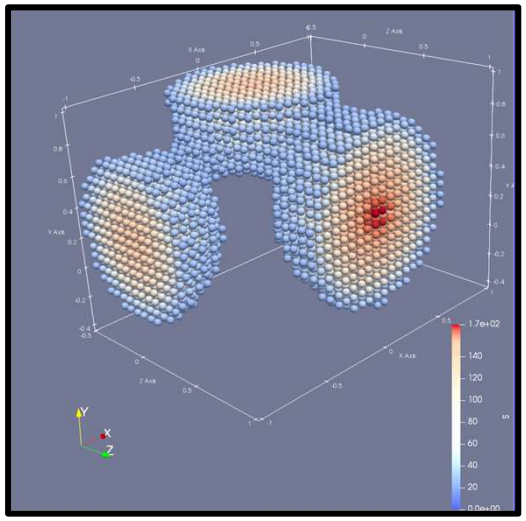
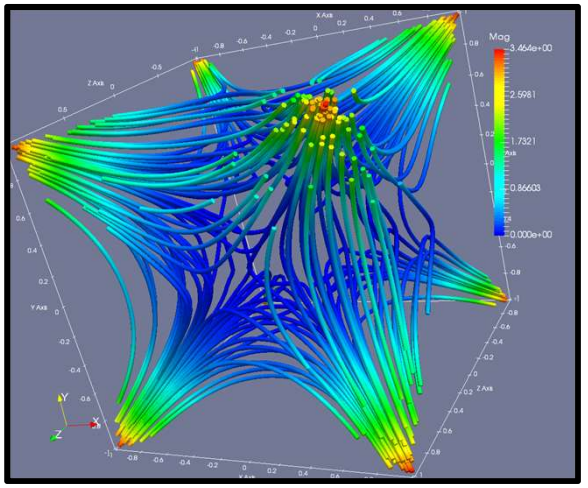
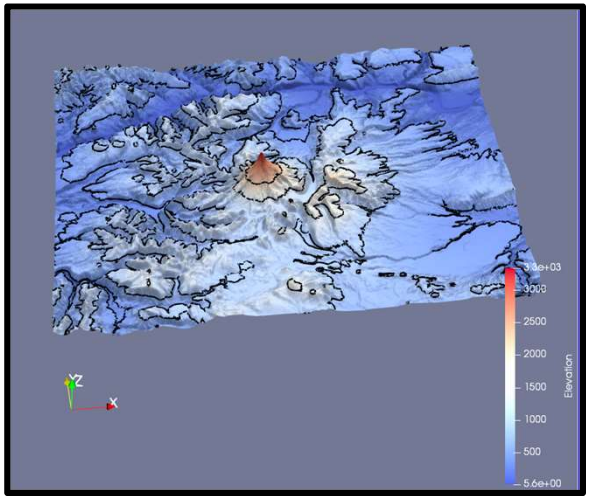
ParaView

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







Oregon State University

Mike Bailey

mjb@cs.oregonstate.edu

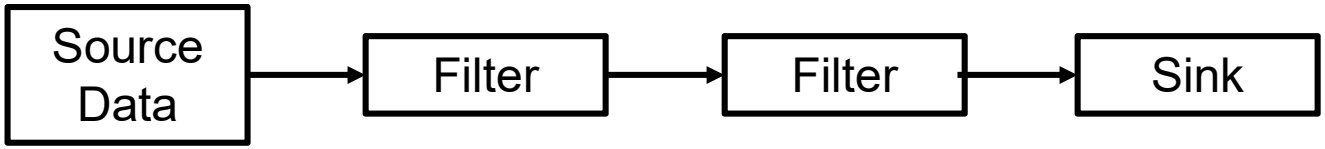
Oregon State University
Computer Graphics

What is ParaView?

ParaView is a free interactive visualization package produced by **KitWare,** **https://www.kitware.com/**

It is built upon VTK, the Visualization Toolkit, https://vtk.org/

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.

Oregon State University
Computer Graphics

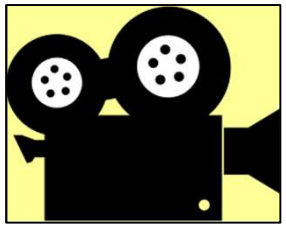
These notes have been written against ParaView version 5.11

http://www.paraview.org

Click here to download ParaView

In these notes, what do these icons mean?

scalar.csv



scalar.ogv

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

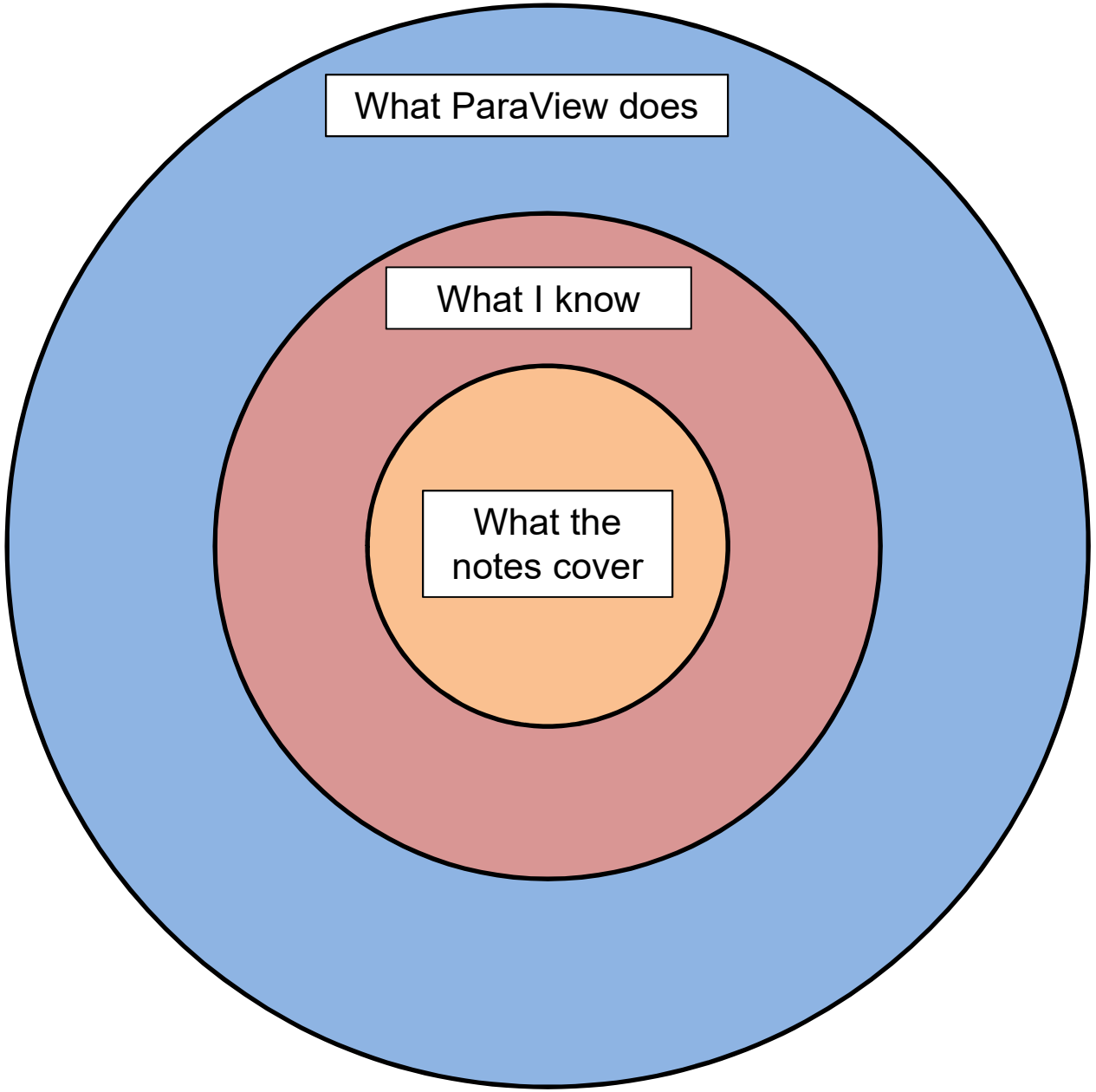
http://cs.oregonstate.edu/~mjb/paraview

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.

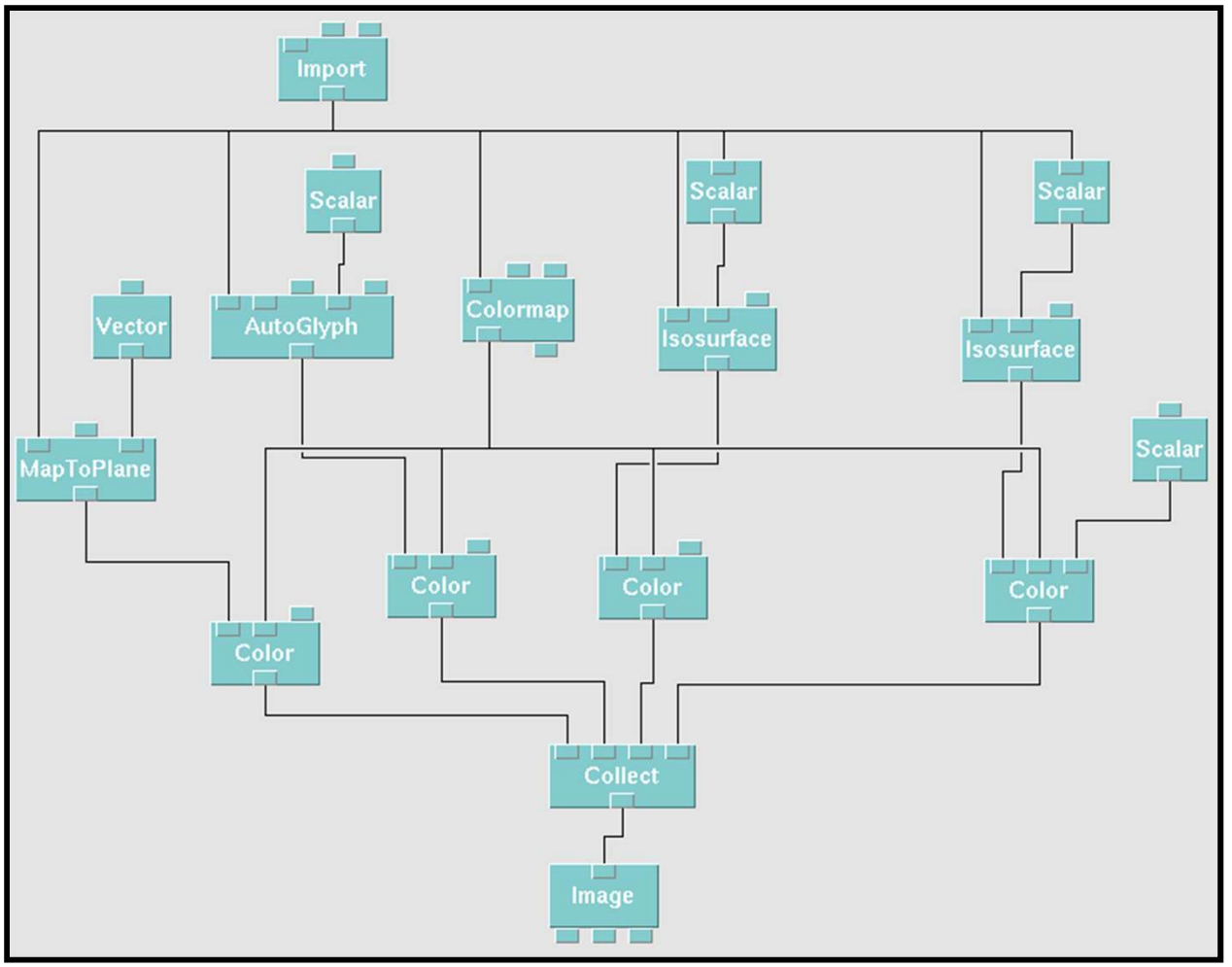
You can play an .ogv movie file right from your browser so that you can see how these examples look without having to run ParaView at all.

A warning about me and the Notes



Screen Layout, Color Editor, and 3D Display

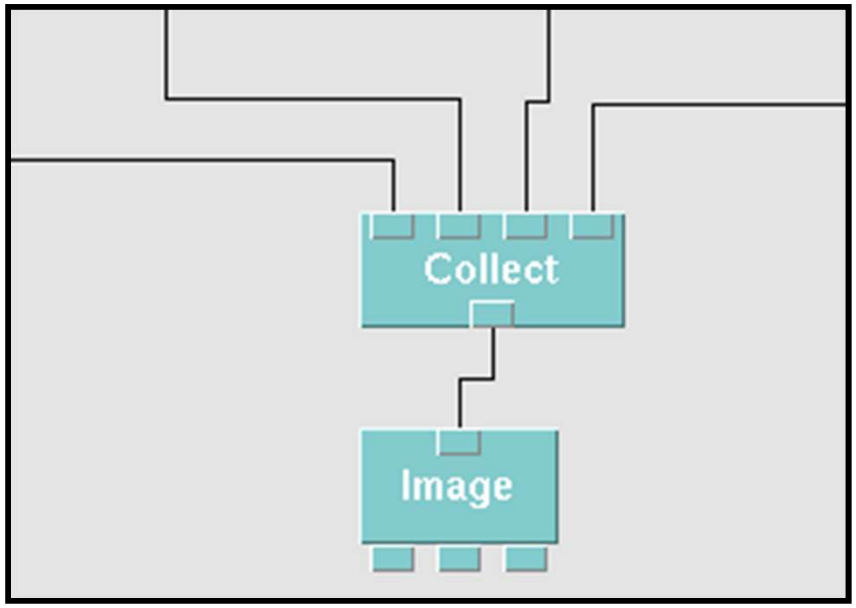
In the Beginning, there was OpenDX …



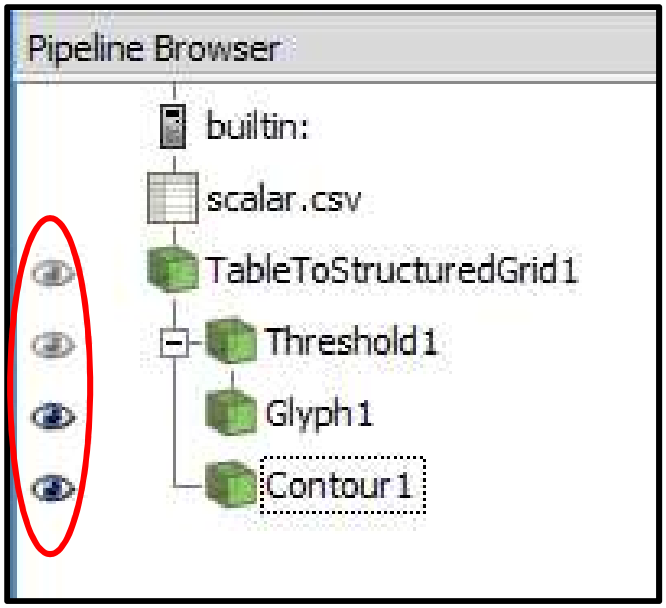
"DX" stands for "IBM Data Explorer". Like the name implies, it let you ***explore***! But, once it became "open" instead of commercial, all reliable support went away. Also, it required a lot of screen area just to hold the block diagram.

Fan-In to the Full Scene

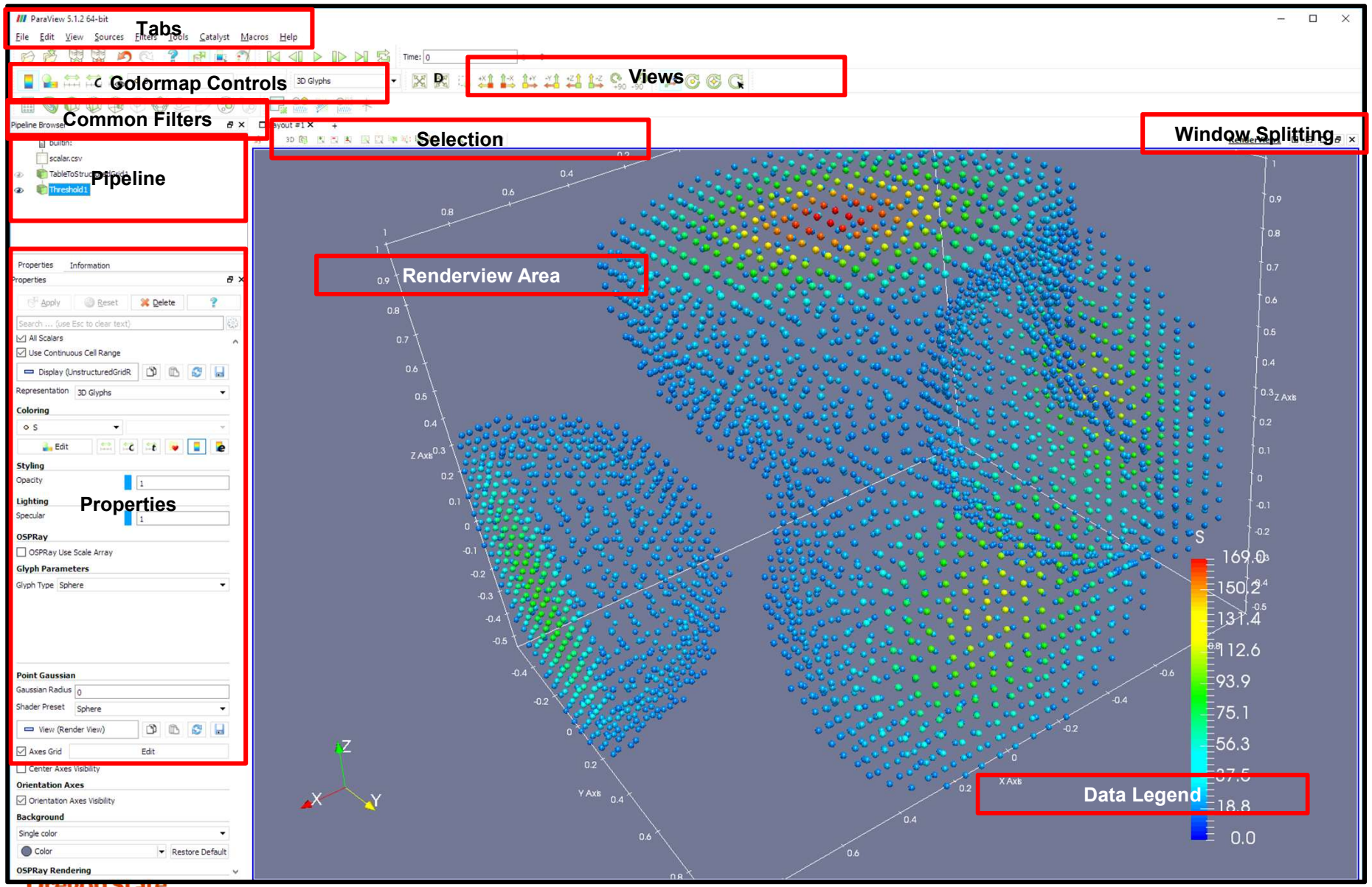
OpenDX:



ParaView:

