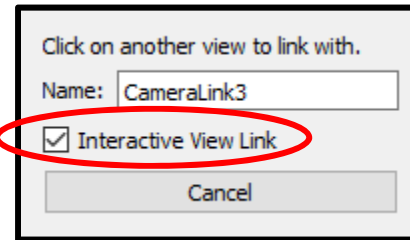


Your Windows now share a single transformation

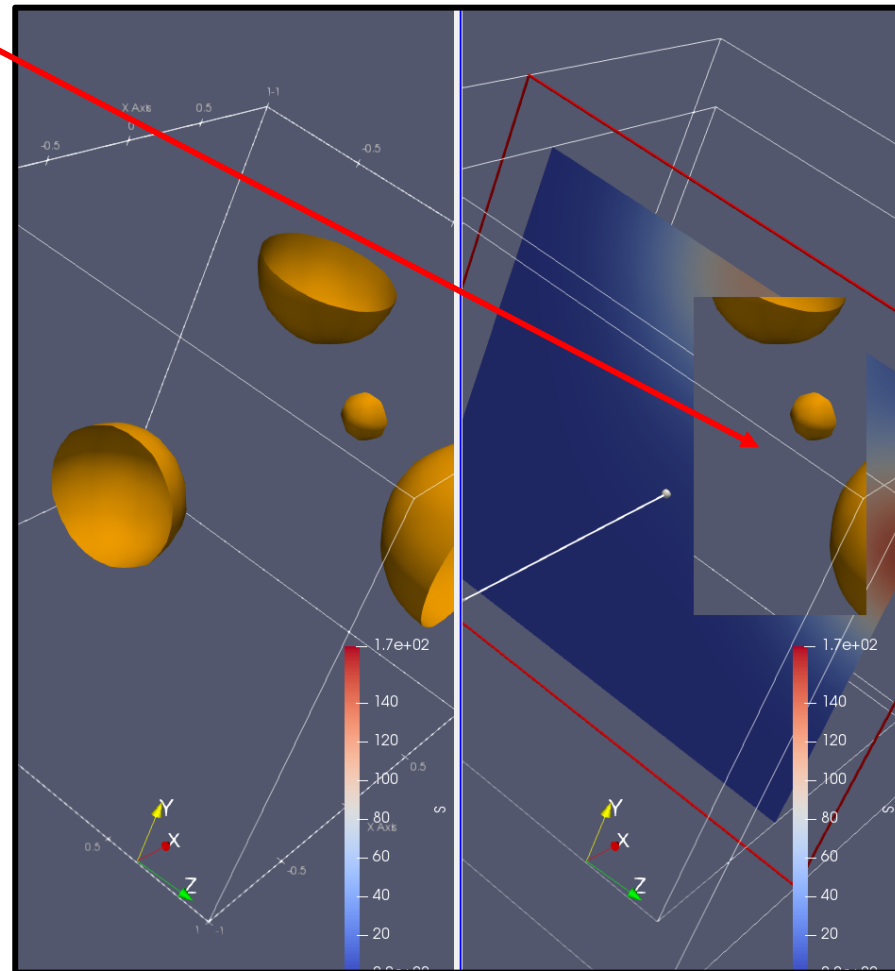


Multiple Views with Linked Viewing Transformations

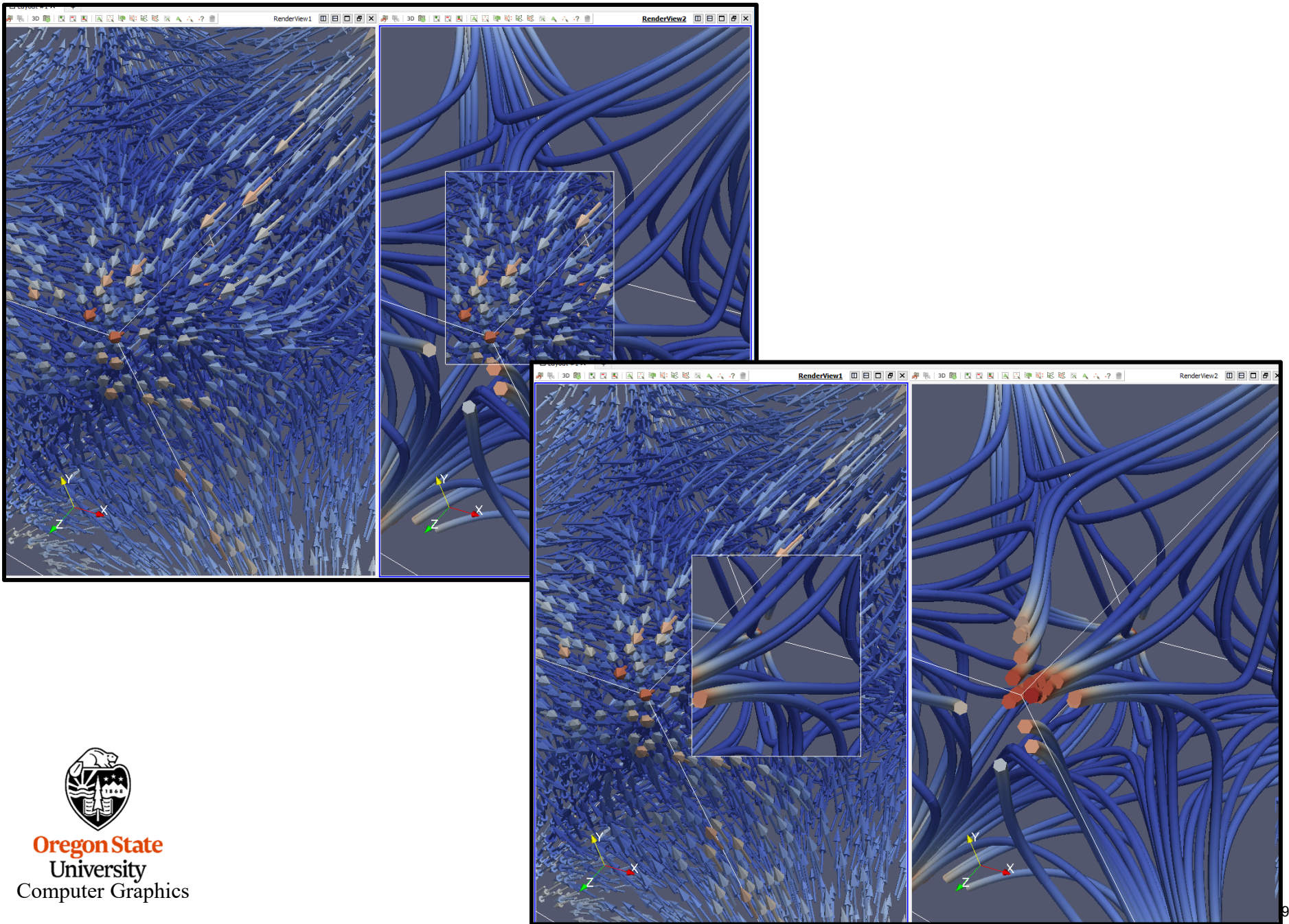
If you click on this checkbox and then click in another Window ...



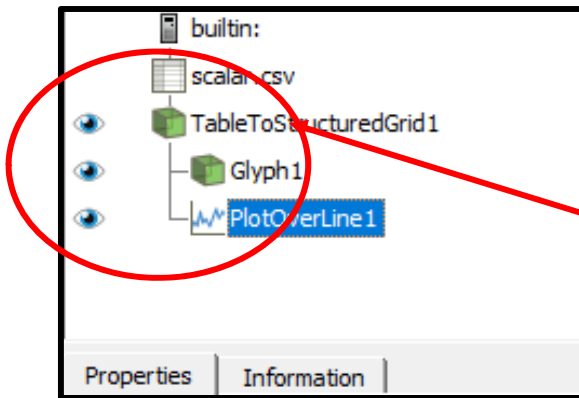
... you get a Magic Lens



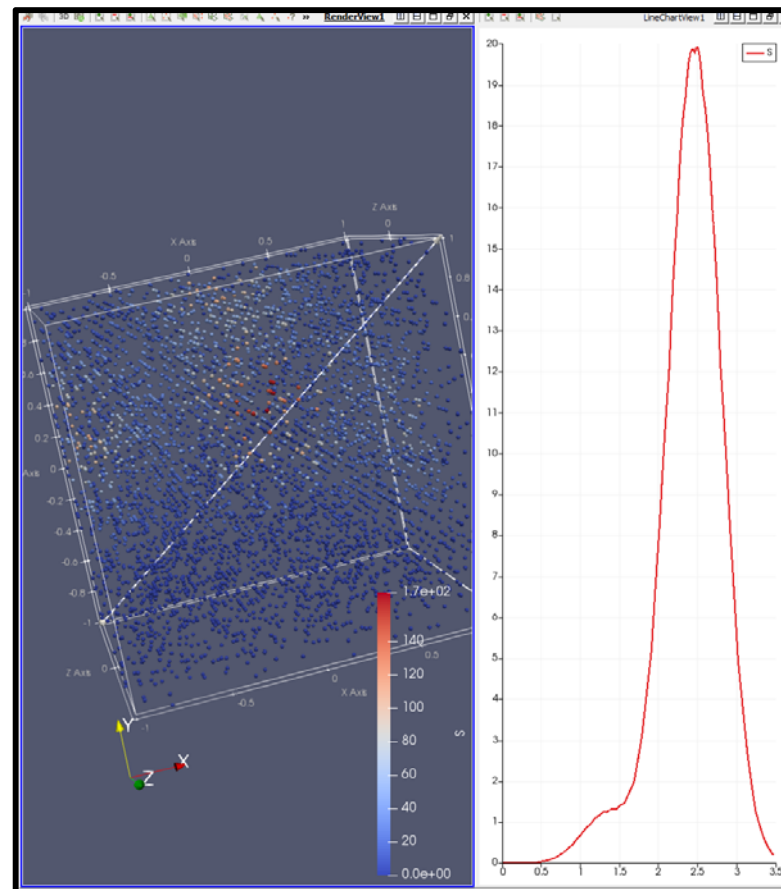
Order Matters!



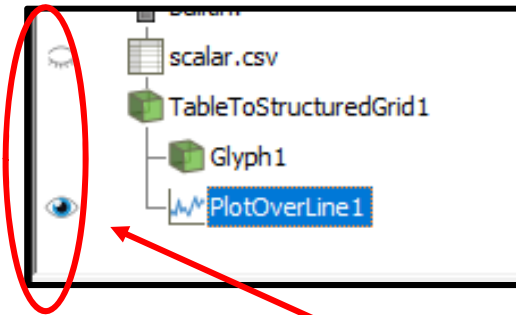
Using Plot Over Line



1. Create this Pipeline
2. Split the Render window and ask for a **Line Chart** window
3. When you click in the Render window, make the eyeballs look like this, with the **TableToStructuredGrid** representation set to **Outline** and the **Glyph** representation set to **Surface**



Using Plot Over Line



1. When you click in the **Line Chart display** window, make the eyeballs look like this
2. Setup the Properties like this
3. Be sure **Auto-Apply** is turned on

Search ... (use Esc to clear text)

Series Parameters

Variable	Legend Name
<input type="checkbox"/> Points_Mag...	Points_Magnitude
<input type="checkbox"/> Points_X	Points_X
<input checked="" type="checkbox"/> Points_Y	Points_Y
<input type="checkbox"/> Points_Z	Points_Z
<input checked="" type="checkbox"/> S	S
<input type="checkbox"/> arc_length	arc_length
<input type="checkbox"/> vtkValidPoin...	vtkValidPointMask

Line Thickness: 2
Line Style: Solid
Marker Style: None
Chart Axes: Bottom-Left

View (Line Chart View)

Title

Chart Title: Use \${TIME} to display current time

Annotation

Show Legend
 Sort By X-Axis

Left Axis

Left Axis Title: [Empty]

Left Axis Range

Left Axis Log Scale
 Left Axis Use Custom Range
Left Axis Range Minimum: 0
Left Axis Range Maximum: 20

Bottom Axis

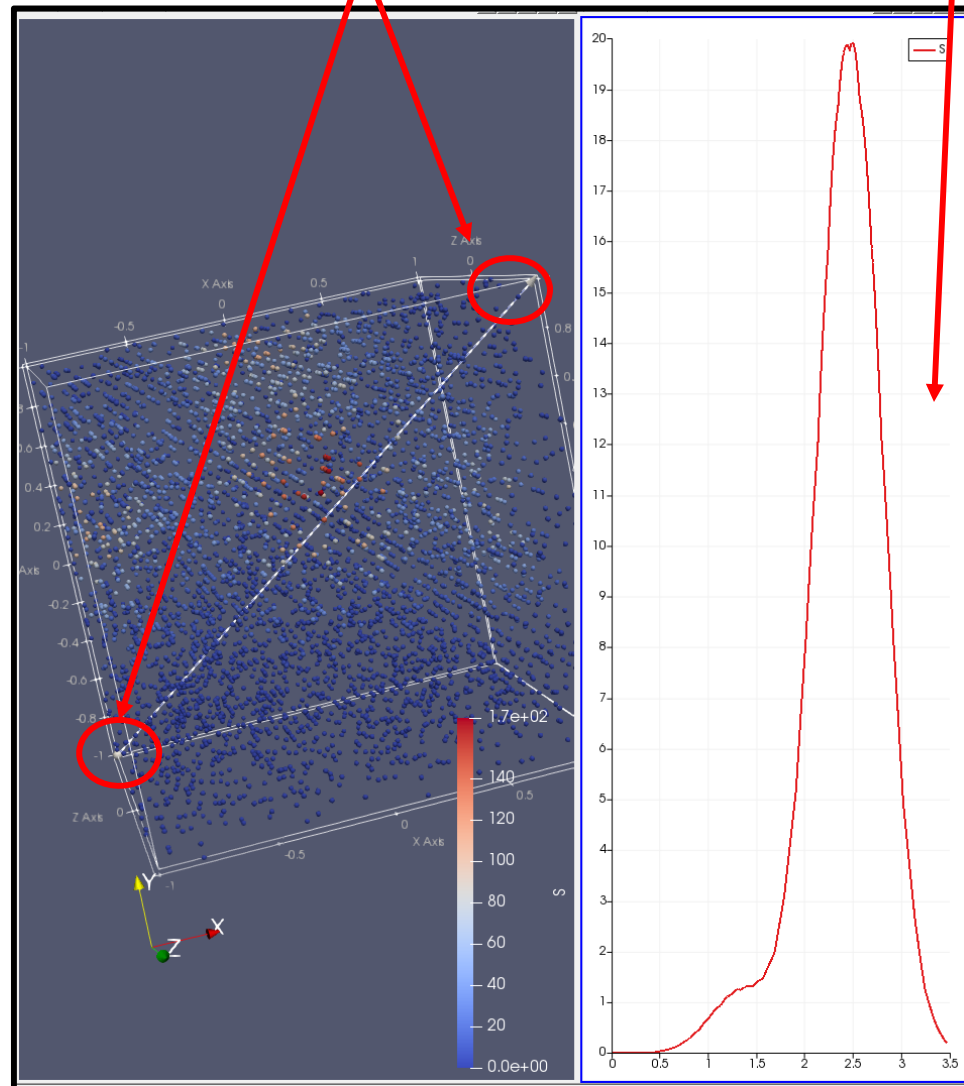
Bottom Axis Title: [Empty]

Bottom Axis Range

Bottom Axis Log Scale
 Bottom Axis Use Custom Range
Bottom Axis Range Minimum: 0
Bottom Axis Range Maximum: 3.5

Using Plot Over Line

Now, when you click on the **Line** endpoints and move them, the graph changes



Comparative Visualization

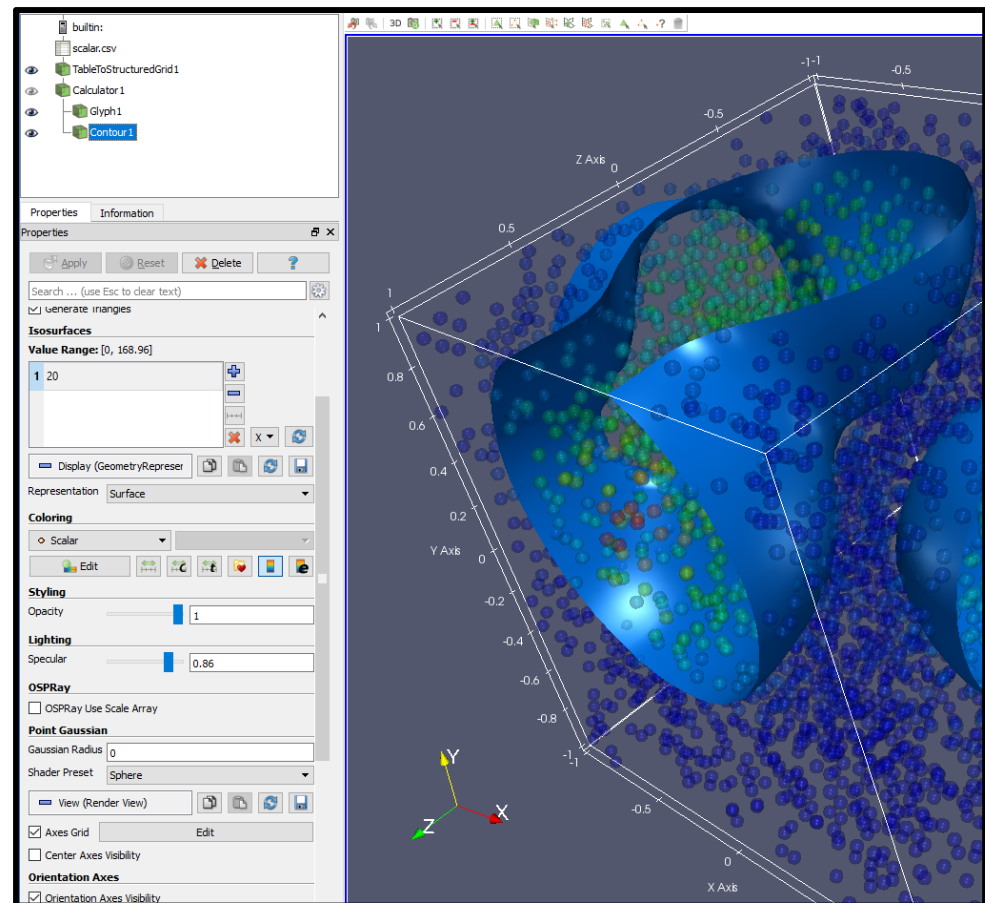


scalarcompare.pvsm

Comparative Visualization

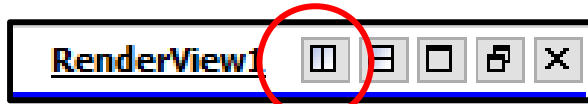
ParaView can setup a side-by-side visualization comparison with different vis parameters in each view.

Start by creating a 3D Render view visualization. This case is using the isosurface demonstration shown earlier.

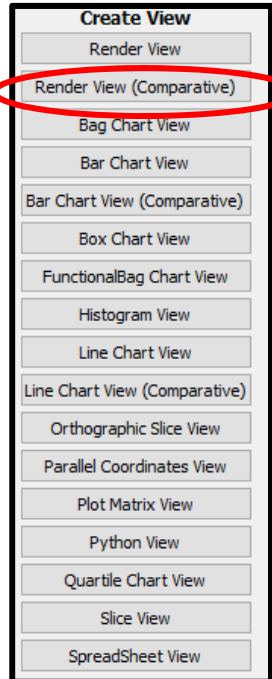


Comparative Visualization

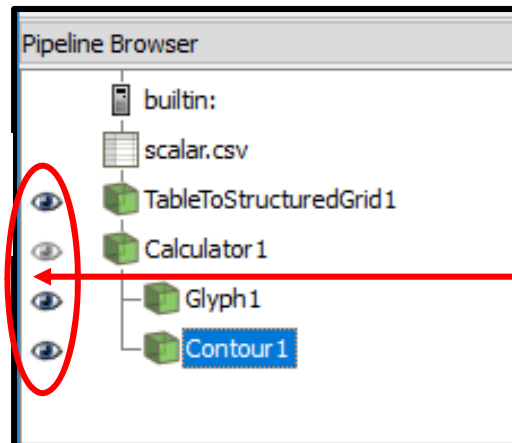
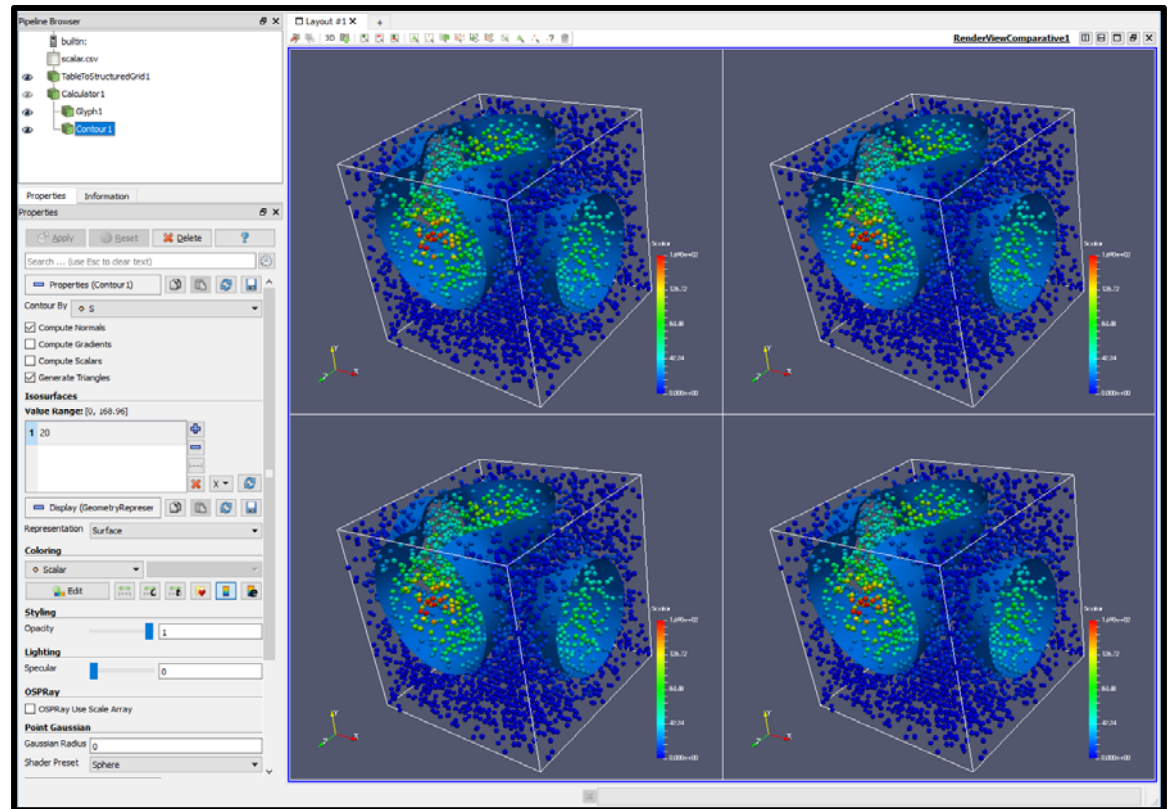
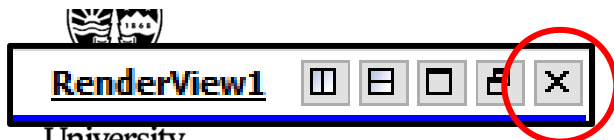
Now, split the window



and select:
**Render View
(Comparative).**

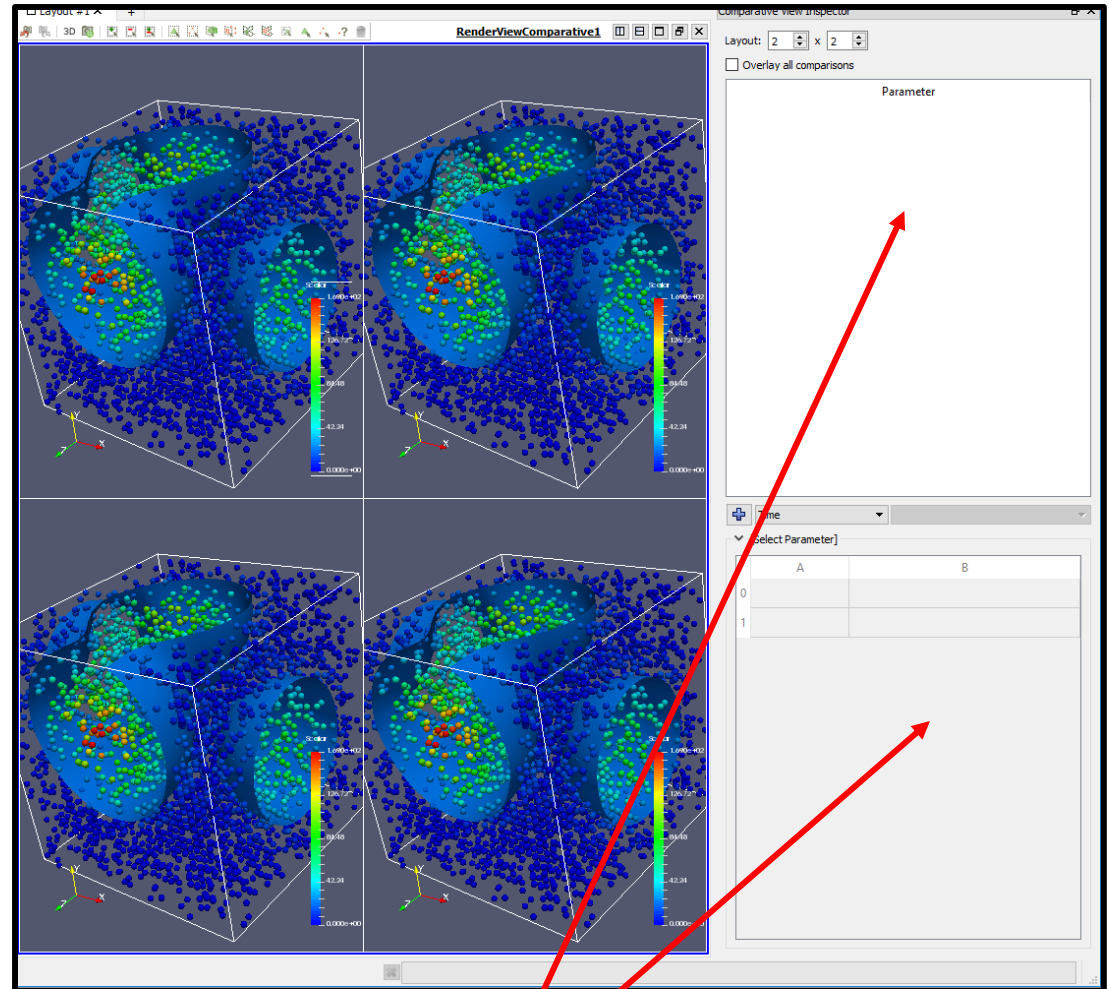
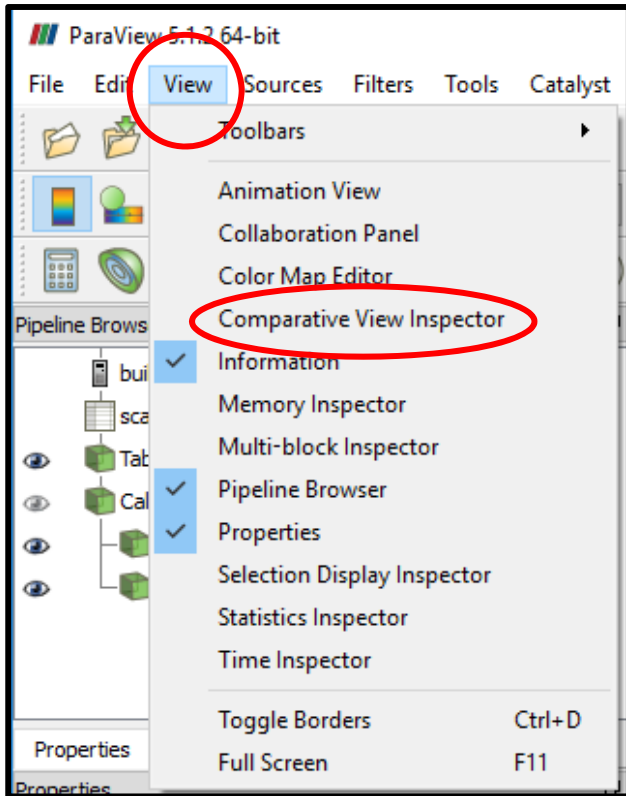


You can now eliminate the left-hand window if you want.



Click all the eyeballs on for the visualization features you want to see.