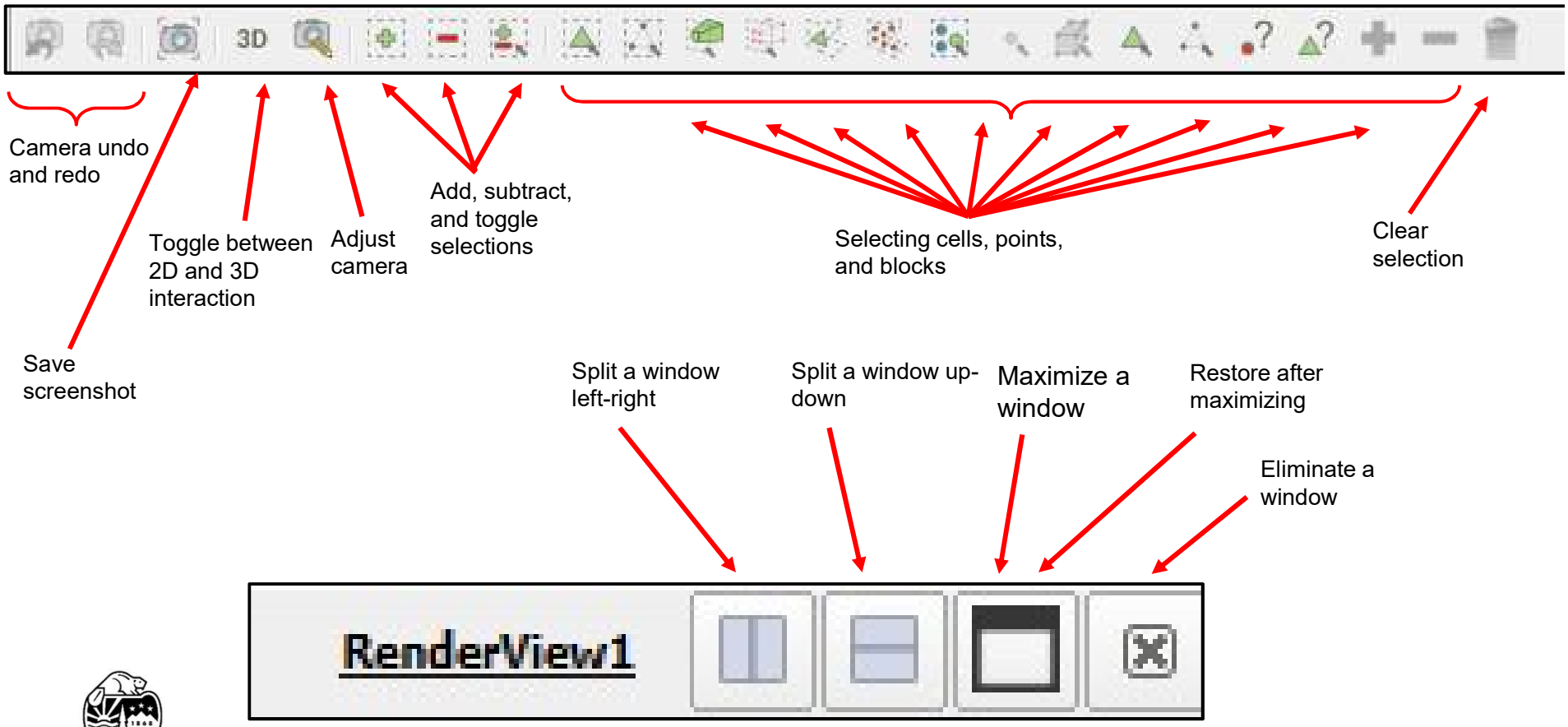


ParaView Screen Layout



ParaView Menus

Window Layout Menu



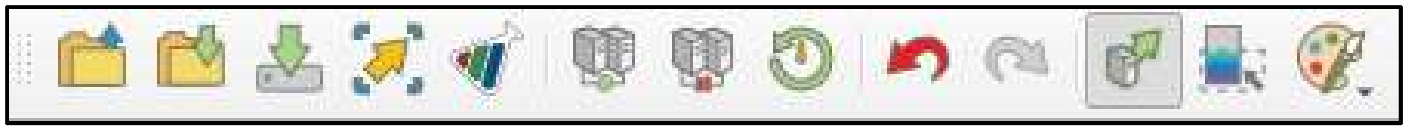
Commonly-used Filters Menu

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

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



Open

Save

Save State

Save Extracts

Save Catalyst State

Connect

Disconnect

Reset Session

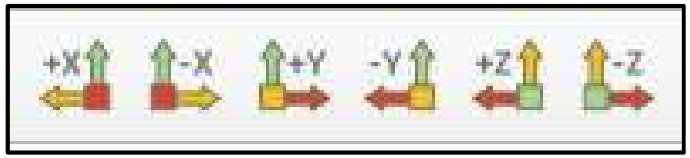
Undo

Redo

Apply Changes Automatically

Find Data Matching

Load Color Palette



Directional Camera Positions

ParaView Menus

Color Legend Visibility

Use Separate Color Map

Rescale to Custom Data Range

Rescale to Visible Data Range

Graphical Representation

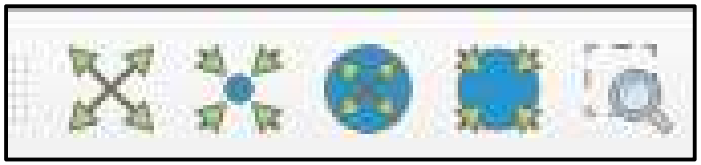
Edit Color Map

Rescale to Data Range

Rescale to Data Range over all Time Steps

What to Color Based On

ParaView Menus



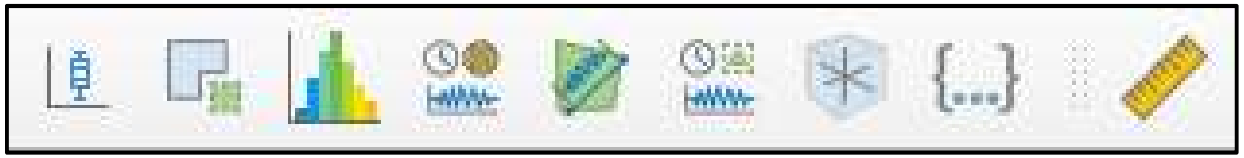
Reset

Zoom to Data

Reset Camera Closest

Zoom Closest to Data

Zoom to Box



Compute Quantities

Extract Selection

Histogram

Plot Variables Over Time

Plot Over Line

Plot Selection Over Time

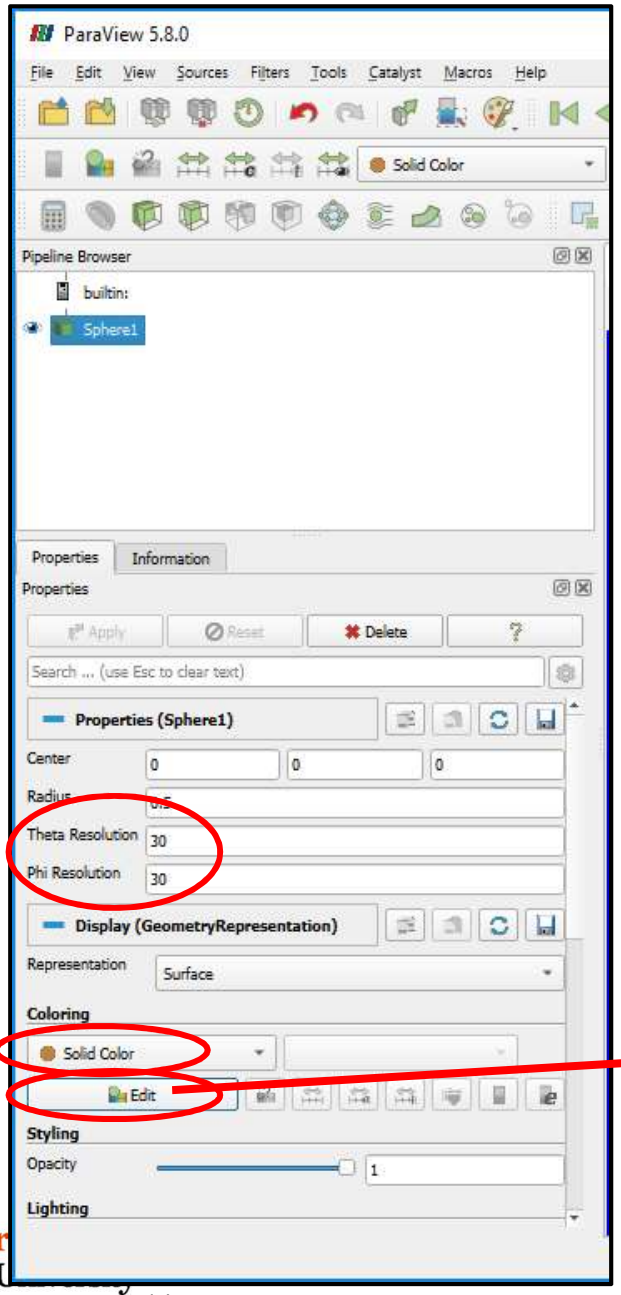
Probe Location

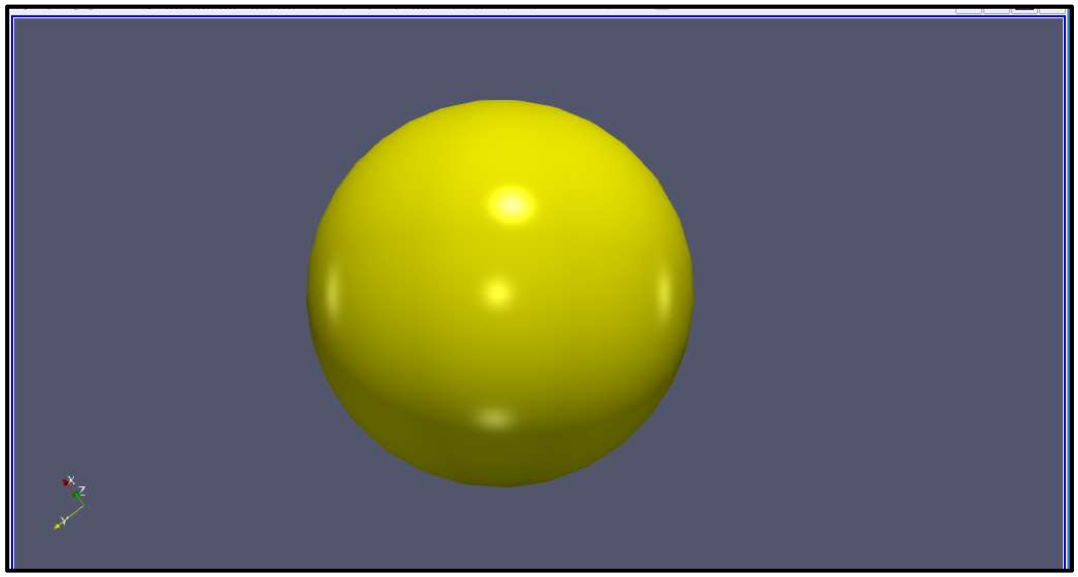
Programmable Filter

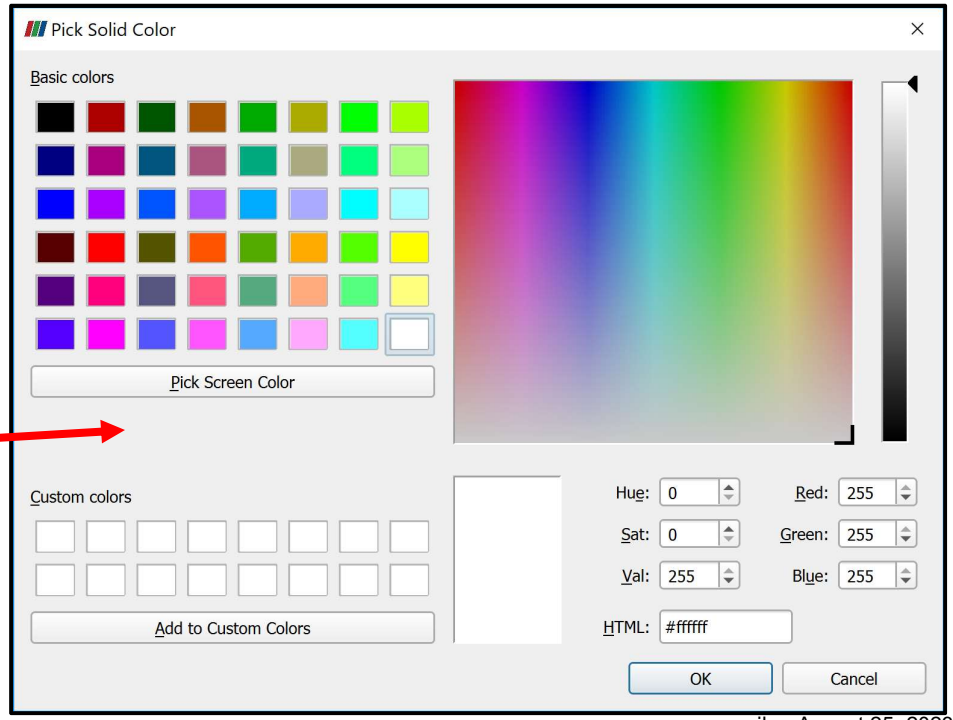
Ruler

3D Scene Manipulation

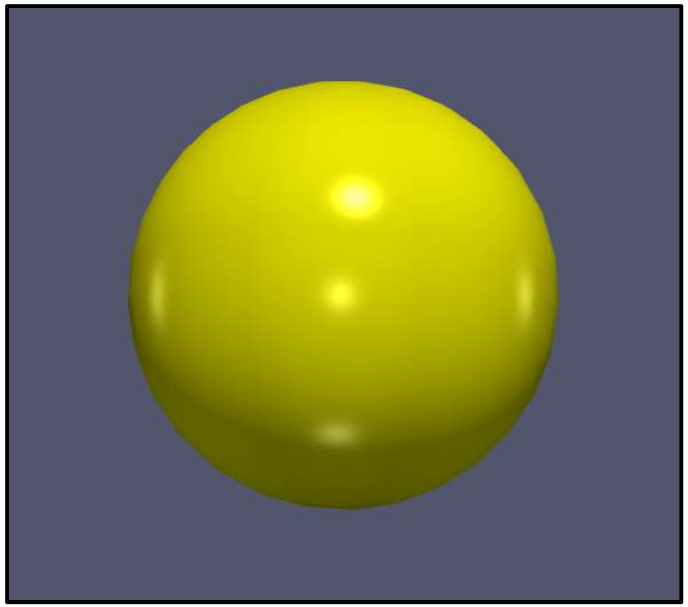
Sources → Geometric Shapes → Sphere





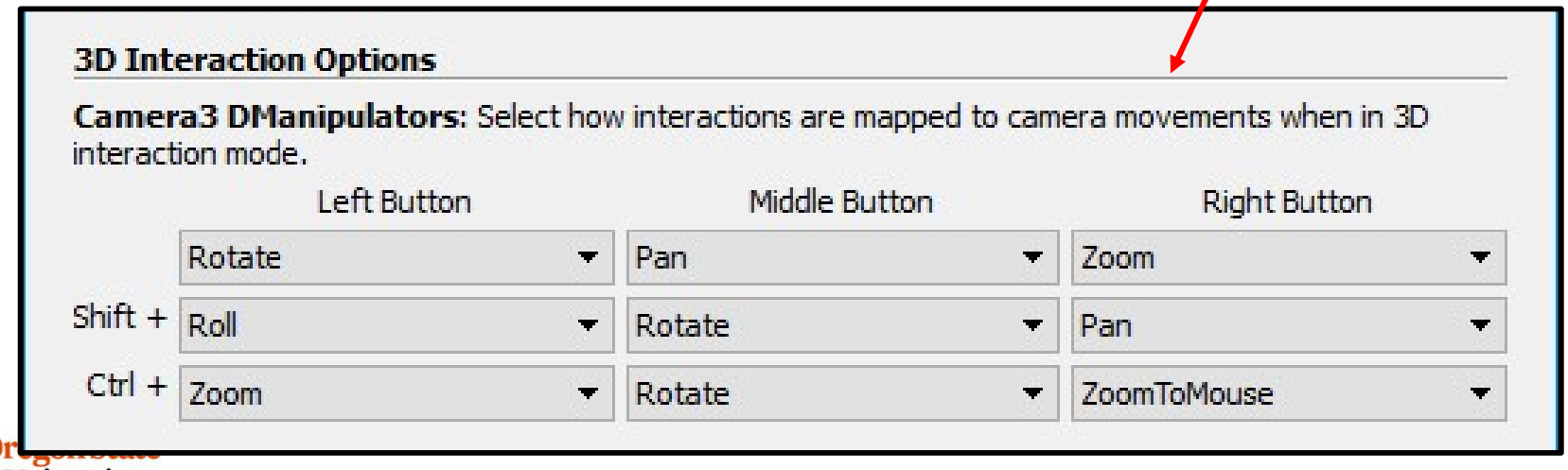


3D Scene Manipulation

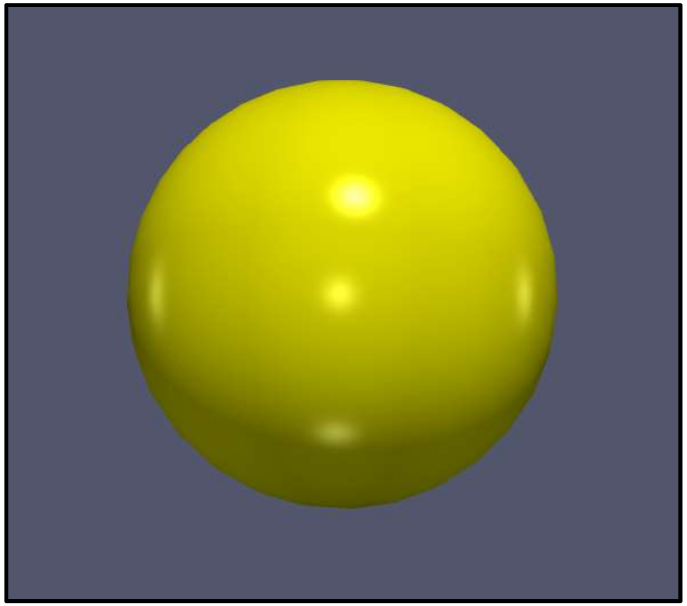


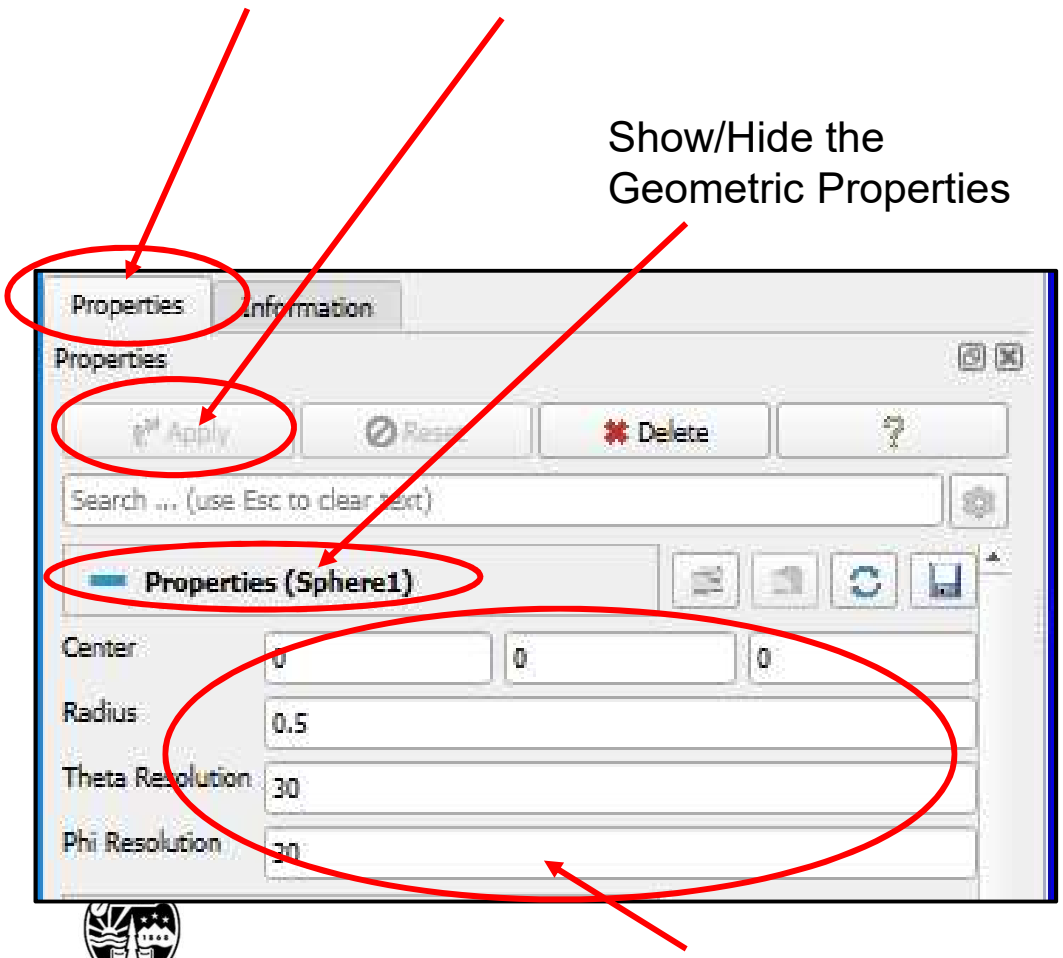
By default, these are the 3D Scene Manipulators (plus the mouse wheel, which is also a Zoom):

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

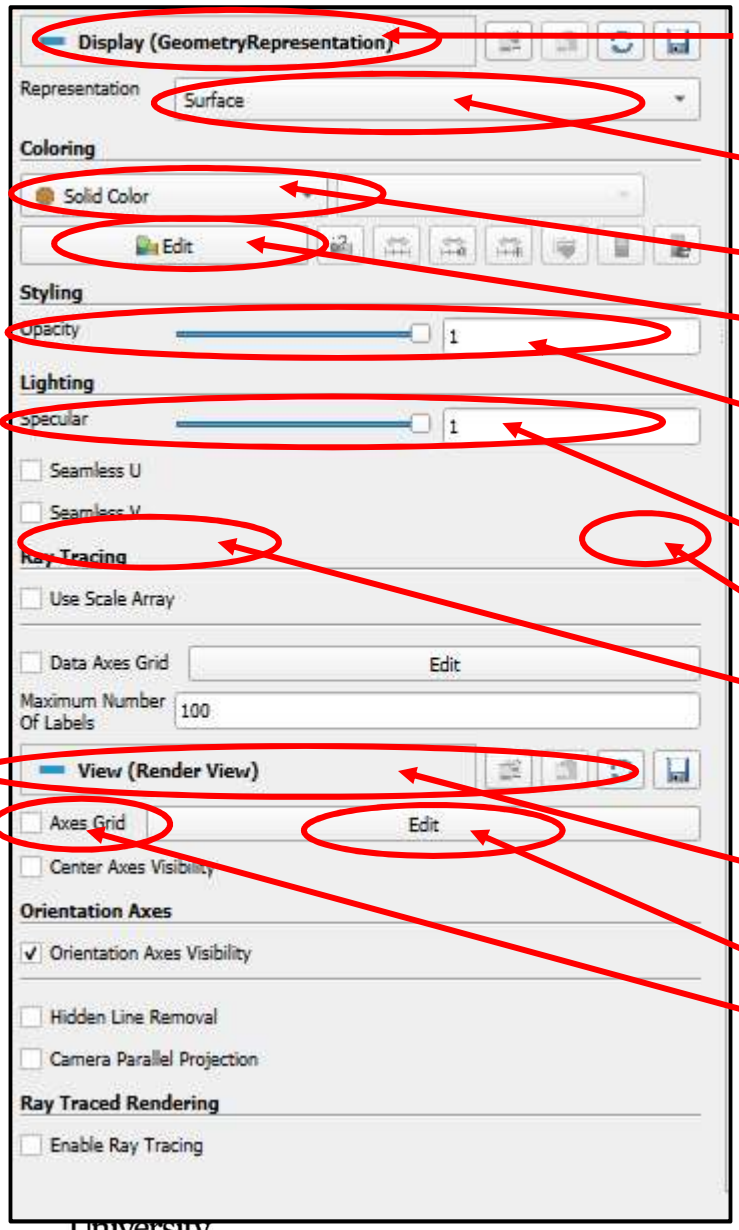


You Can Change Sphere Properties





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

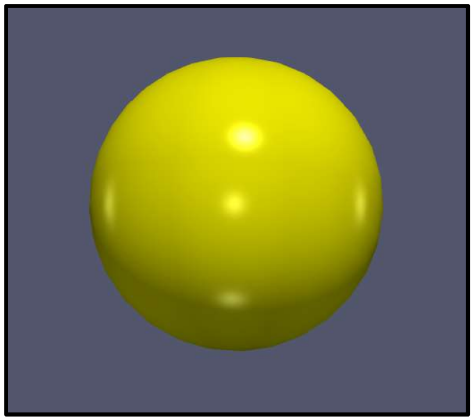
Bring up other Features to Color-Edit

Edit the Edge Color

Show/Hide the Render View Properties

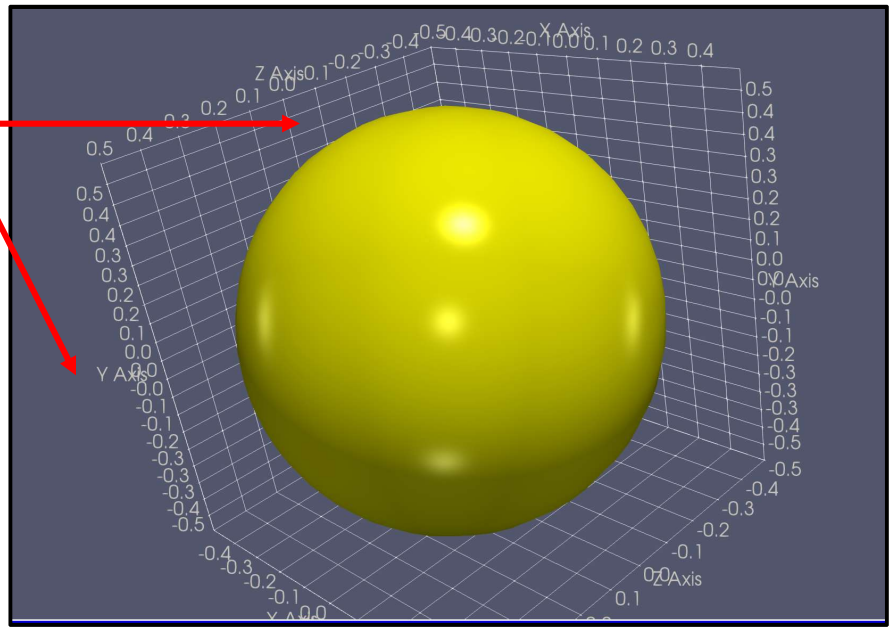
Edit the Features of the **Axes Grid**

Turn on/off the **Axes Grid**



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.



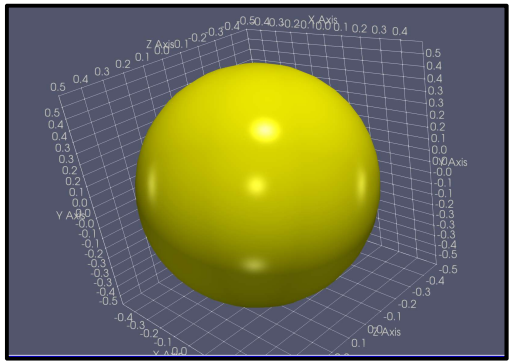
Editing the Axes Grid

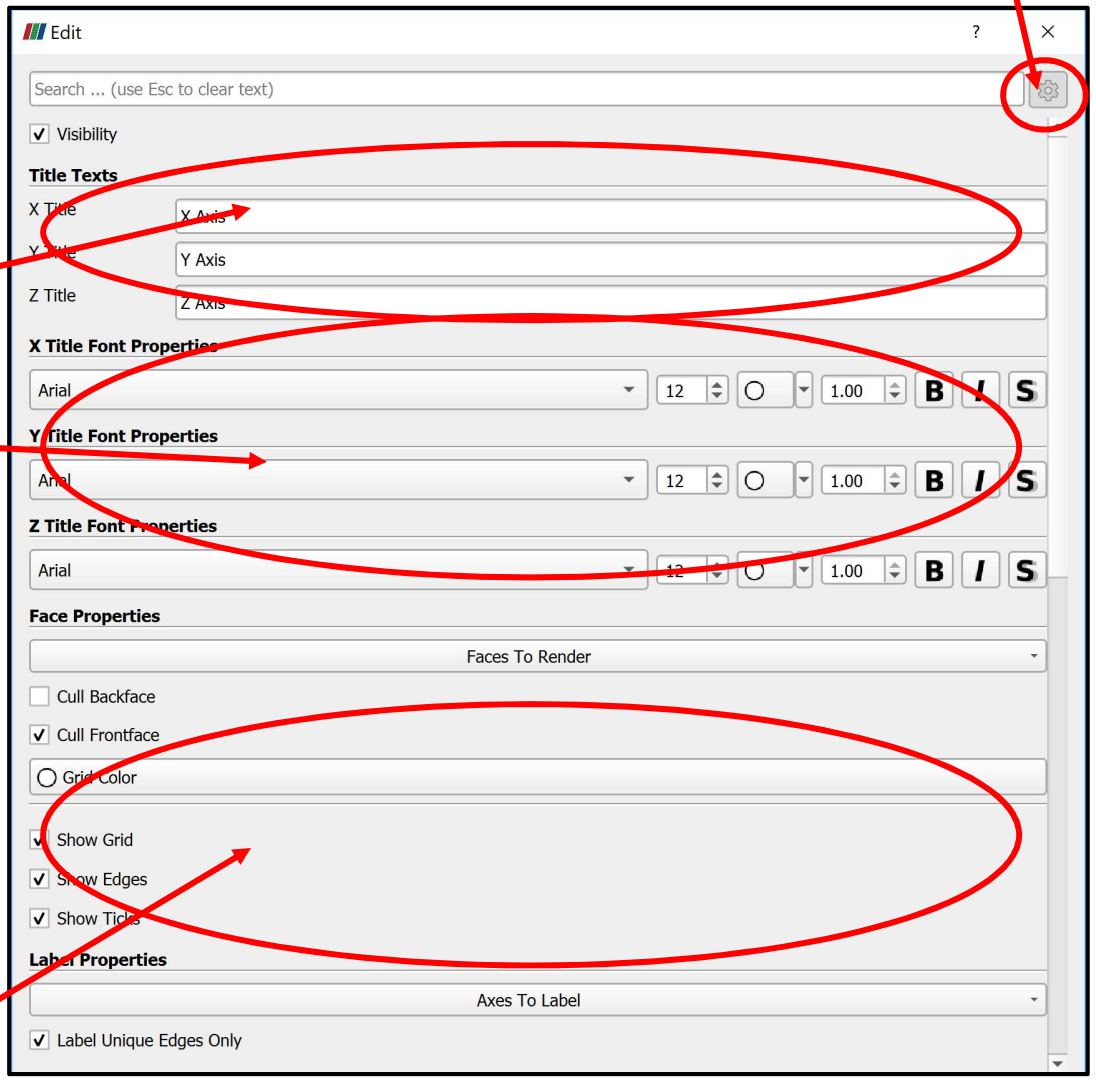
Show more/less options

Titles for the axes

Title font styles

Number label font styles

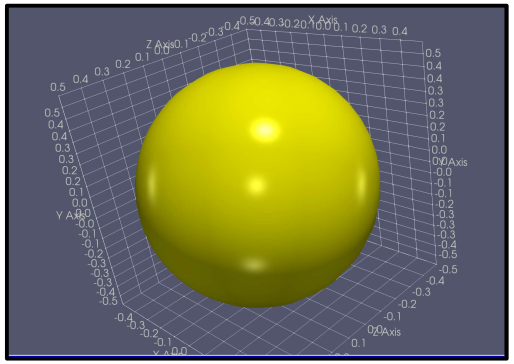




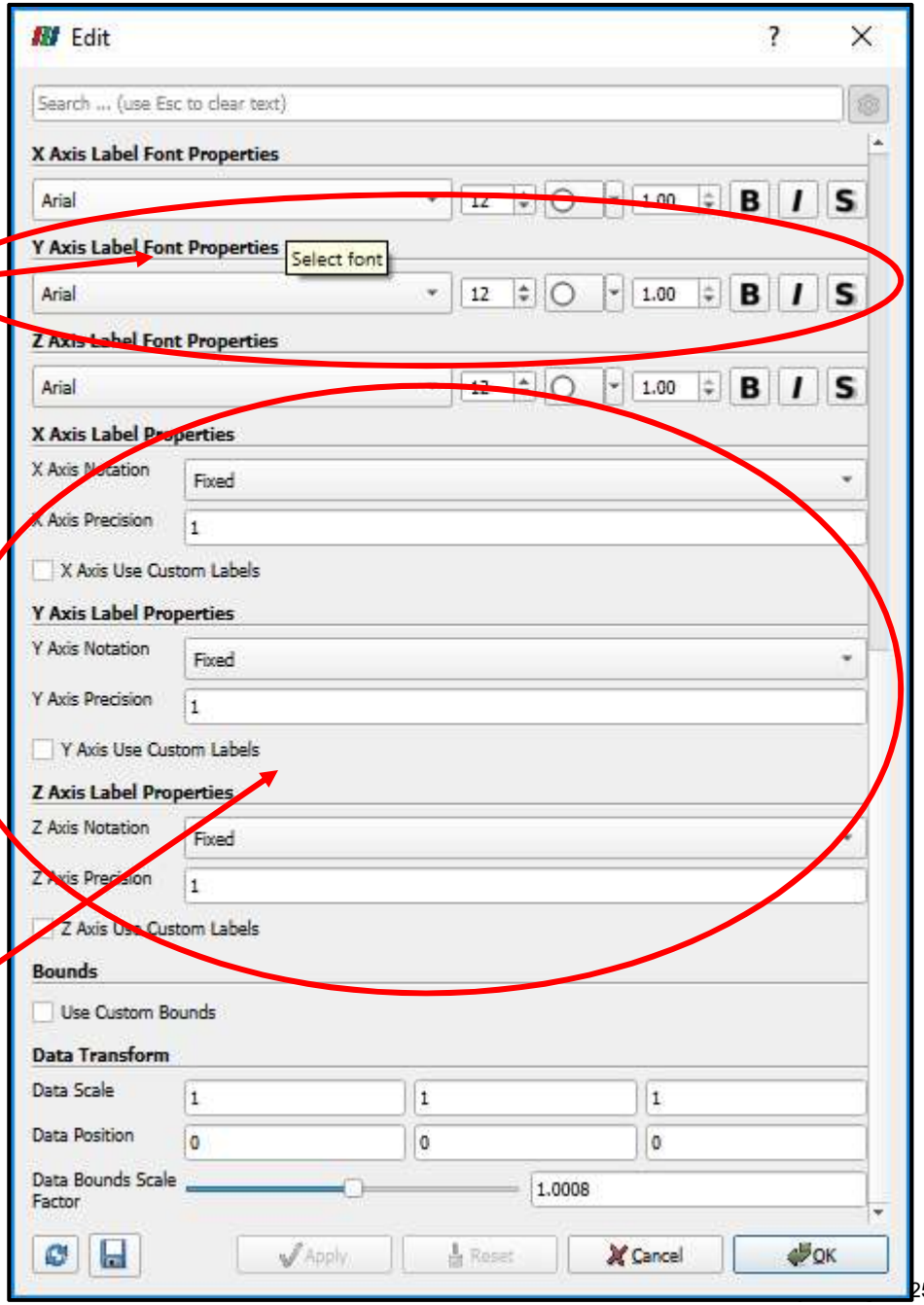
Oregon State University
Computer Graphics

Editing the Axes Grid

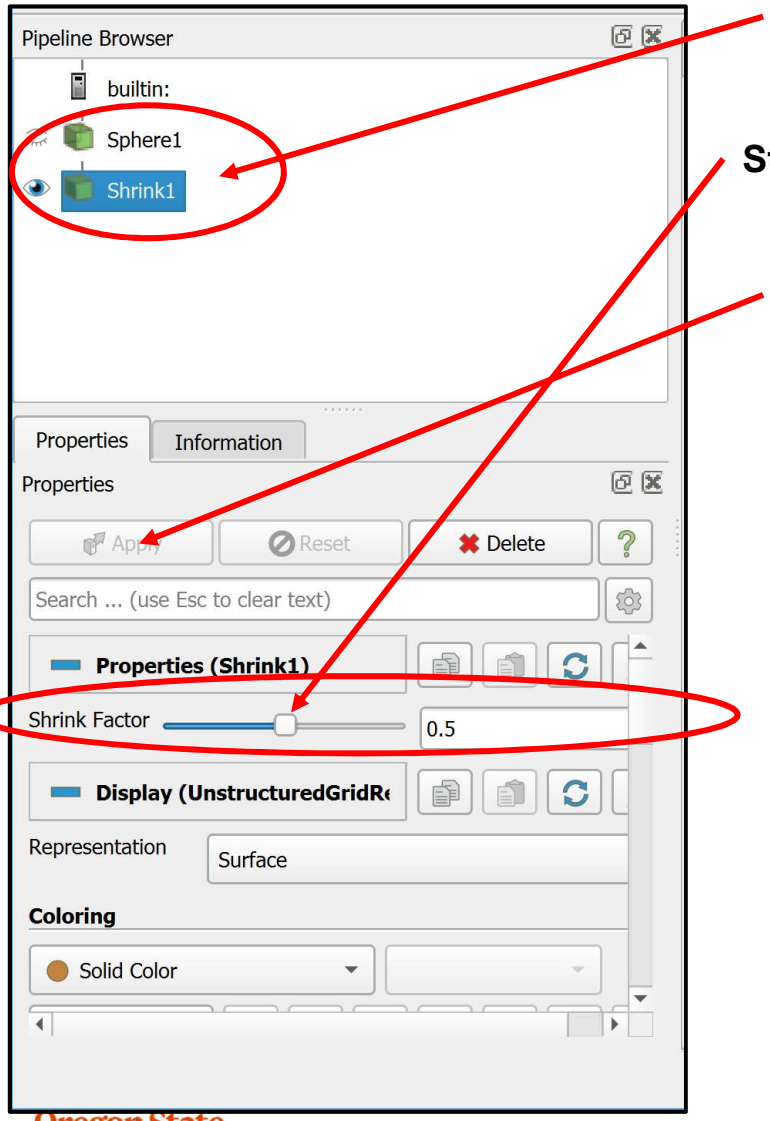
Title font styles



Number label font styles



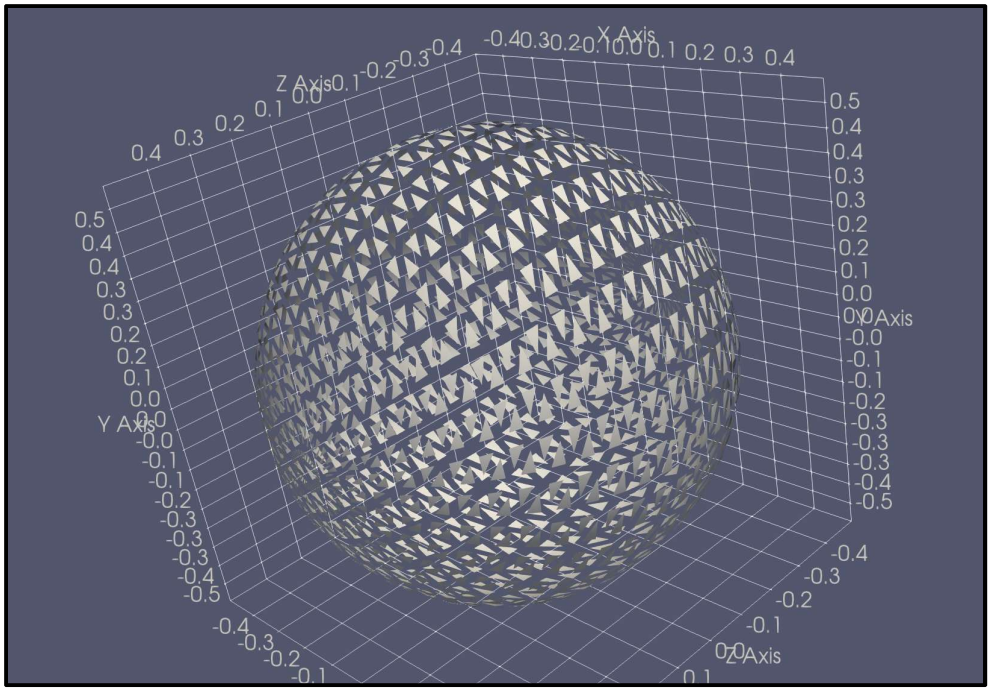
Filters → Alphabetical → Shrink



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

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

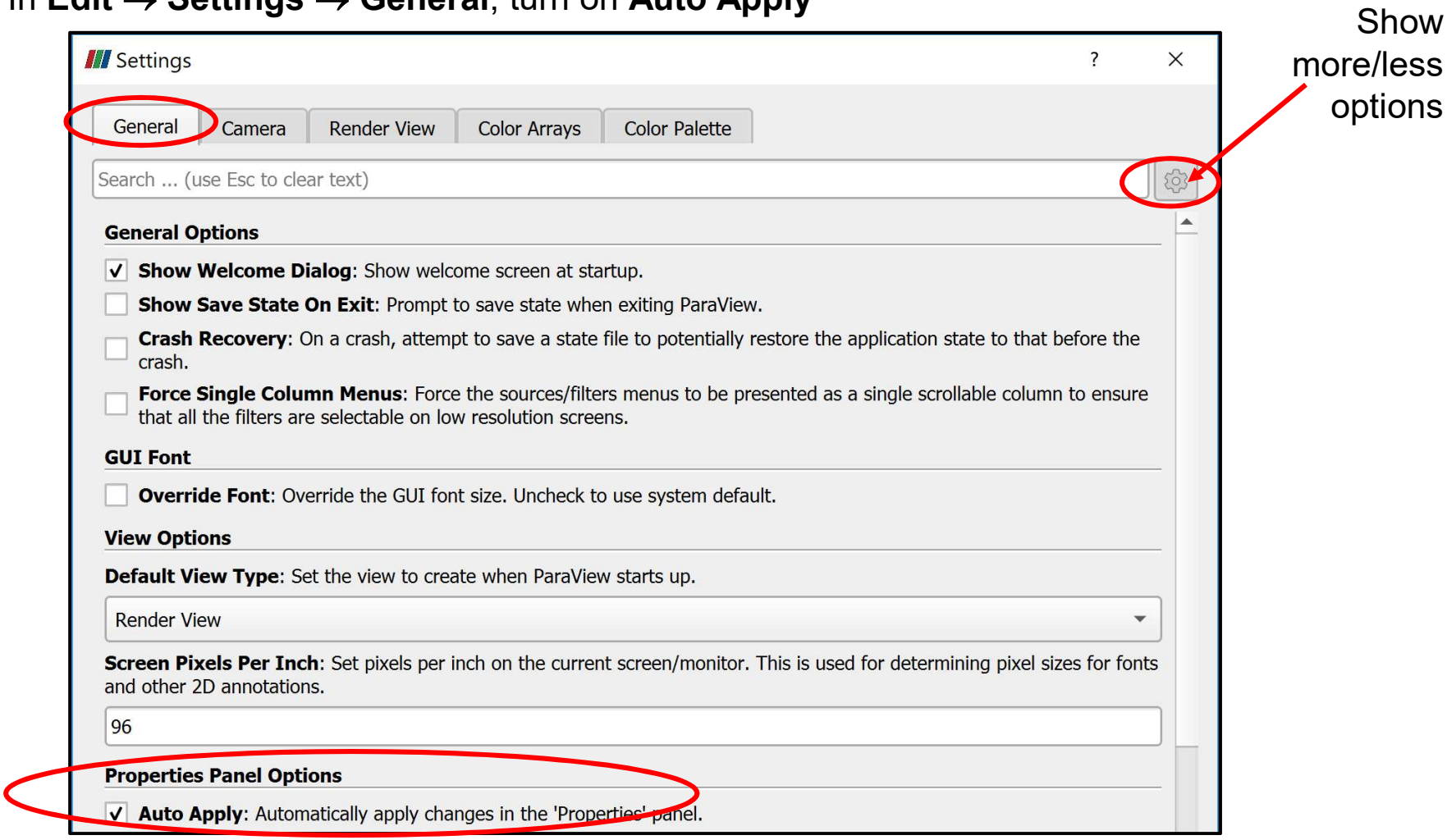
Step #2: Hit **Apply**



Oregon State University
Computer Graphics

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

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



Show more/less options

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	ENZO AMR Particles	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	Nrrd
OpenFOAMReader	PDB	PLOT3D	PLY
PNG Series	PTS	PVD	Particles
Partitioned Legacy VTK	Phasta	ProSTAR (STARCD)	RTXMLPolyDataReader
Restarted Sim	SLAC	Spcth History	STL
Spy Plot	TIFF	Tecplot	Unstructured NetCDF POP
VPIC	VRML	Wavefront OBJ	WindBlade
XDMF	XML	XYZ	

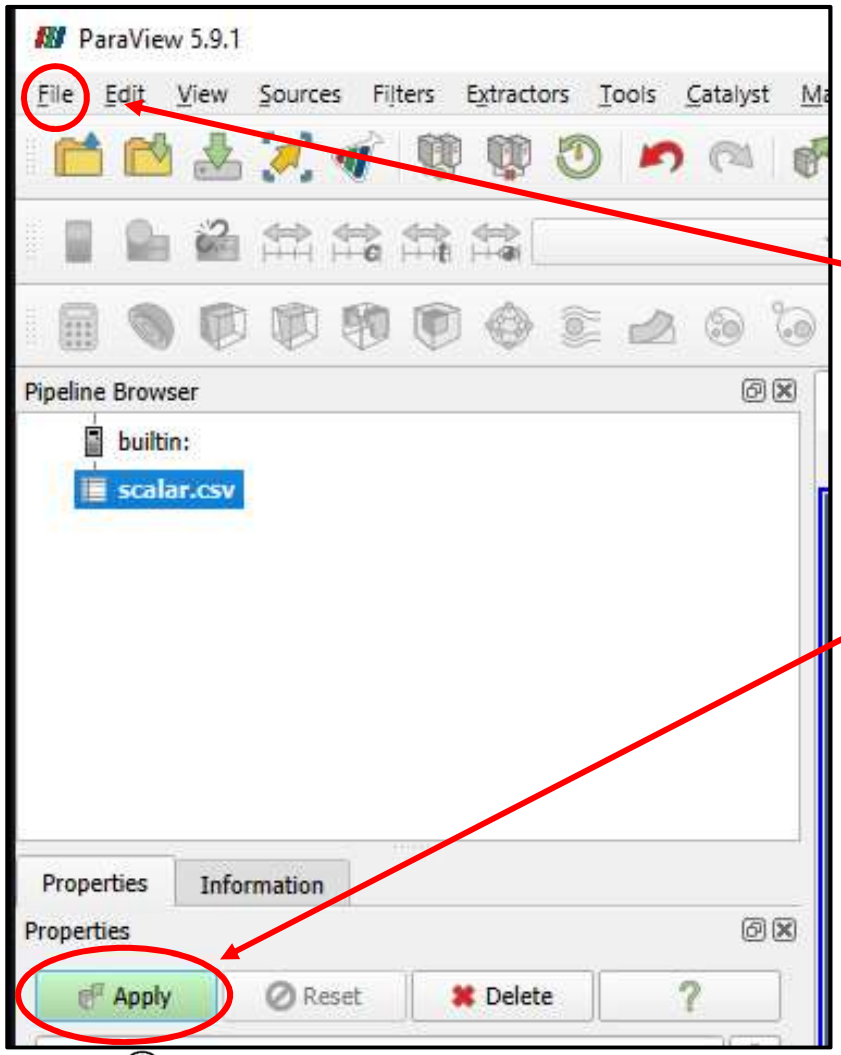
Creating Scalar Data in a CSV File

```
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
```

Go to the **Edit** → **Settings** menu and turn on **Auto-Apply.**
Do a **File** → **Open** and navigate to your CSV file.
Hit the **Apply** button to actually do the read.

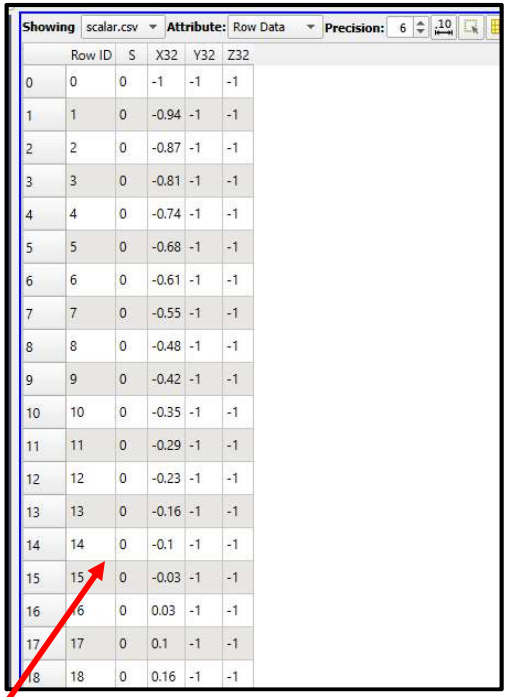
scalar.csv

Reading and Converting the CSV File



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

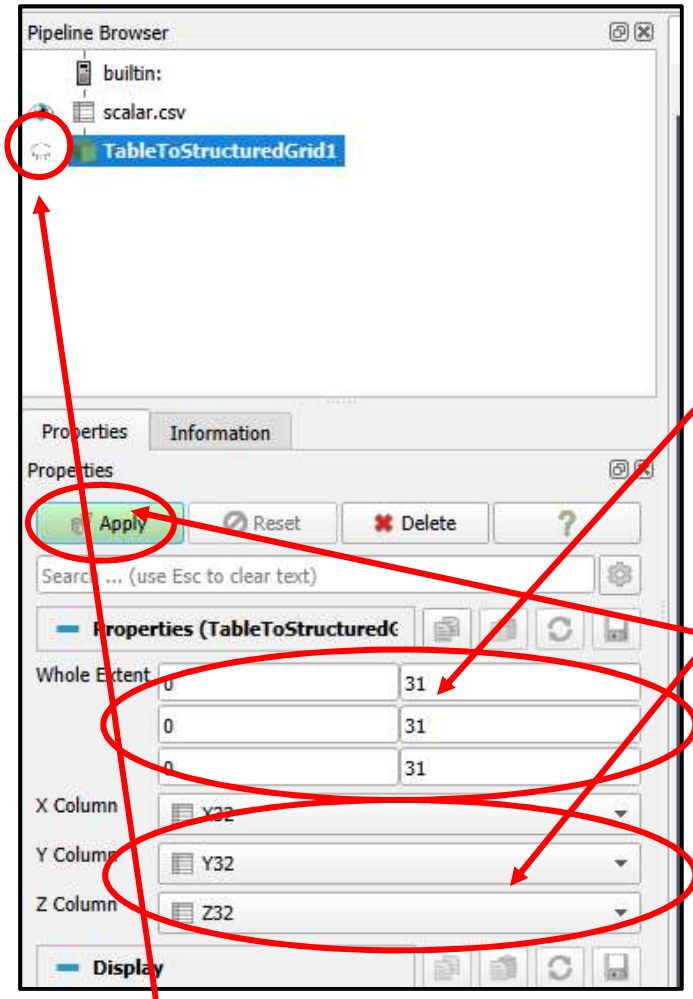
2. Then, click **Apply**



	Row ID	S	X32	Y32	Z32
0	0	0	-1	-1	-1
1	1	0	-0.94	-1	-1
2	2	0	-0.87	-1	-1
3	3	0	-0.81	-1	-1
4	4	0	-0.74	-1	-1
5	5	0	-0.68	-1	-1
6	6	0	-0.61	-1	-1
7	7	0	-0.55	-1	-1
8	8	0	-0.48	-1	-1
9	9	0	-0.42	-1	-1
10	10	0	-0.35	-1	-1
11	11	0	-0.29	-1	-1
12	12	0	-0.23	-1	-1
13	13	0	-0.16	-1	-1
14	14	0	-0.1	-1	-1
15	15	0	-0.03	-1	-1
16	16	0	0.03	-1	-1
17	17	0	0.1	-1	-1
18	18	0	0.16	-1	-1

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



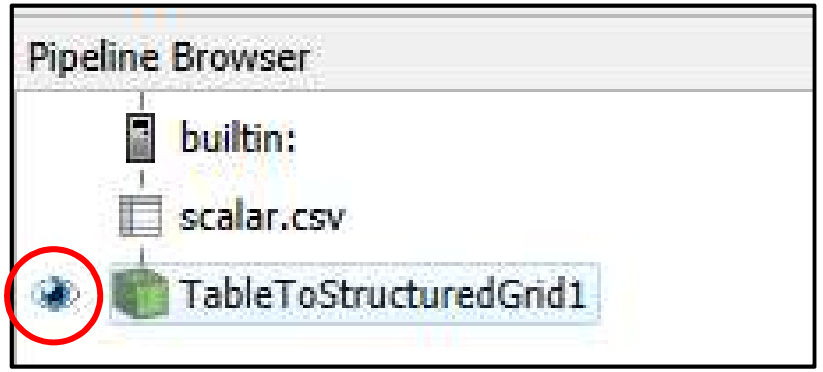
4. Now, go to
Filters → Alphabetical → TableToStructuredGrid

5. 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). In this case, the numbers are **0** and **31**.

6. Fill in the **{X,Y,Z} Column** information so ParaView knows how to make your 3D display. In this case, the names are **X32**, **Y32**, and **Z32**.

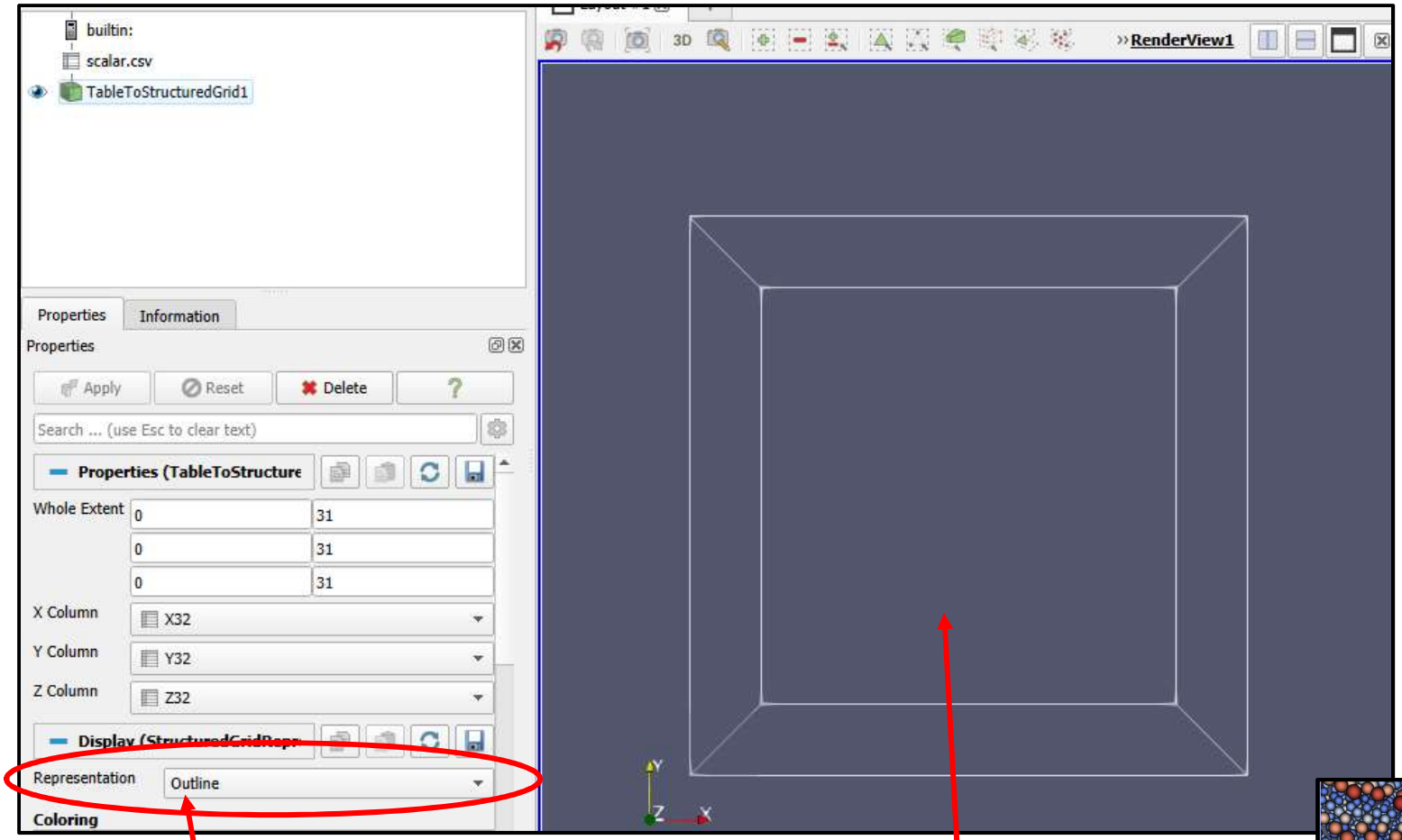
7. Hit the **Apply** button to actually do the conversion.