Comparative Visualization

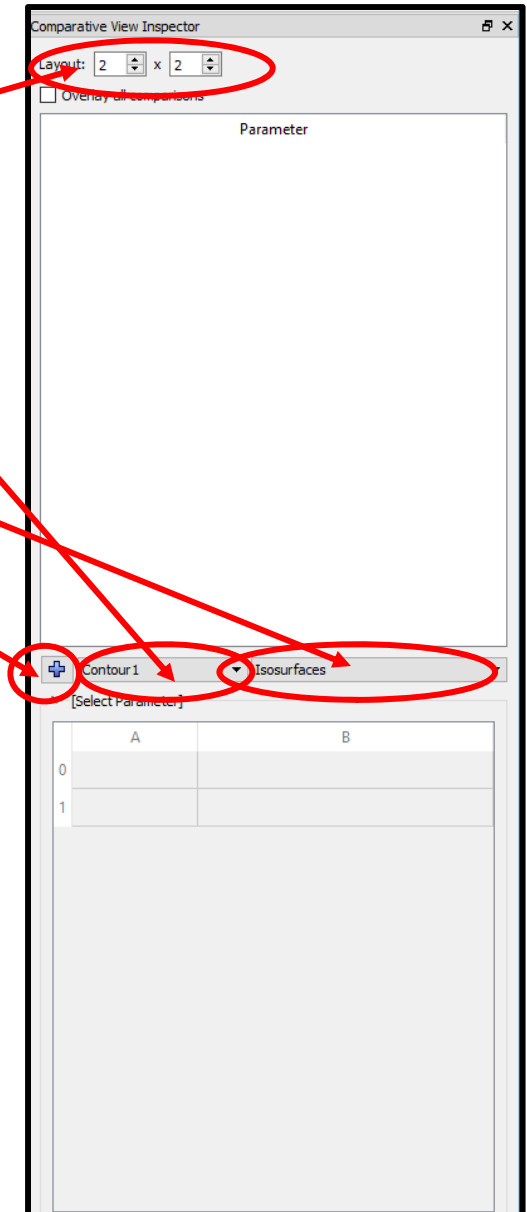
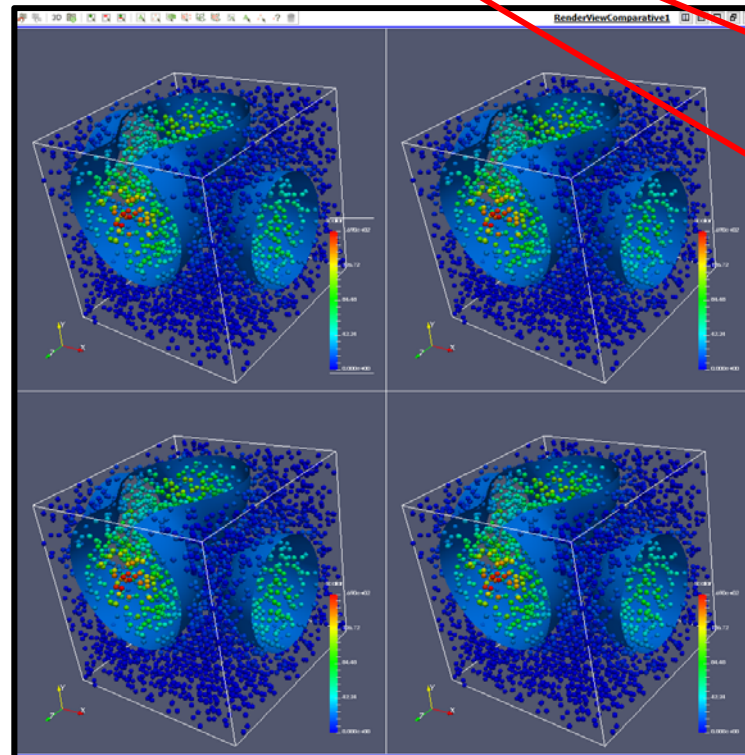
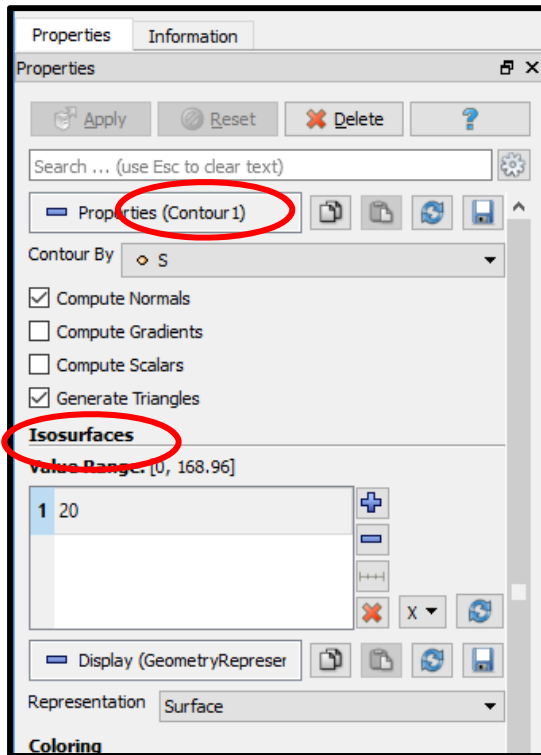


Select **View** → **Comparative View Inspector**

Comparative Visualization

Here's where you get to select how to vary the parameter(s).

1. Select the layout dimensions of the comparative window grid
2. Select the pipeline module that owns the parameter
3. Select the parameter
4. Hit the Big Plus Sign



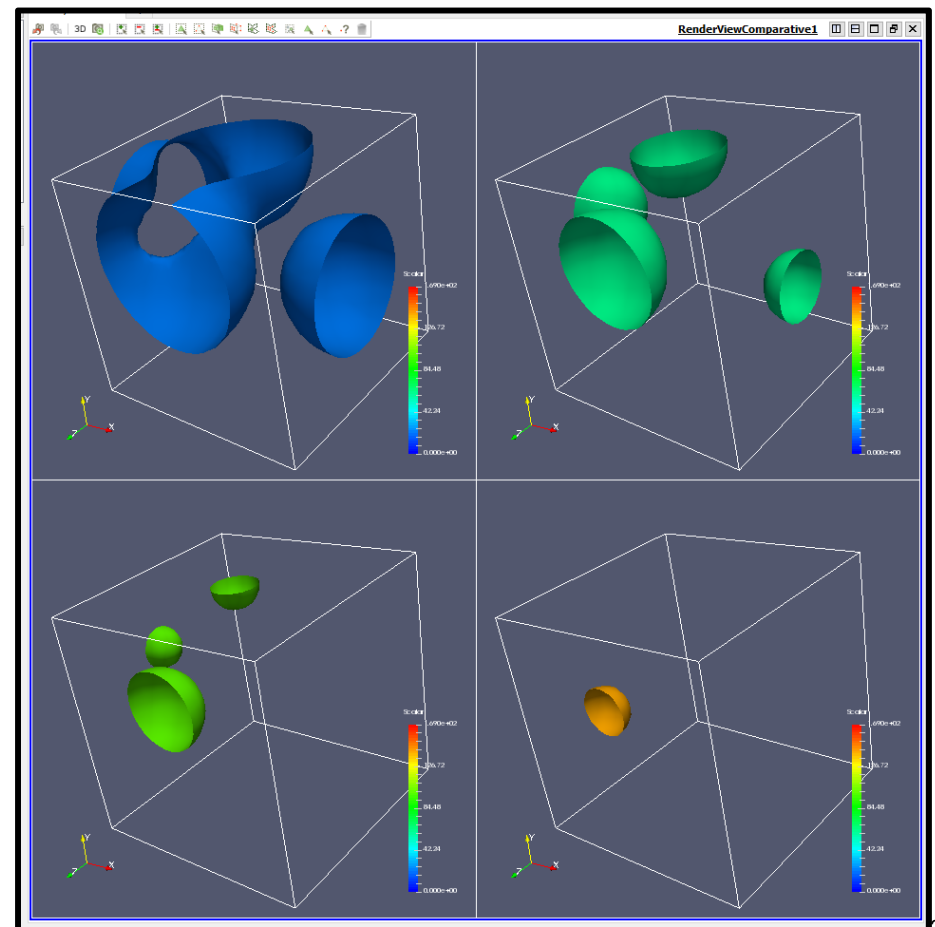
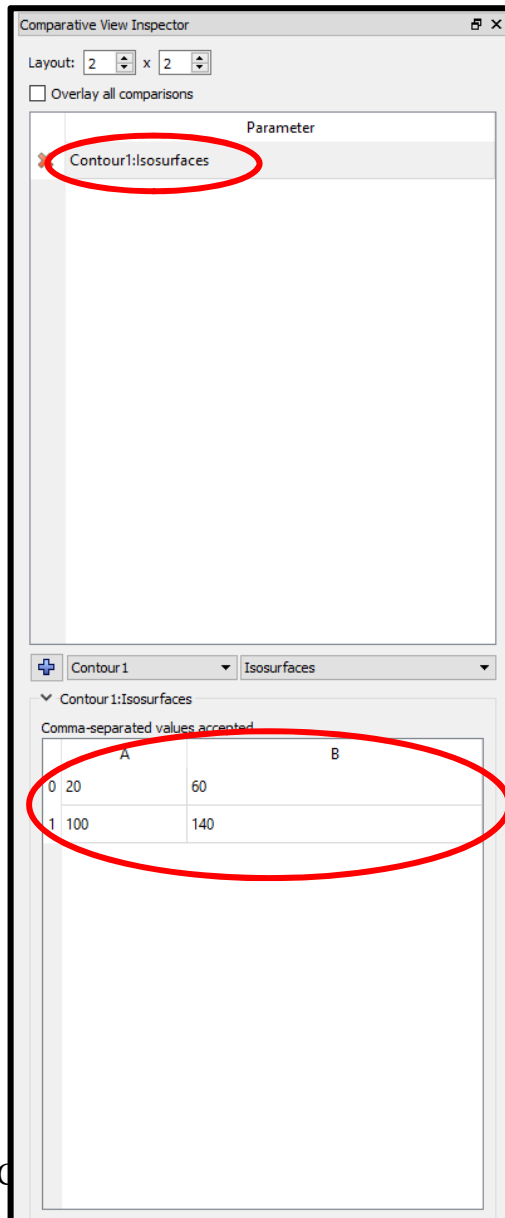
Comparative Visualization

ParaView stocks the number grid with evenly-spaced values and applies them to each visualization, respectively.

You can type your own numbers in each cell

(I eliminated the Glyphs to better see the isosurfaces)

The windows are all transform-linked

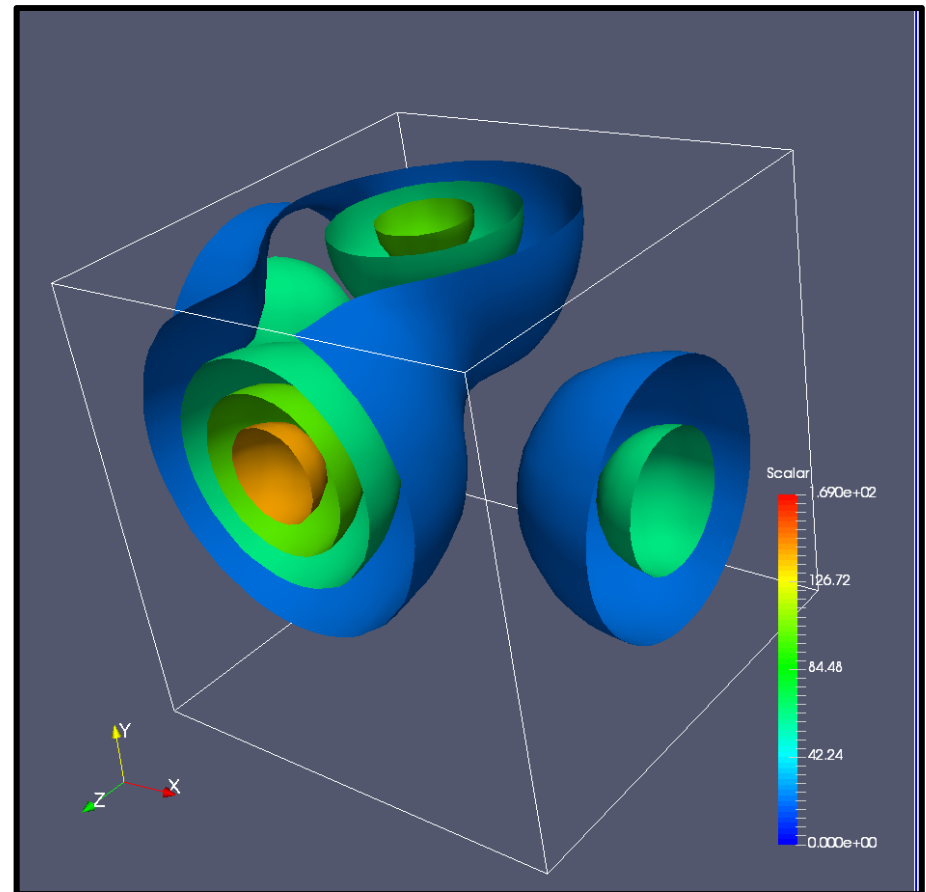
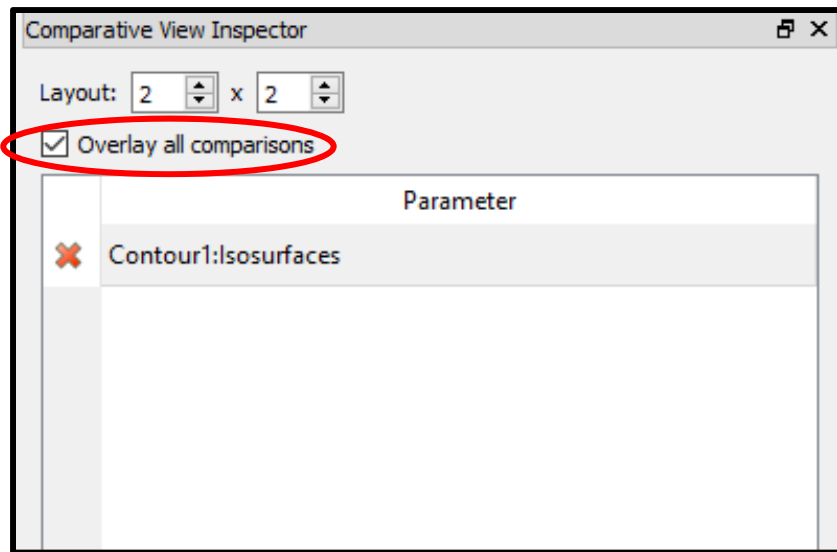


Comparative Visualization

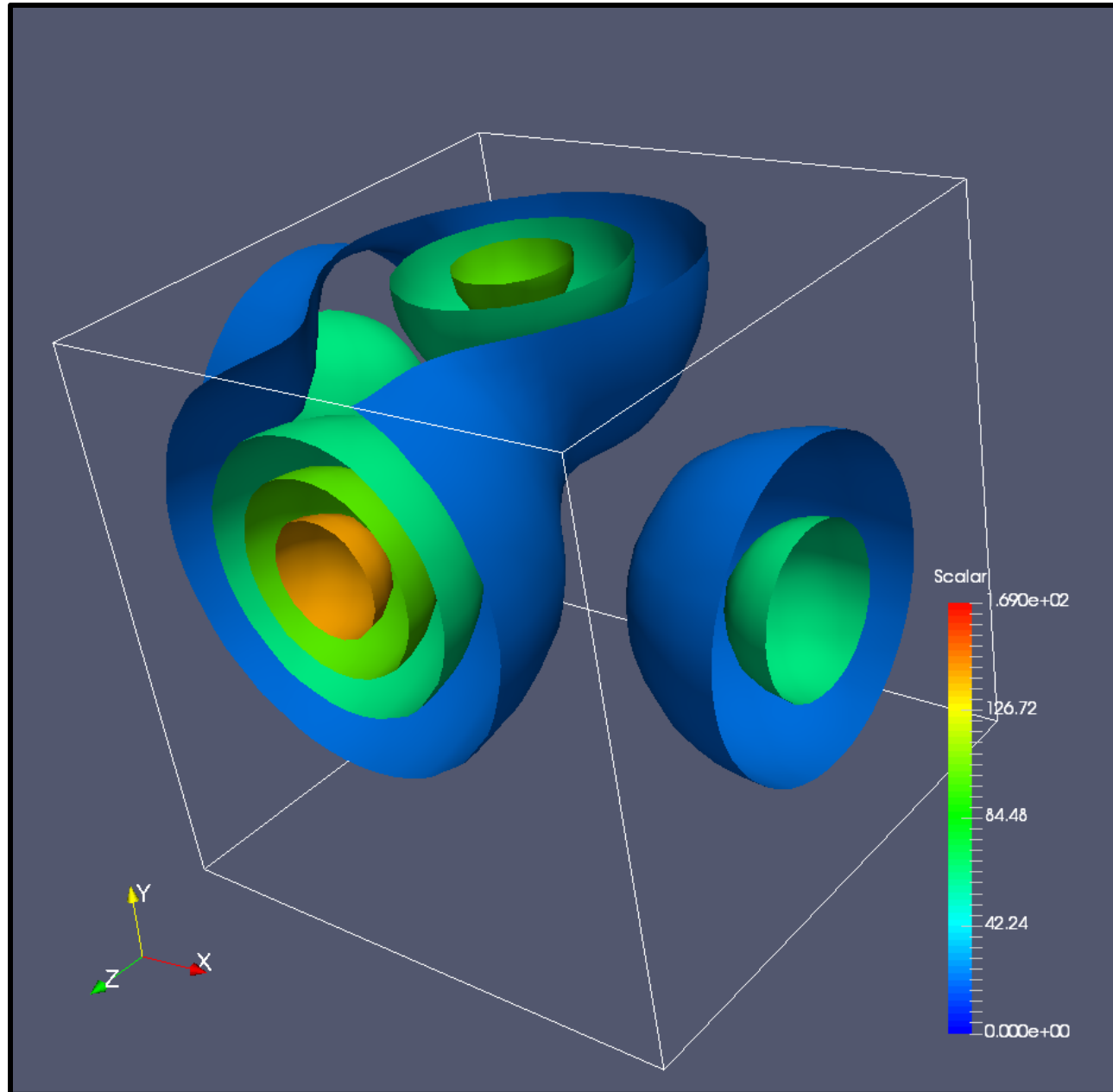
Clicking **Overlay all comparisons**, well, overlays all comparisons

You can vary multiple parameters – just setup multiple pipeline elements and parameters and put numbers separated by commas in the cells

In this case, now could be a good time to also vary the opacity of the isosurfaces



Comparative Visualization



Visualizing Vector Data



vector.csv

Creating Vector Data in a CSV File

```
X32, Y32, Z32, Vx, Vy, Vz
-1.00, -1.00, -1.00, 2.00, 2.00, 2.00
-1.00, -1.00, -0.94, 1.75, 1.75, 2.00
-1.00, -1.00, -0.87, 1.53, 1.53, 2.00
-1.00, -1.00, -0.81, 1.33, 1.33, 2.00
-1.00, -1.00, -0.74, 1.15, 1.15, 2.00
-1.00, -1.00, -0.68, 0.99, 0.99, 2.00
-1.00, -1.00, -0.61, 0.84, 0.84, 2.00
-1.00, -1.00, -0.55, 0.71, 0.71, 2.00
-1.00, -1.00, -0.48, 0.60, 0.60, 2.00
-1.00, -1.00, -0.42, 0.49, 0.49, 2.00
-1.00, -1.00, -0.35, 0.40, 0.40, 2.00
-1.00, -1.00, -0.29, 0.31, 0.31, 2.00
-1.00, -1.00, -0.23, 0.24, 0.24, 2.00
-1.00, -1.00, -0.16, 0.17, 0.17, 2.00
-1.00, -1.00, -0.10, 0.10, 0.10, 2.00
-1.00, -1.00, -0.03, 0.03, 0.03, 2.00
```

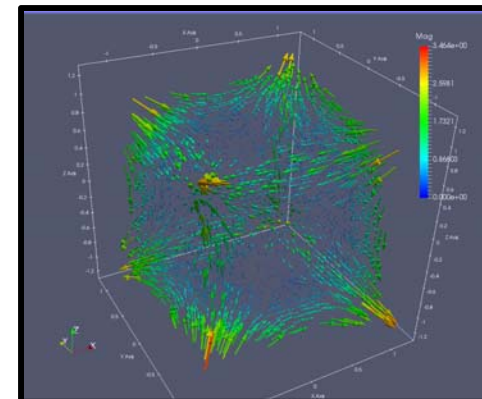
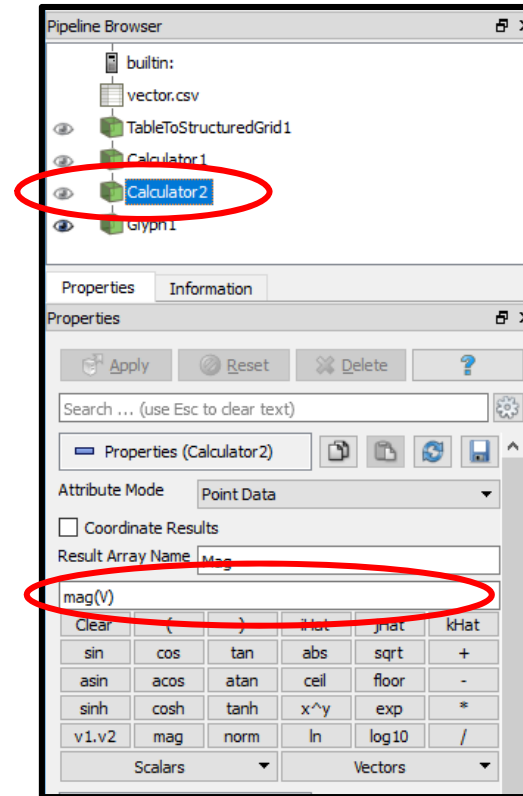
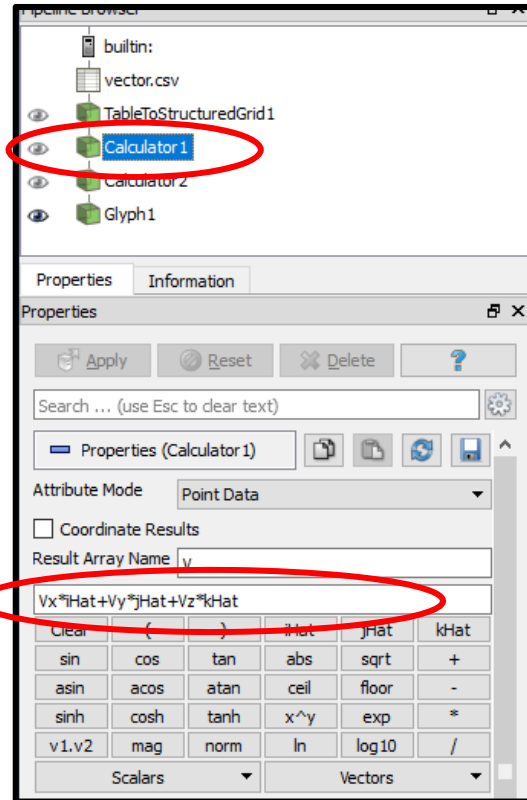
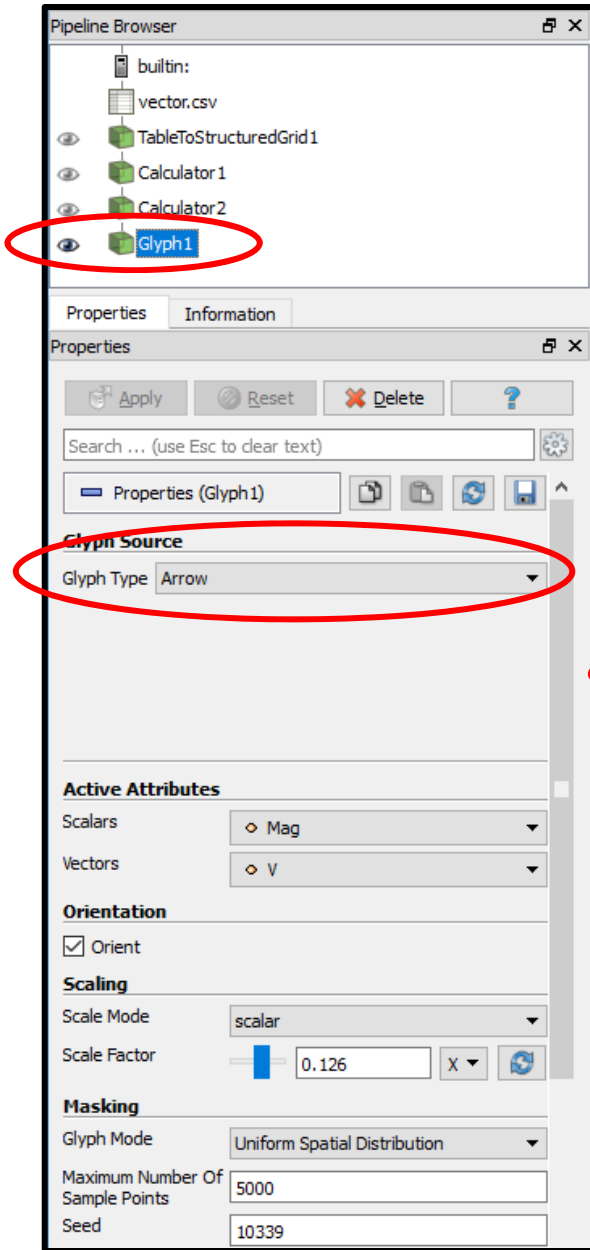
Do a **File** → **Open** and navigate to your CSV file.

Hit the **Apply** button to actually do the read.

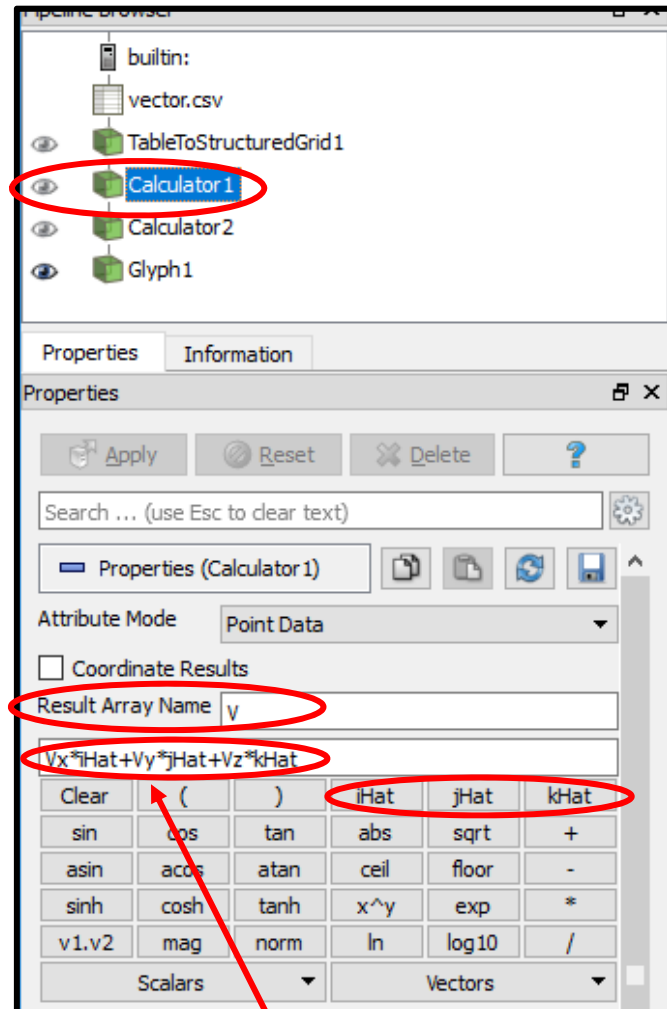
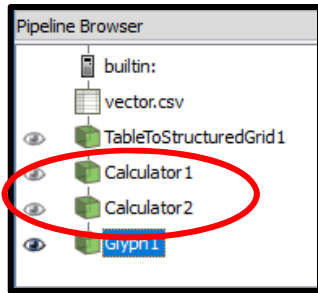


vector.csv

As Glyphs

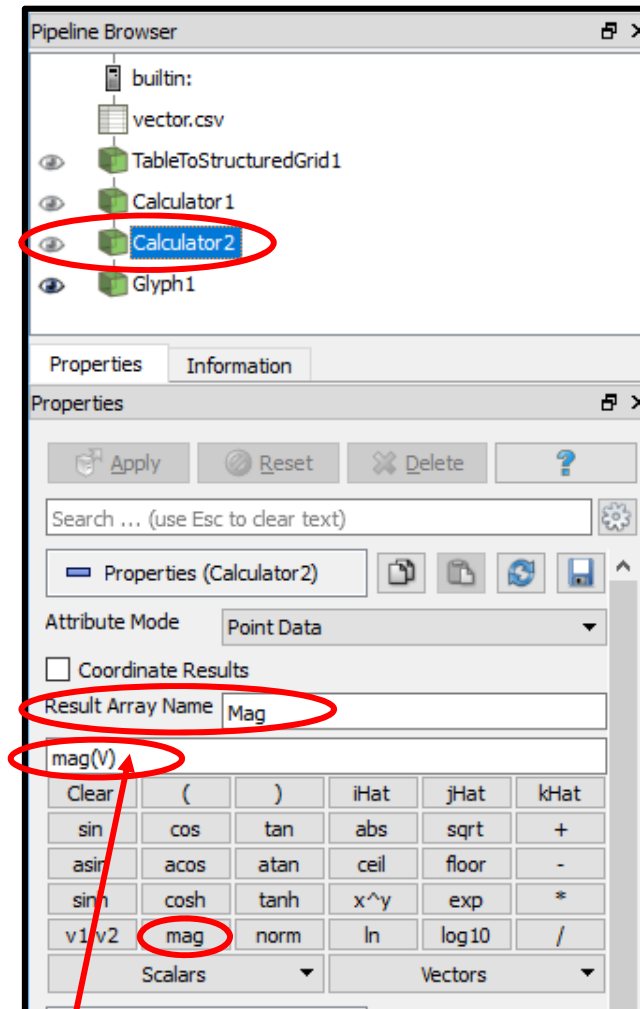


Why Are the Two Calculator Filters There?