Turn on the "eyeballs" so that you can view this data

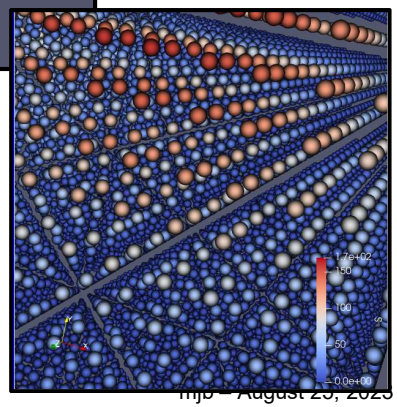


Oregon University
Computer Graphics

Reading and Converting the CSV File

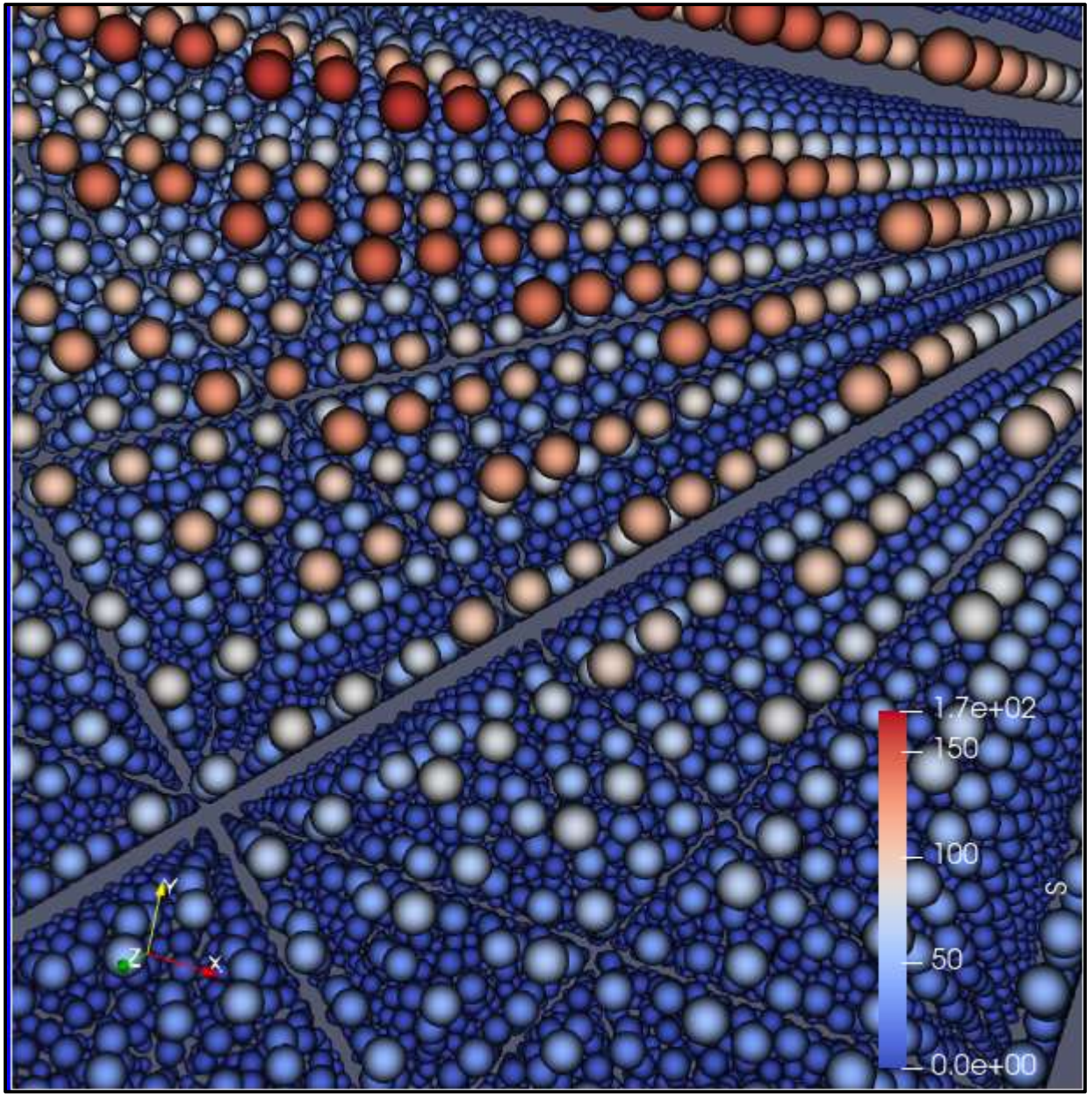


The default Display Representation is **Outline**.
Click here and try some of the others. **Point Gaussian** is cool!



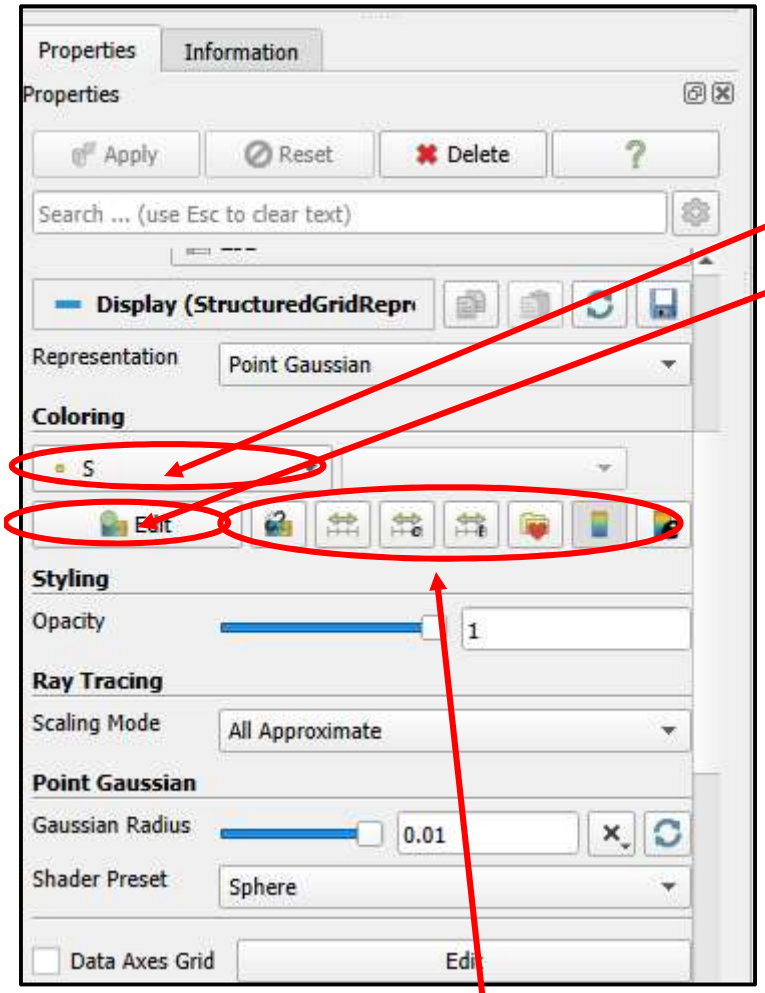
At this point, you should probably go to the **Edit → Settings** menu and turn off **Auto-Apply**

As Point Gaussian



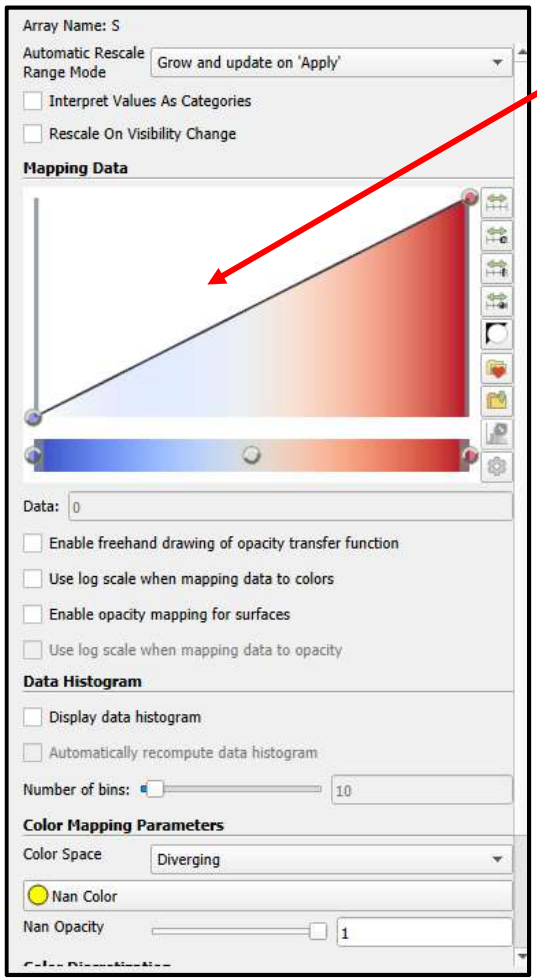
A Side Trip: Choosing Colors

Turning on Color



The default coloring is by scalar value, **S** in this case.
You can also click here and change it to **Solid Coloring**.

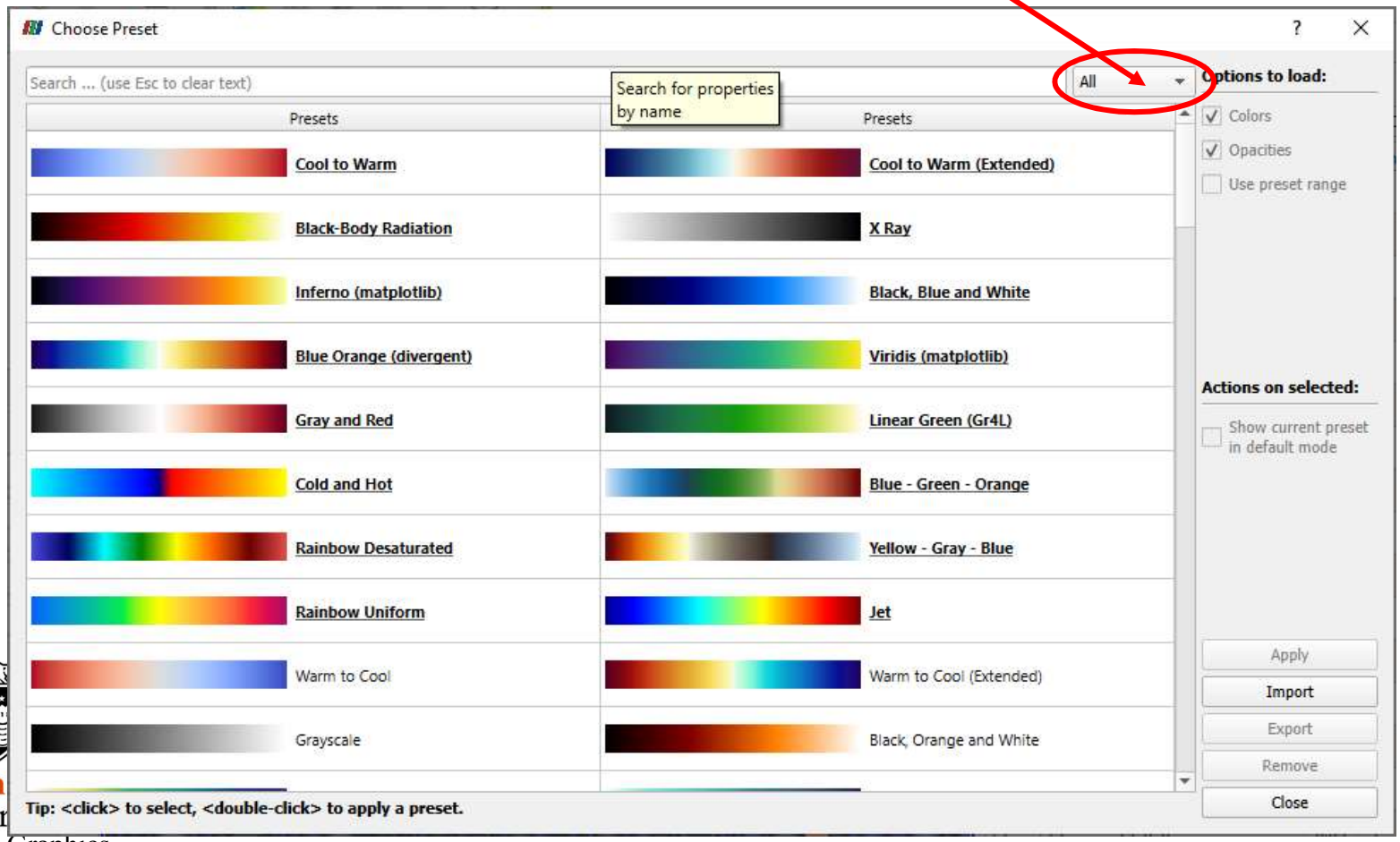
The **Edit** button will bring up a color map editor



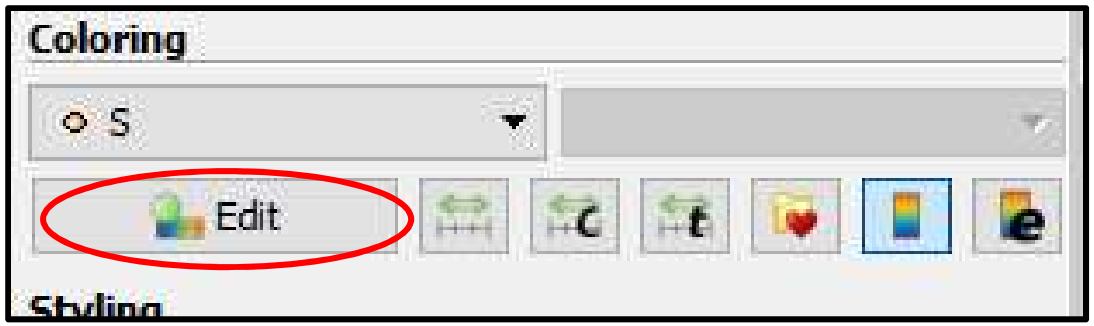
This is a row of color options.

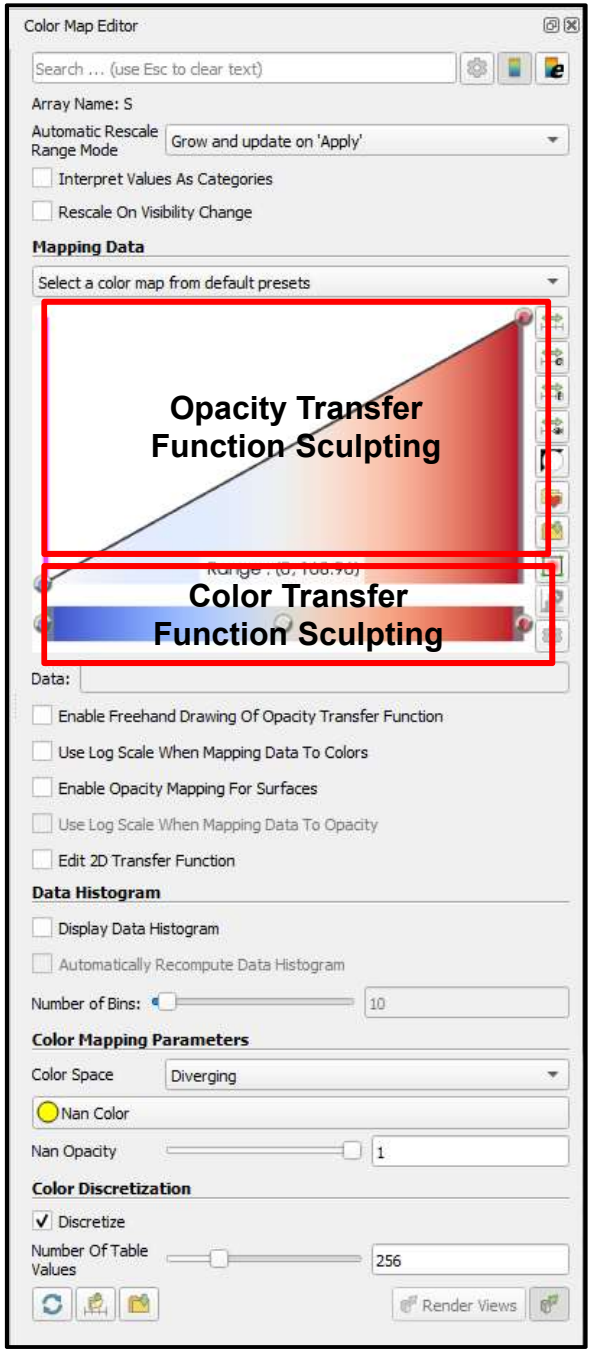
Choose Among Standard Color Transfer Functions

Click here to see all the categories of Transfer Functions available to you. Click **All** to see them all at once. (You will need to scroll down a lot.)

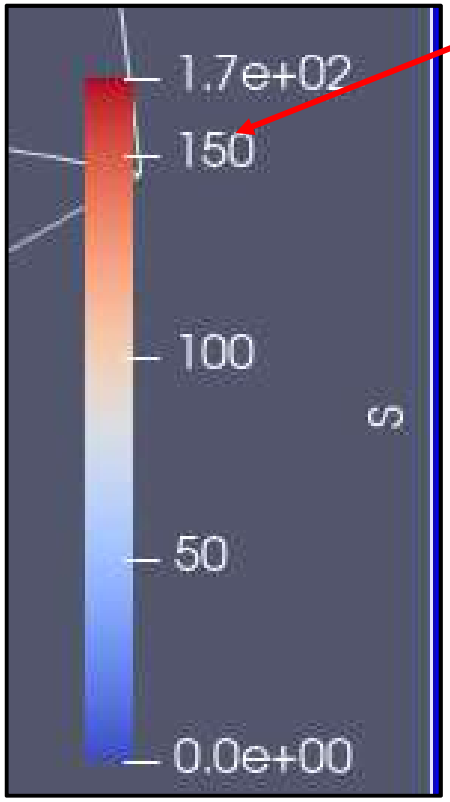


Color Map Editor



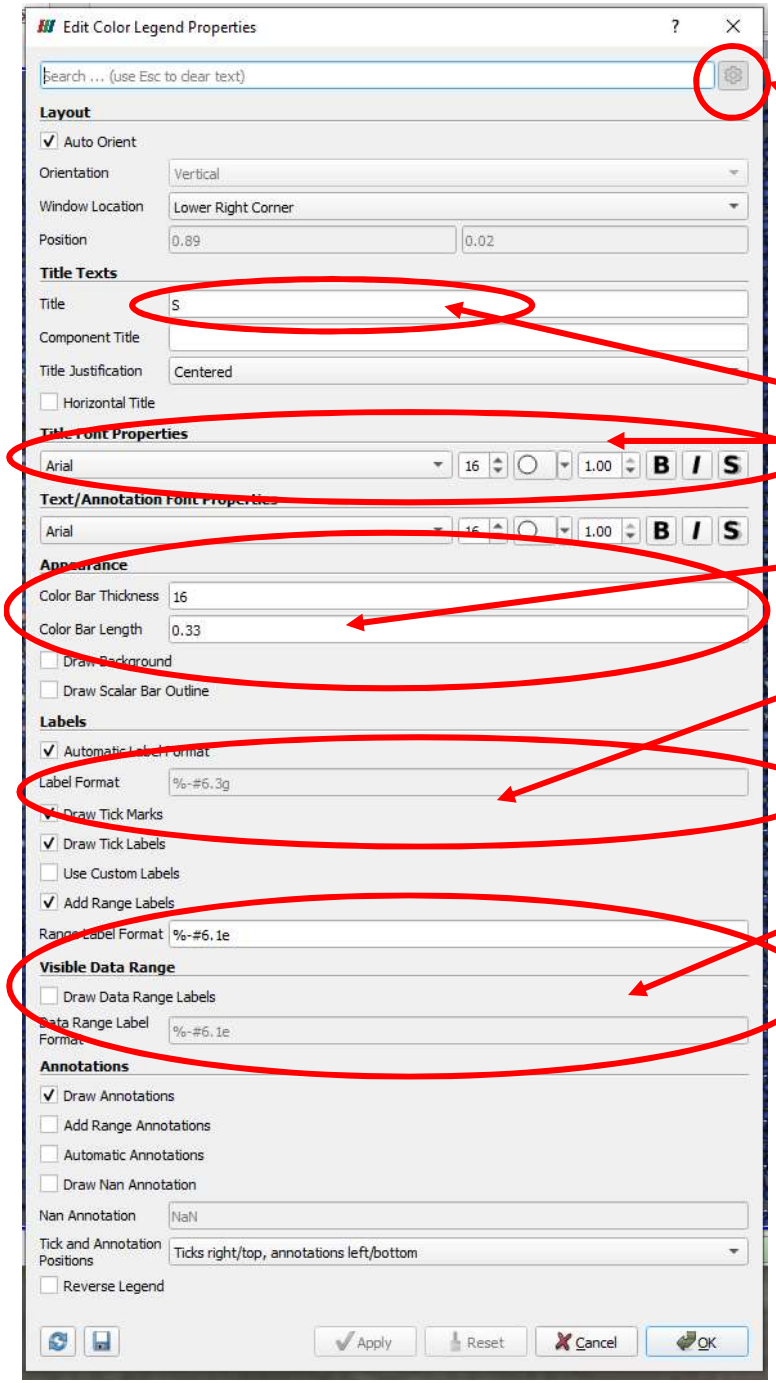


Changing the Legend



The default legend is good, but you can make it better. Start by clicking here.

Changing the Legend



Click on the “gear” to bring up *all* of the options.
(This is a good idea on *all* ParaView dialog boxes.)

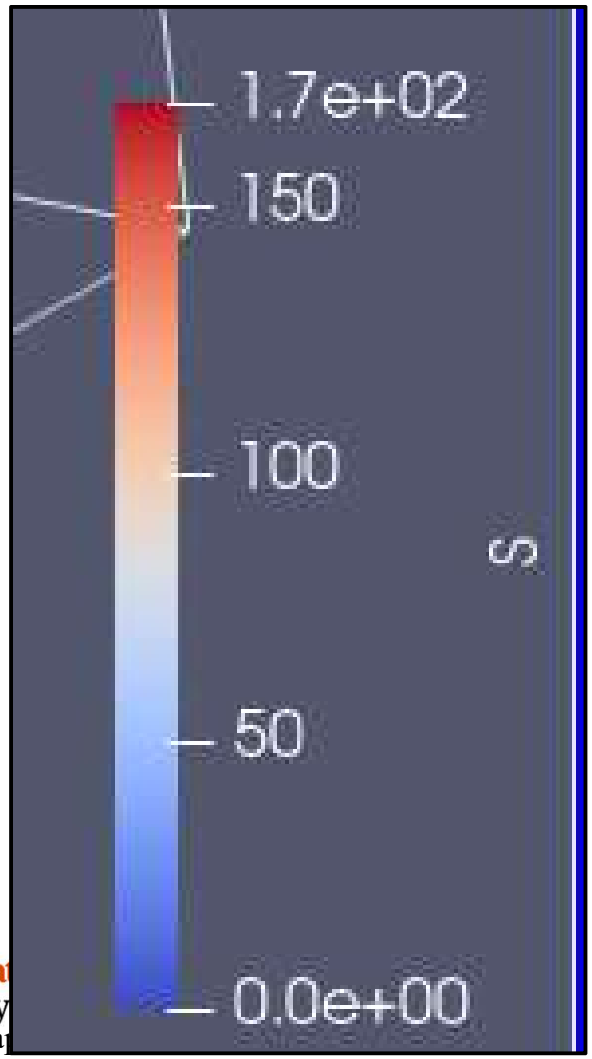
Legend title and font

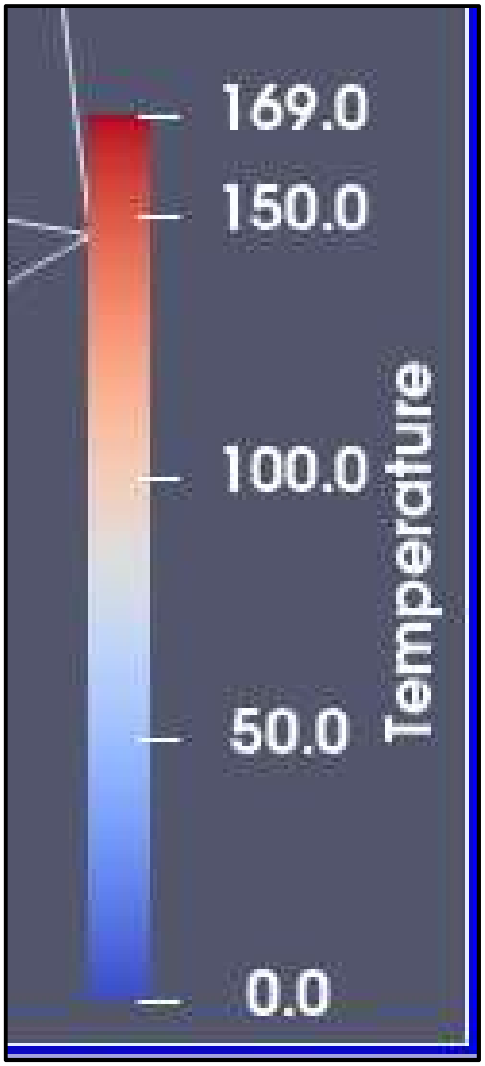
Color bar

Tick mark font and number format (“printf-style”)

Range numbers at the end of the legend

From this, to this

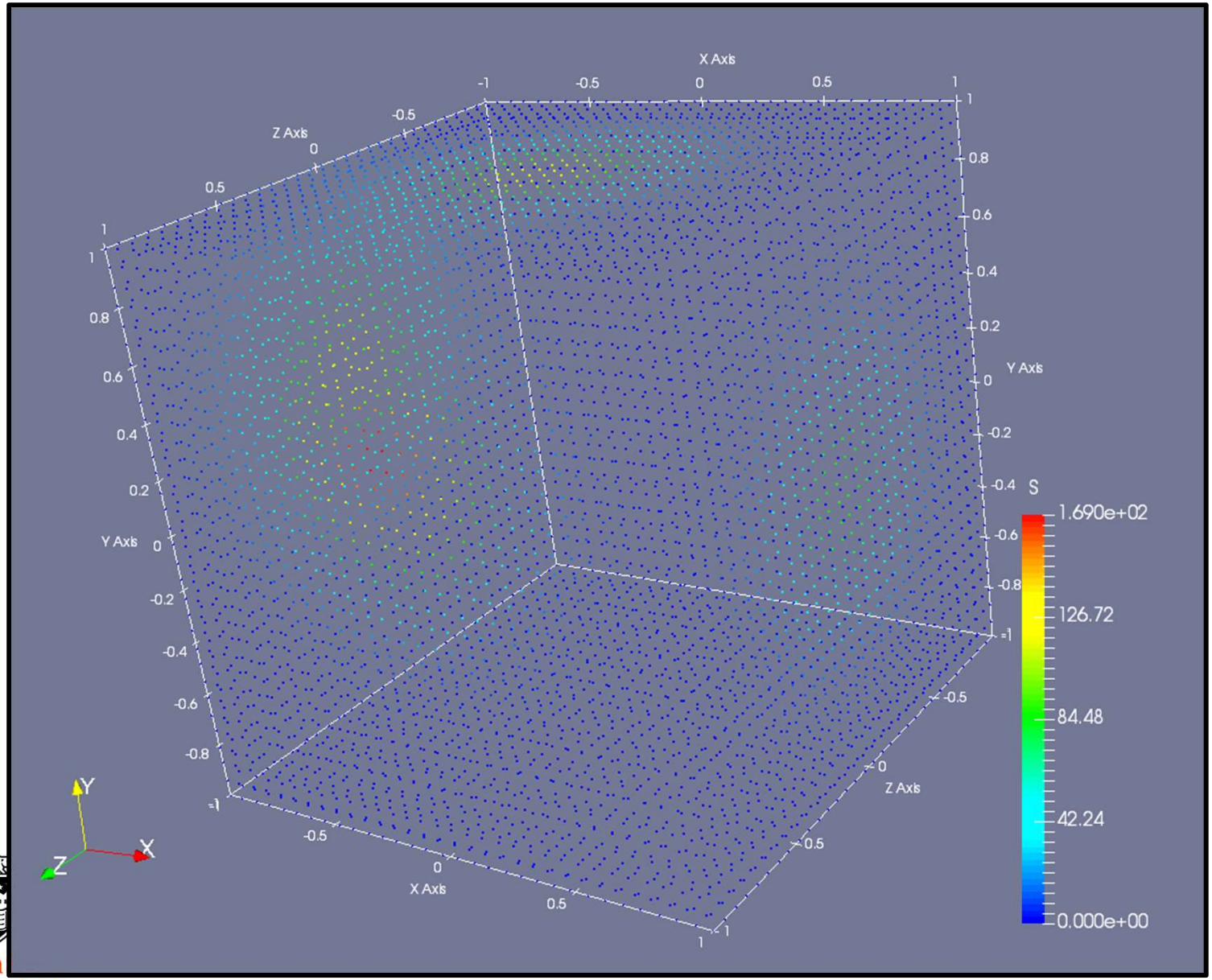




Visualizing Scalar Data, II

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.