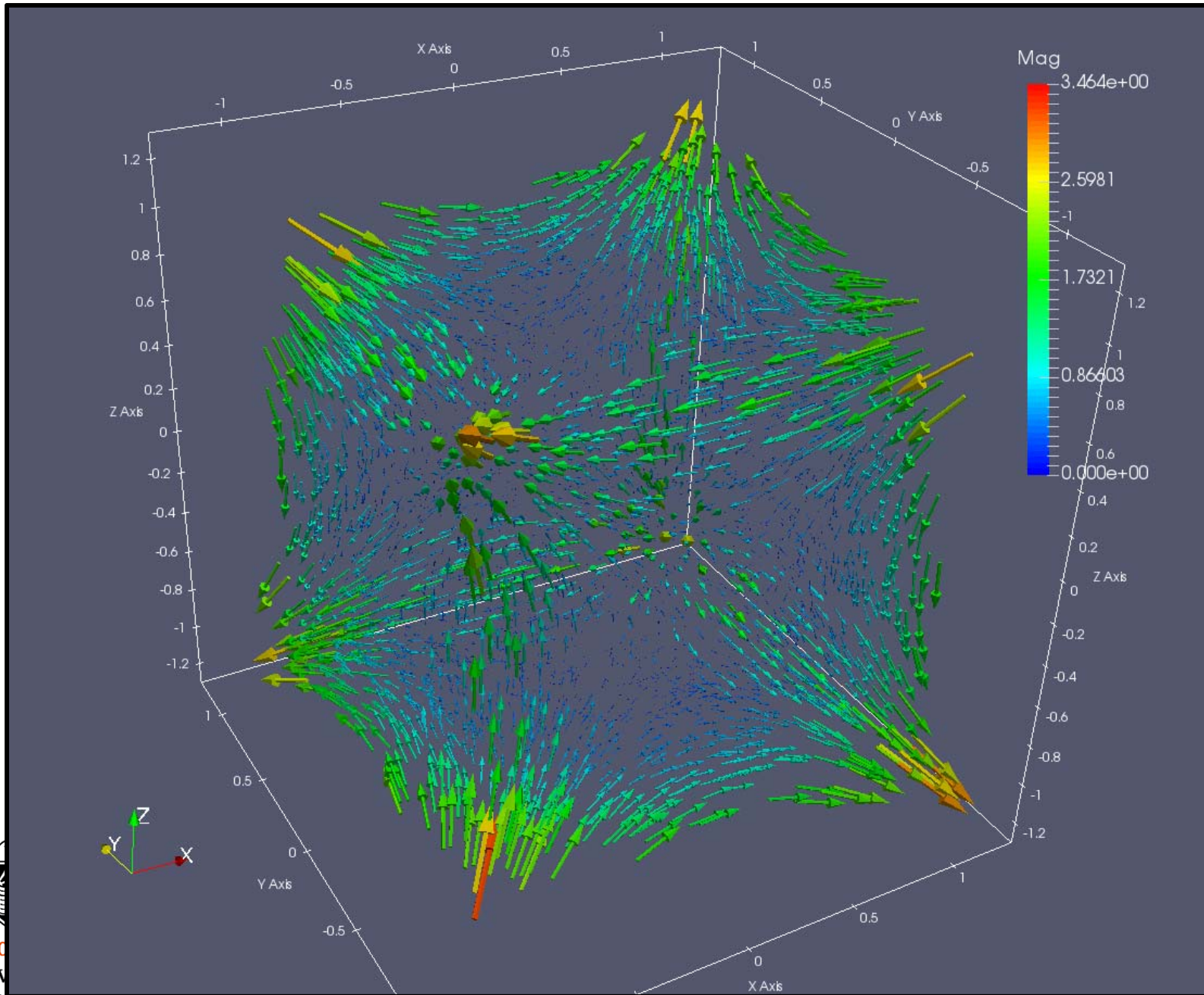
The **vector.csv** file brought in the three vector components **V_x**, **V_y**, and **V_z**. ParaView's vector vis filters want a 3-element vector instead. **Calculator1** is used to create that 3-element vector using the **iHat**, **jHat**, and **kHat** buttons (unit vectors in x, y, and z):

$$V = V_x \hat{i} + V_y \hat{j} + V_z \hat{k}$$

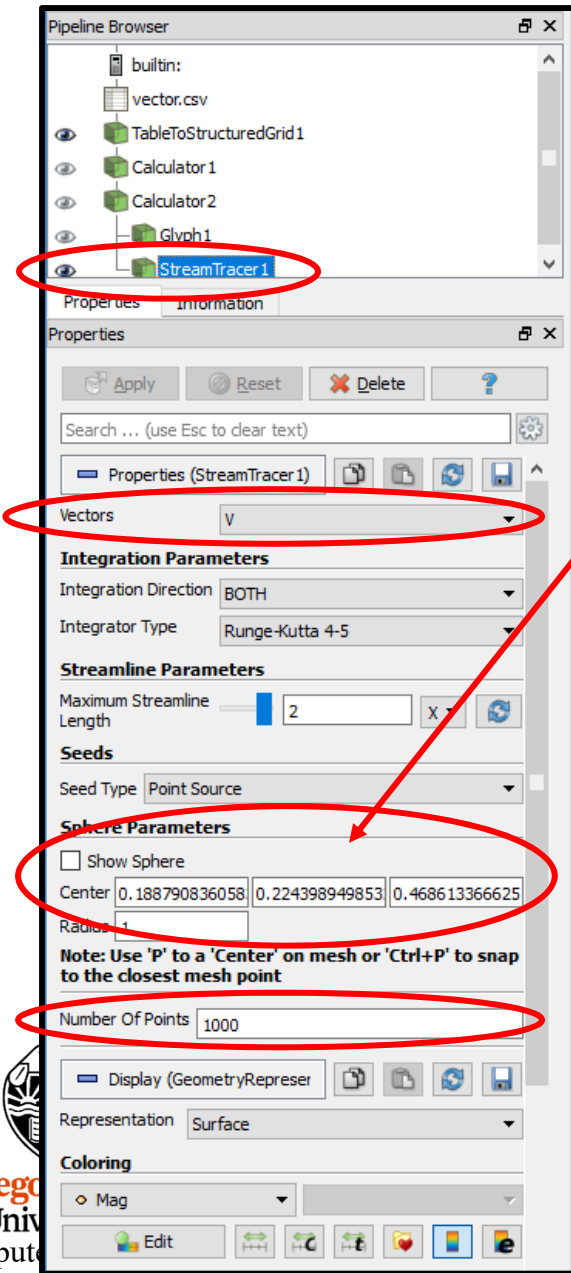


We want to color the vector visualizations by the magnitude of the vector. **Calculator2** computes that magnitude using the **mag** button:

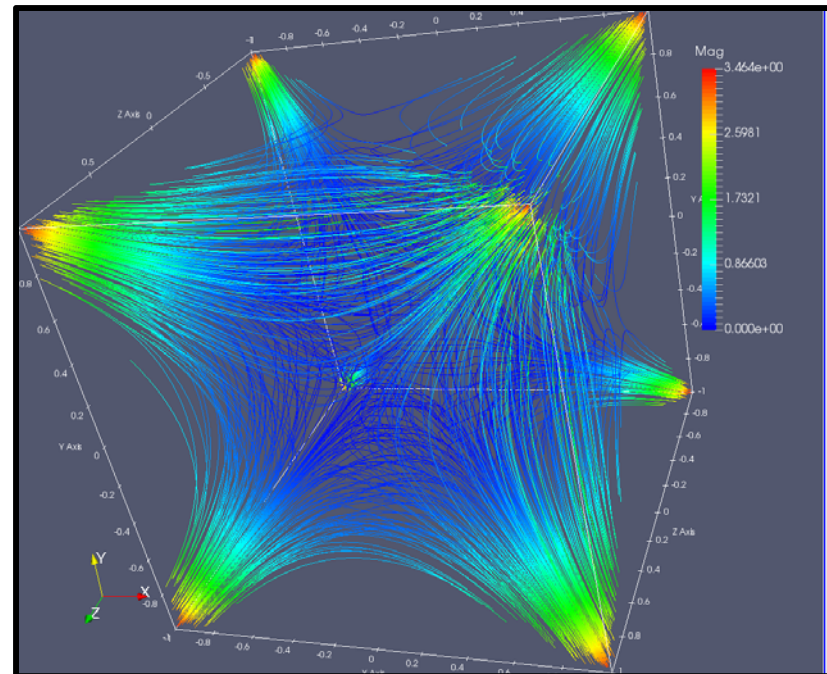
$$Mag = \|V\|$$

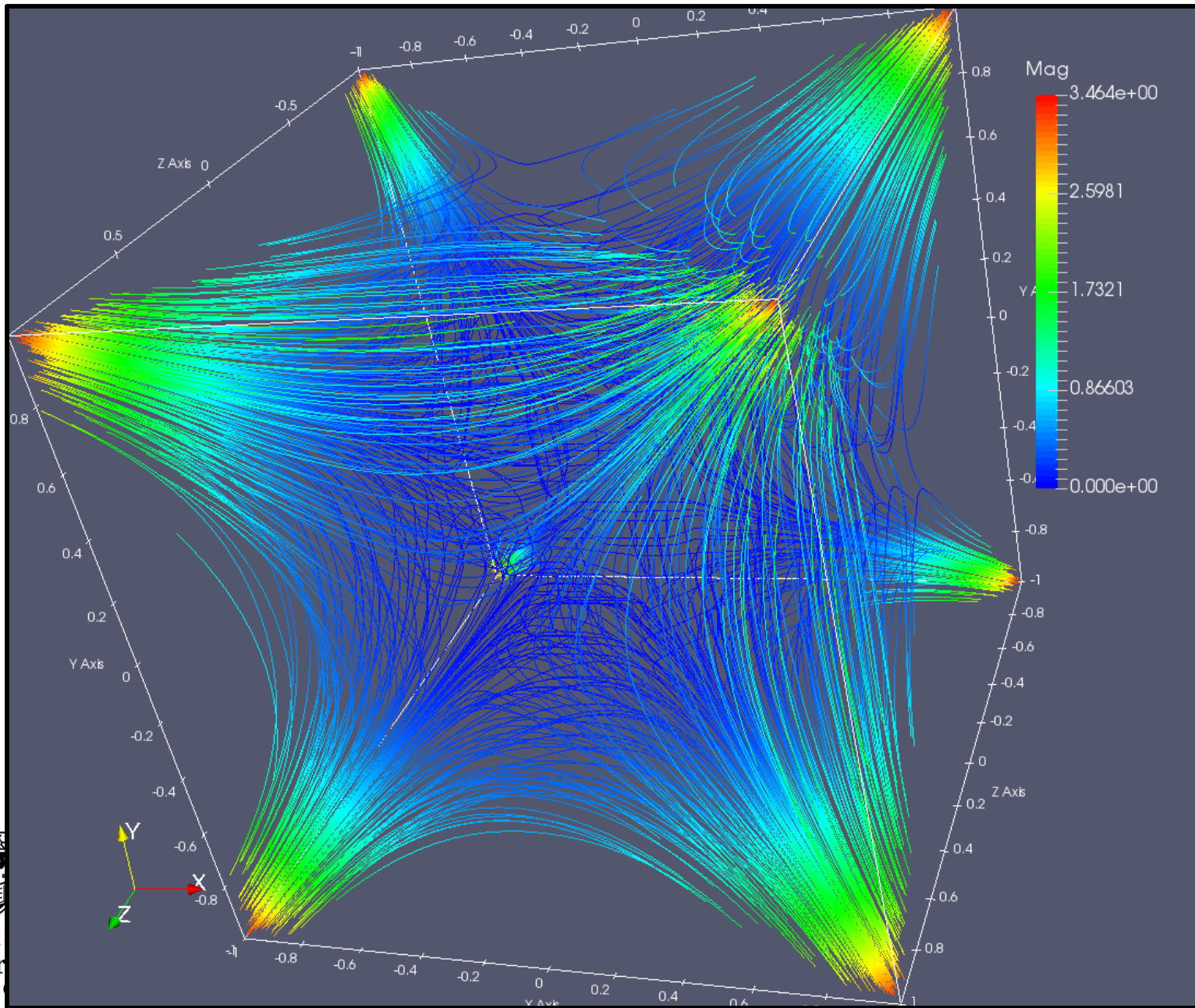


As Streamlines

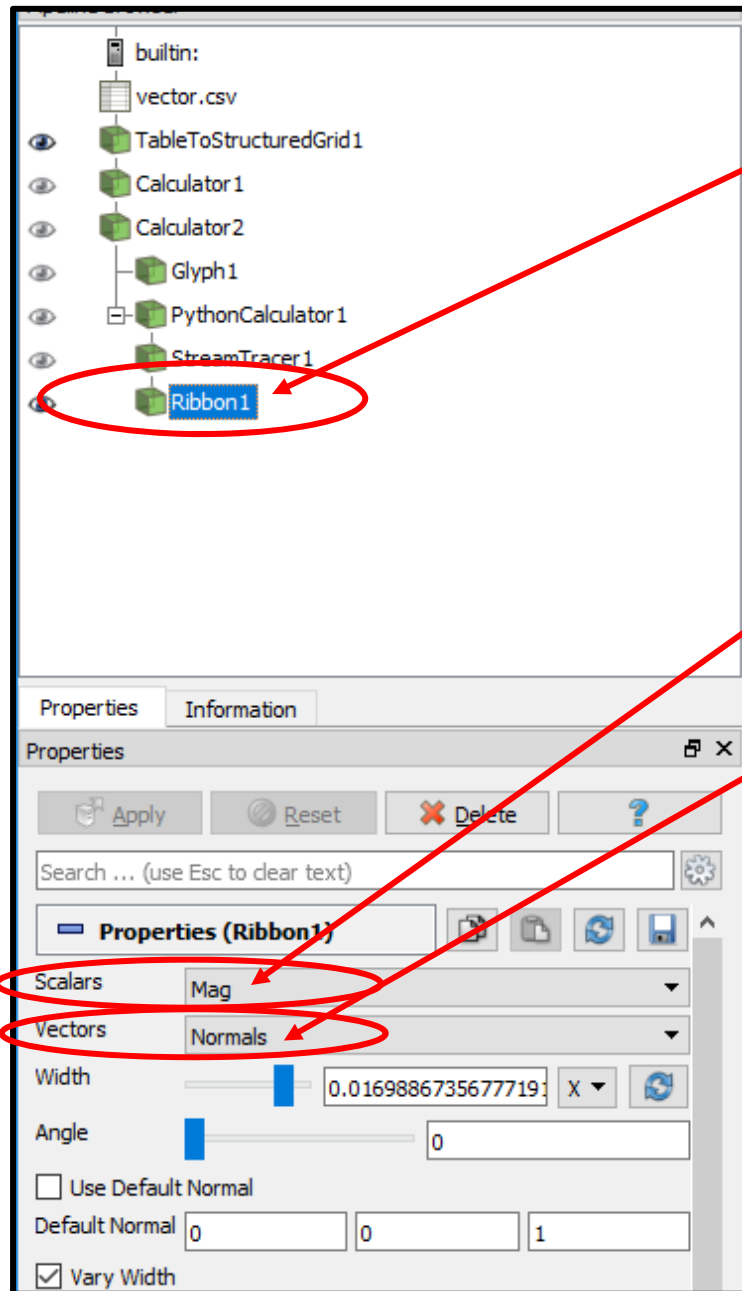


Will start the streamlines from within this sphere. You can move it and resize it.





As Ribbon Traces

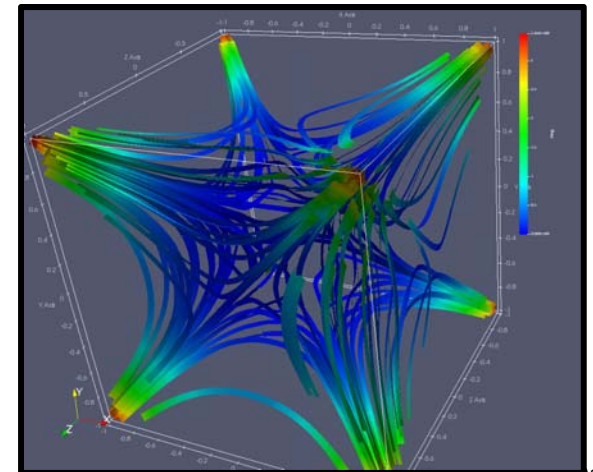


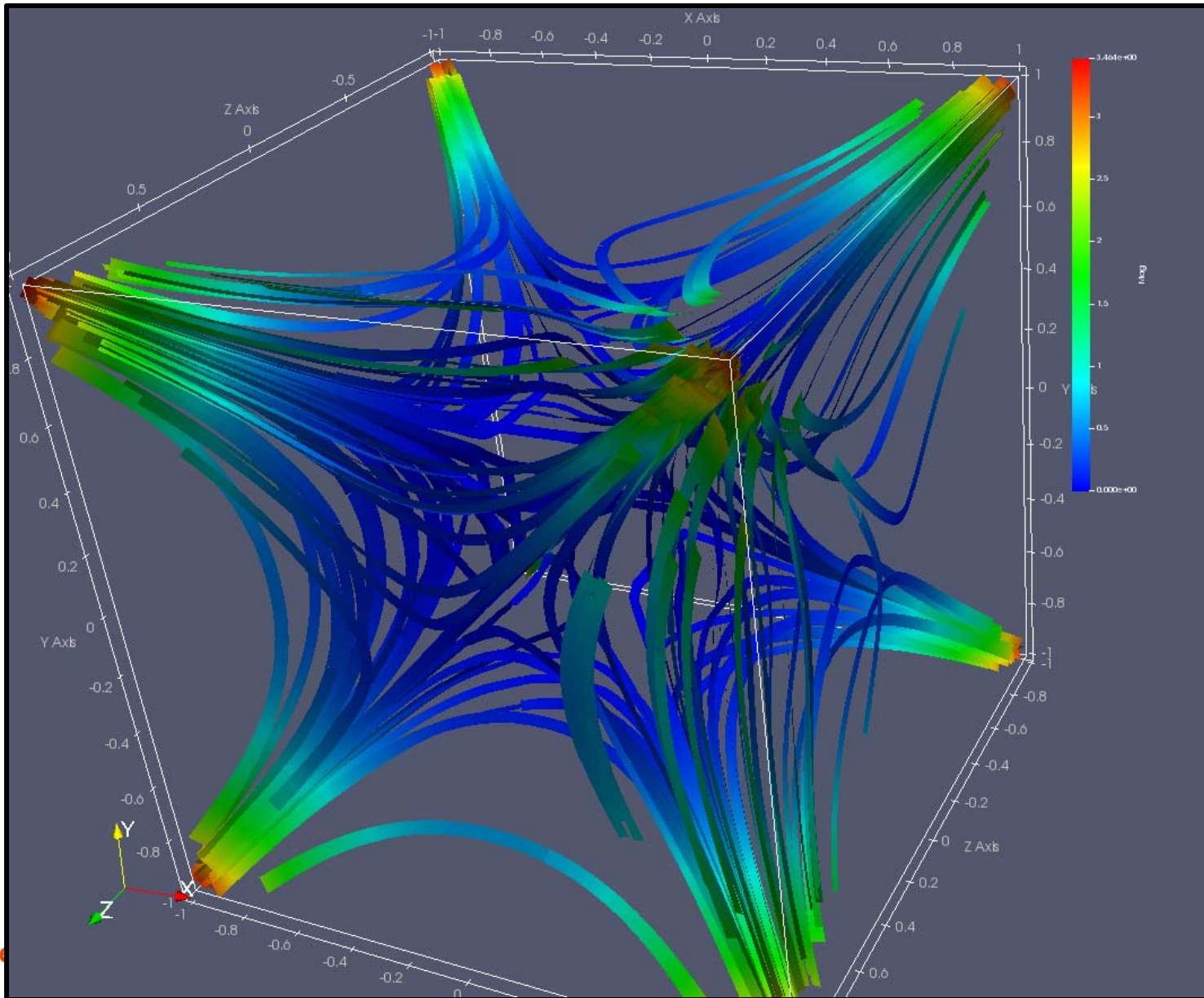
Note – **Ribbon** is parented from **StreamTracer**.

Ribbon Traces are especially good for showing *twisting* in the vector field. This dataset is not a great example of that.

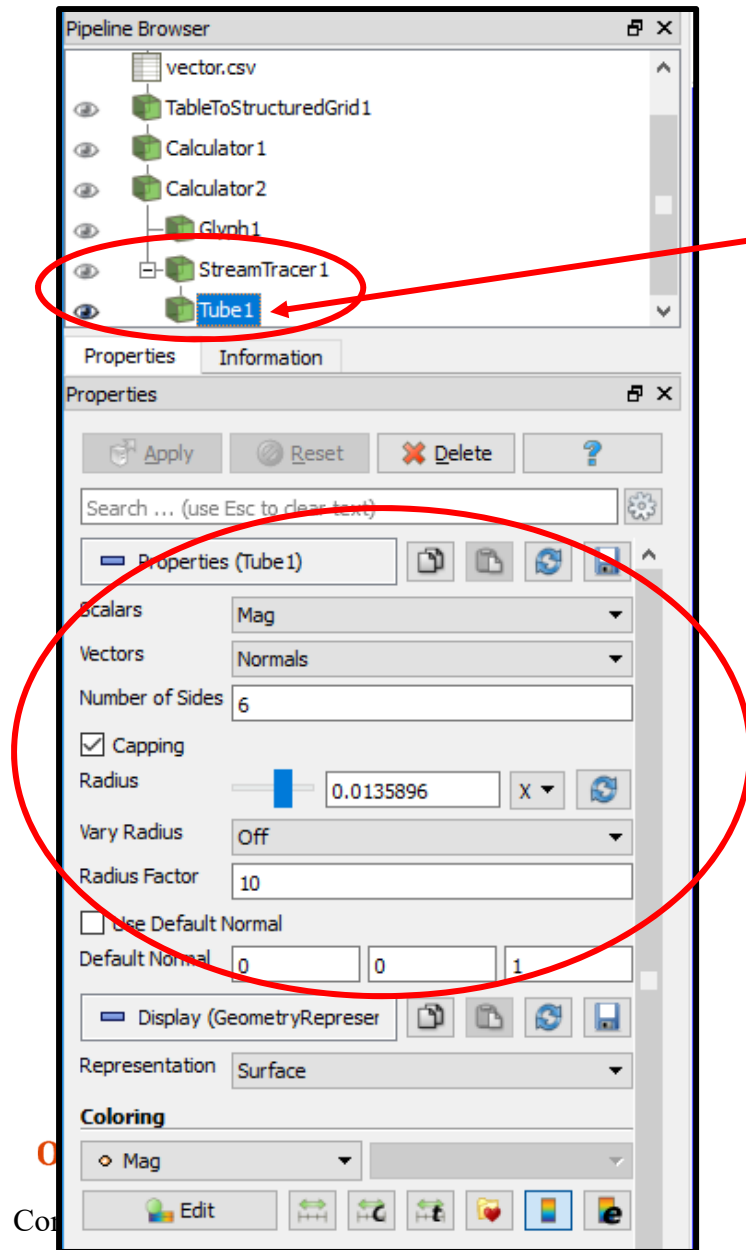
The **Scalar** setting tells what will be used to size the width of the ribbons.

The **Vector** setting tells what will be used to decide which way the ribbon is facing.

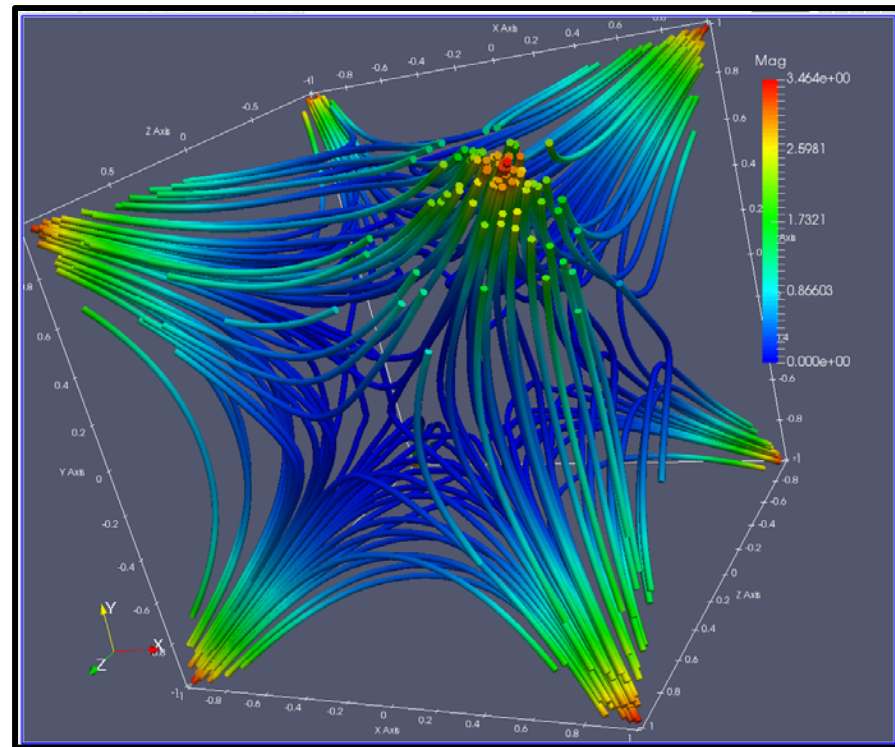




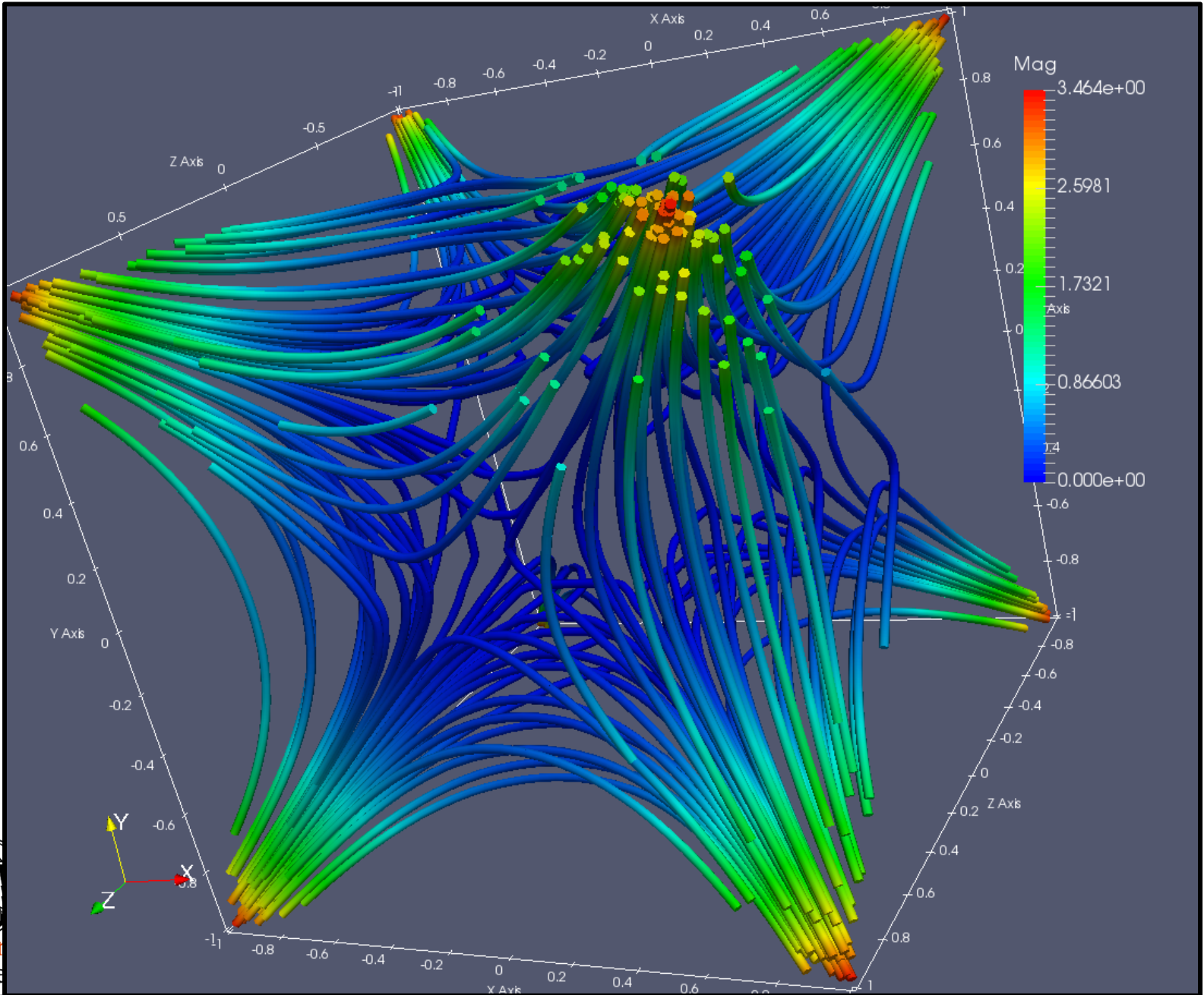
As Streamtubes



Note – Tube is parented from **StreamTracer**.