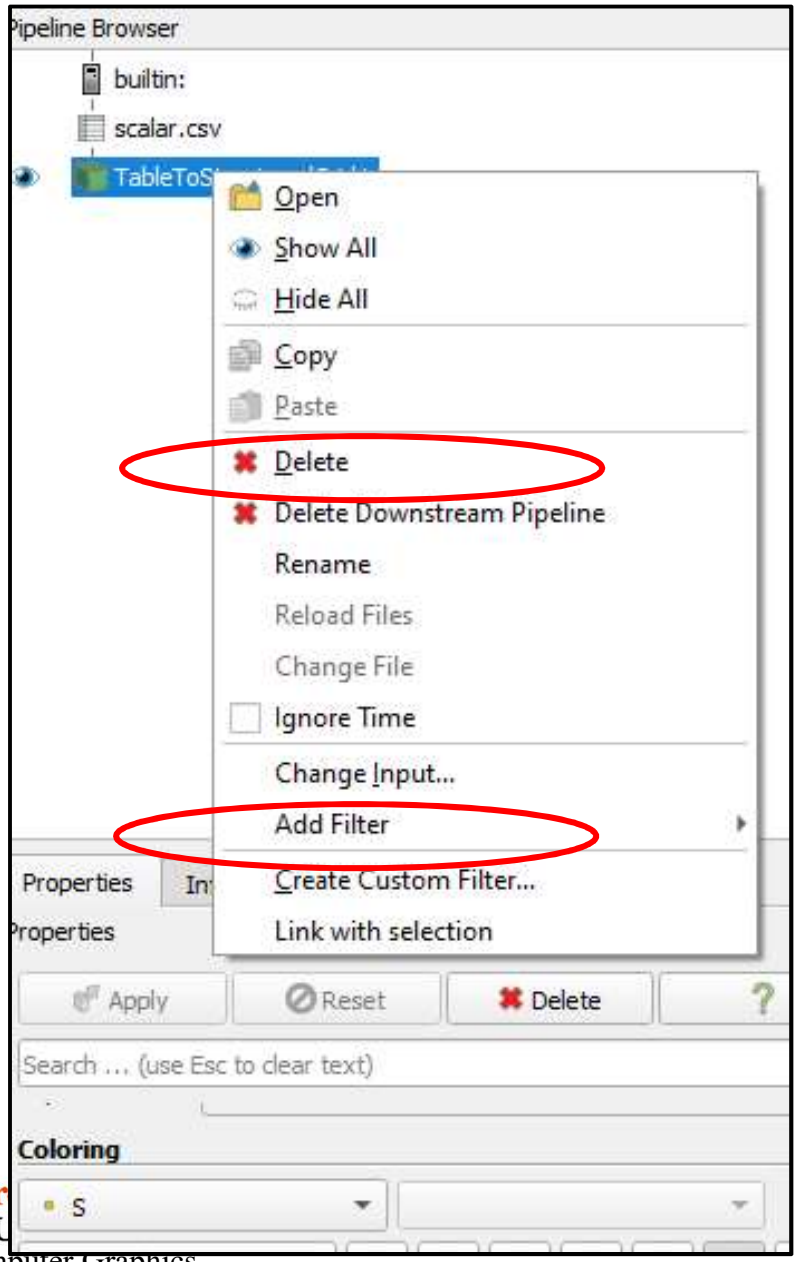
- For data-size reasons, it is helpful if any datasize reduction Filters are included early in the pipeline.

- 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.

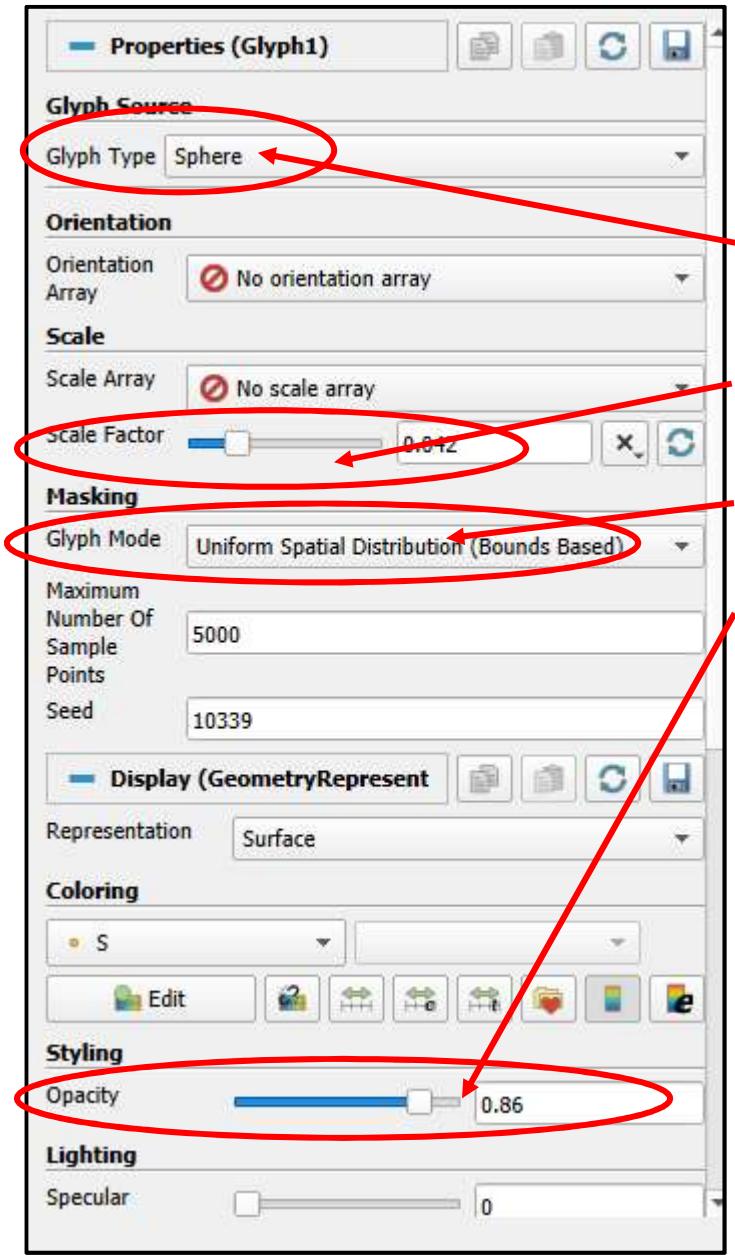
- Whatever "eyeballs" you have clicked on, that's what pipeline elements' visual representations you will see in the display.

- 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



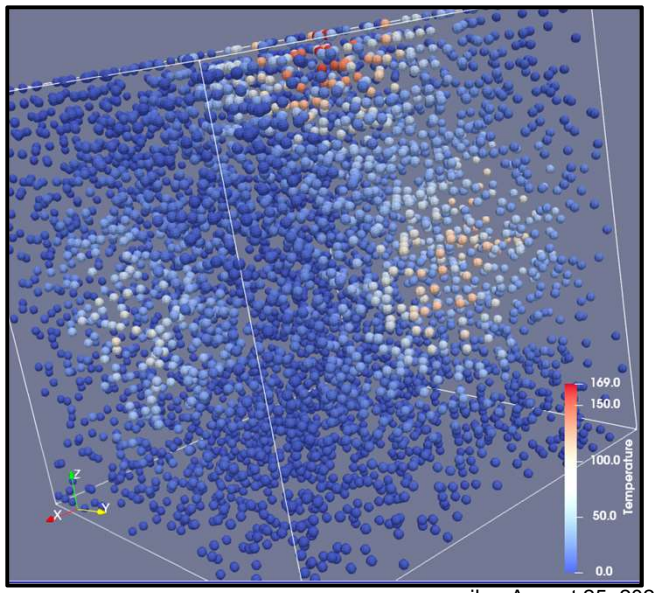
Filters → **Alphabetical** → **Glyph** adds the glyph cloud to the pipeline. Hide the **TableToStructuredGrid** (click off the eyeball) and un-hide the **Glyph.**

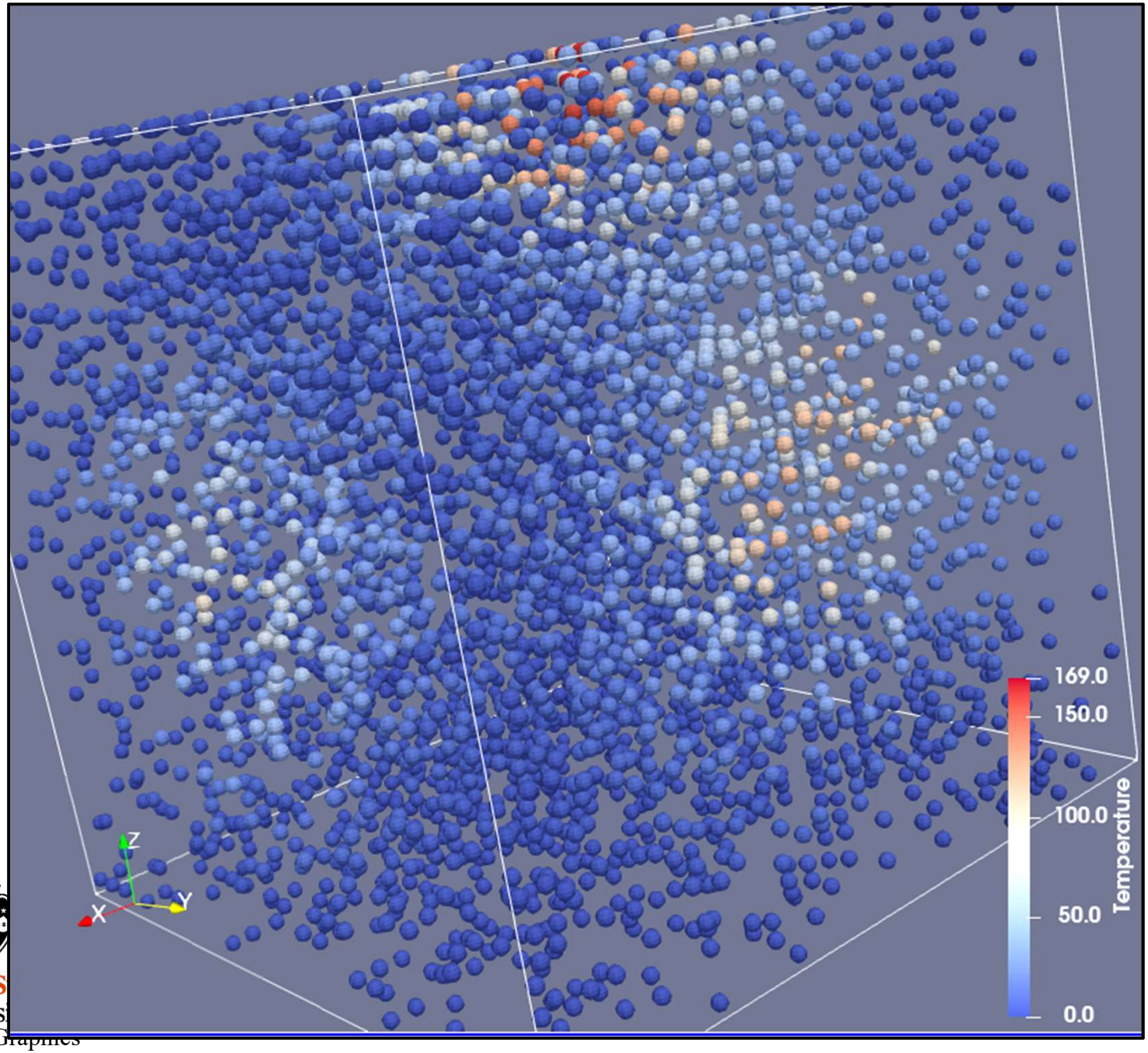
Set the **Glyph Type**

Play with the **Scale Factor**

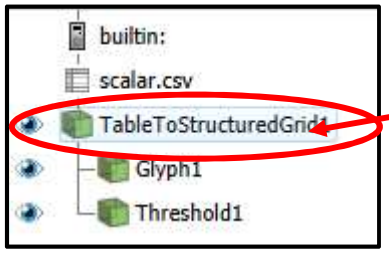
Play with the **Glyph Mode**

Play with the **Opacity**



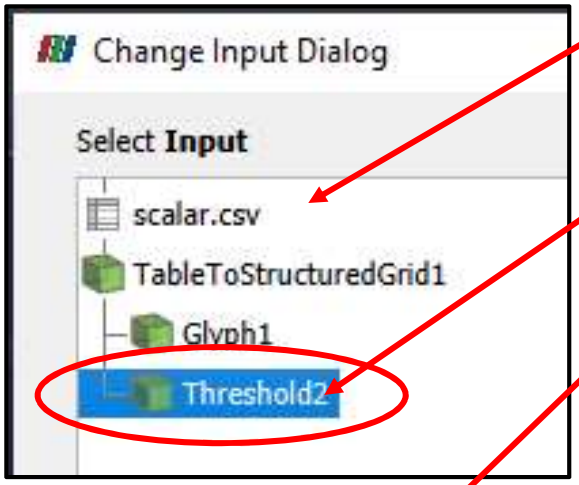

169.0
150.0
100.0
50.0
0.0
Temperature
Z
Y
X

As a Threshold Glyph Cloud



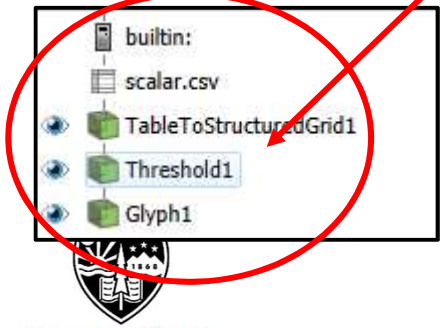
Click on **TableToStructuredGrid**, then **Right-click → Add Filter → Alphabetical → Threshold**.

But, this is the wrong order. We want to threshold the data first. So, select **Glyph1**, right-click on it, select **Change Input…**



Then select **Threshhold1**

You now have this order

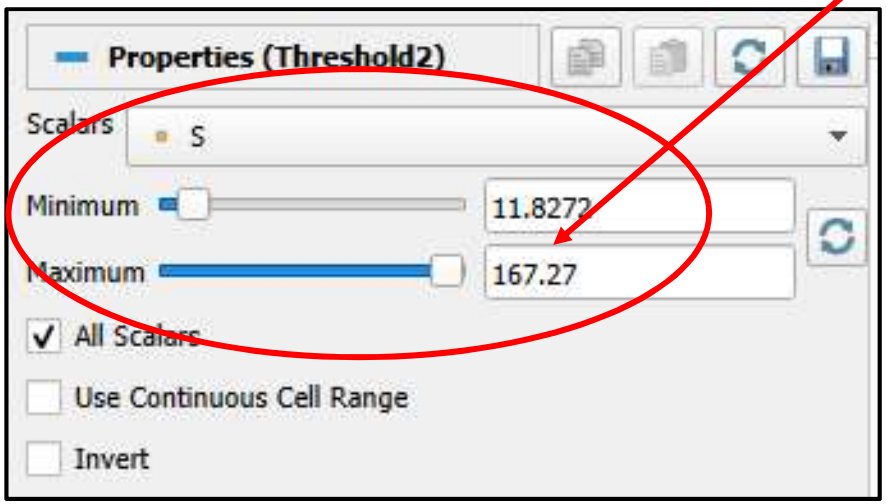


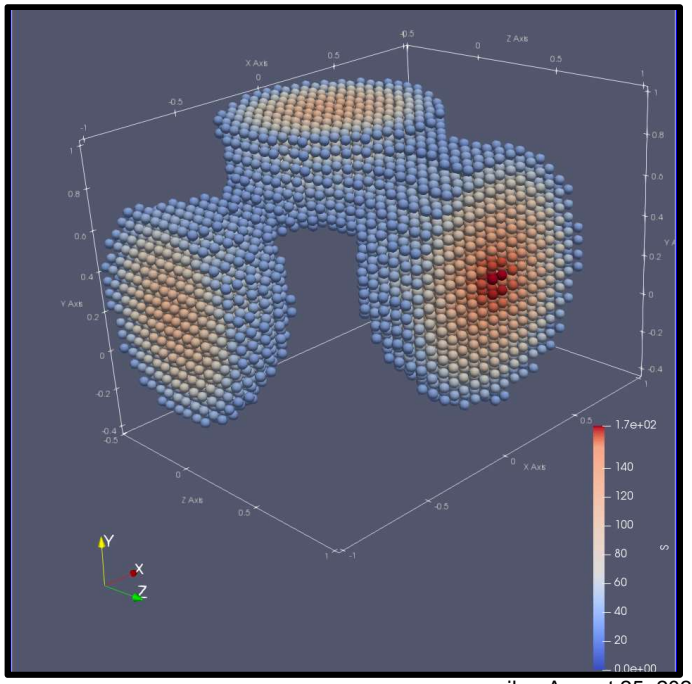
Oregon State University
Computer Graphics

As a Threshold Glyph Cloud

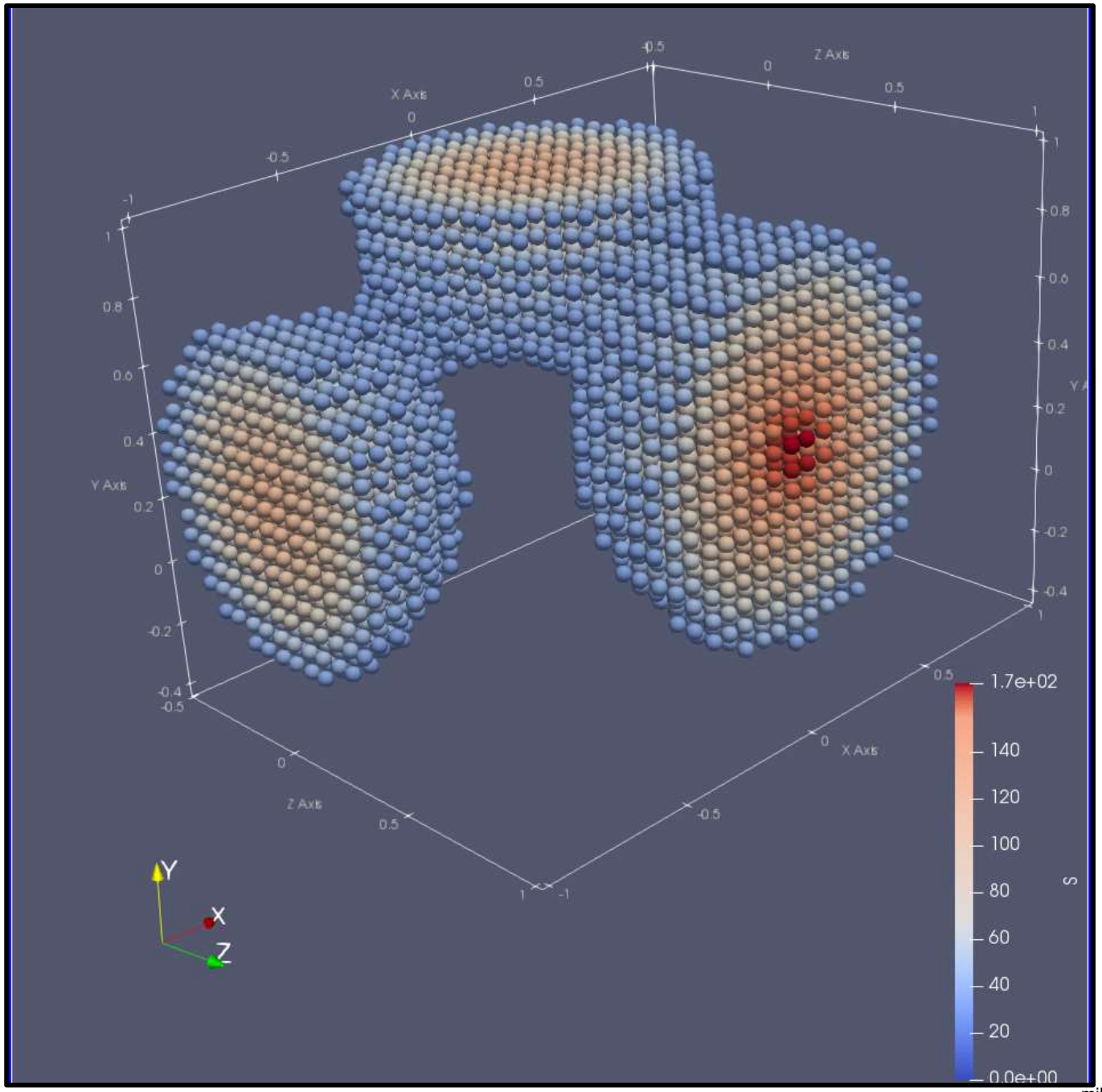
Hide the **TableToStructuredGrid** and the **Threshold**, then un-hide the **Glyph.**

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

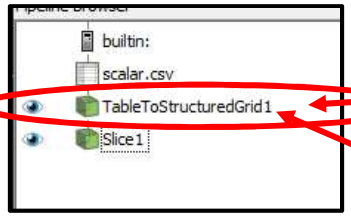




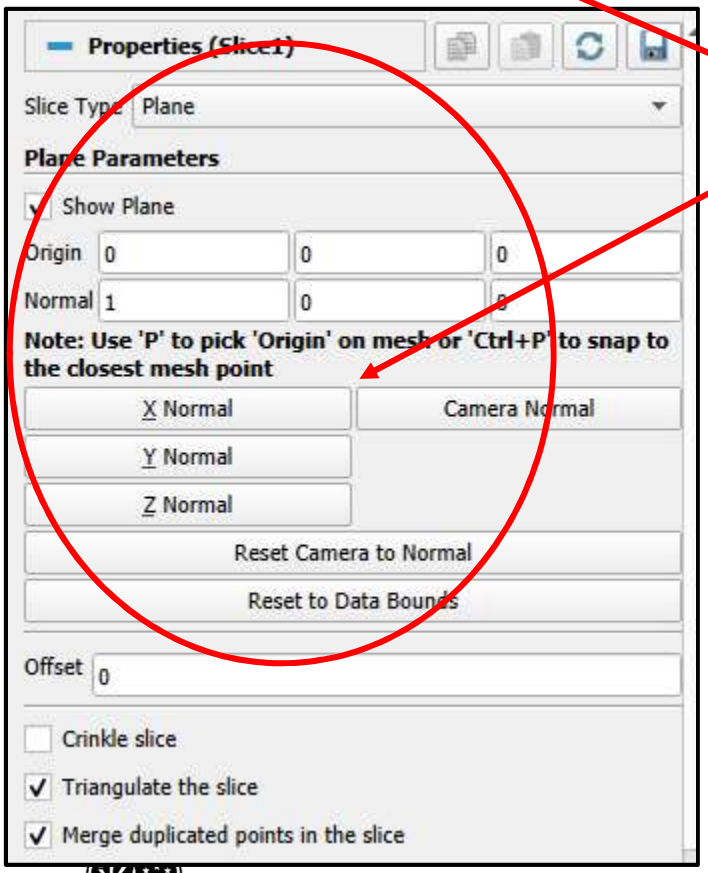
Oregon State University
Computer Graphics