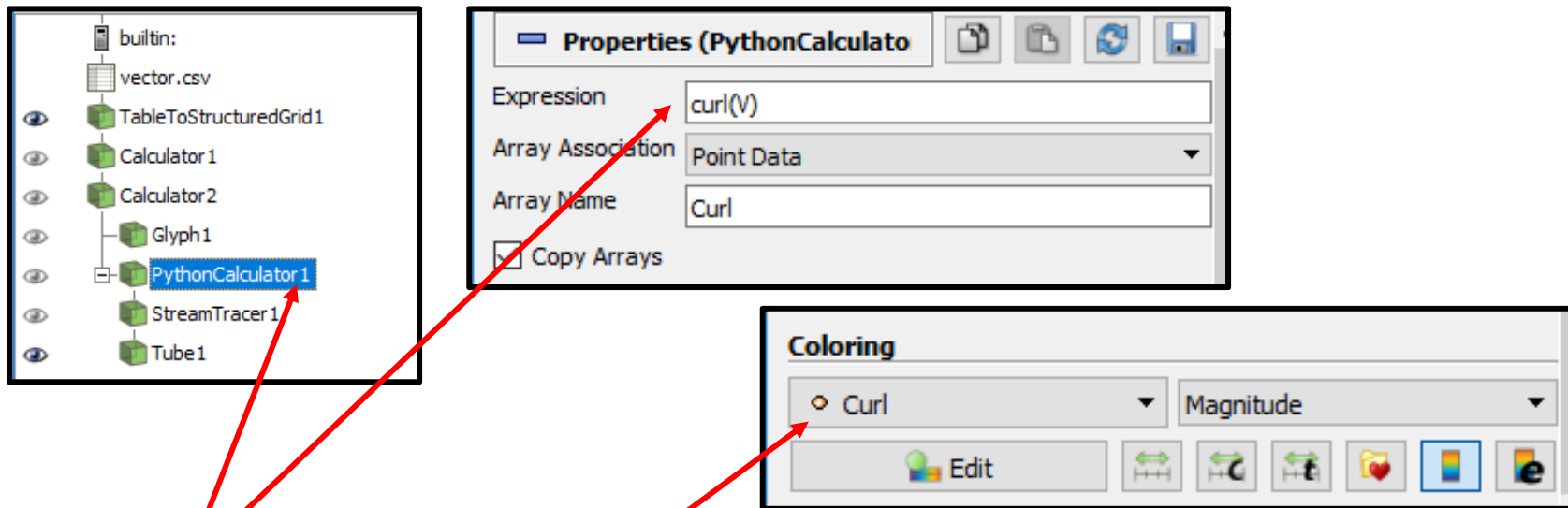


0
Con

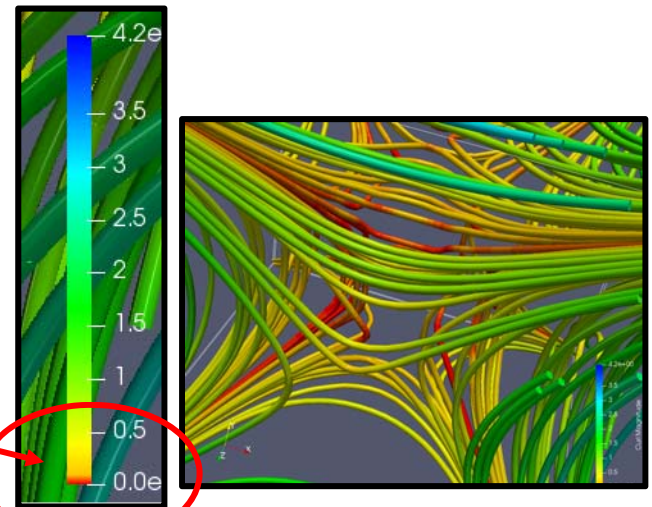


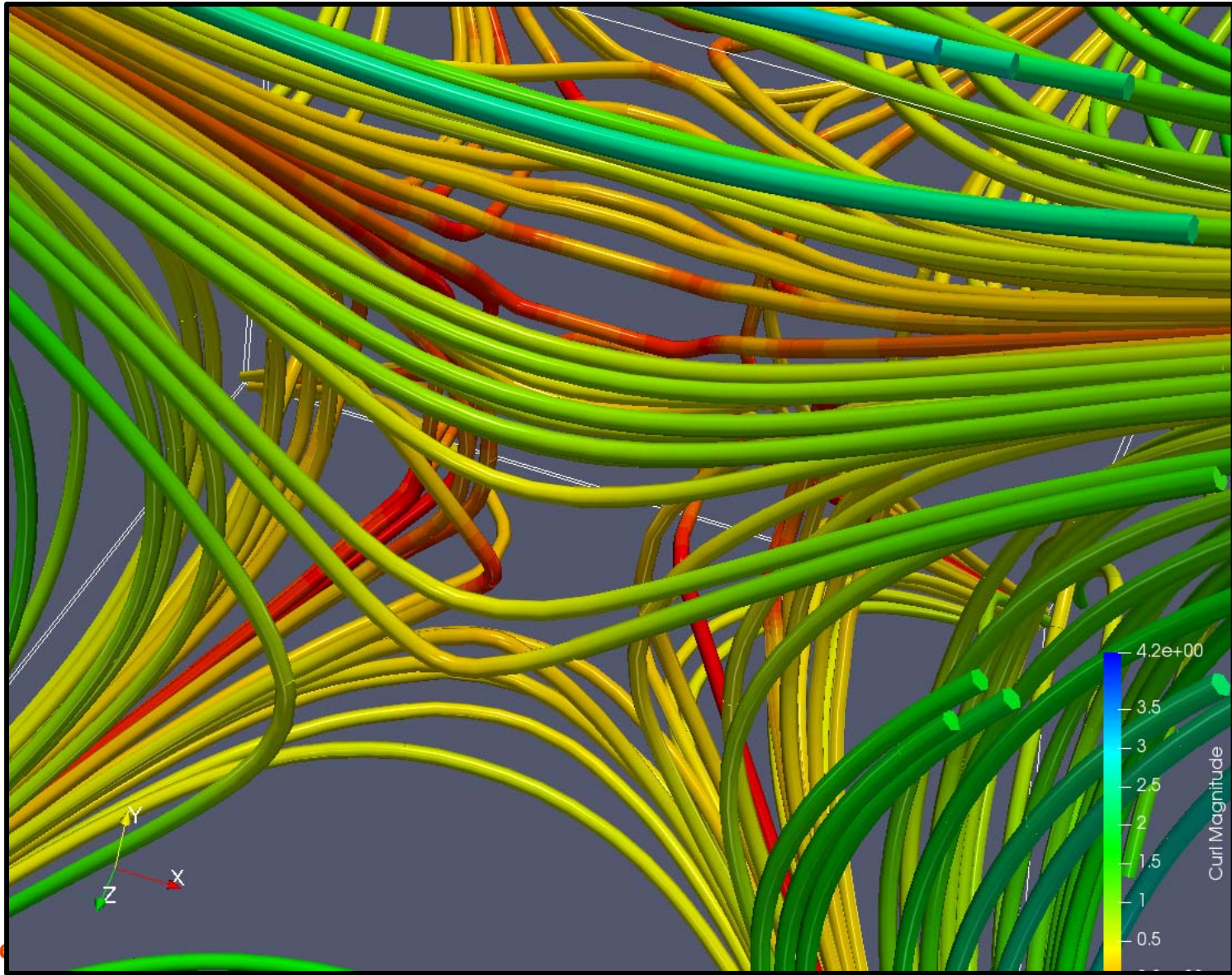
Streamtubes are Especially Useful if You Want to Map Scalar Values to the Streamlines

In this case, we will map curvature (defined by the curl of the vector field)



- The **Python Calculator** filter was used to produce the **Curl** of the vector field (it has a built-in `curl()` function – the Calculator does not)
- The StreamTube's coloring was changed from **Mag** to **Curl**
- The color mapping was changed to cut down on the amount of red (lots of low curl values)





Functions Available in the Python Calculator

- `area(dataset)`
- `aspect(dataset)`
- `cos(array)`
- `cross(X,Y)` where X and Y are two 3D vector arrays
- `curl(array)`
- `divergence(array)`
- `dot(a1,a2)`
- `eigenvalue(array)`
- `eigenvector(array)`
- `gradient(array)`
- `max(array)`
- `mean(array)`
- `min(array)`
- `norm(array)`
- `sin(array)`
- `strain(array)`
- `volume(array)`
- `vorticity(array)`

From: https://www.paraview.org/Wiki/Python_calculator_and_programmable_filter



Visualizing Terrain Data



terrain.csv

Creating Terrain Data in a CSV File

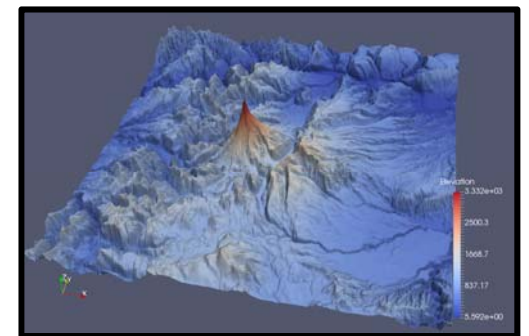
```

UTMx,UTMy,Z,Longitude,Latitude,Elevation
-6909.865,-6870.170,1174.991,-122.200,45.010,1174.991
-6882.896,-6870.356,1268.436,-122.198,45.010,1268.436
-6855.759,-6870.542,1308.478,-122.196, 5.010,1308.478
-6828.789,-6870.728,1266.755,-122.193,45.010,1266.755
-6801.820,-6870.911,1203.239,-122.191,45.010,1203.239
-6774.682,-6871.095,1127.675,-122.189,45.010,1127.675
-6747.544,-6871.279,1074.388,-122.187,45.010,1074.388
-6720.575,-6871.461,1060.748,-122.185,45.010,1060.748
-6693.606,-6871.642,1056.135,-122.182,45.010,1056.135
-6666.468,-6871.823,1050.158,-122.180,45.010,1050.158
-6639.499,-6872.002,1029.548,-122.178,45.010,1029.548
-6612.361,-6872.182,1001.763,-122.176,45.010,1001.763
-6585.391,-6872.360,975.069,-122.174,45.010,975.069
-6558.254,-6872.539,980.551,-122.172,45.010,980.551
-6531.284,-6872.715,1029.739,-122.169,45.010,1029.739

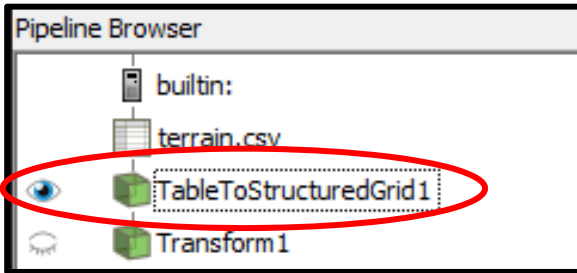
```

Do a **File** → **Open** and navigate to your CSV file.
Hit the **Apply** button to actually do the read.

UTM data is in meters, which makes a more reality-looking base than longitude and latitude do. It is good to have both Z and Elevation, even though they are the same number because once you use a variable for a geometric dimension, you can't also use it again for a data value (e.g., to color or contour by elevation).

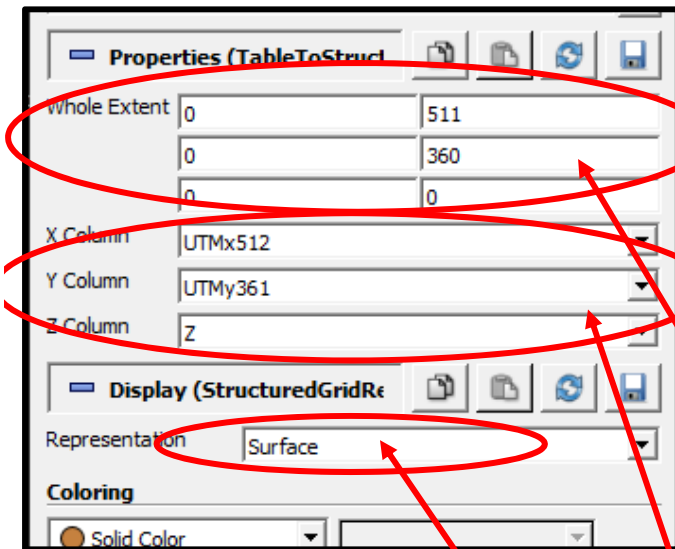


Reading and Converting the CSV File



This will bring up a table window to confirm that the data has been read properly. You can delete this now if you want.

Row ID	Elevation	Latitude	Longitude	UTMx	UTMy
0	1174.99	45.01	-122.2	-6909.86	-6870.17
1	1268.44	45.01	-122.198	-6882.9	-6870.36
2	1308.48	45.01	-122.196	-6855.76	-6870.54
3	1266.76	45.01	-122.193	-6828.79	-6870.73
4	1255.51	45.01	-122.191	-6801.82	-6870.91
5	1127.67	45.01	-122.189	-6774.68	-6871.1
6	1074.39	45.01	-122.187	-6747.54	-6871.28
7	1060.75	45.01	-122.185	-6720.57	-6871.46
8	1056.13	45.01	-122.182	-6693.61	-6871.64
9	1050.16	45.01	-122.18	-6666.47	-6871.82
10	1029.55	45.01	-122.178	-6639.5	-6872
11	1001.76	45.01	-122.176	-6612.36	-6872.18
12	975.069	45.01	-122.174	-6585.39	-6872.36
13	980.551	45.01	-122.172	-6558.25	-6872.54
14	1029.74	45.01	-122.169	-6531.28	-6872.72
15	1077.68	45.01	-122.167	-6504.31	-6872.89
16	1128.52	45.01	-122.165	-6477.38	-6873.07
17	1167.35	45.01	-122.163	-6450.21	-6873.24
18	1156.65	45.01	-122.161	-6423.07	-6873.42
19	1120.73	45.01	-122.158	-6396.1	-6873.59
20	1157.74	45.01	-122.156	-6368.96	-6873.76
21	1224.09	45.01	-122.154	-6341.99	-6873.94
22	1301.08	45.01	-122.152	-6314.86	-6874.11
23	1336.25	45.01	-122.15	-6287.89	-6874.28
24	1320.58	45.01	-122.147	-6260.75	-6874.45
25	1286.73	45.01	-122.145	-6233.78	-6874.61
26	1318.22	45.01	-122.143	-6206.61	-6874.78
27	1408.39	45.01	-122.141	-6179.67	-6874.95



Now, go to **Filters** → **Alphabetical** → **TableToStructuredGrid**

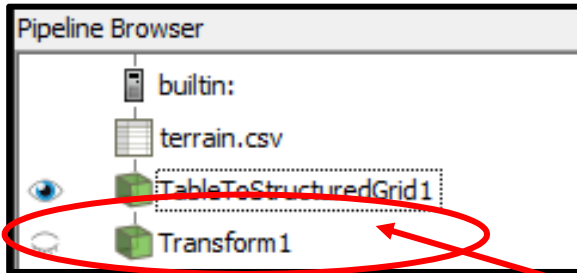
Fill in the **Whole Extent** boxes showing the first and last index in each dimension (the last index is one less than the number of points in that dimension).

Fill in the **{X,Y,Z} Column** information so ParaView knows how to make your 3D display.

Hit the **Apply** button to actually do the conversion.

Change **Representation** to **Surface**

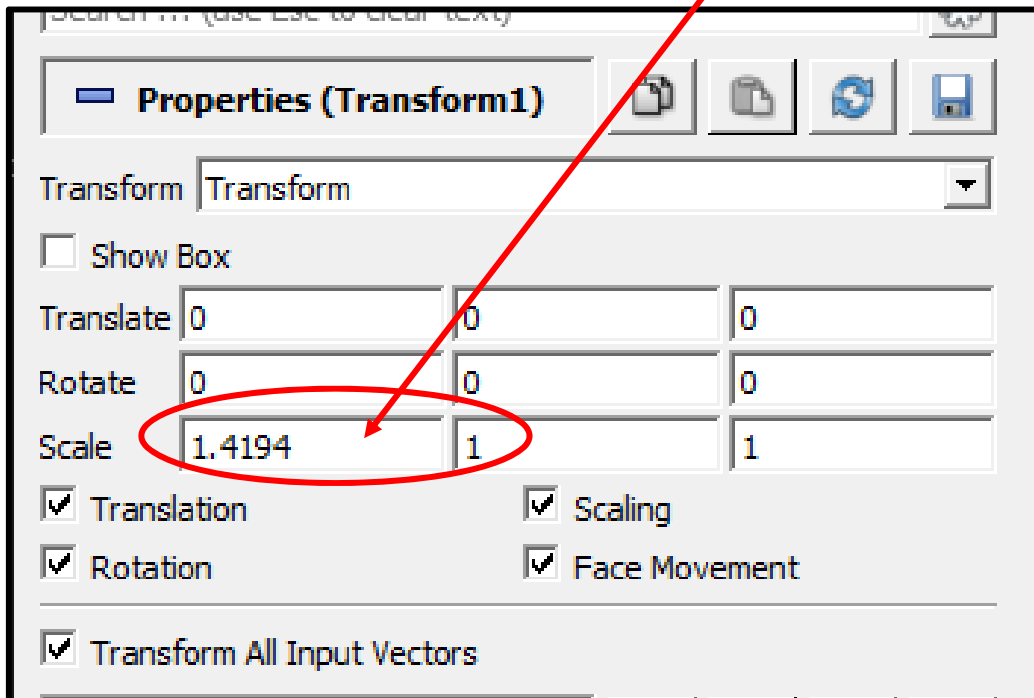
The Correct Scale Factor

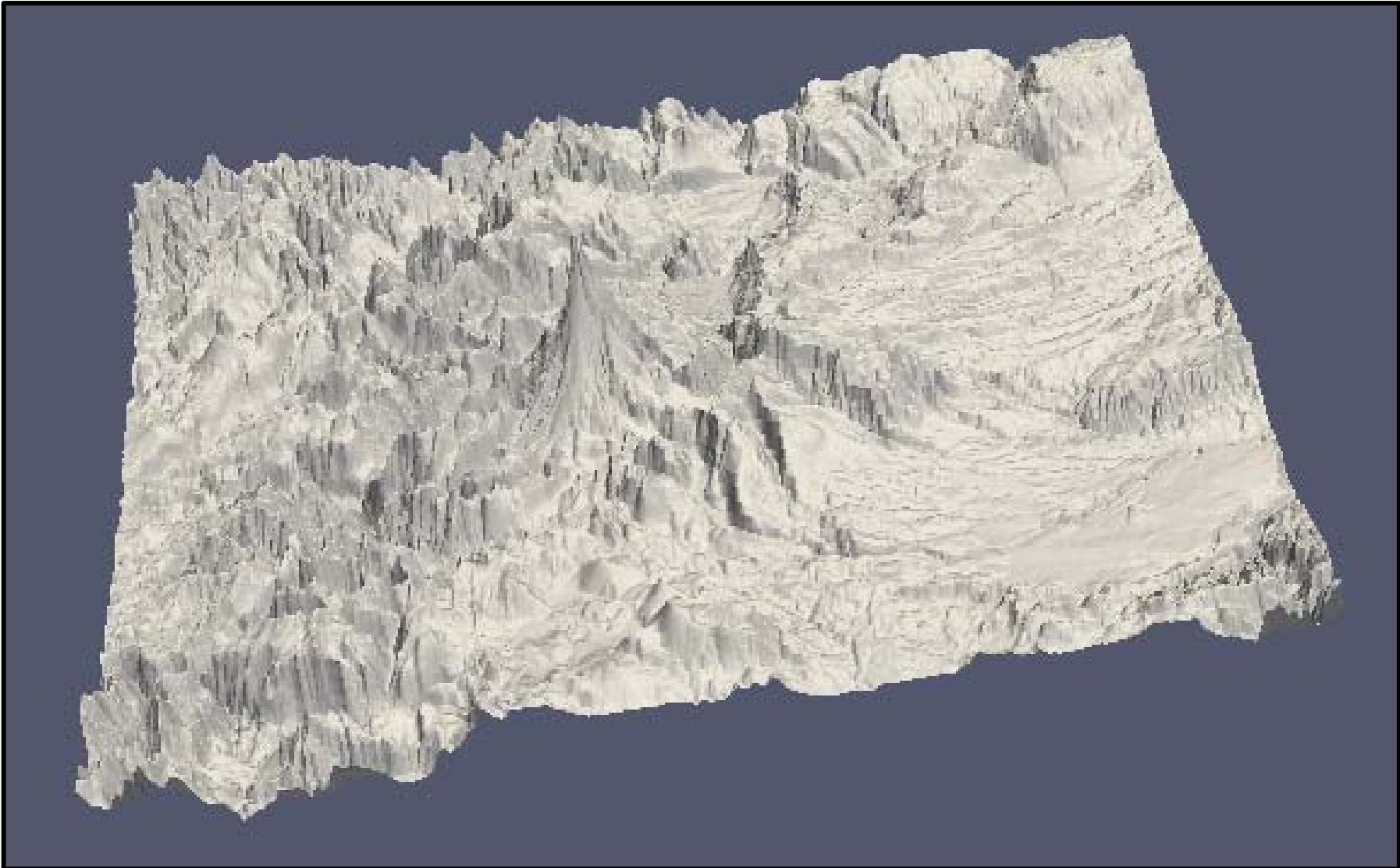


This will bring up a square terrain, which isn't what we want. We notice that the UTM coordinates are 511 and 360, so we really want to scale by $511/360 = 1.4194$ in the X direction.

Now, go to
Filters → **Alphabetical** → **Transform**

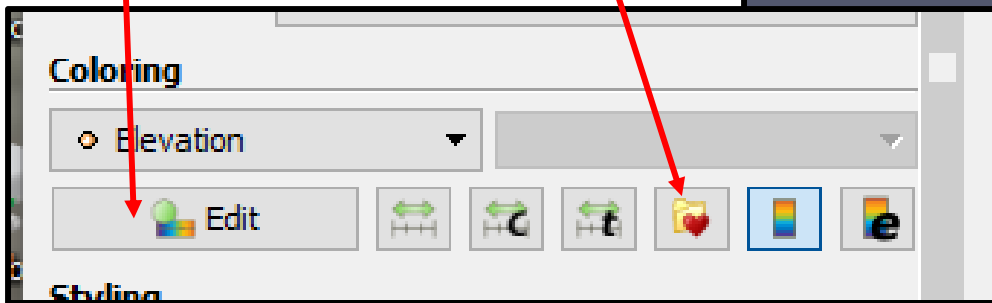
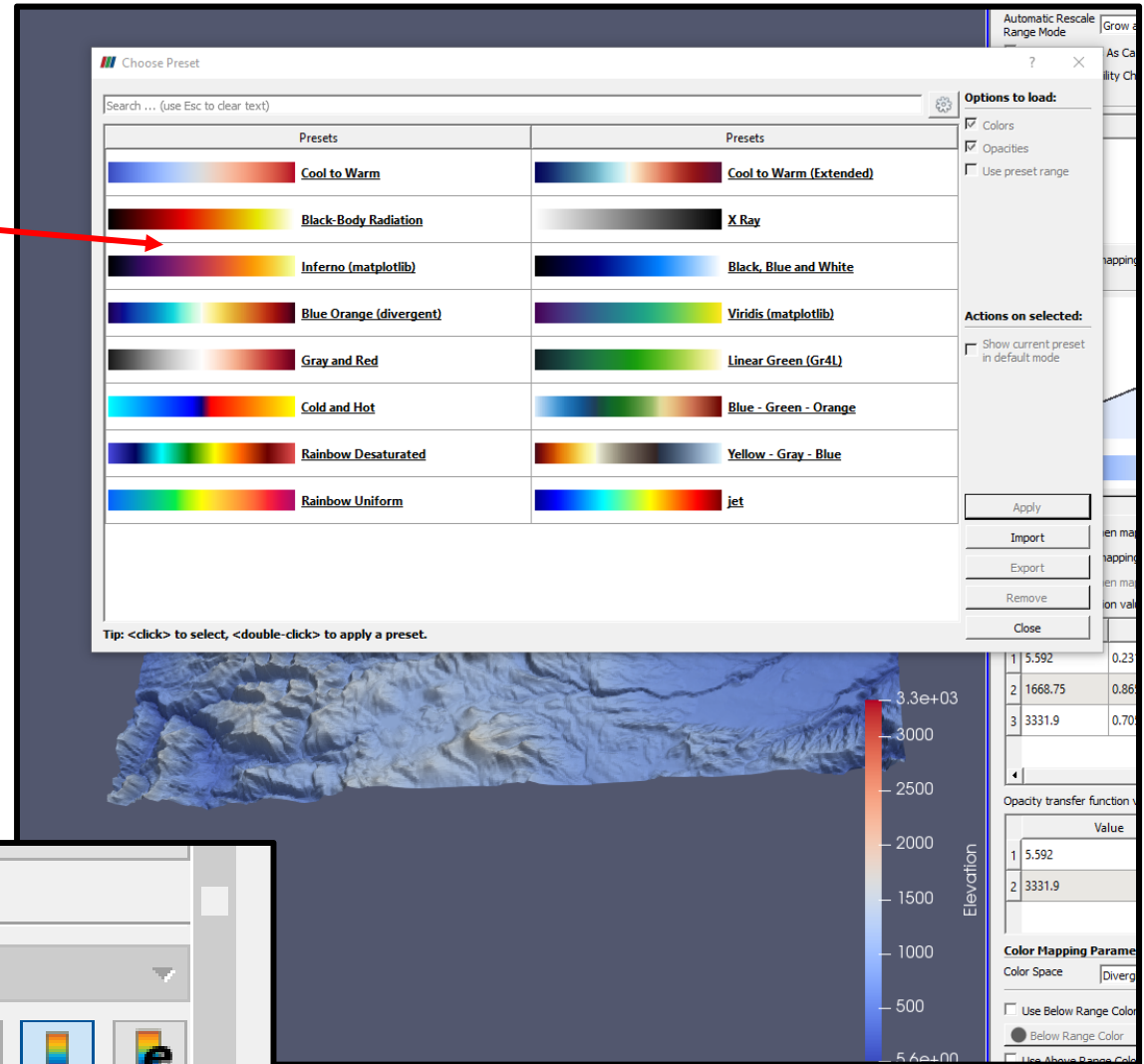
Set the X scale factor to 1.4194

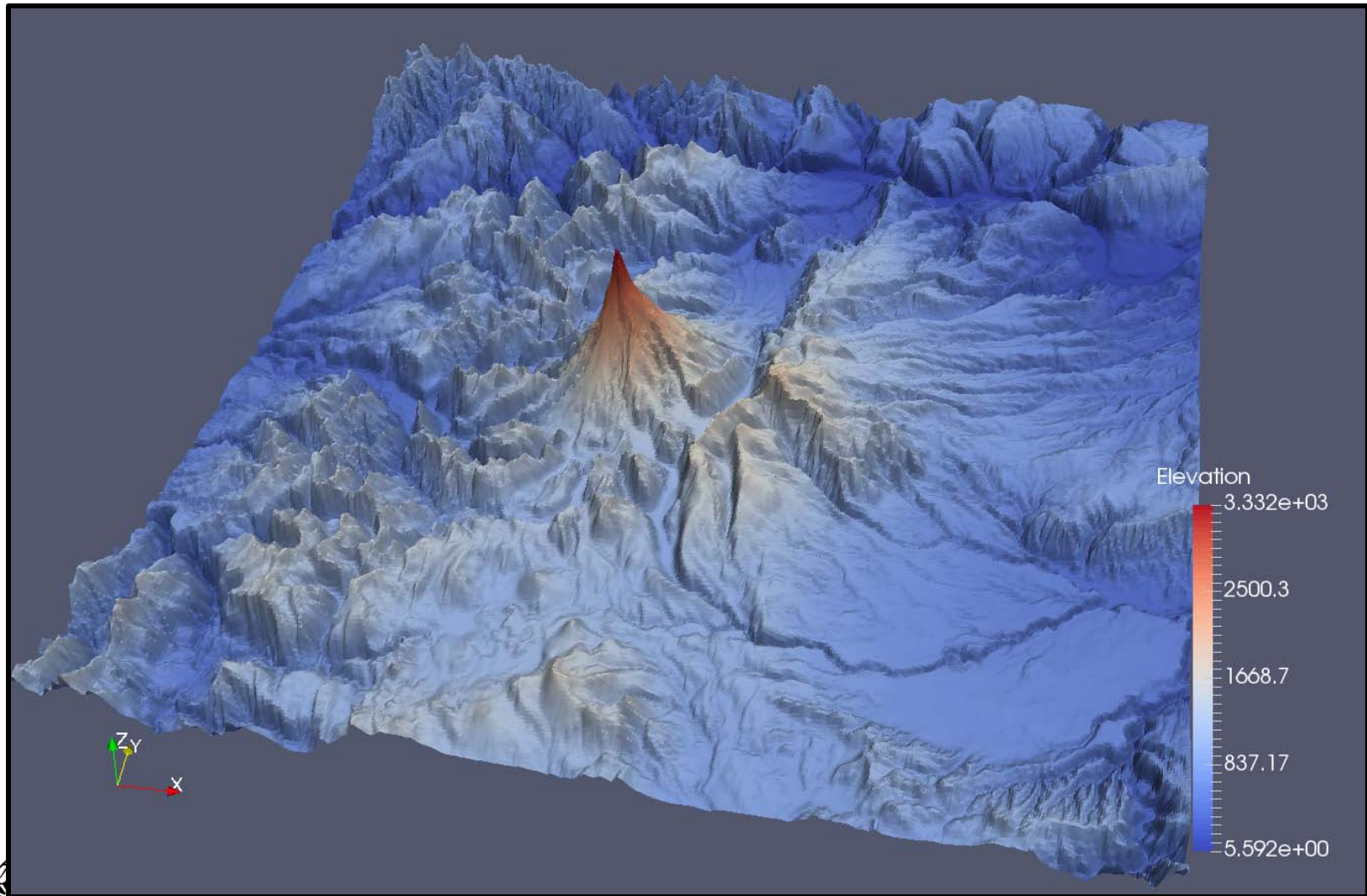




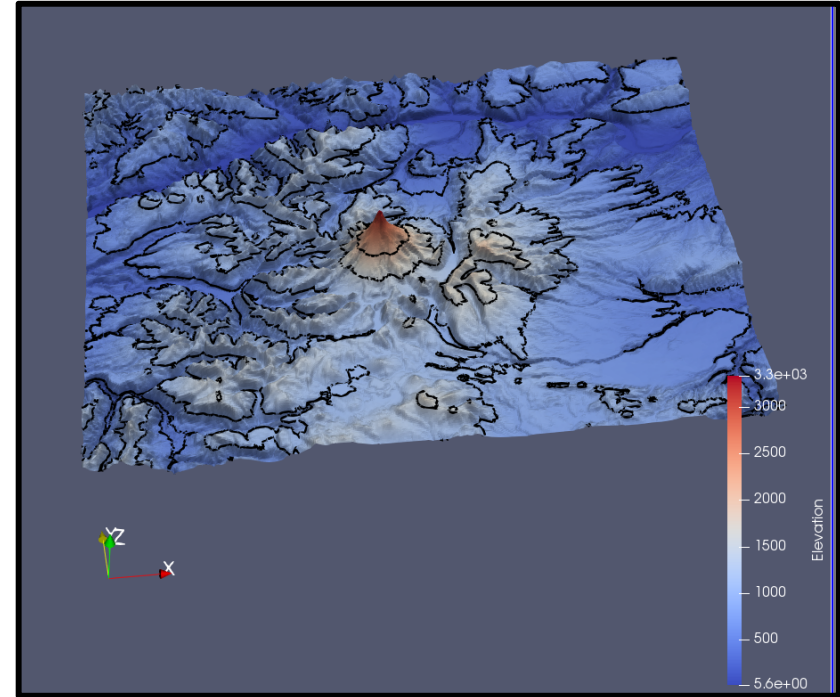
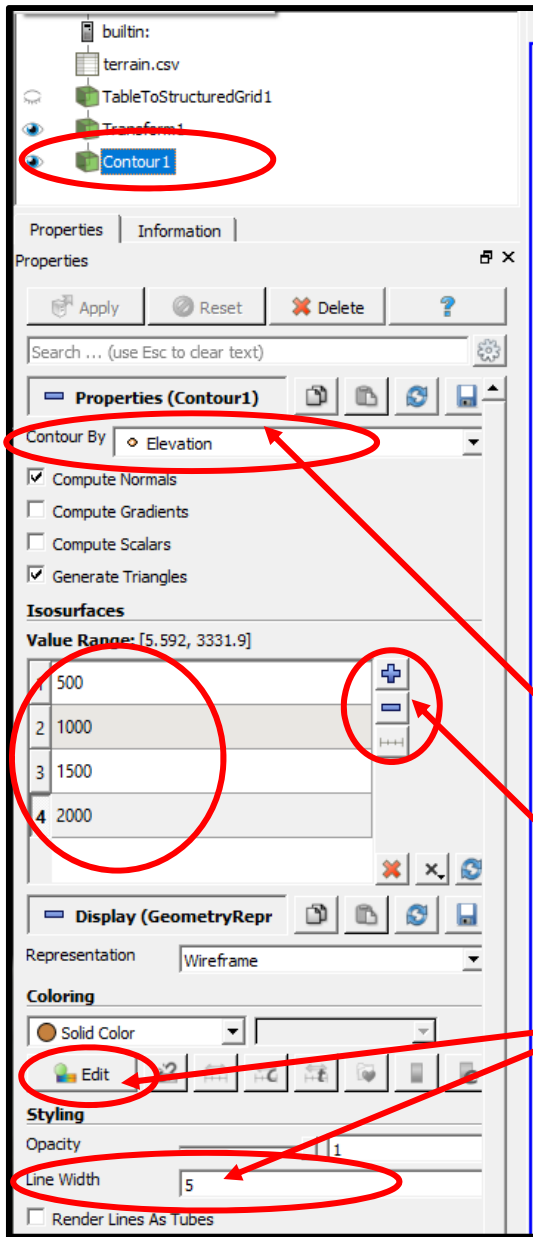
Color by Elevation

Try coloring by **Elevation**.
The heart icon brings up popular color scales. You can pick one of these or sculpt your own.





Contouring



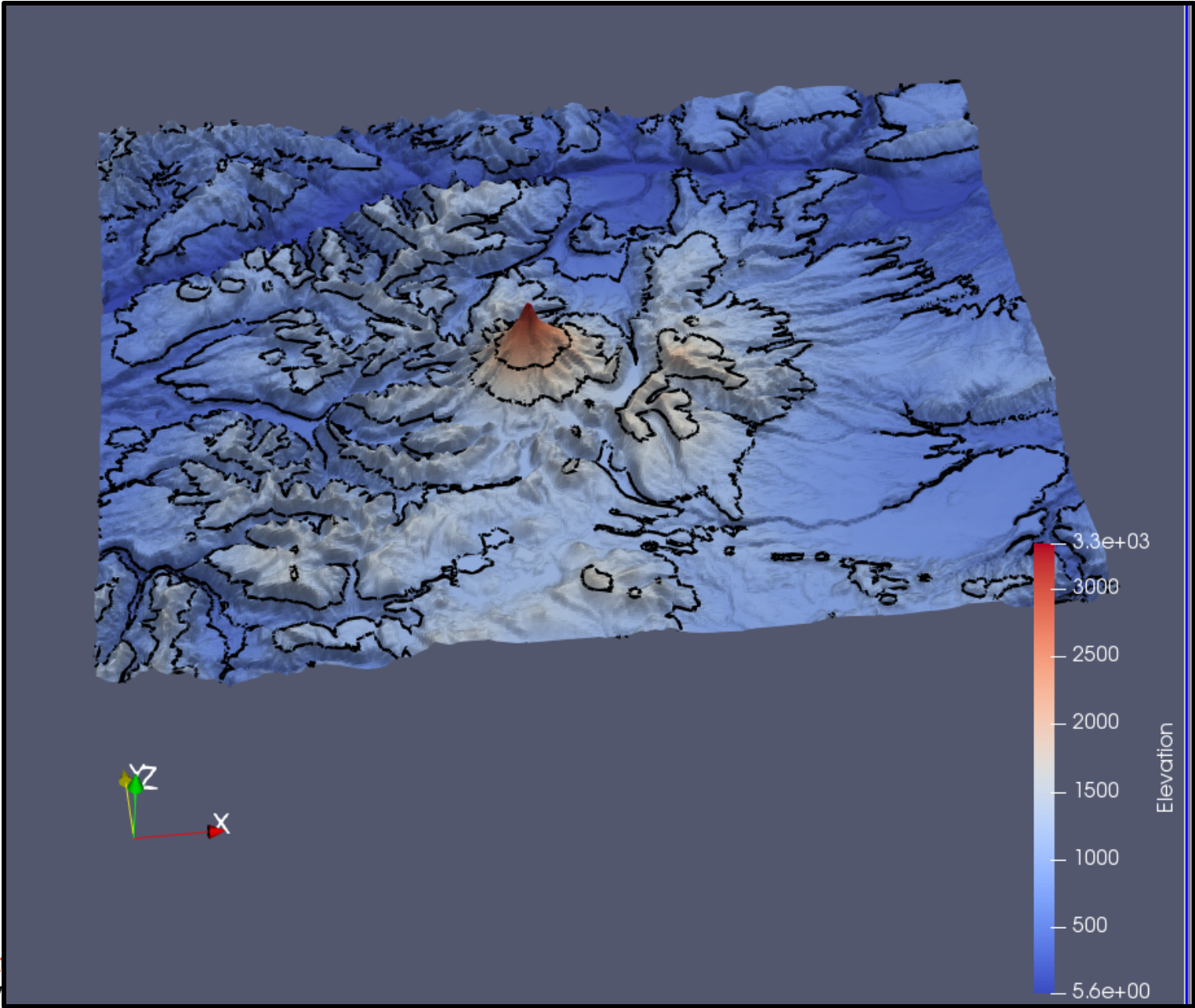
Now, go to **Filters** → **Alphabetical** → **Contour** and select **Contour by Elevation**

ParaView gives one default contour elevation, but you can add more.

Edit to select a contour color and enter a **Line Width**.

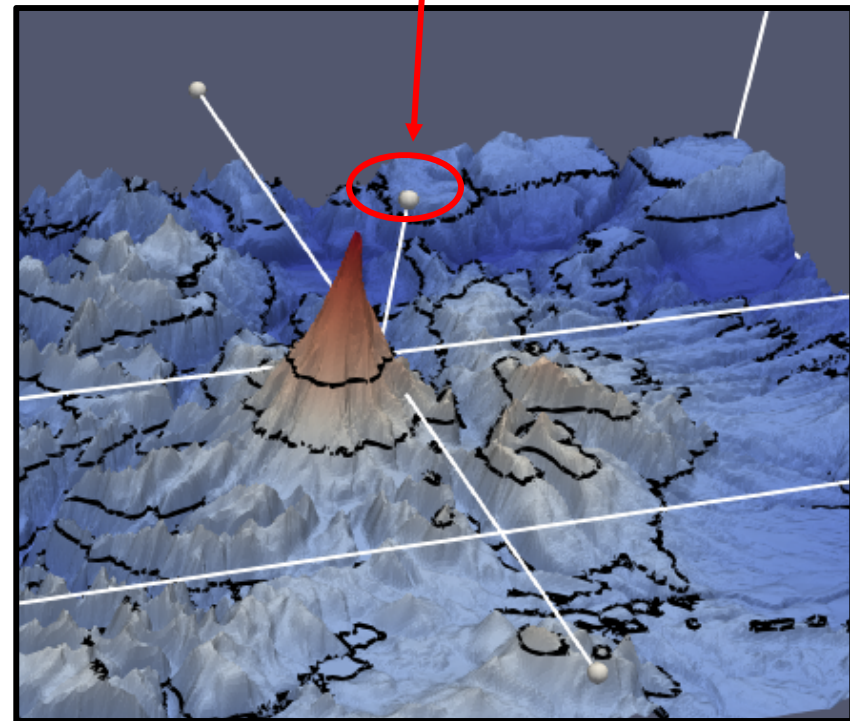
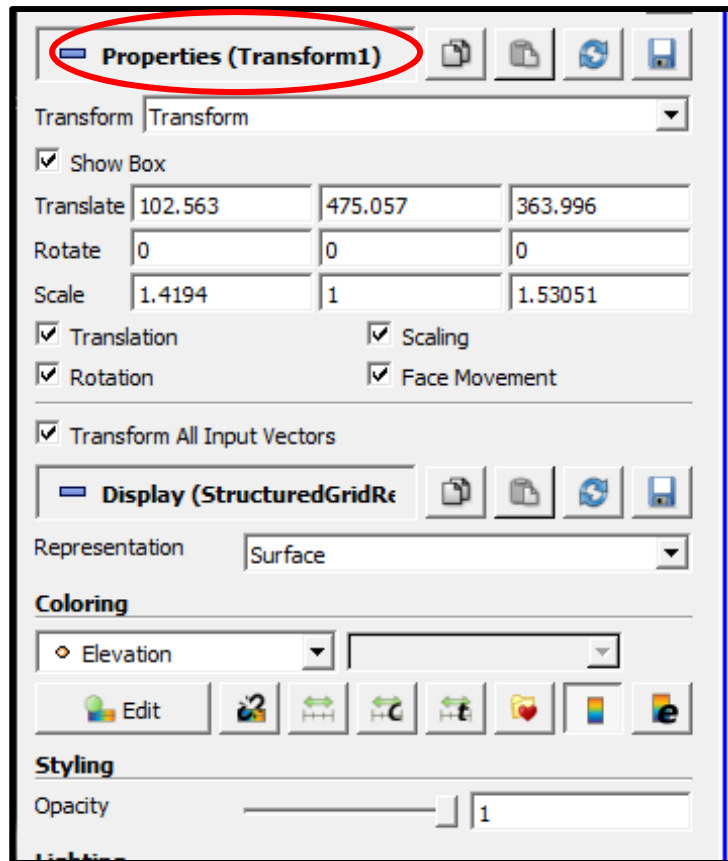
Hit the **Apply** button.

Be sure the eyeballs are turned on.



Changing the Vertical Exaggerations

Re-click on the **Transform** filter, turn on the Box, and move the scaling knob



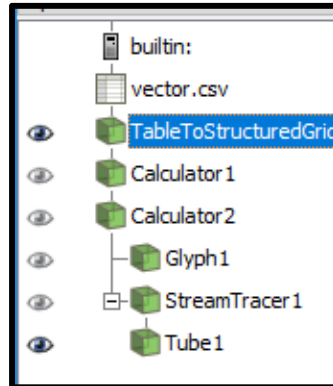
Parallel Coordinates



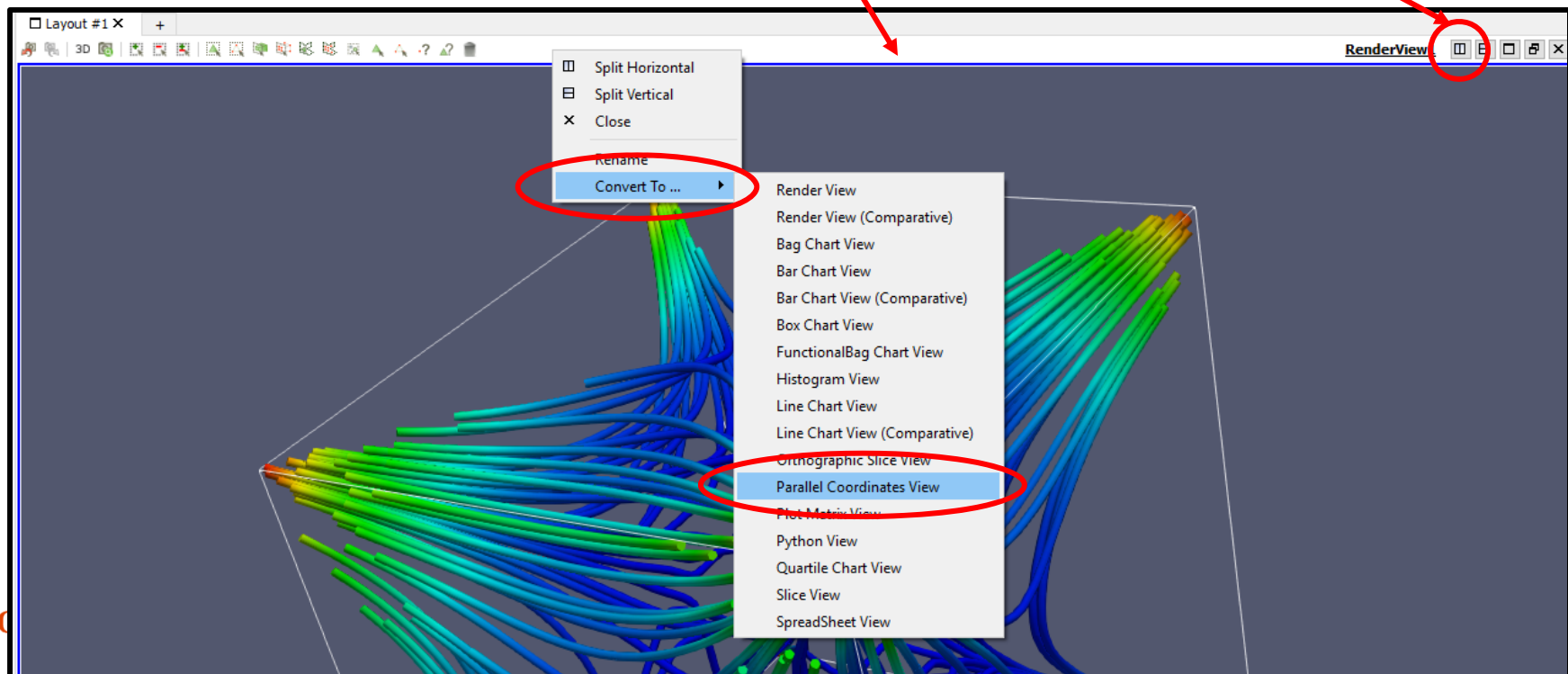
vector.pvsm
parallelcoords.pvsm

Parallel Coordinates – Correlating Fields

Let's say you were to start with this:

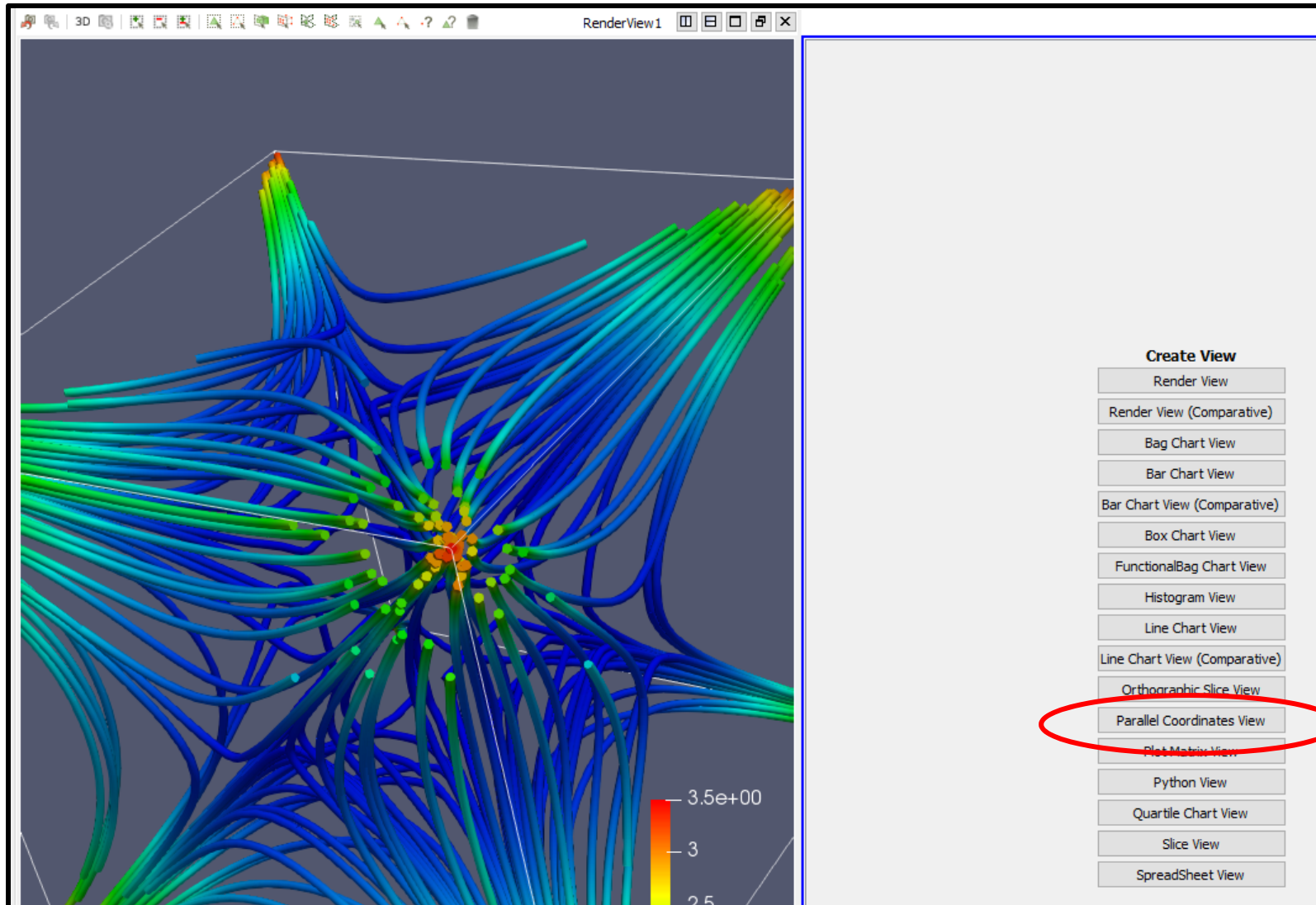


Either convert the **Render View** window to a **Parallel Coordinate View** window by **right-clicking** anywhere in the window header bar, or by splitting the window

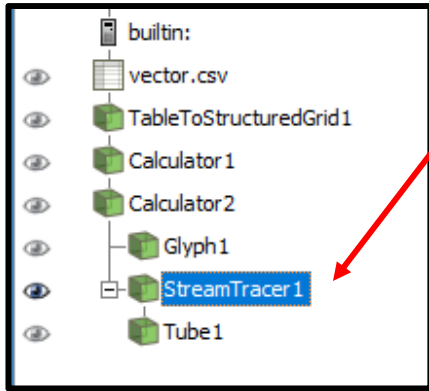


Parallel Coordinates

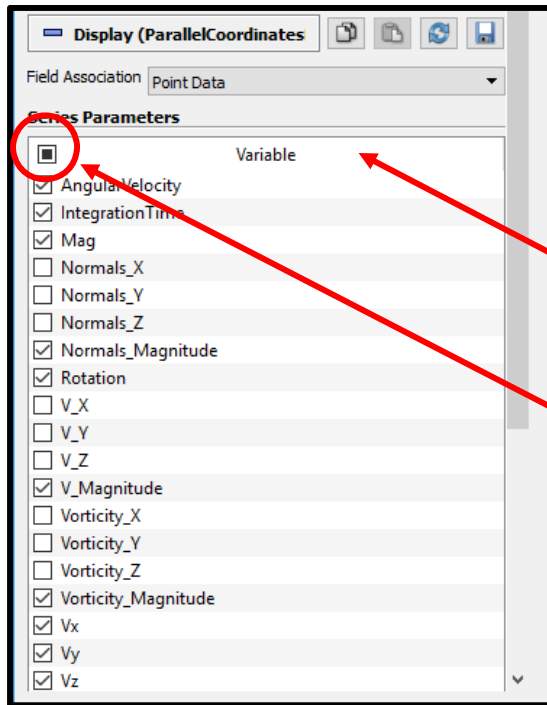
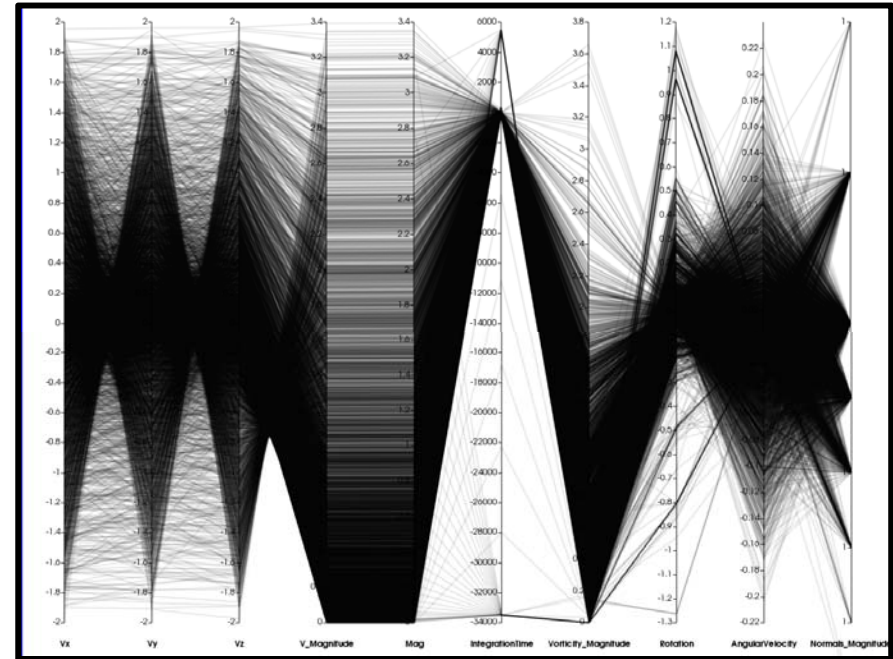
Splitting the window looks like this. Select **Parallel Coordinates View**.



Parallel Coordinates



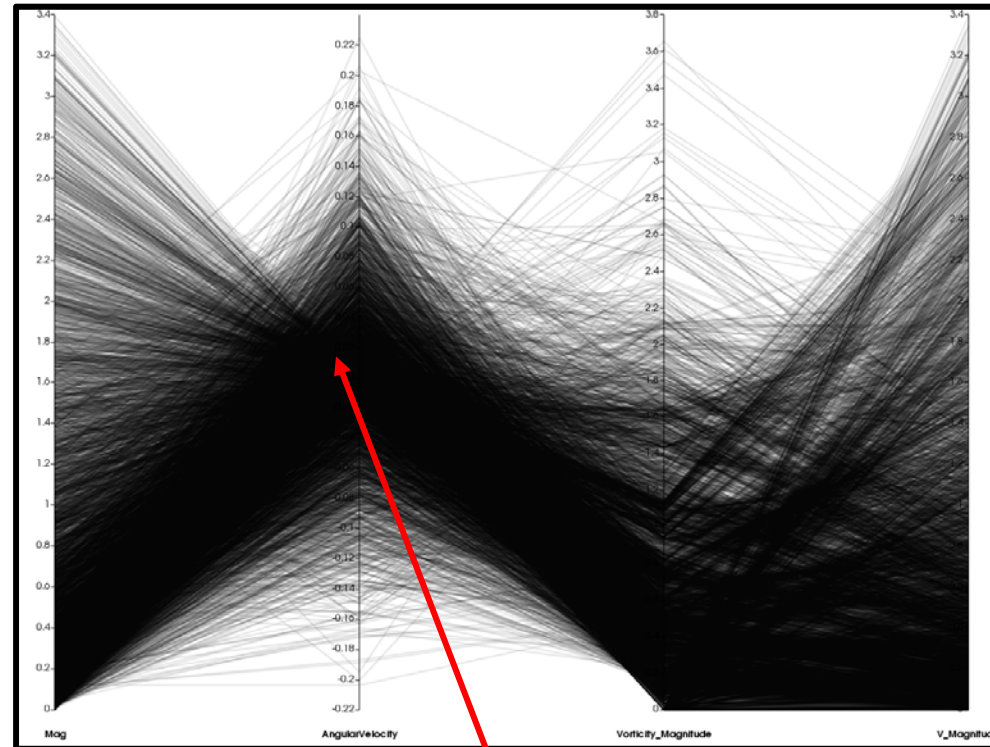
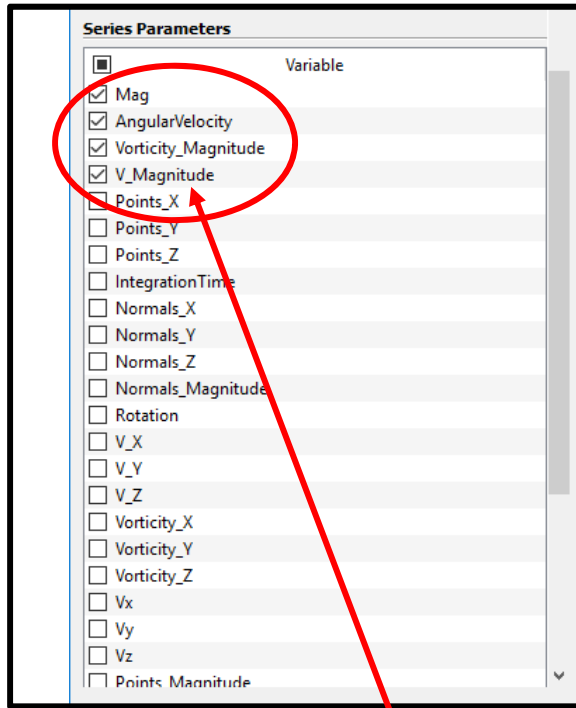
Turn the eyeballs on for the **StreamTracer**. It turns out StreamTracer creates a bunch of derived variables, so this will give us more to look at.



The **Parallel Coordinates Display Properties** shows what variables will be displayed. No matter what, they are probably not exactly the variables you wanted to see and they are not in the desired horizontal order.