X Axis
Z Axis
Y Axis
1.7e+02
140
120
100
80
60
40
20
0.0e+00
S
Y
X
Z

As a Colored Cutting Plane

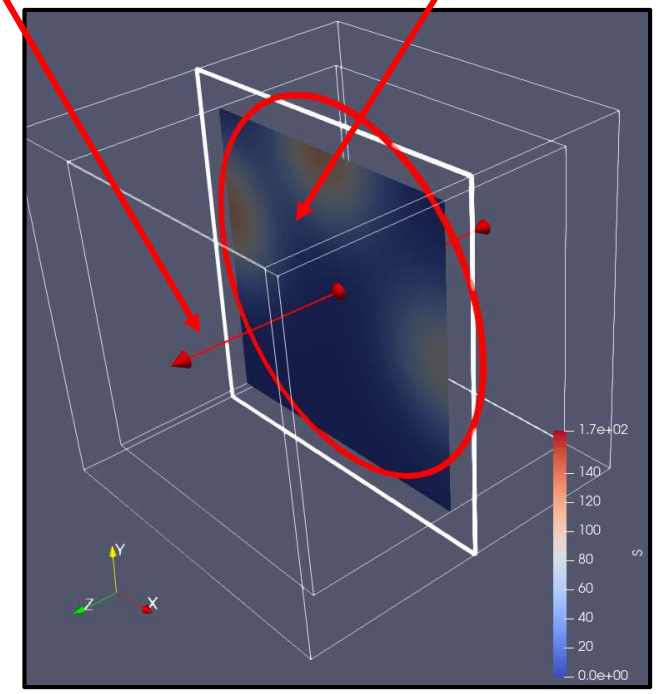


ParaView trick – turn on the **TableToStructuredGrid** display and set the Representation to **Outline**. That keeps ParaView from displaying the plane as 2D-only

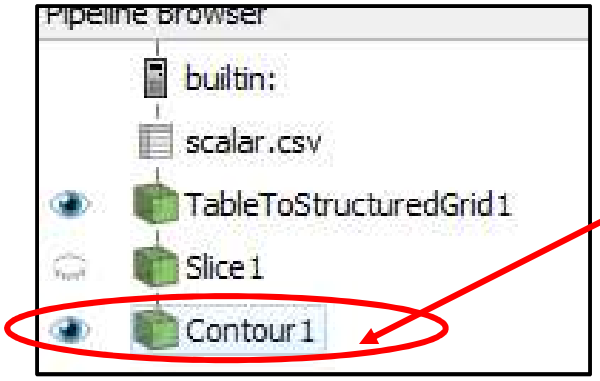


Right-click on **TableToStructuredGrid**, then select **Add Filter → Alphabetical → Slice**

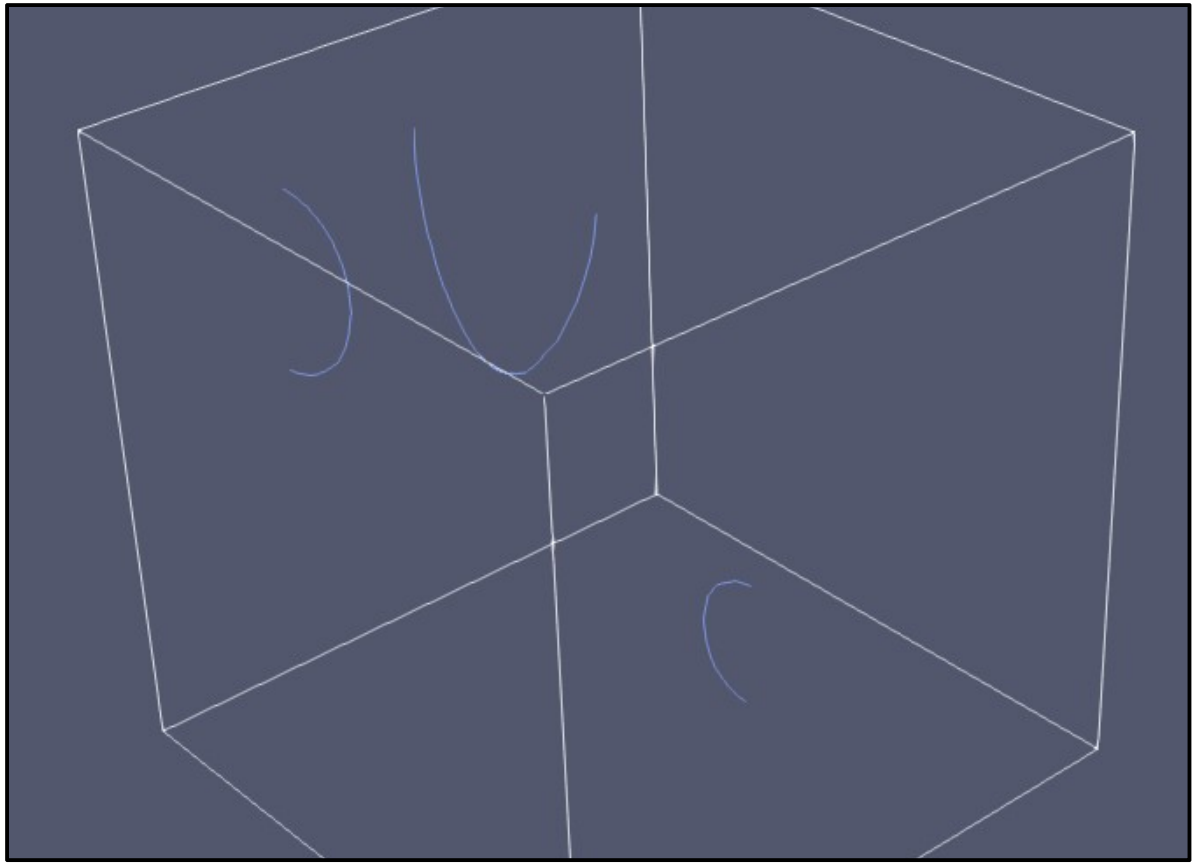
Click in here to change the slice parameters.
Click on the colored plane itself to move the plane.
Click on the arrow to rotate the plane.



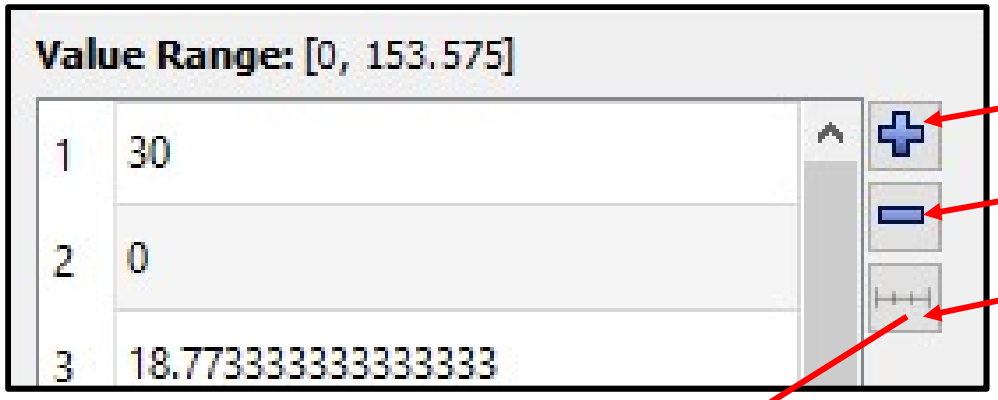
Turning the Slice into Contours



Right-click on **Slice1**, then select
Add Filter → Alphabetical → Contour



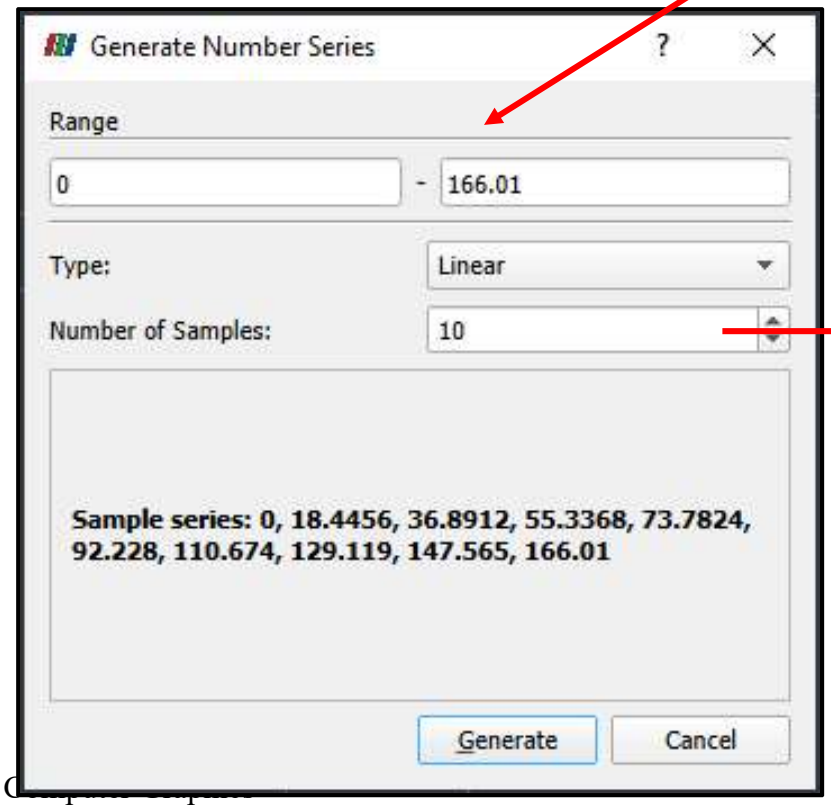
Changing the Contour Isovalue

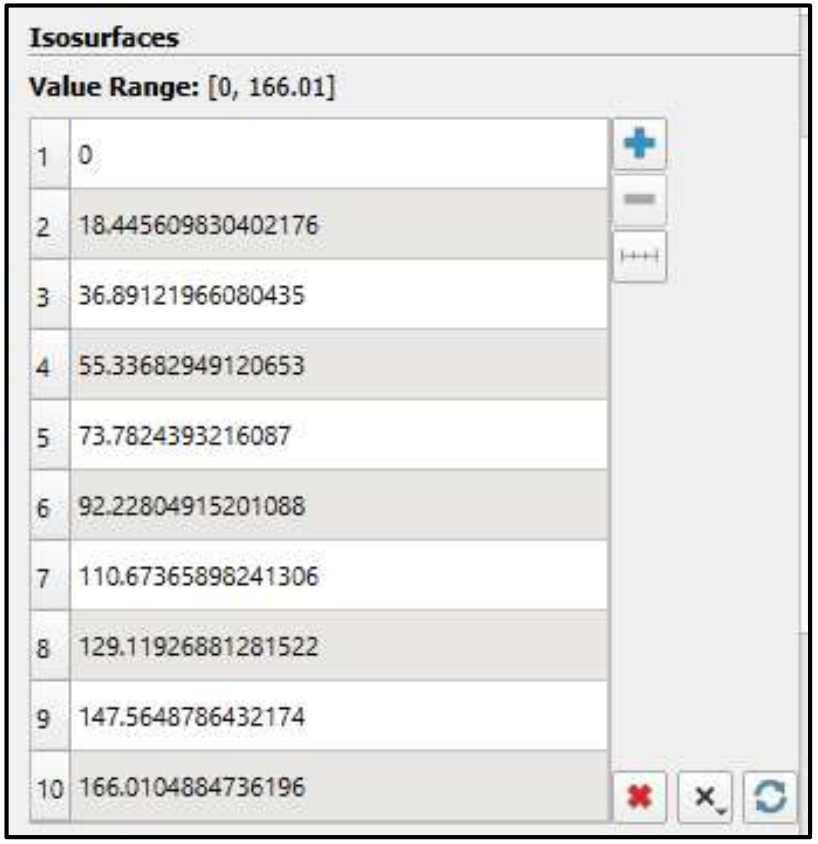


Add a contour isovalue

Delete a contour isovalue

Add a range of contour isovalues



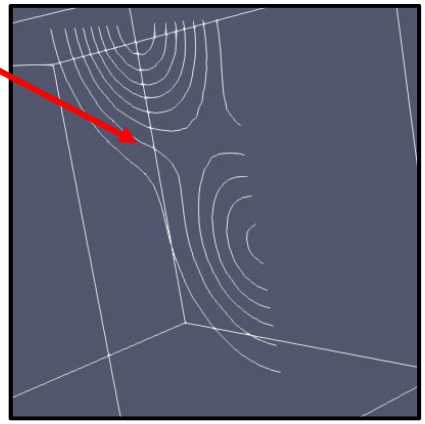


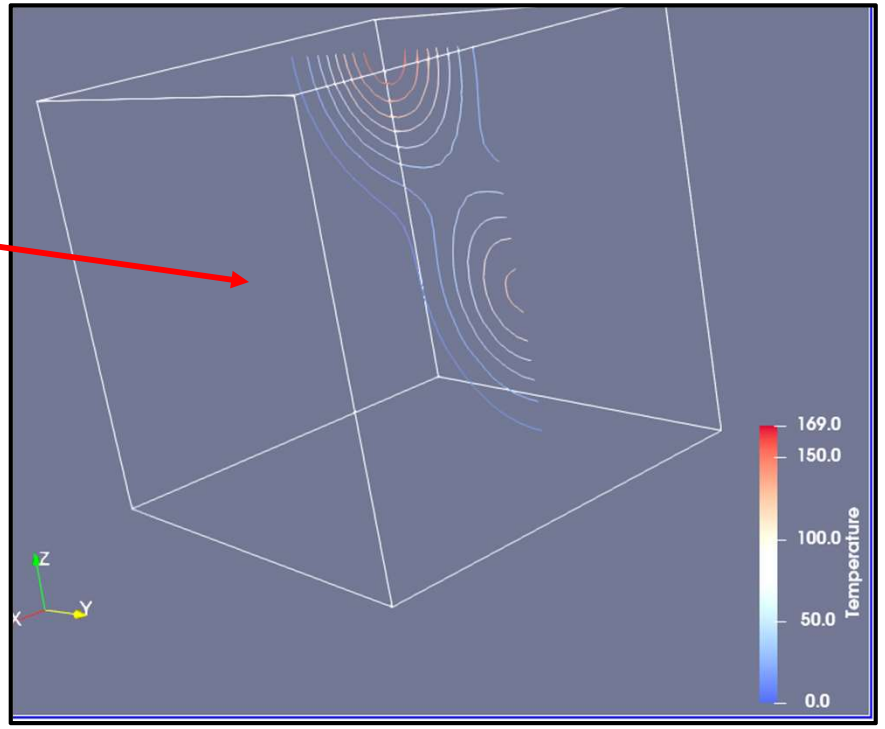
As Contours

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

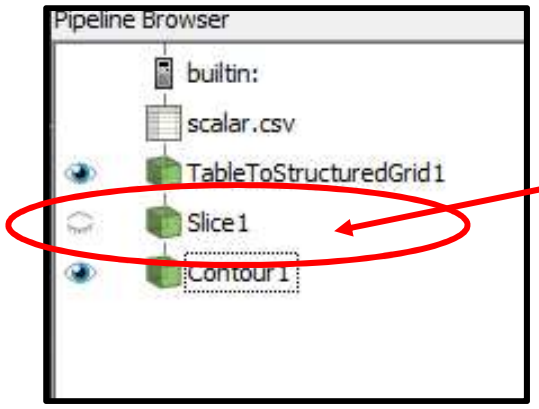
Coloring by **S** will give you colored contour lines.

Coloring by **Solid Color** will give you a single color.



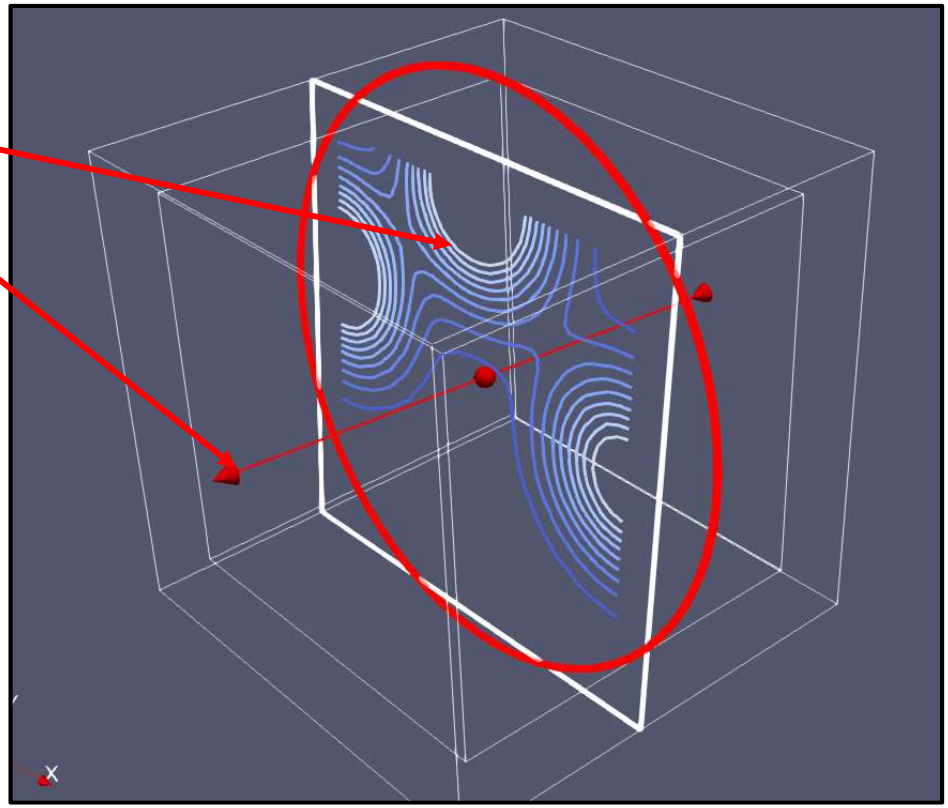


As Contours

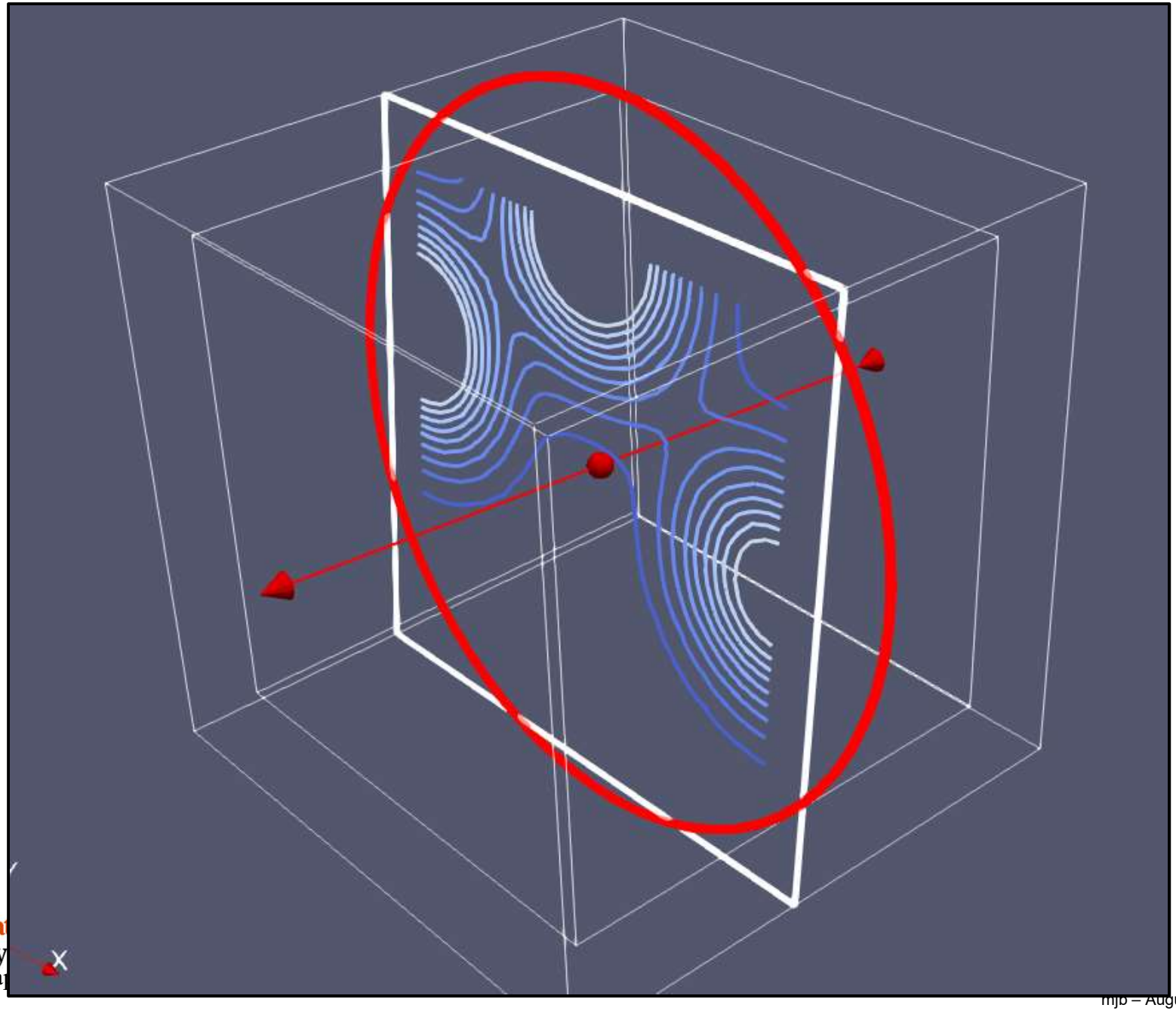


Clicking on the Slice filter will bring up these slice handles so that you can move and re-orient the slice plane

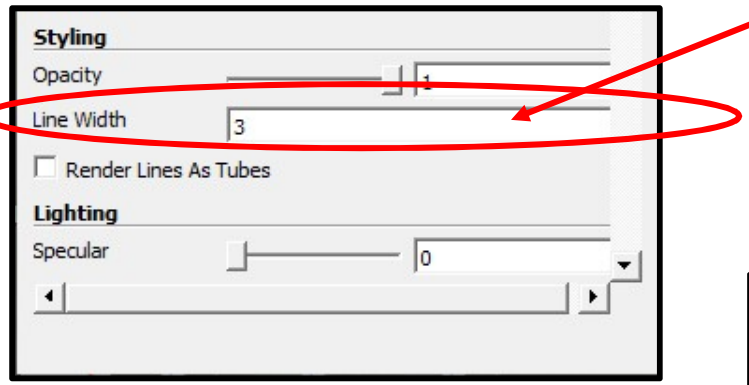
Click on the plane itself to move the plane.
Click on the arrow to rotate the plane.