So, click them all off and turn them back on in the horizontal order you want to see them.

Parallel Coordinates



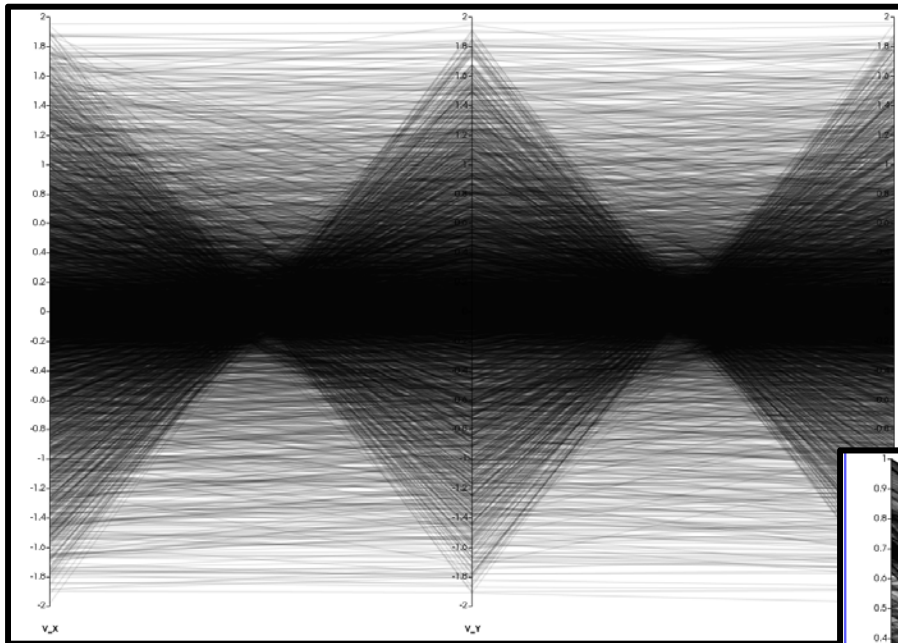
So, click them all off and turn them back on in the horizontal order you want to see them.

You can left-click-drag them to a new vertical position in the list to make re-clicking on them in a different order much easier.

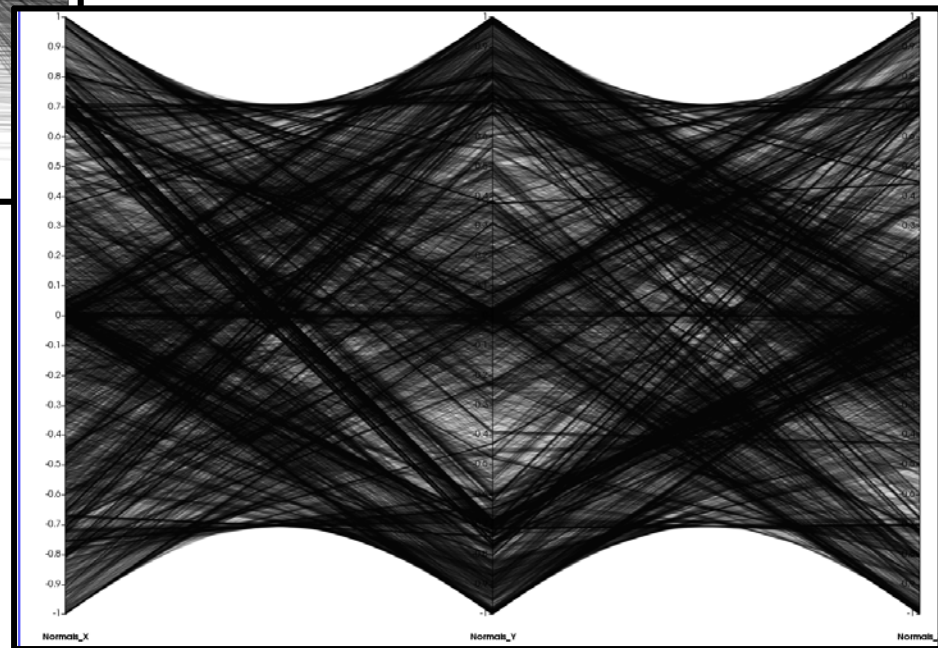
The narrowness of the bundle of lines shows the strength of the positive and negative correlations.

Parallel Coordinates

Lots of (negative) correlation

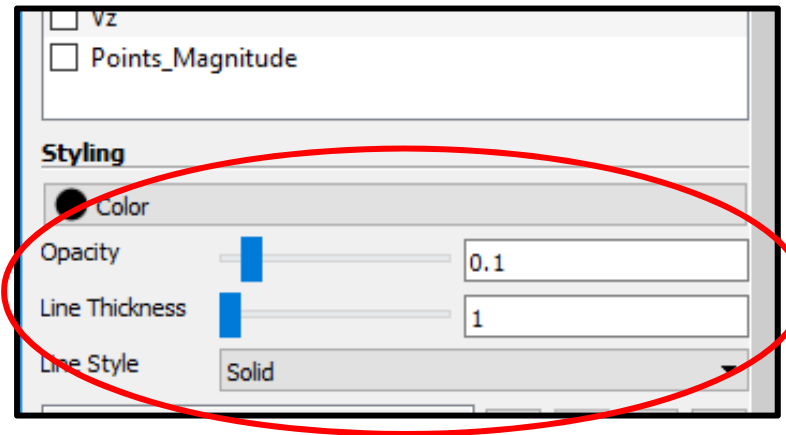


Little correlation

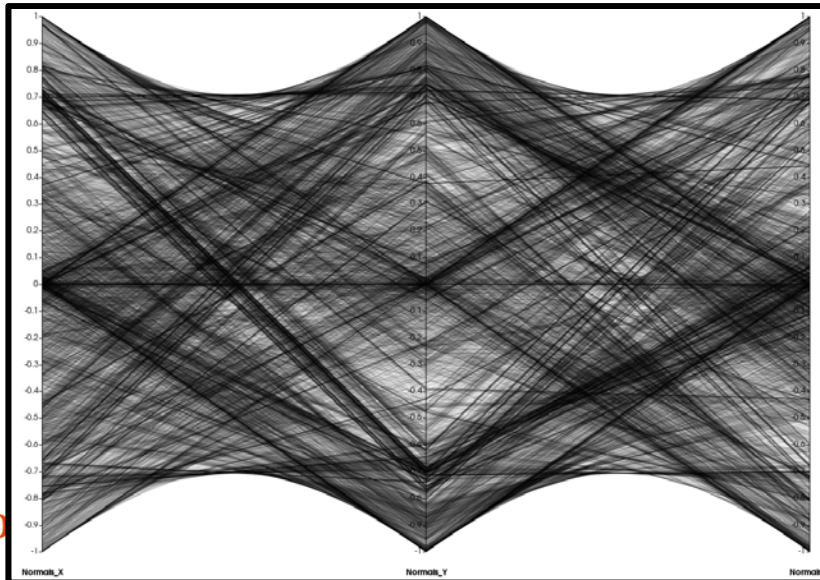


Parallel Coordinates

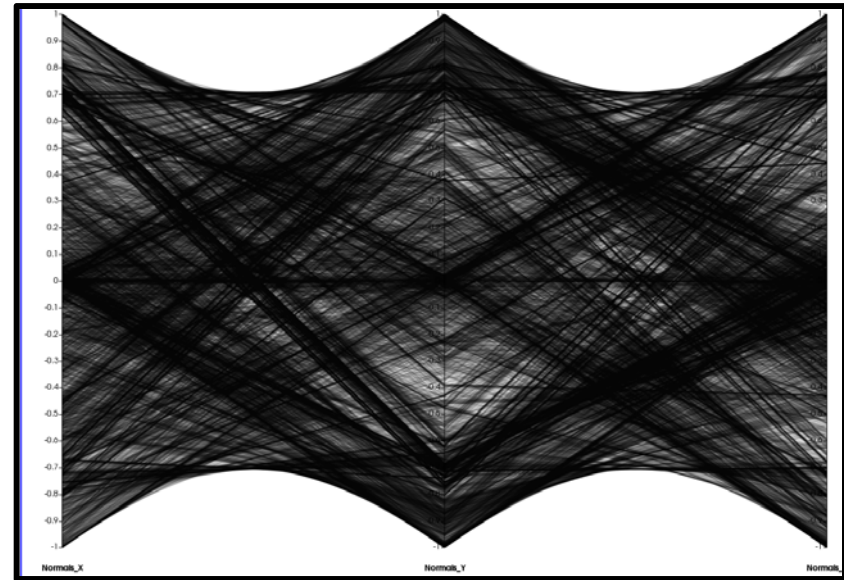
Scroll down a little more in the properties menu and you will find the **Parallel Coordinates Styling** menu:



Line Thickness = 1



Line Thickness = 2

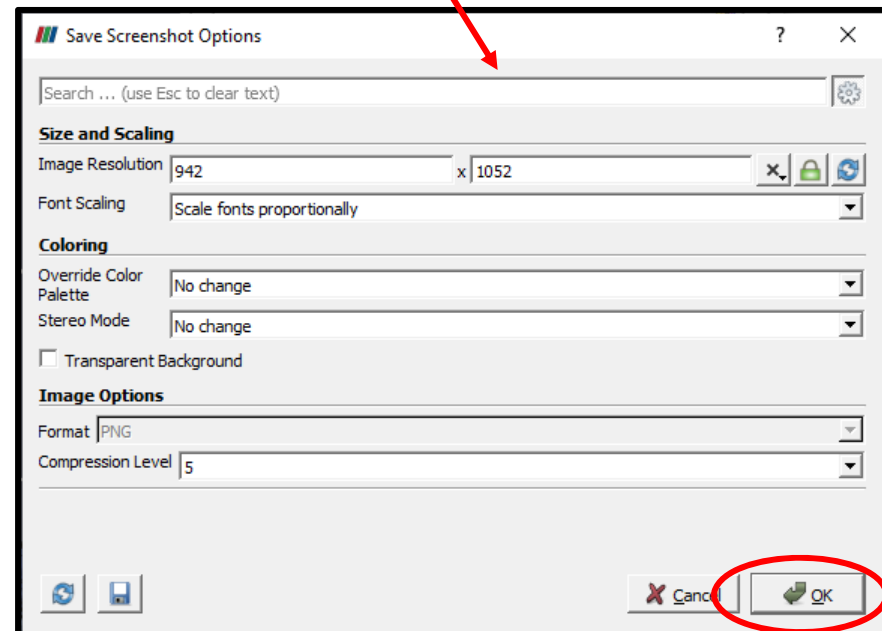
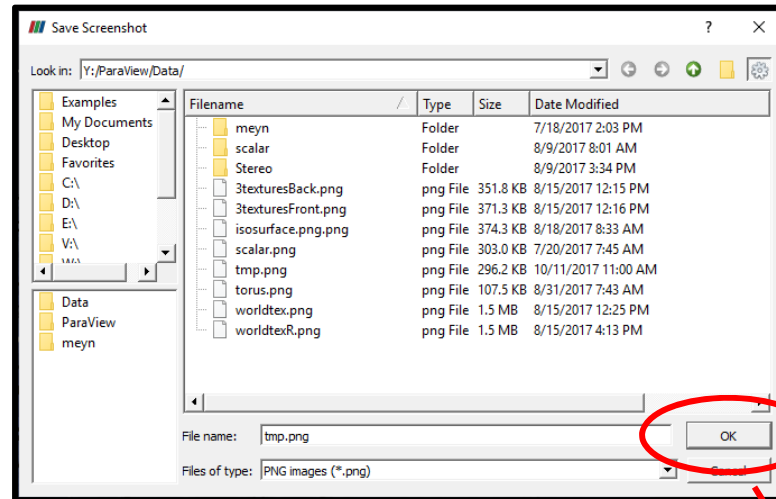
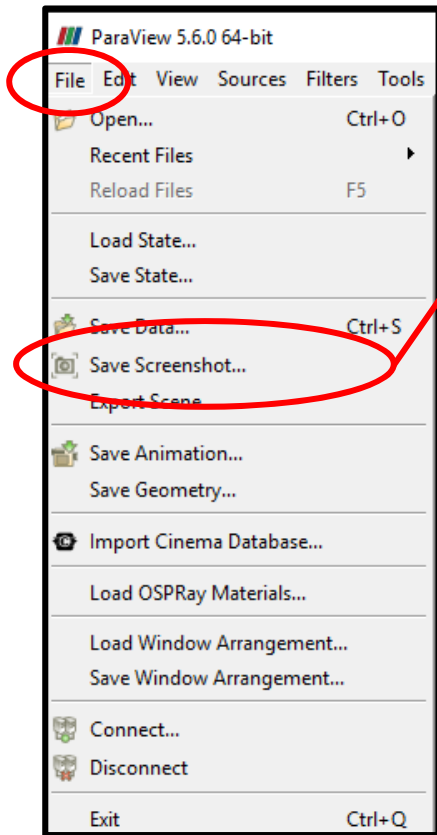


Saving an Image of the Screen



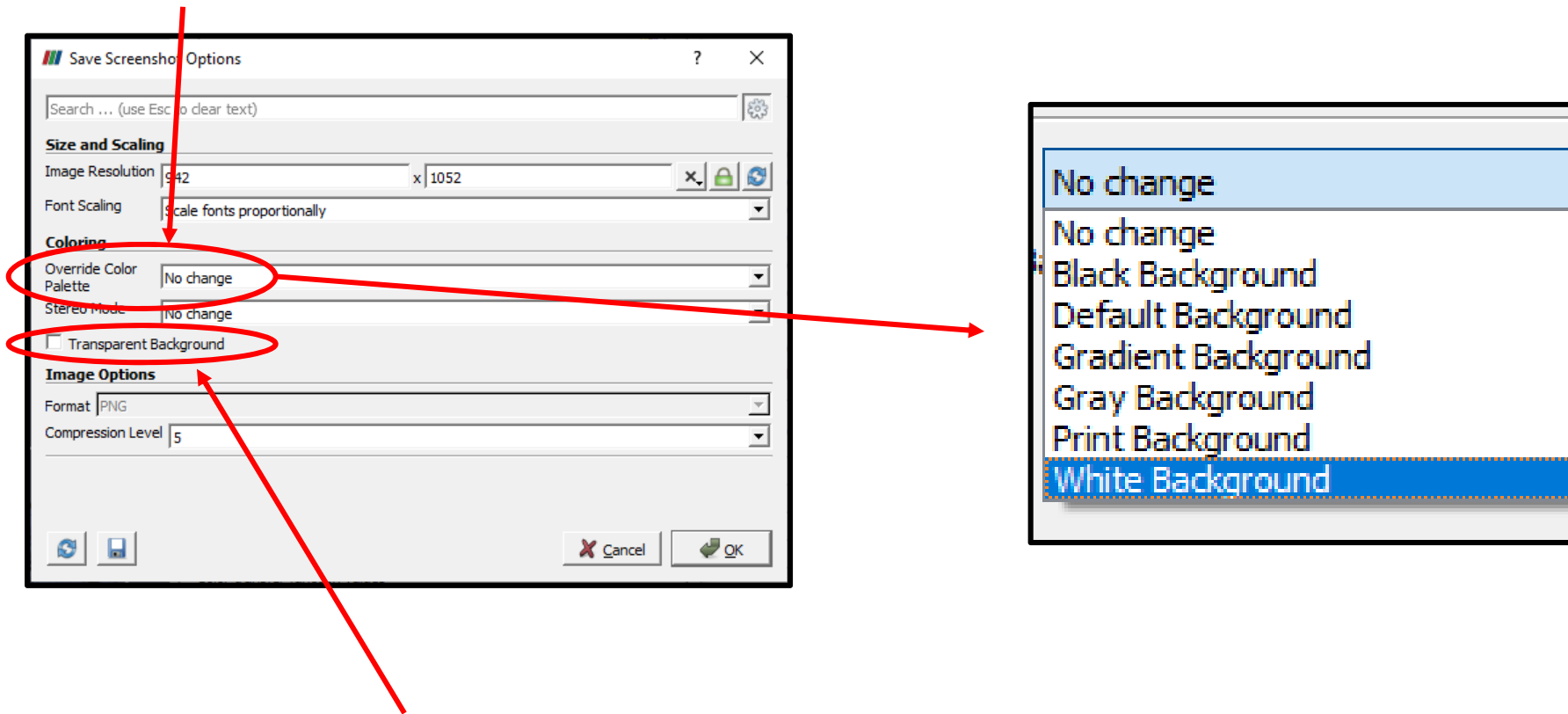
scalar.pvsm

File → Save Screenshot



Changing the Background Color

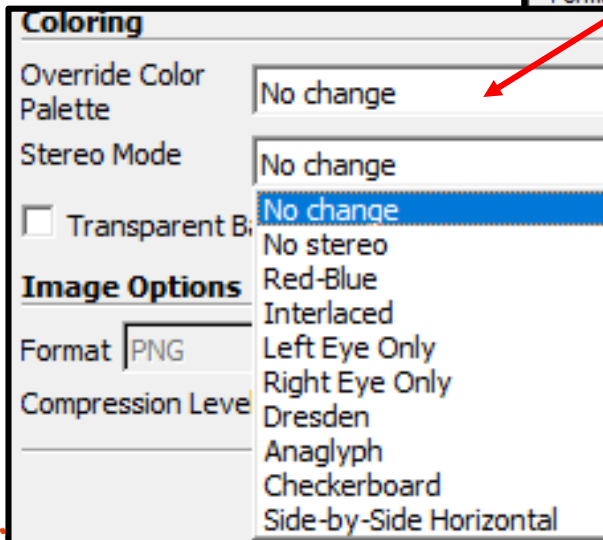
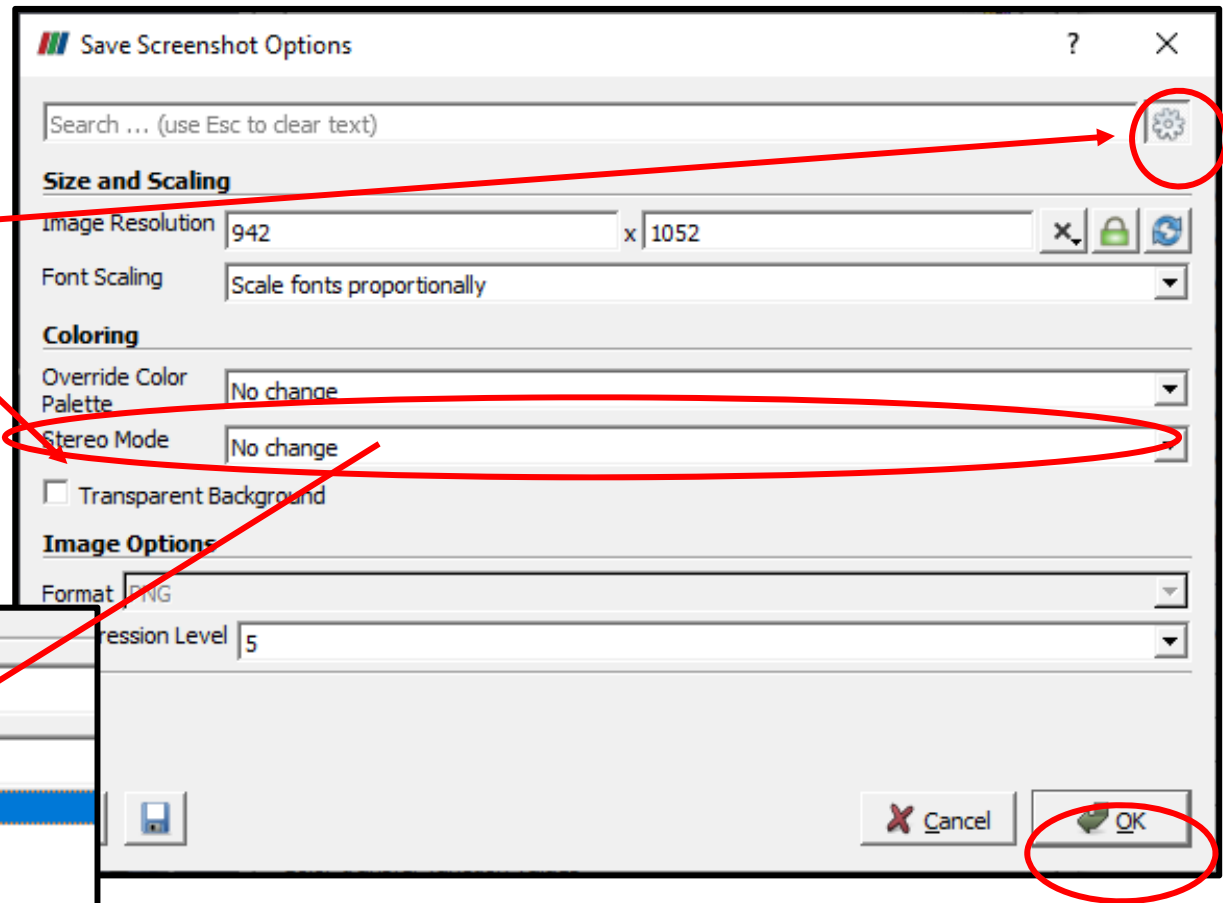
You can override the existing background color just long enough to create the screenshot



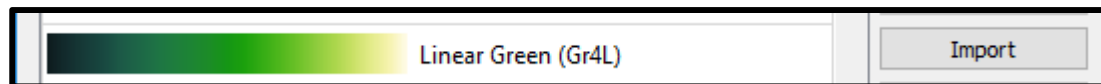
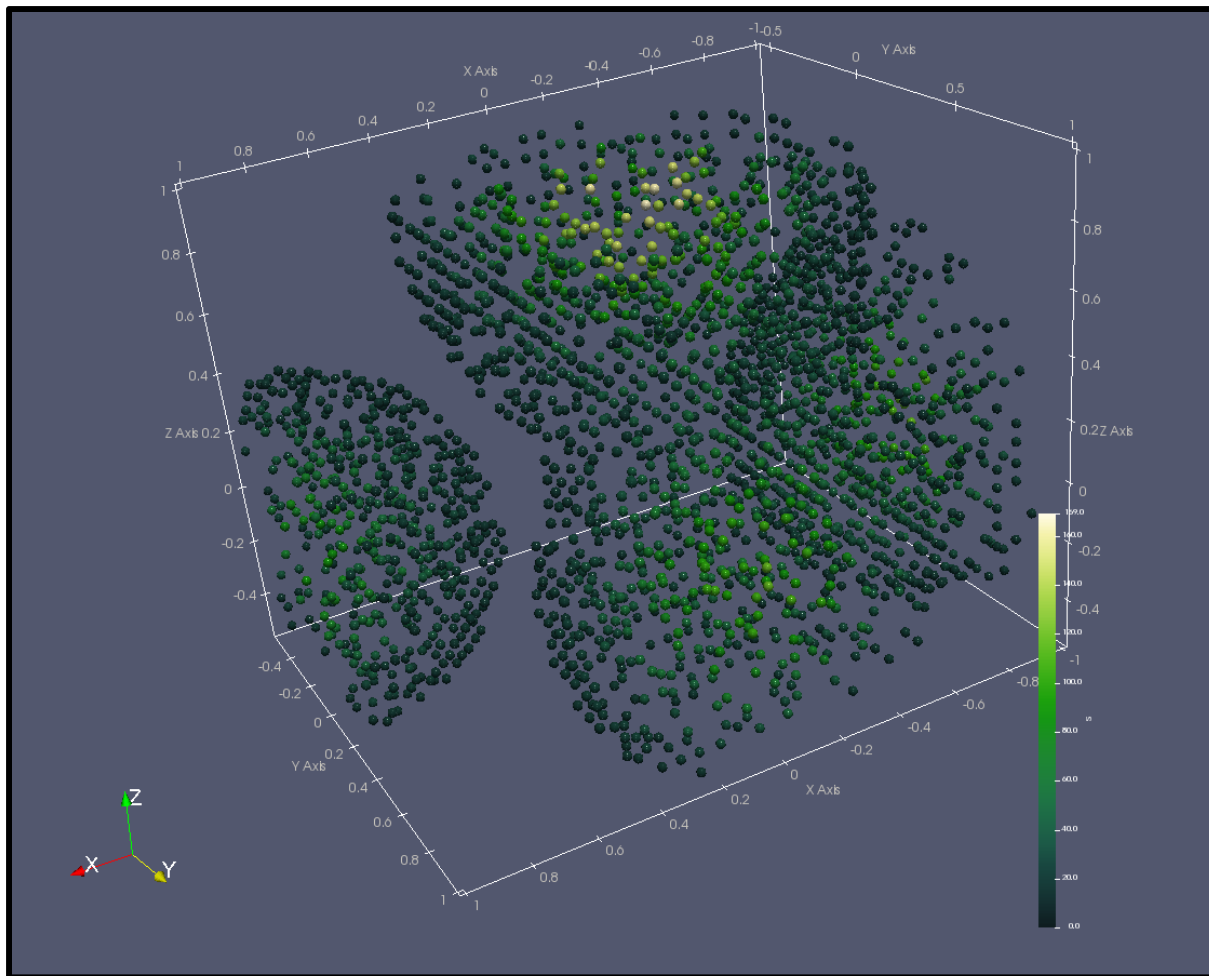
You can also force the image background to be transparent. (This only works on some image file formats, such as PNG.)

Creating Stereographics Images

Turning on the **Advanced Settings** enables **Stereo Mode**

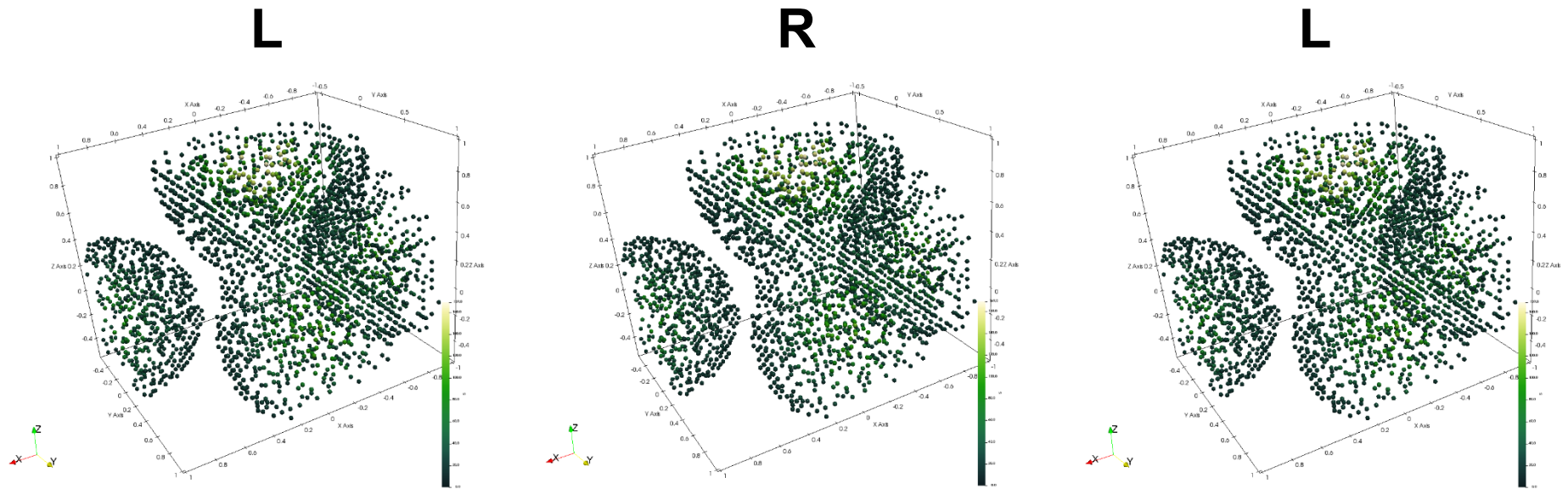


The Original Visualization



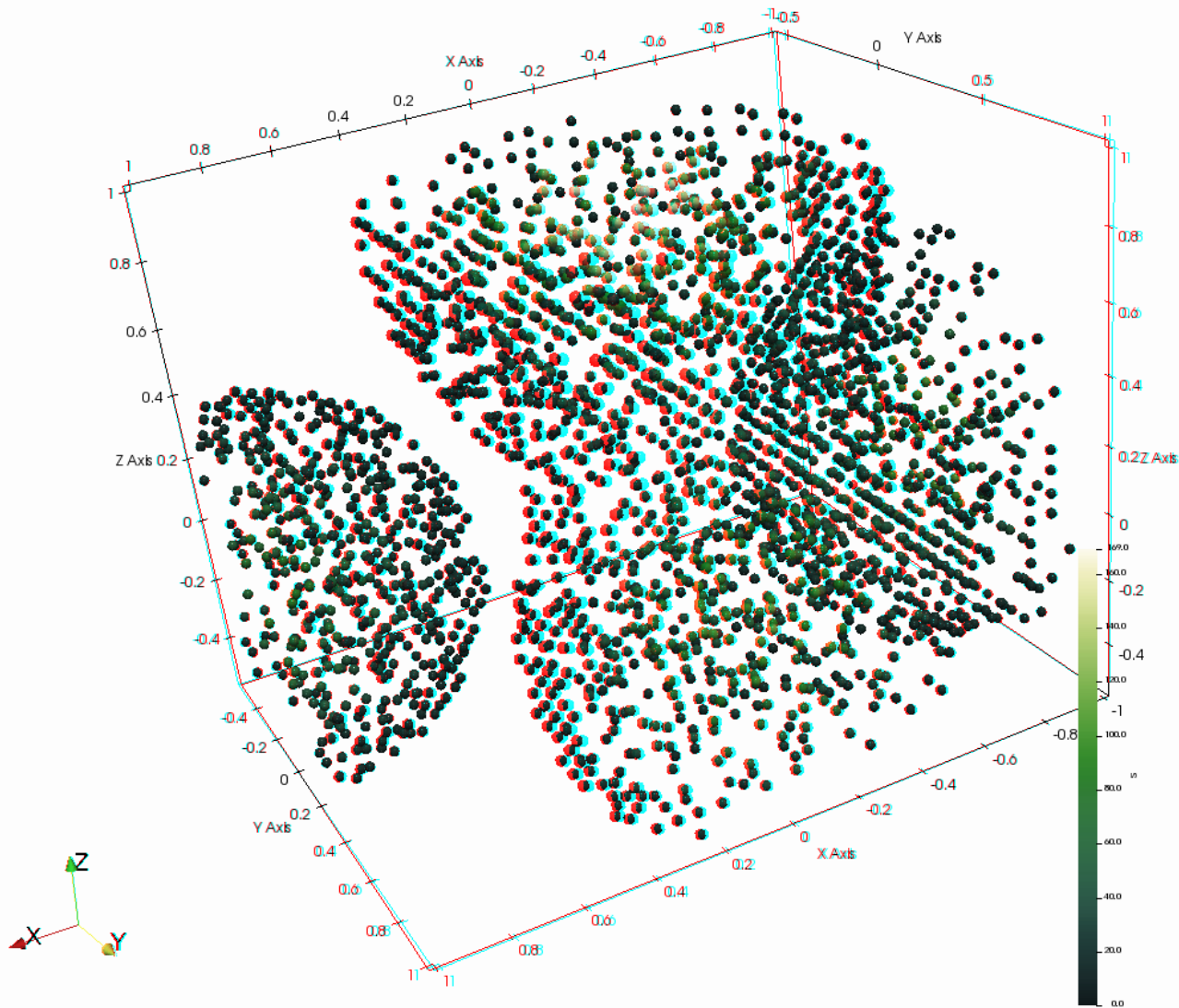
This is using the **Linear Green** color scale because it seems to work better for Red-Cyan Anaglyphs than do color scales with blue or red in them

Side-by-Side Stereopairs



If you can parallel freeview, use the left two images.
If you can cross-eyes freeview, use the right two images
If you can't do either, then never mind

Red-Cyan Anaglyph



The Left Two Images Work Well Together in my Handheld Stereo Viewer

L

R

L



Print this page and cut out the left two images



Note to self: don't resize these images, as much as you are tempted to – they fit perfectly in the viewer as they are now.

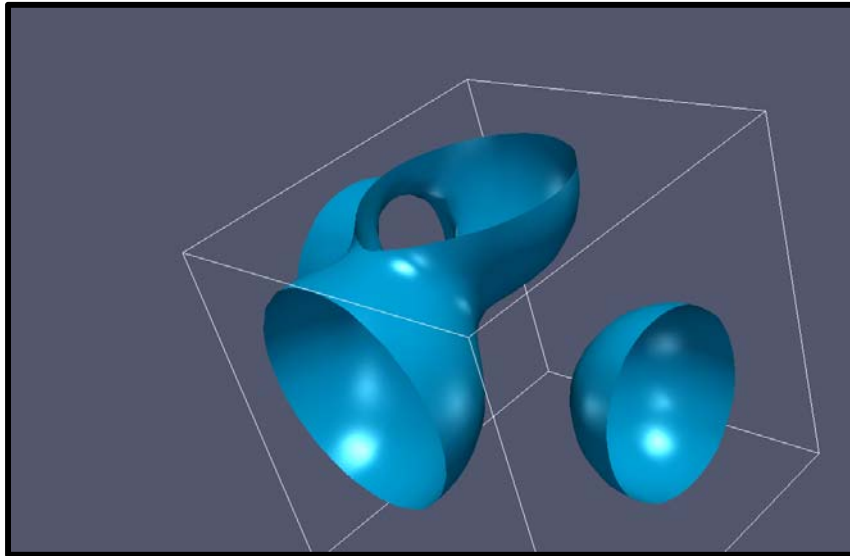
Animation in ParaView



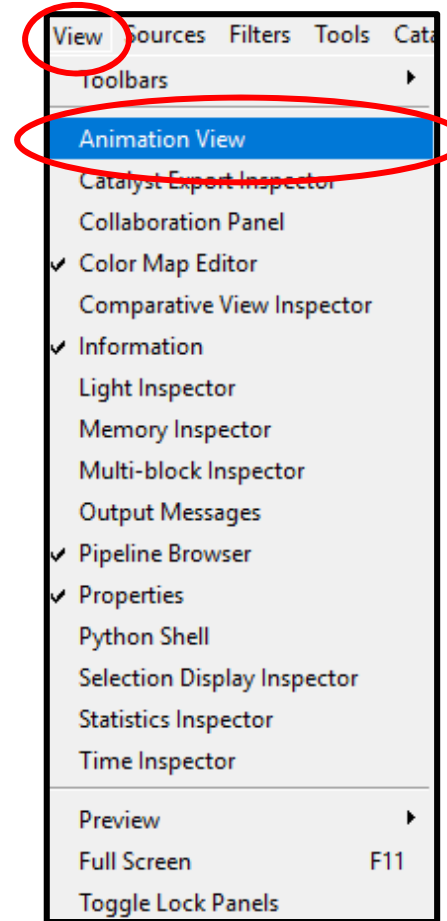
anim.pvsm

Animation in ParaView

Start with this:

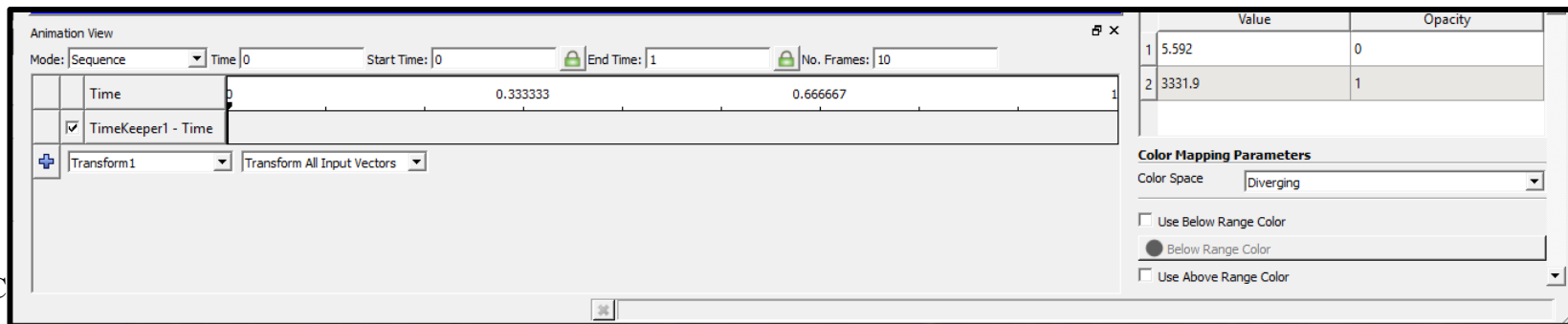


Select this:



anim.pvsm

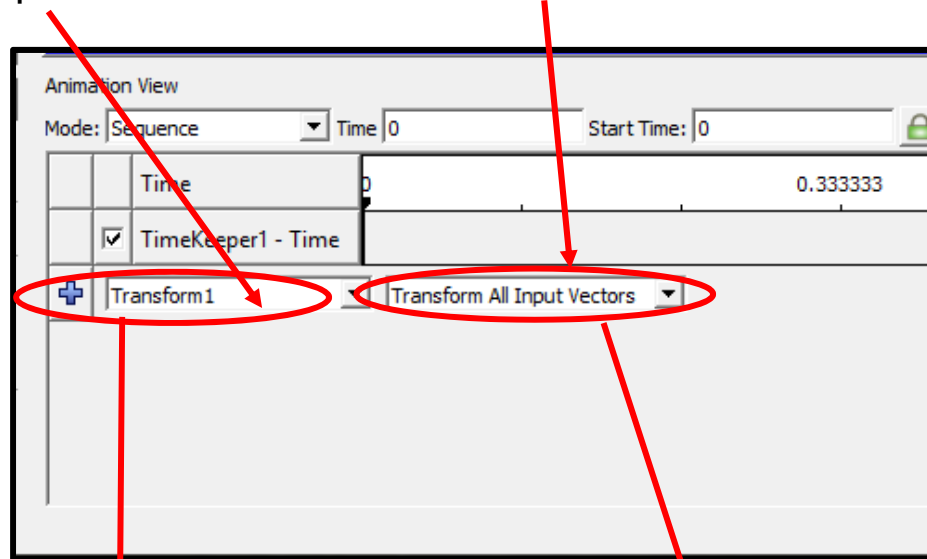
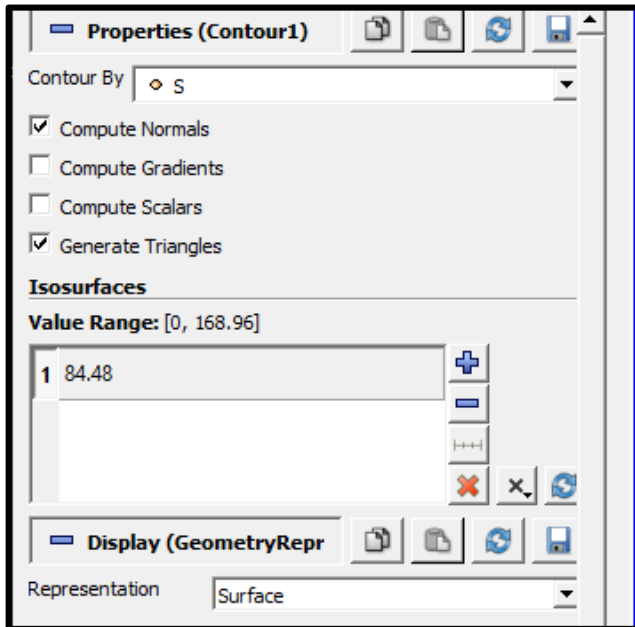
And this appears at the bottom:



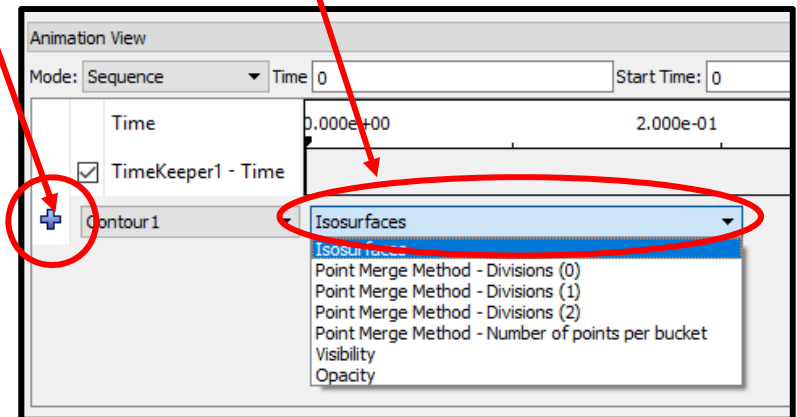
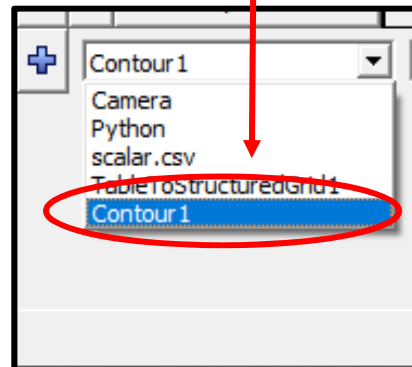
Animation in ParaView – Pick Something to Animate

Conveniently, the user interface for animation in ParaView looks a lot like the user interface for Comparative Visualization:

Select a Pipeline Element and a Parameter within that Element

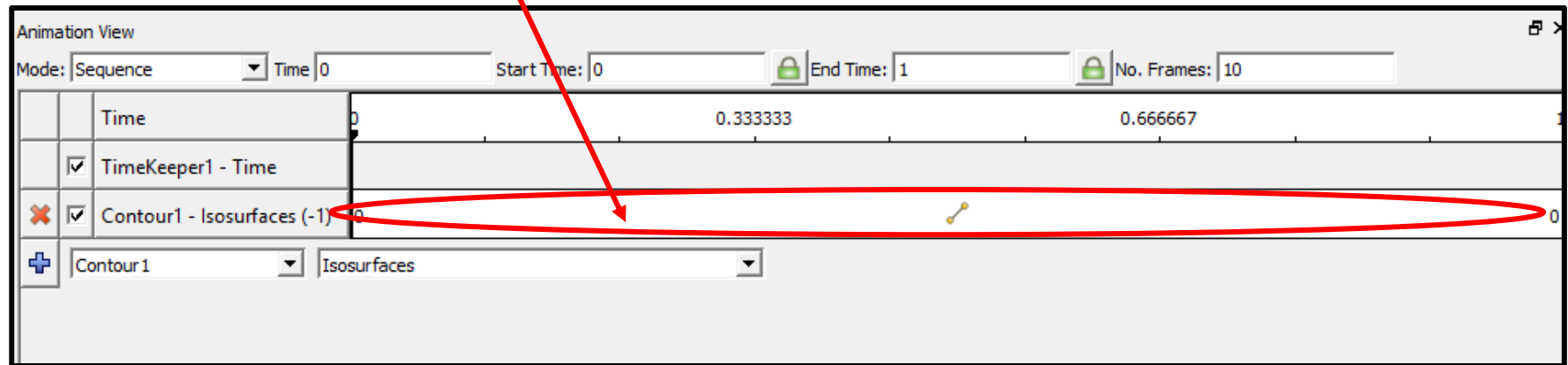


Hit the + when you are done

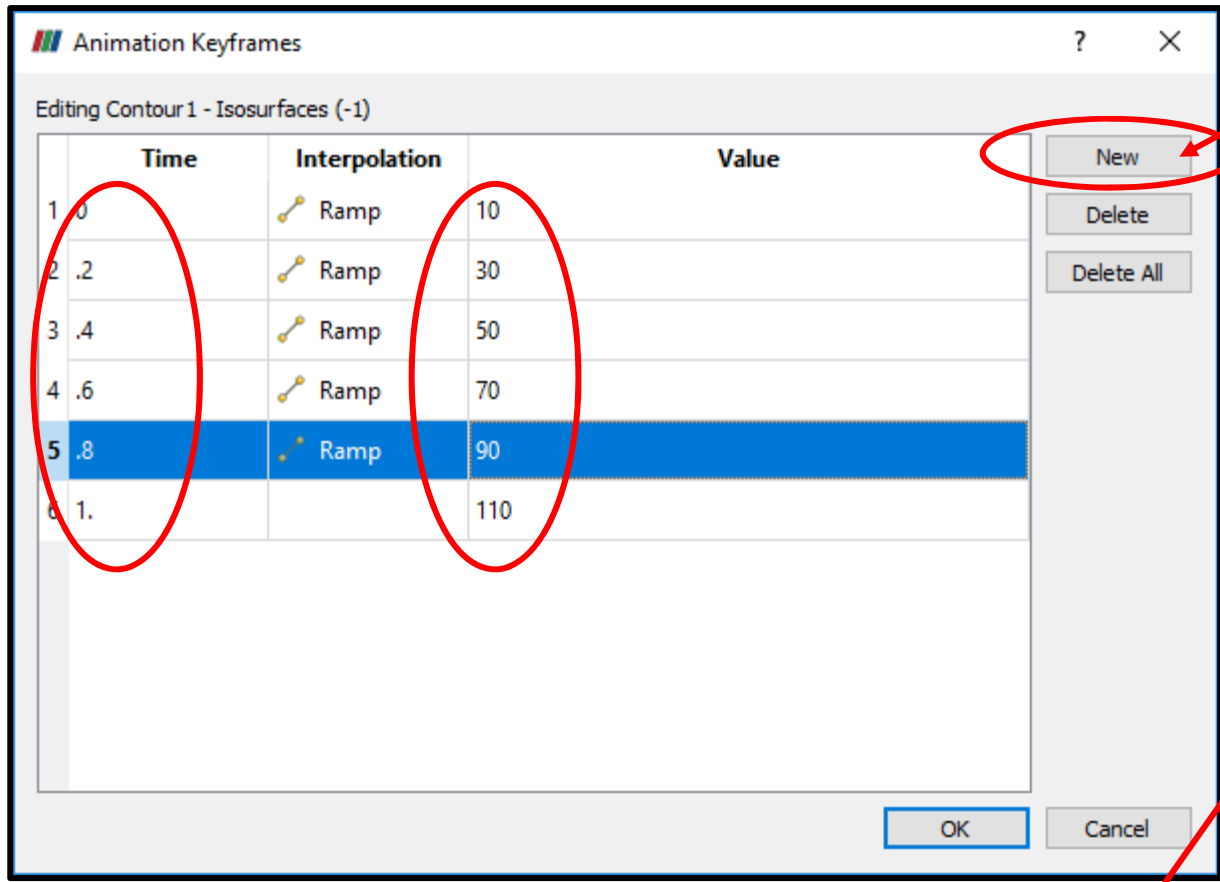


Animation in ParaView – Bring up a Keyframe Menu

The, double-click in the white space to the right of the Property-Parameter you selected:



Animation in ParaView – Setting Parameter Keyframes

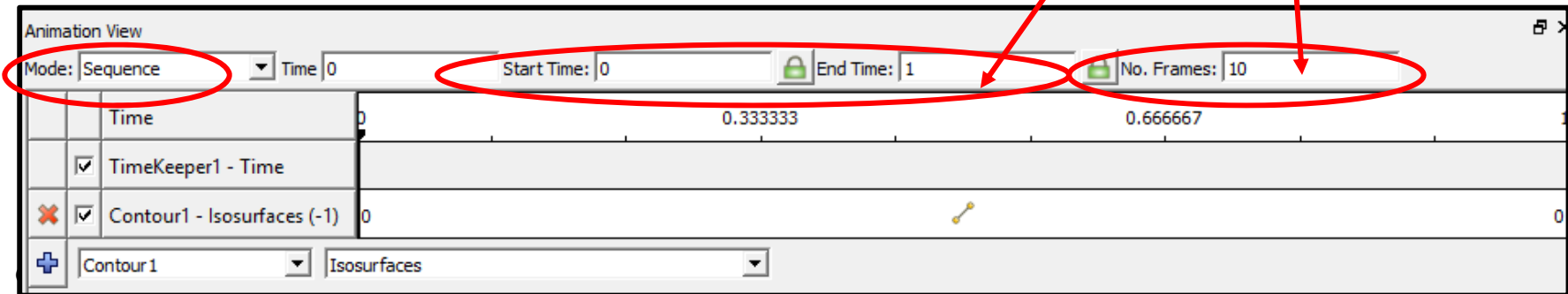


Click **New** to add a new row

The first column is the **Time**, the third column is the **Parameter** value at that time.

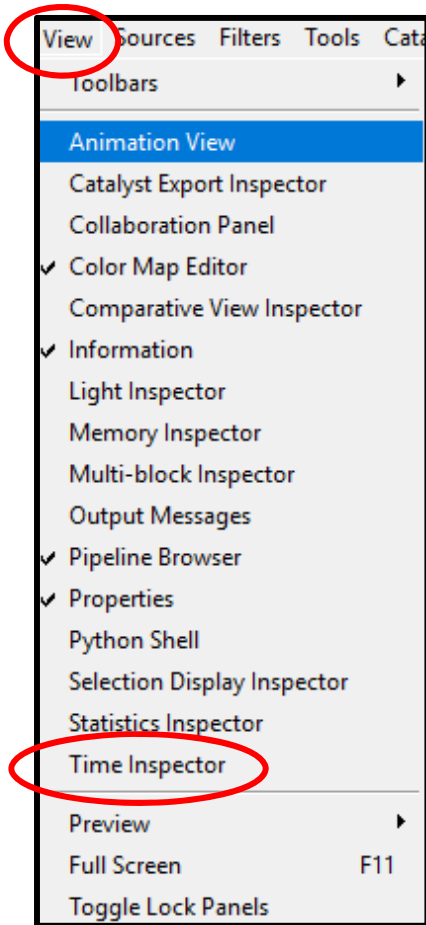
By default, the **Time** starts at 0. and goes to 1. – I just left it that way.

I did change the 10 frames to 1000 frames, though.

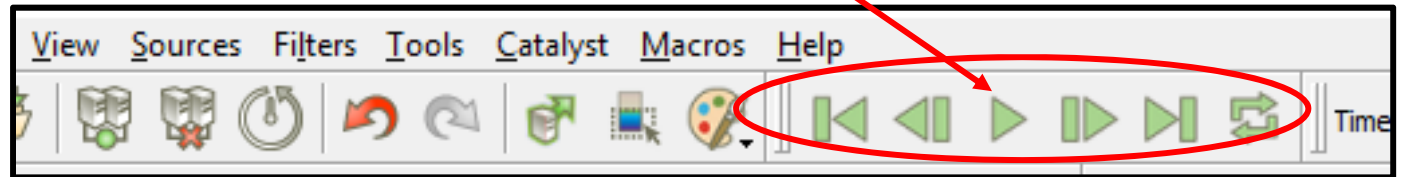


Animation in ParaView – the Time Inspector

Select this:

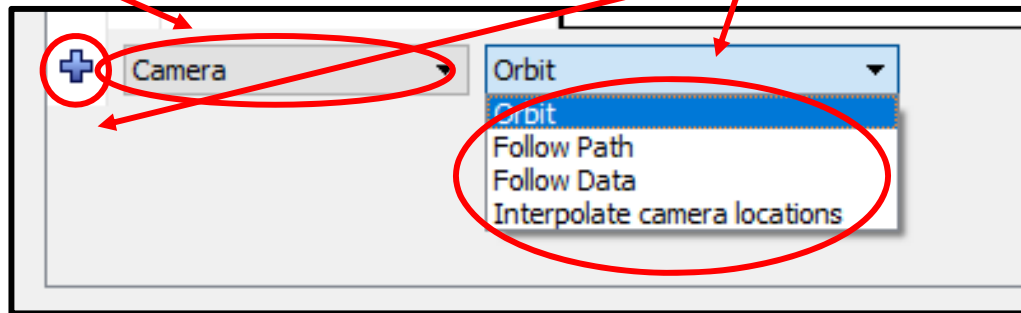


Unless you've been living in a cave, you know what to do with these – hit **Play**:



Animation in ParaView -- Animating the Camera

Here's how to animate the **Camera** – select **Camera** from the list of **Properties** and select one of these from the list of **Parameters**, then hit the **+**:



Orbit: animate the camera in a circle around a specific point

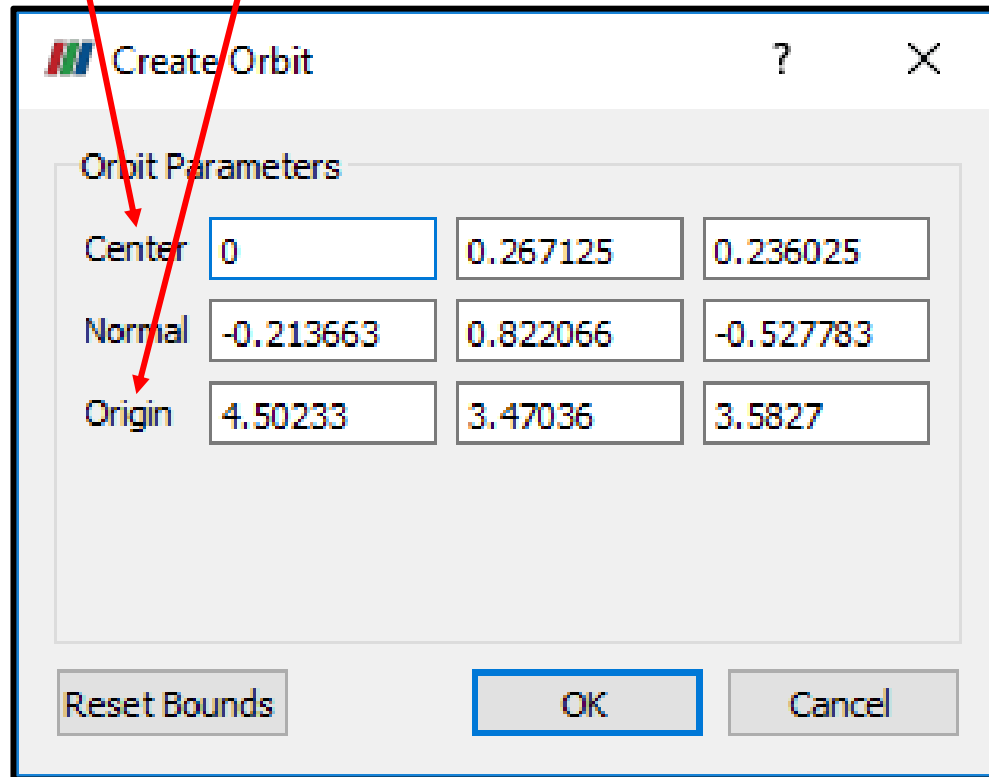
Follow Path: set keyframes for the camera position and look-at point

Follow Data: ??

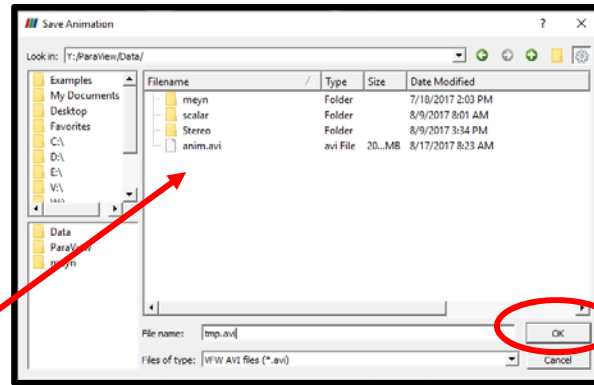
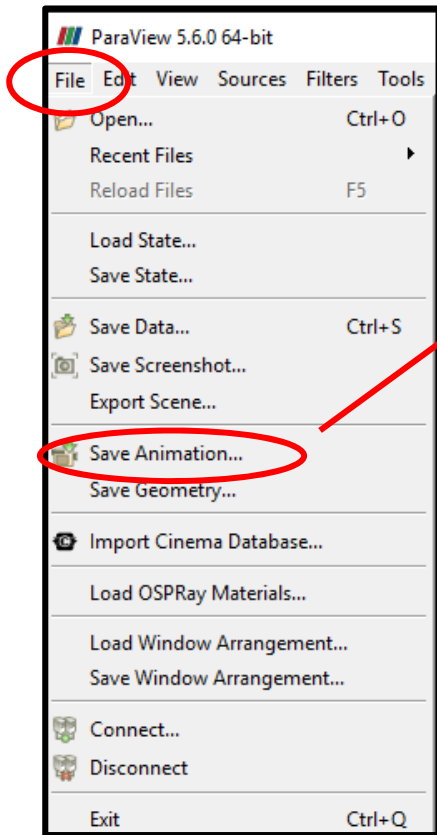
Interpolate camera locations: Manually specify keyframe camera locations

Animation in ParaView -- Orbiting the Camera

By default, the **Center** (look-at point) is the center of the data currently selected in the Pipeline. The Camera starts at its **Origin** and orbits at its current radius around that point.

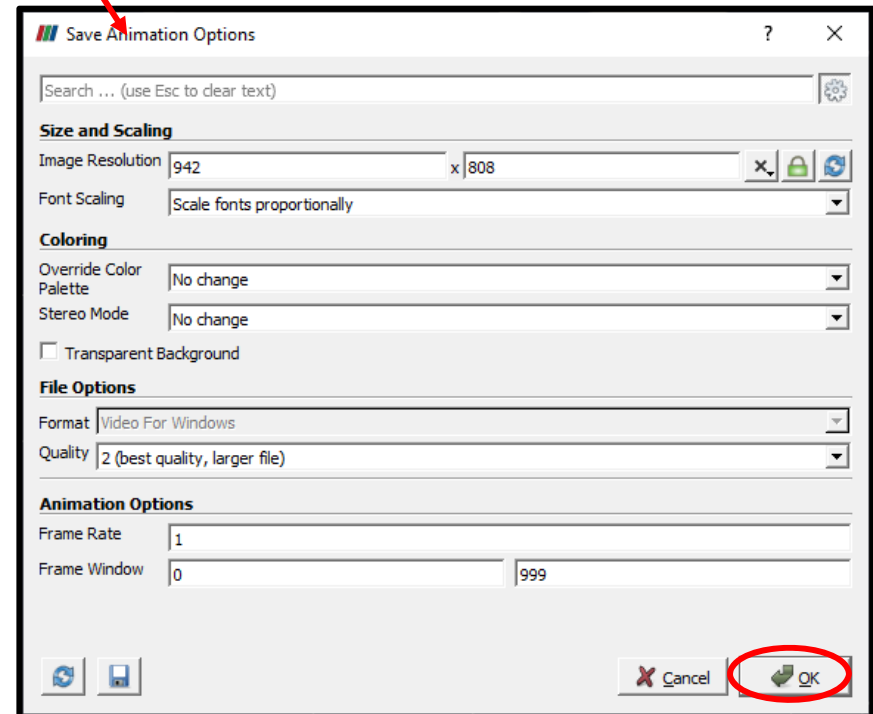


Saving the Animation



Clicking **Save Animation** brings up a file navigator dialog. You can save the animation in either **AVI** or **OGV** formats.