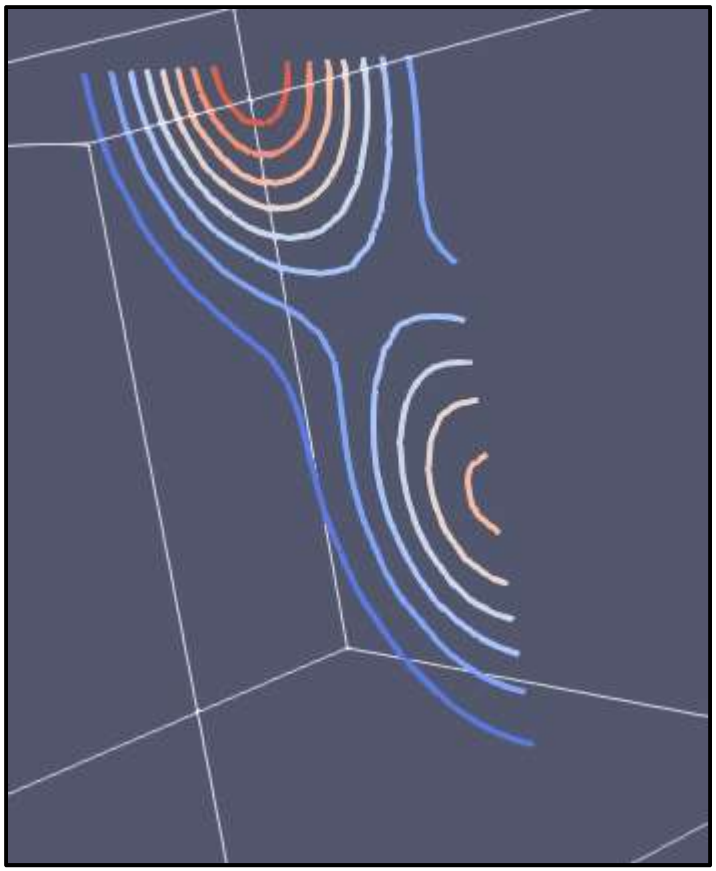
As Contours



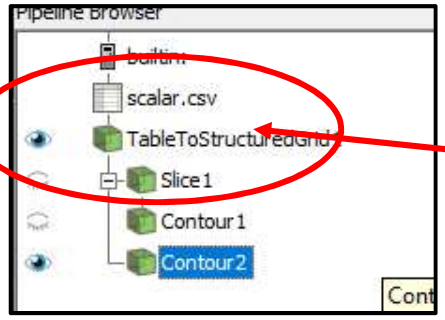
As Contours



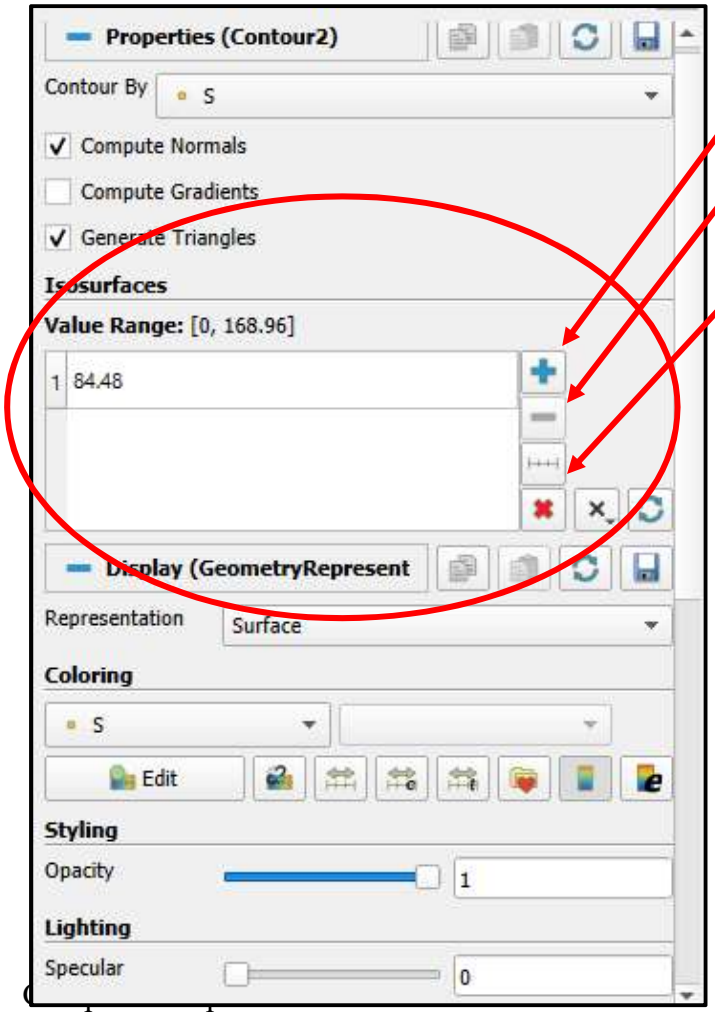
Adjusting the **Line Width**



As 3D Isosurfaces



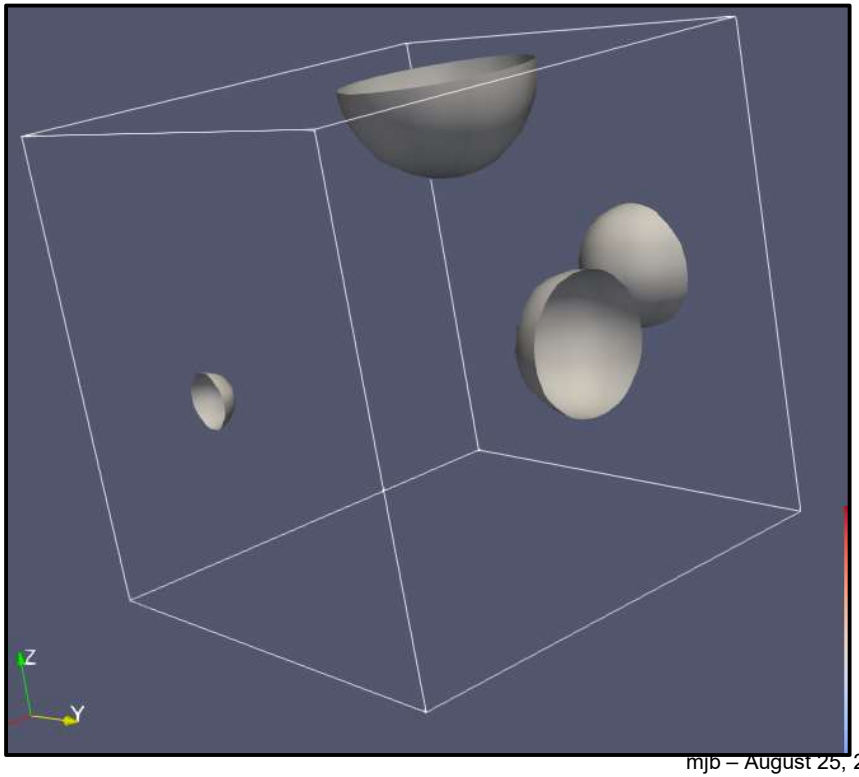
Note – This instance of **Contour** needs to be parented from **TableToStructuredGrid**, not **Slice**

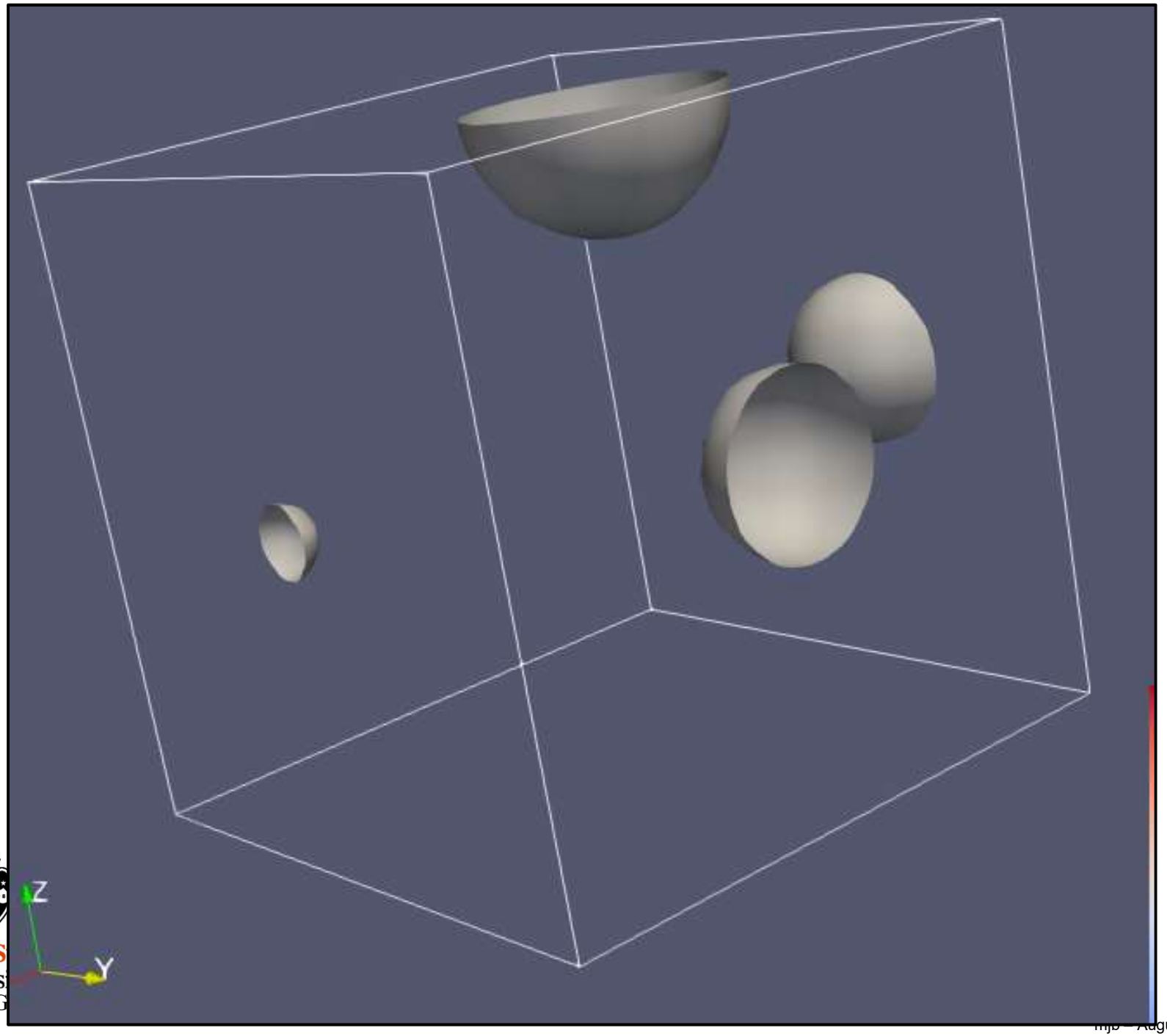


Add an isosurface isovalue

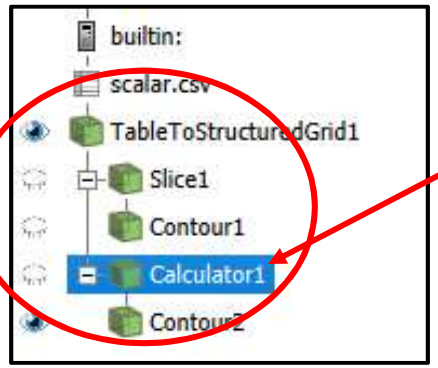
Delete an isosurface isovalue

Add a range of isosurface isovalues



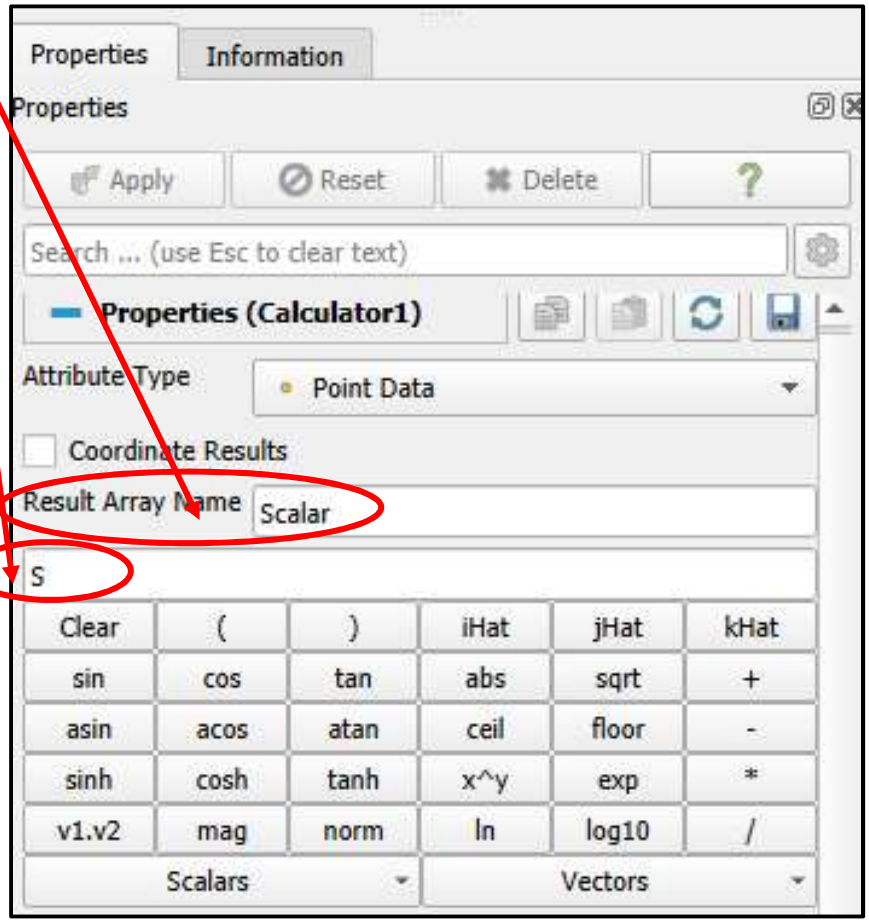

Z
Y

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



Add a Calculator filter parented by **TableToStructuredGrid.** The isosurface **Contour2** should be parented by the **Calculator**.

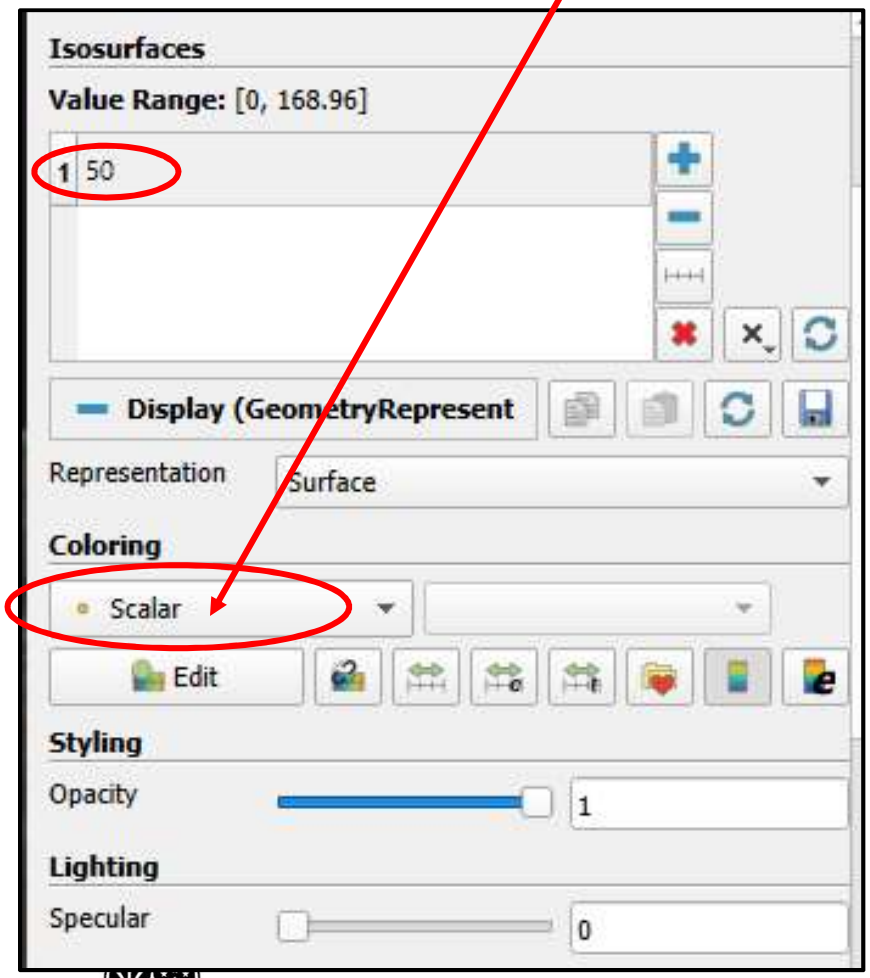
In the **Calculator**, this is like saying: ***Scalar = S***

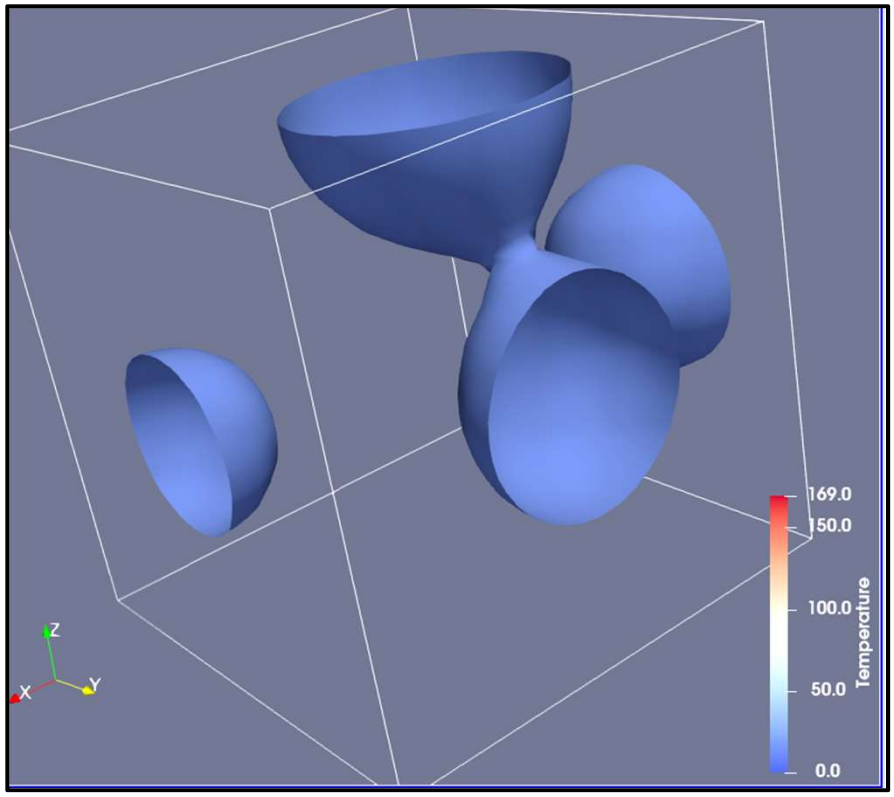


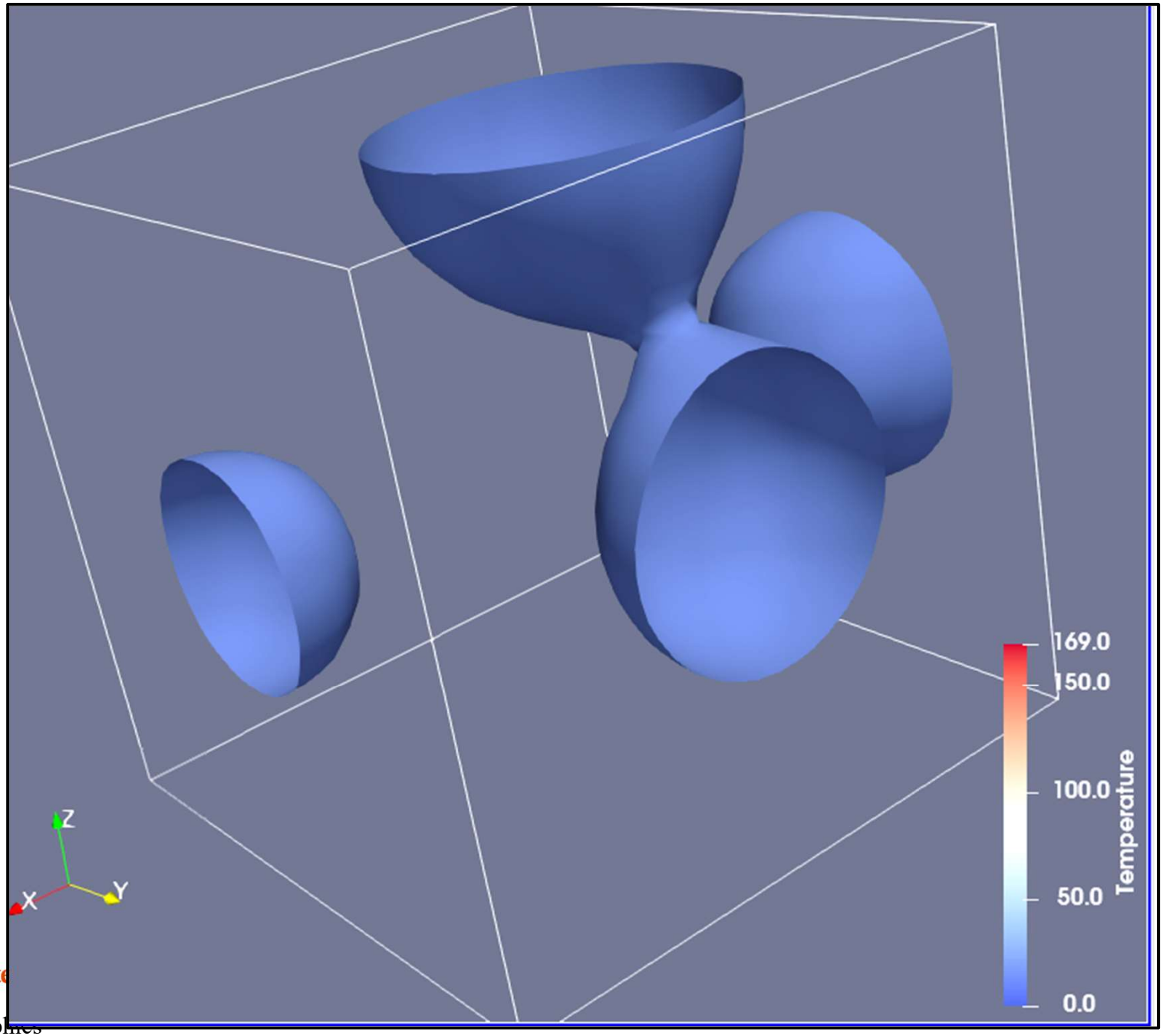
Oregon State University
Computer Graphics

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

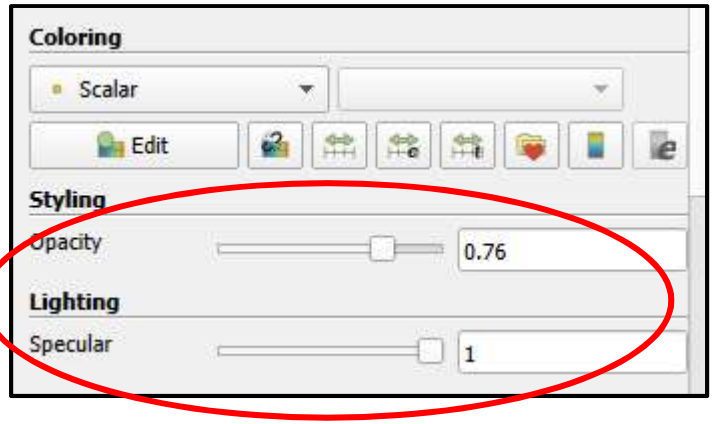
Now change the **Coloring** to color by **Scalar** instead of **S**.