You can then set some animation parameters.

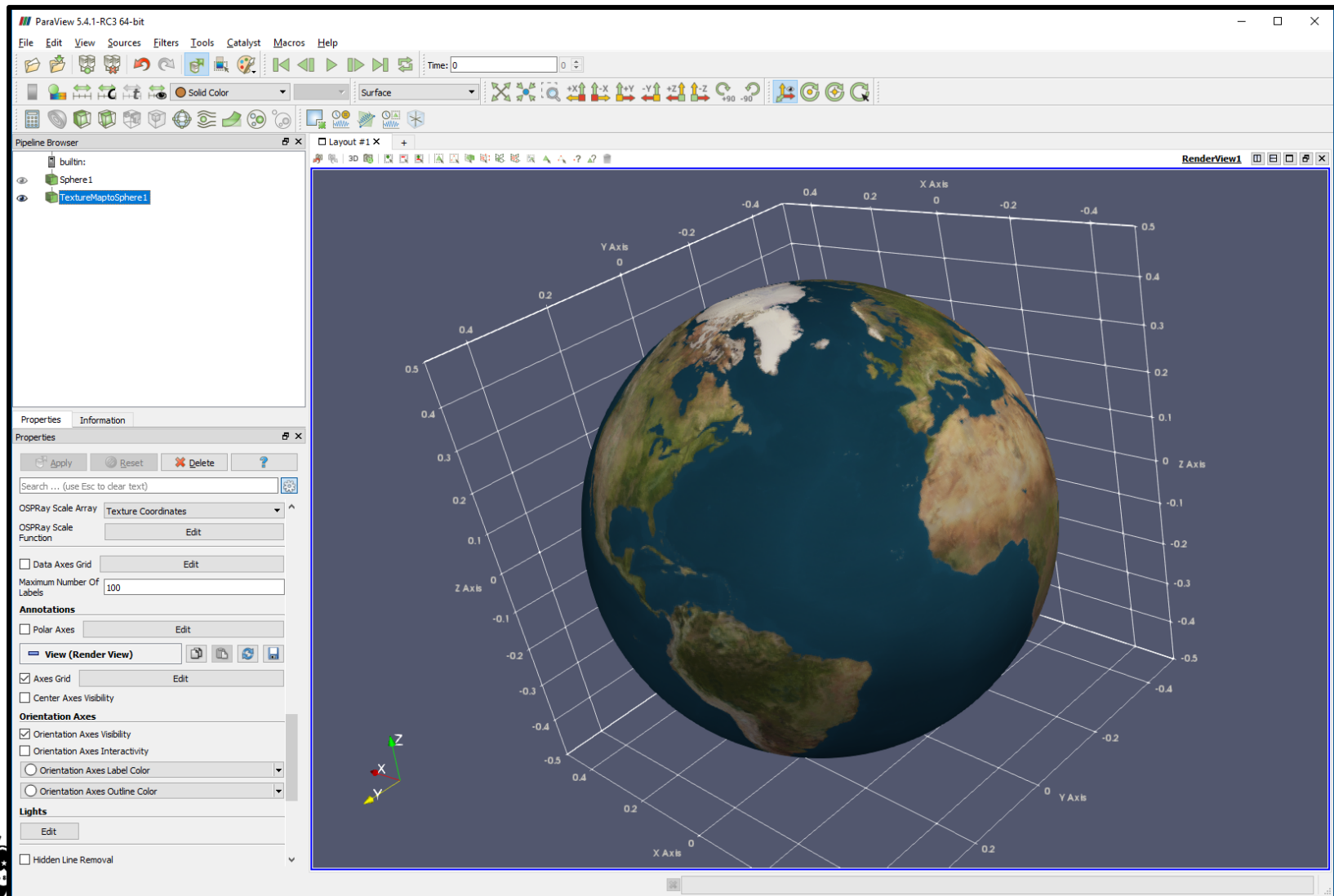


I haven't done an exhaustive study of this, but I can tell you that OGV files play in Firefox, Edge, and Chrome – but not in PowerPoint. AVI files play in PowerPoint. The OGV files are much smaller than the AVI files.

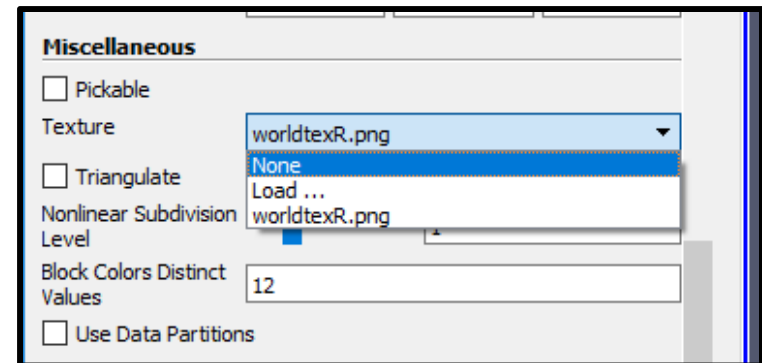
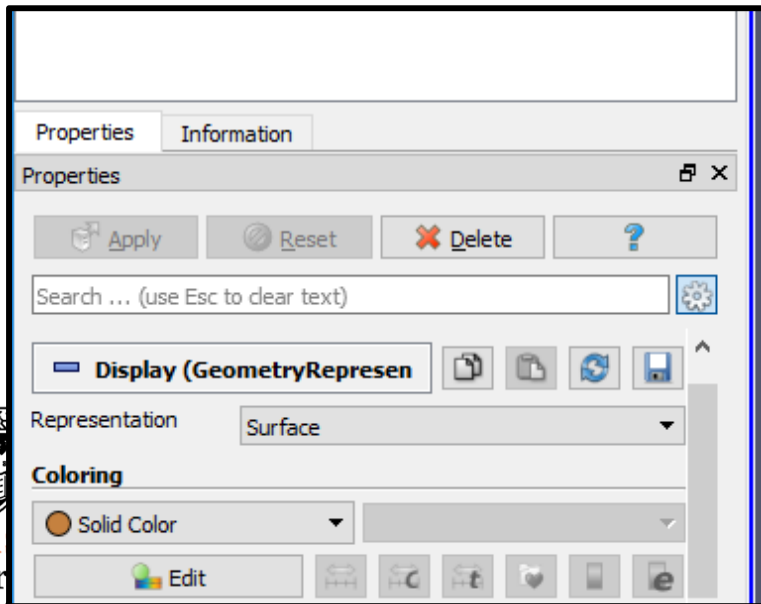
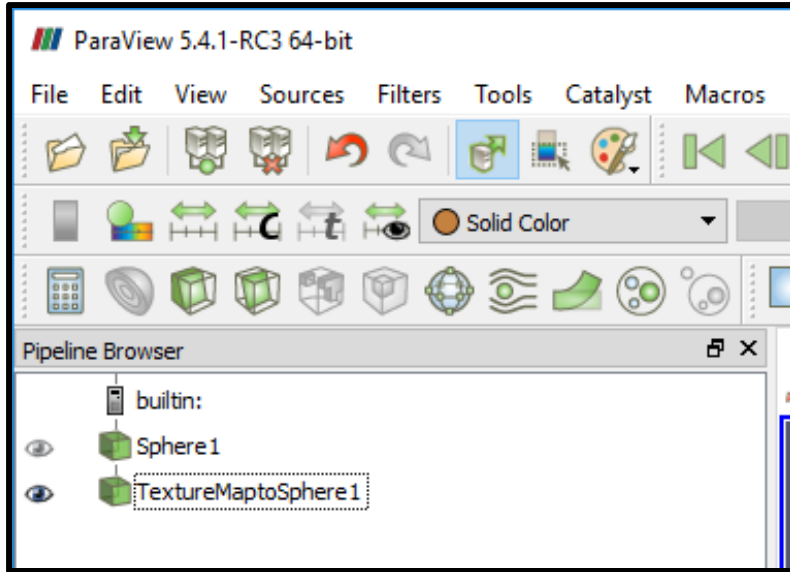
New Stuff

Still working on this...

Yes, you can map texture images to scene geometry

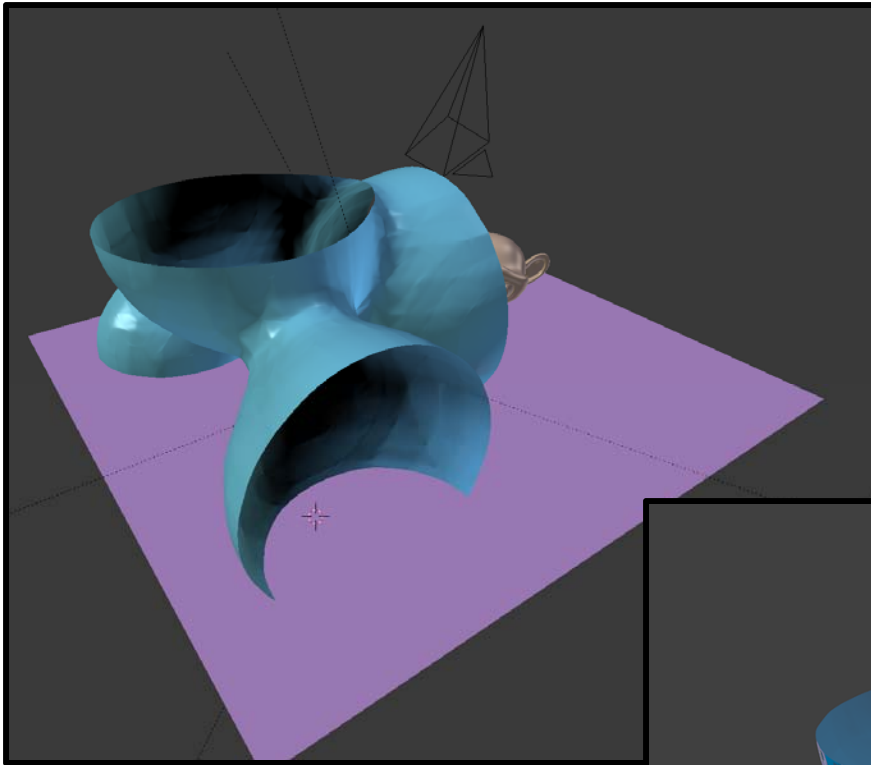


Yes, you can map texture images to scene geometry

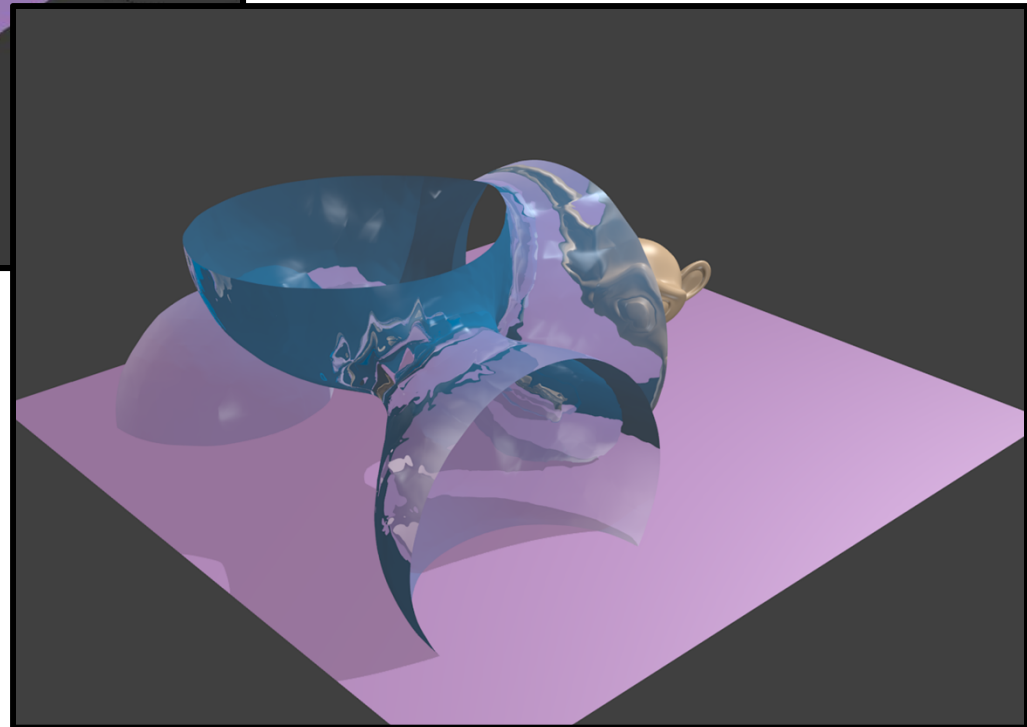


Working on mapping data in lat-long to a sphere with the texture on it

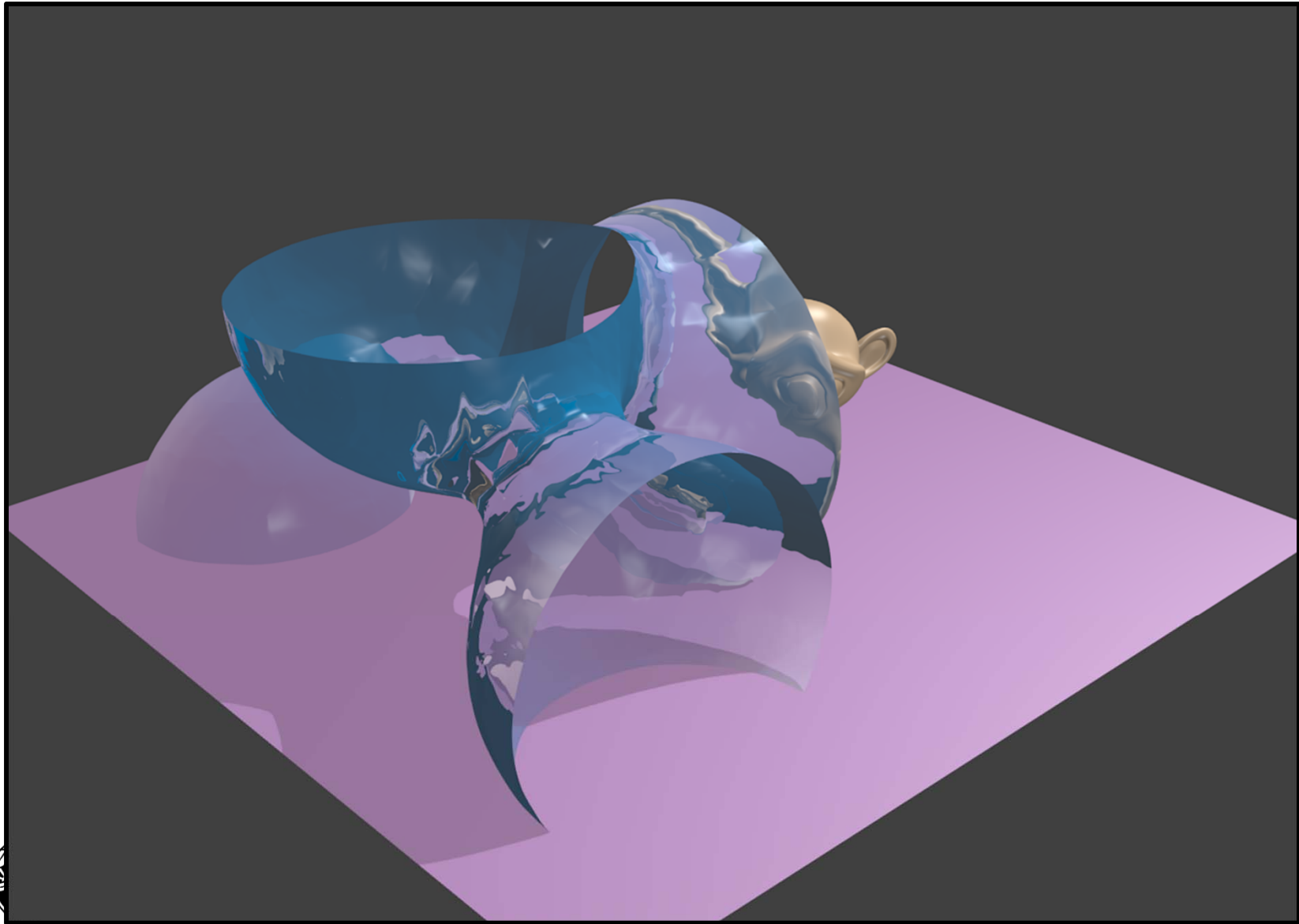
Looks like you can export the scene *geometry* (in this case to Blender) via X3D files



File → Export Scene



Looks like you can export the scene *geometry* (in this case to Blender) via X3D files



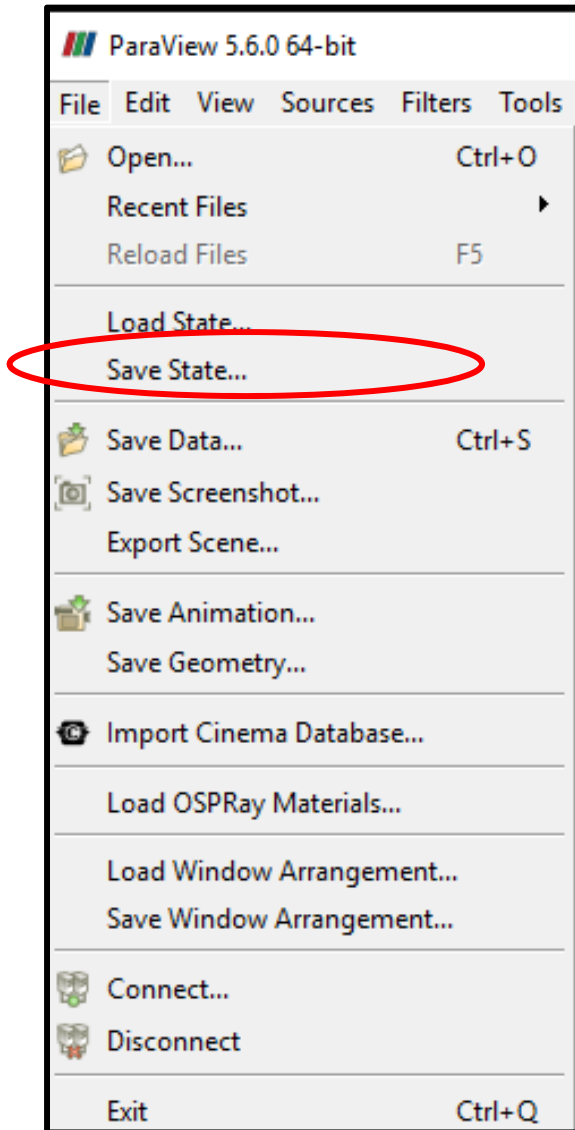
“Should” be able to create STL files from legal solid geometry (e.g., isovolumes) this way, too

ParaView Files

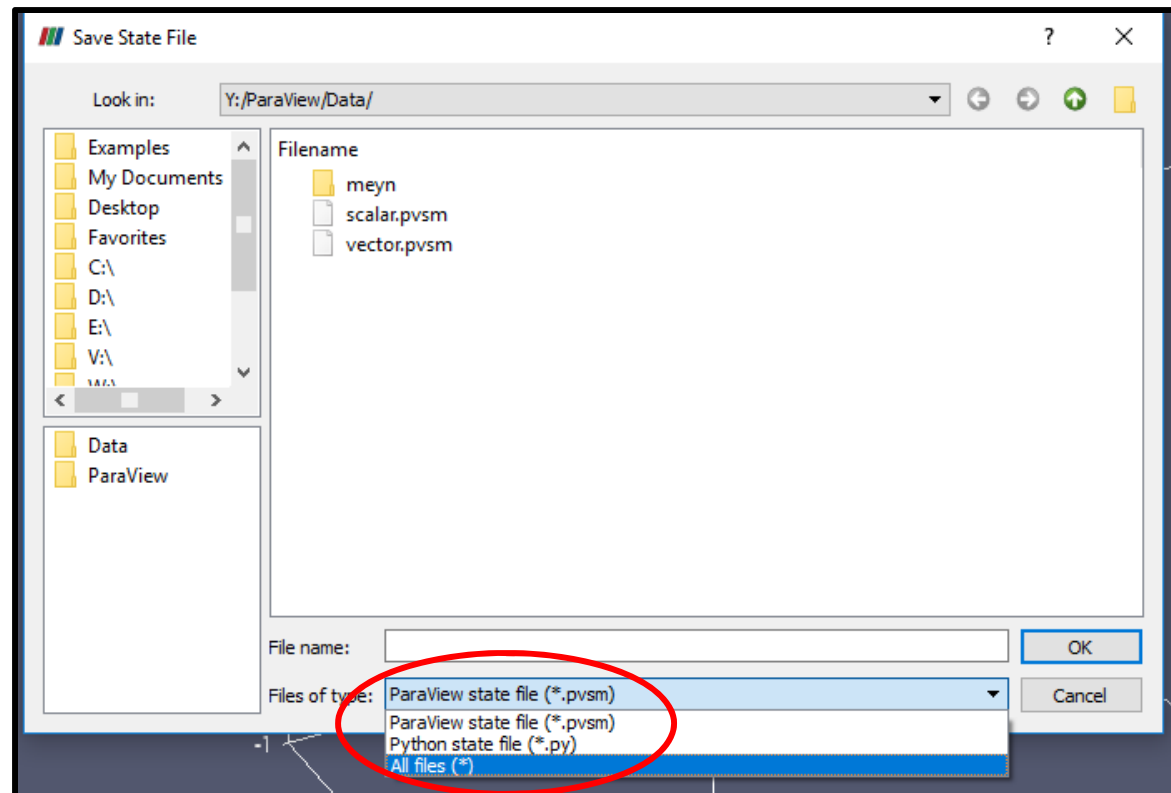


anim.pvsm
scalar.csv
scalar.pvsm
scalar.py
vector.csv
vector.pvsm
vector.py
terrain.csv
terrain.pvsm
terrain.py

Saving the State in Either a Native Format or as a Python Script



Oregon State
University
Computer Graphics



“State” means the entire state of the user interface (pipeline, properties, etc.). The data is not part of the state. When you read the state back in, ParaView will prompt you to show it what data file you want included with this state.

```

# state file generated using paraview version 5.1.2

# -----
# setup views used in the visualization
# -----

#### import the simple module from the paraview
from paraview.simple import *
#### disable automatic camera reset on 'Show'
paraview.simple._DisableFirstRenderCameraReset()

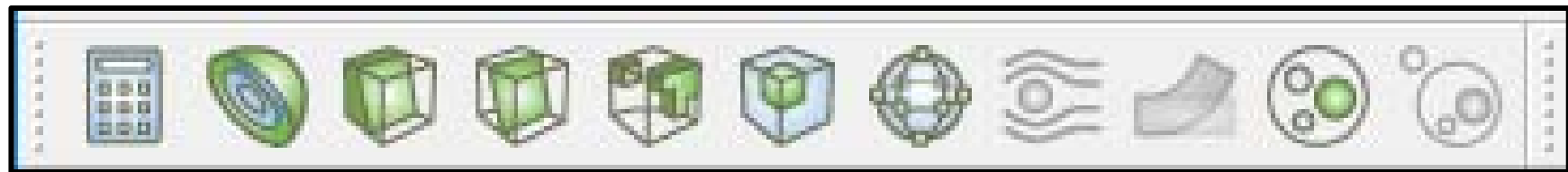
# Create a new 'Render View'
renderView1 = CreateView('RenderView')
renderView1.ViewSize = [1160, 912]
renderView1.AxesGrid = 'GridAxes3DActor'
renderView1.StereoType = 0
renderView1.CameraPosition = [3.76687547966054, 5.62637881722241, 4.44163730510425]
renderView1.CameraFocalPoint = [0.0241978424871666, -0.0474471125809167, 0.0405907851464954]
renderView1.CameraViewUp = [-0.384789750616684, -0.393723993522038, 0.834816305989173]
renderView1.CameraParallelScale = 1.73205080756888
renderView1.Background = [0.32, 0.34, 0.43]
# init the 'GridAxes3DActor' selected for 'AxesGrid'
renderView1.AxesGrid.Visibility = 1
# -----
# setup the data processing pipelines
# -----
# create a new 'CSV'
scalarcsv = CSVReader(FileName=['Y:\\\\ParaView\\Data\\scalar.csv'])
. . .

```

ParaView Menus you will use a Lot

ParaView Menus:

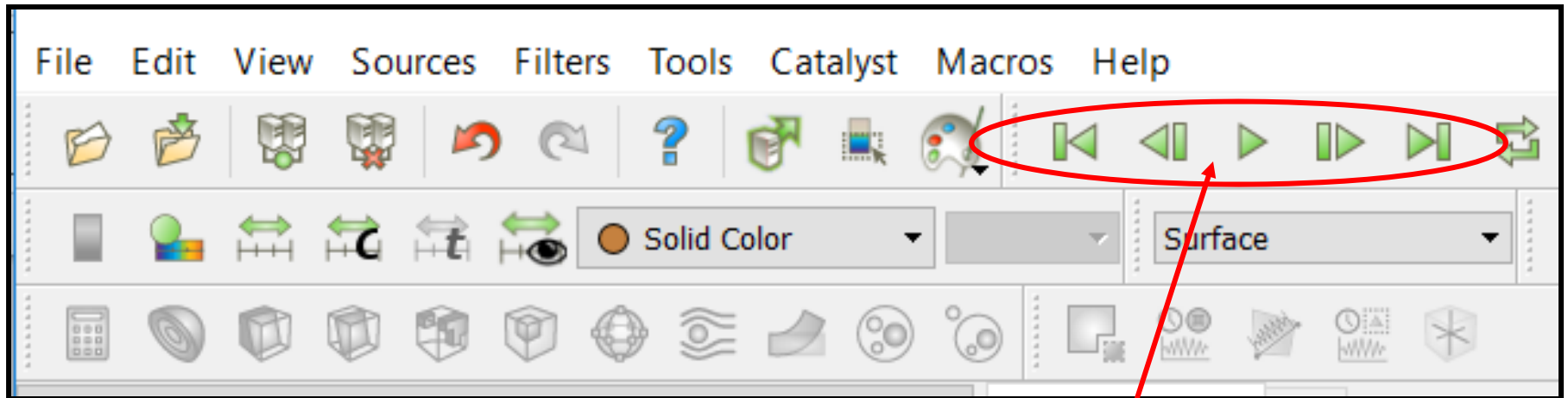
Commonly-used filters



Calculator Clip Threshold Glyph Warp by Vector Extract Level
 Contour Slice Extract Subset Stream Tracer Group Datasets

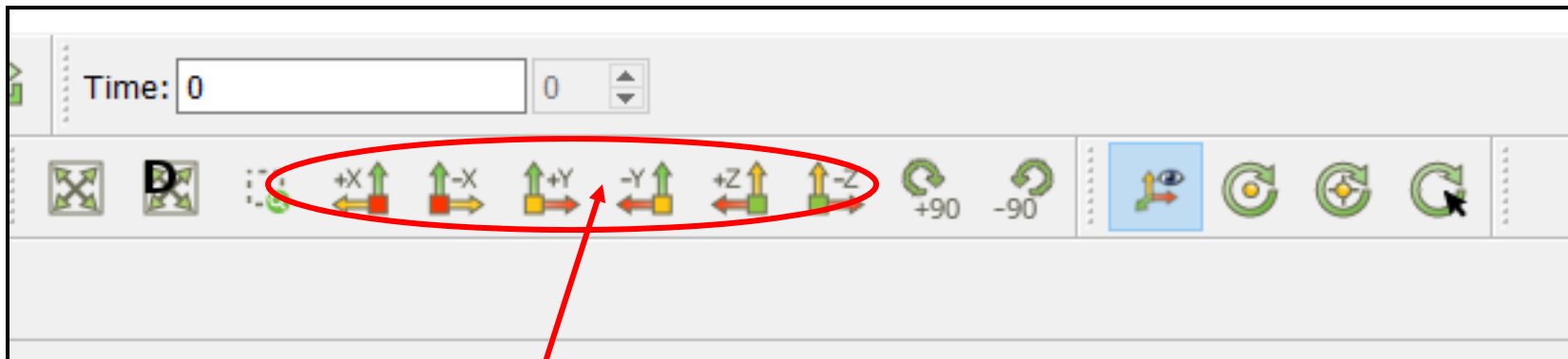
Some will be activated and some will be greyed-out, depending on what data you would be trying to use them for

ParaView Menus:



Animation Controls

ParaView Menus:



Directional Camera Positions

References

<http://cs.oregonstate.edu/~mjb/paraview>

Utkarsh Ayachit. *The ParaView Guide: A Parallel Visualization Application*, Kitware, 2015.

A free PDF of the book can be found here:

<https://www.paraview.org/paraview-guide/>

The ParaView tutorial:

https://www.paraview.org/Wiki/The_ParaView_Tutorial