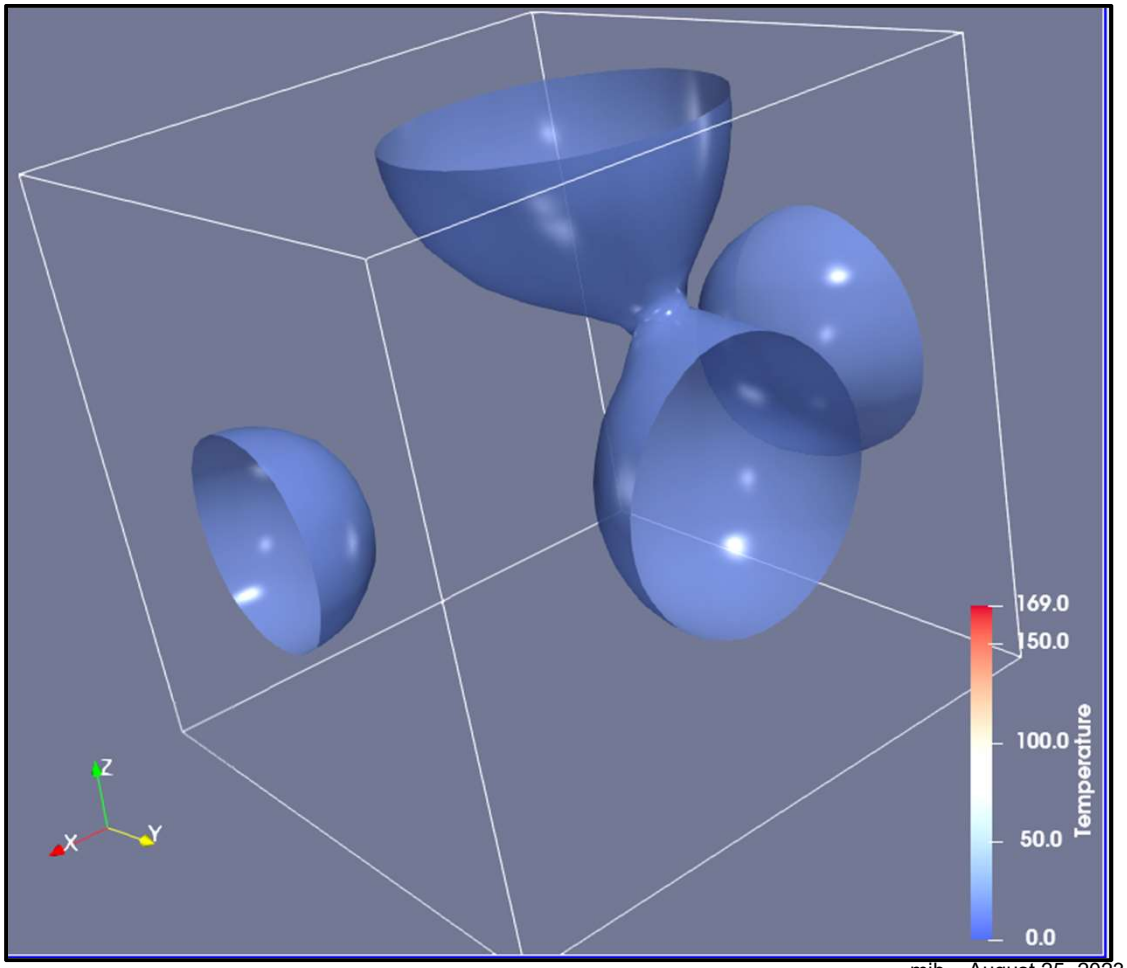




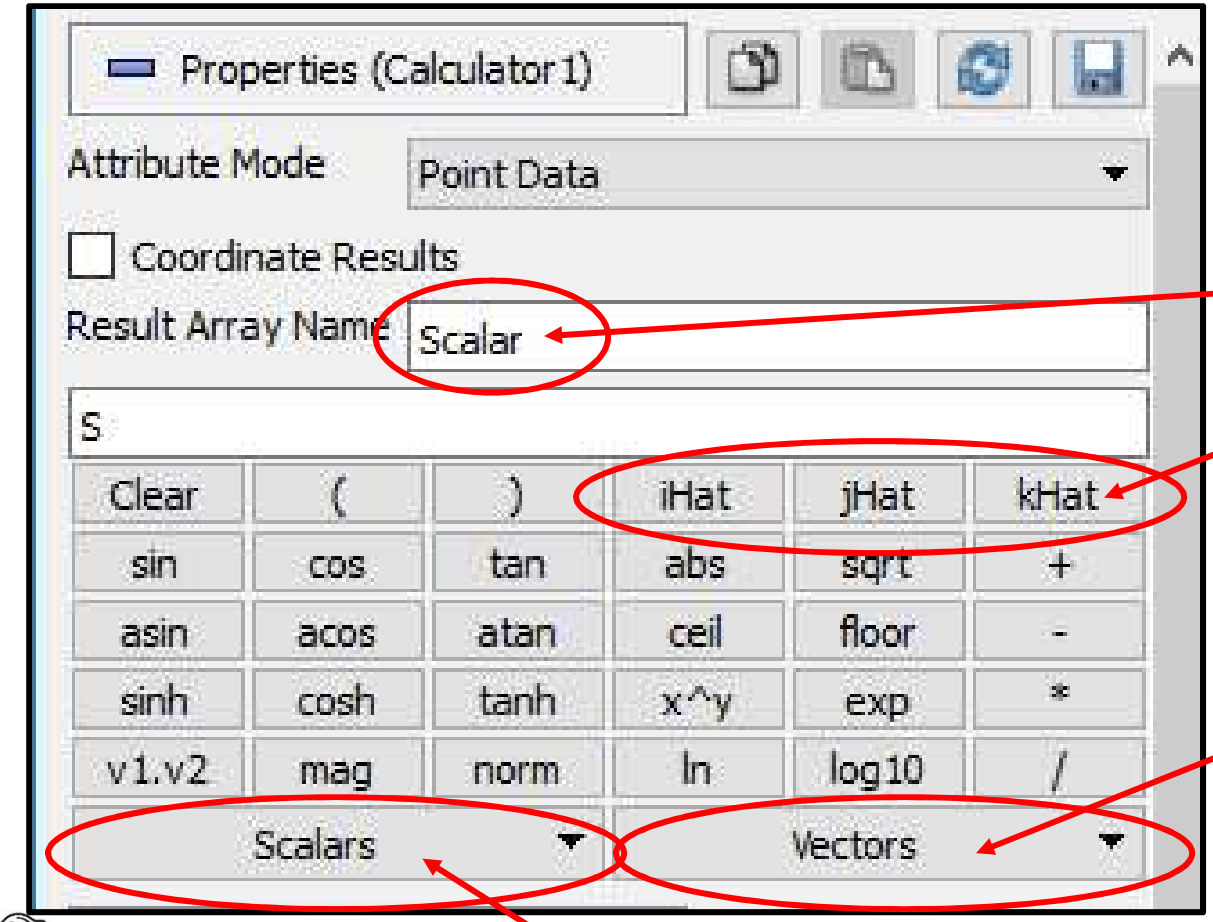

Z
X
Y
169.0
150.0
100.0
50.0
0.0
Temperature

Experimenting with the Opacity and Specular is Fun Too





How the Calculator Works



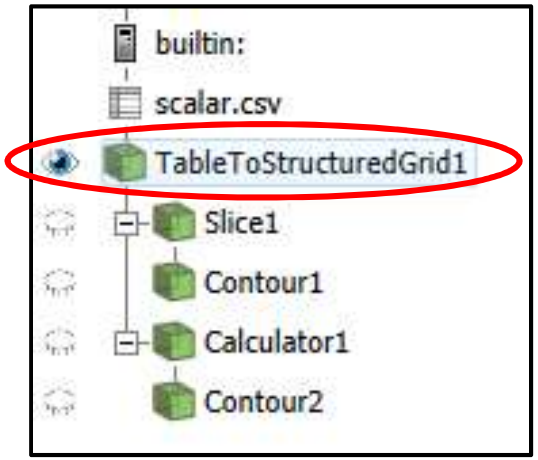
Name of the output field

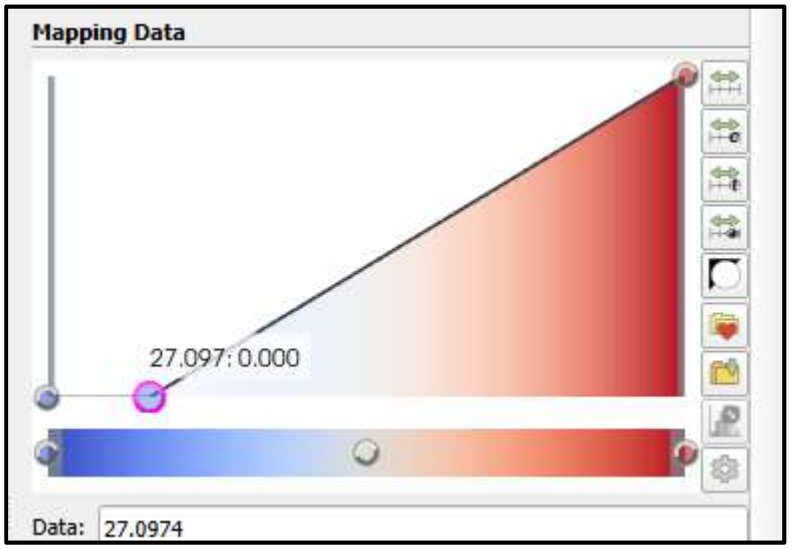
X, Y, and Z unit vectors

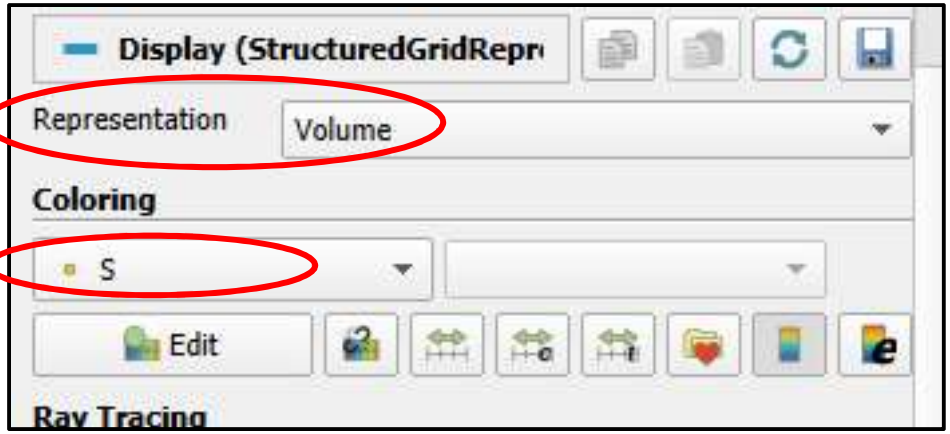
A list of the current vector variables in the dataset

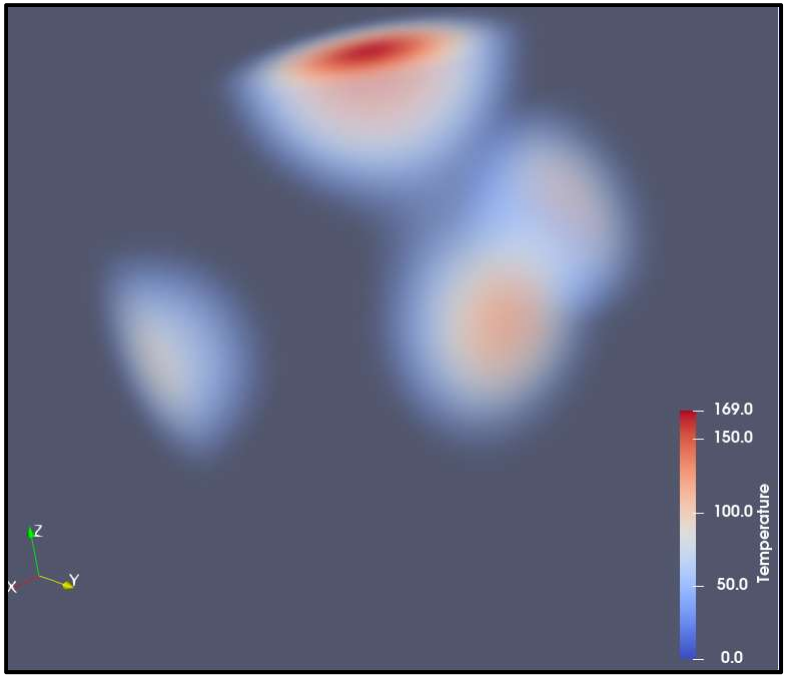
A list of the current scalar variables in the dataset

As a Volume


builtin:
scalar.csv
TableToStructuredGrid1
Slice1
Contour1
Calculator1
Contour2


Mapping Data
27.097: 0.000
Data: 27.0974

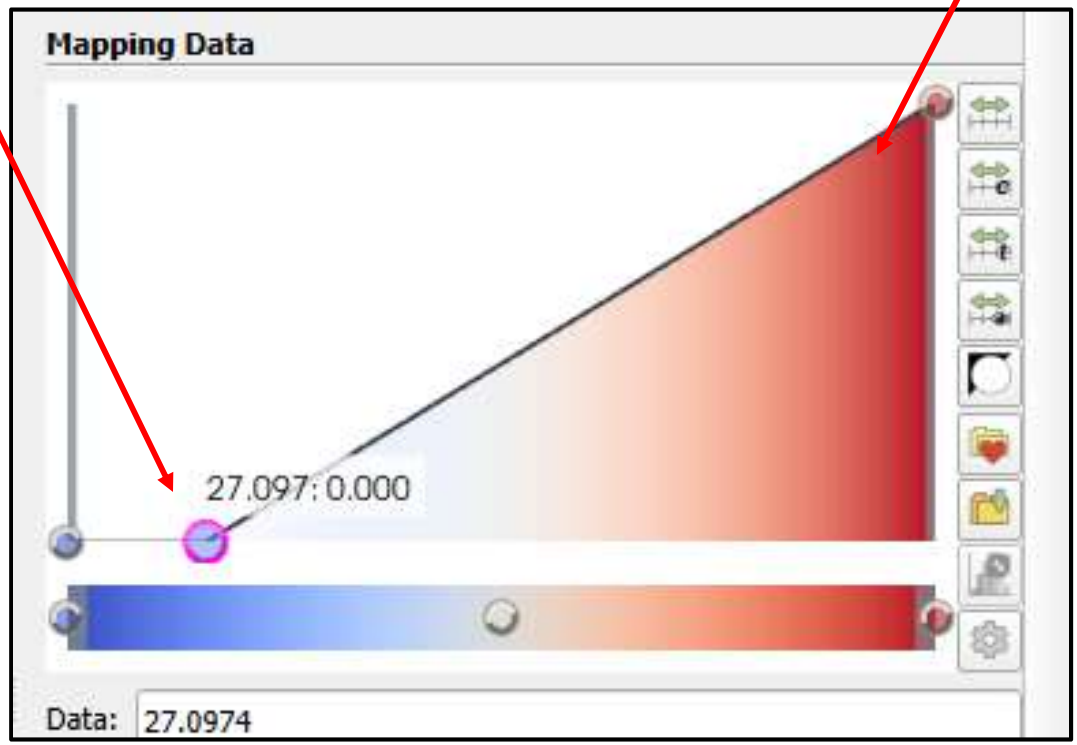

Display (StructuredGridRepr
Representation
Volume
Coloring
S
Edit
Ray Tracing


169.0
150.0
100.0
50.0
0.0
Temperature

Sculpting the Alpha Transfer Function

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

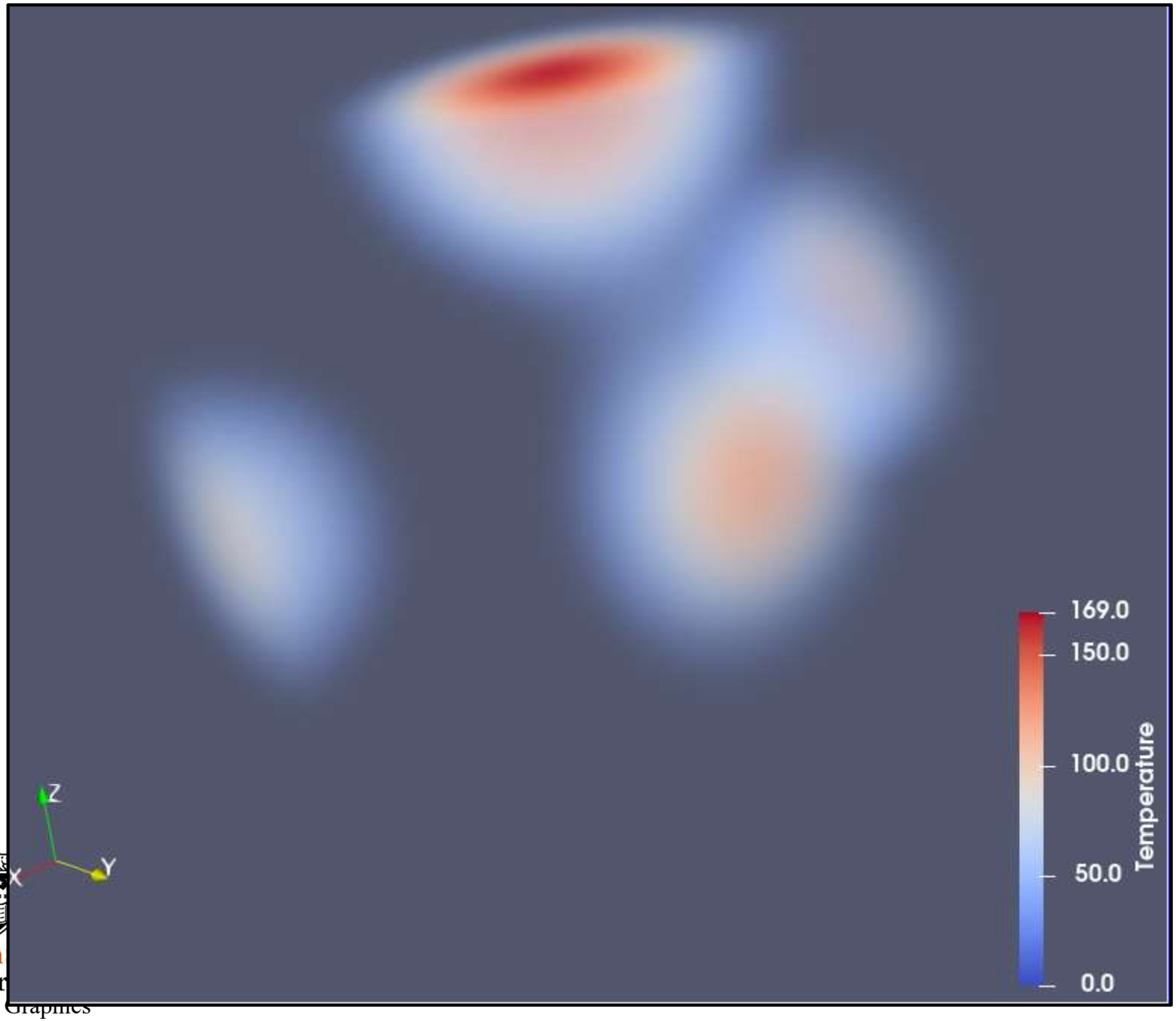
Hover over a point and hit the **Delete** key or **Middle Mouse Button** to delete a point



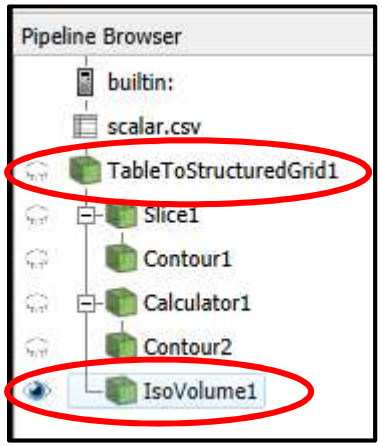
Alpha=1.
(opaque)

Alpha=0.
(transparent)

Data value range


169.0
150.0
100.0
50.0
0.0
Temperature
Z
X
Y

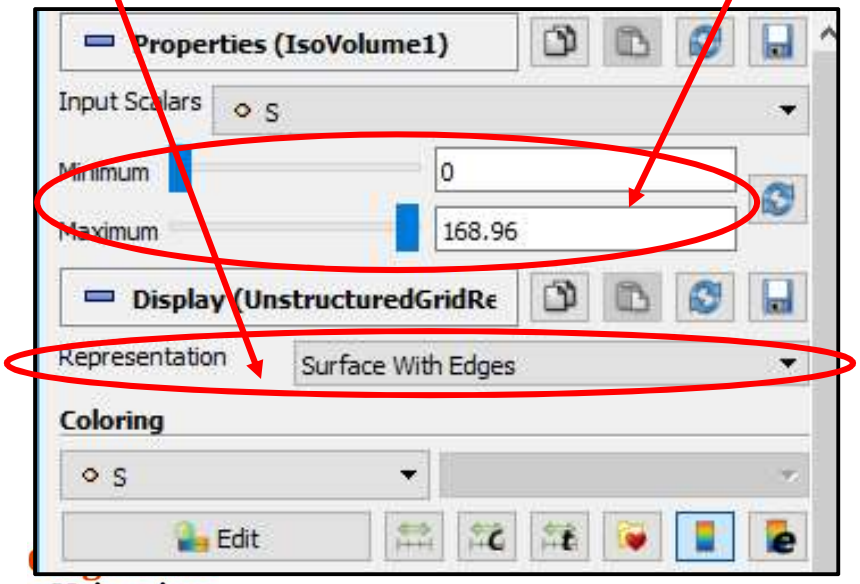
IsoVolumes

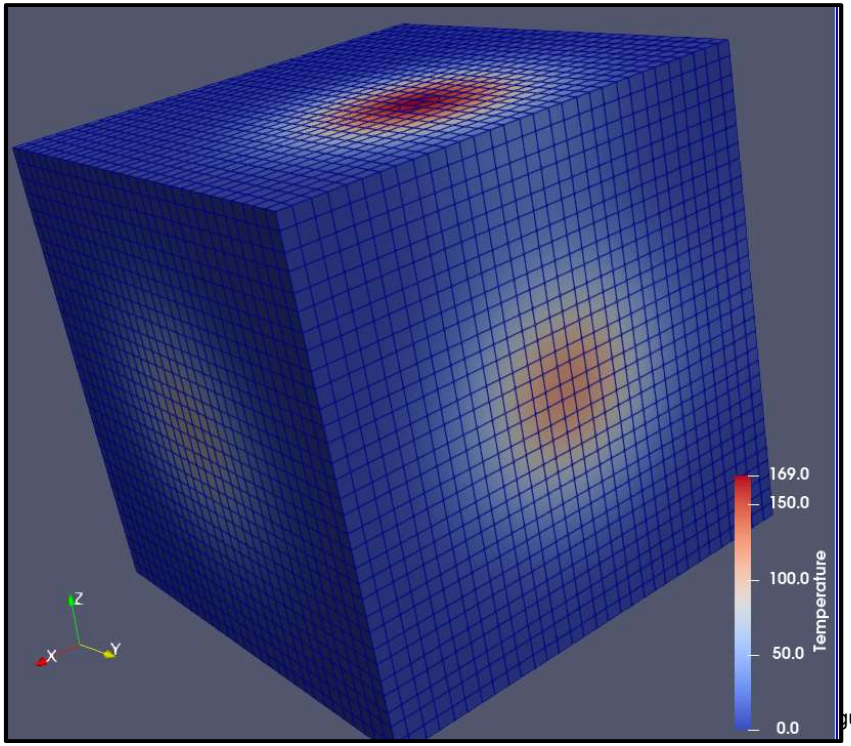


Start with this

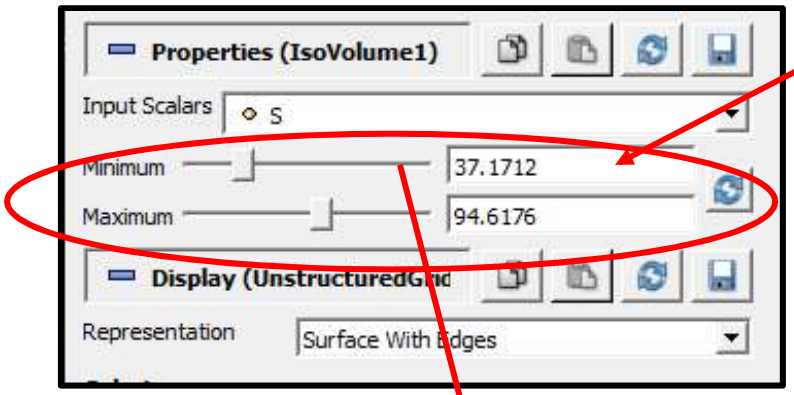
The **IsoVolume** properties start out at “allow all values” to pass through. We’re going to change this.

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

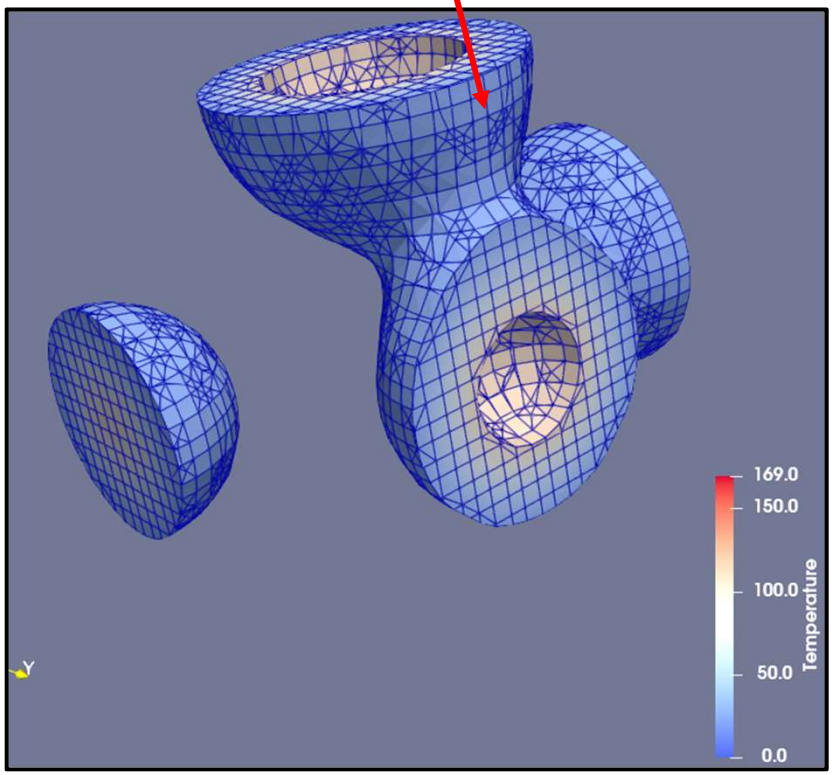




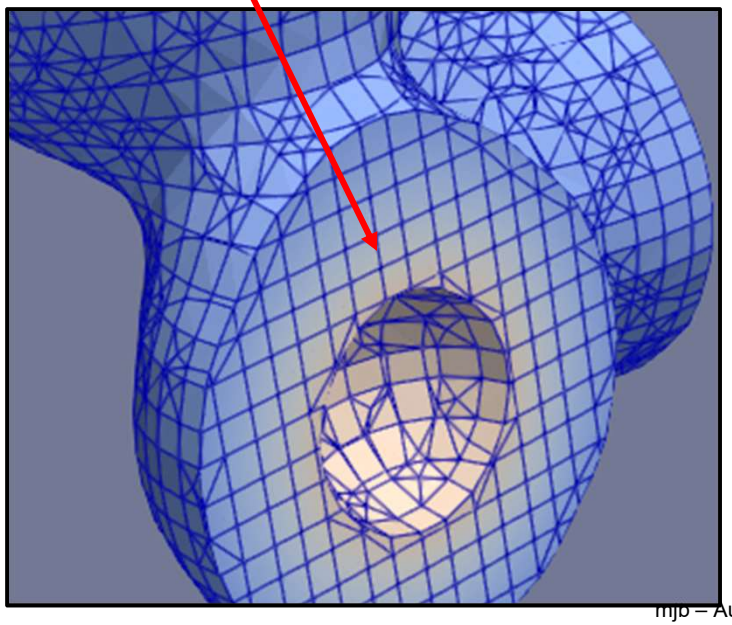
IsoVolumes



Now adjust the Minimum and Maximum to something else.

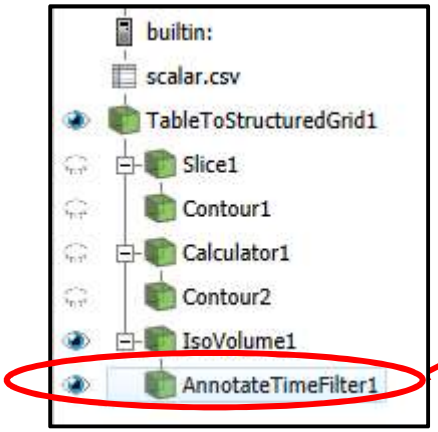


Note that the **IsoVolume** filter turned your nice, efficient structured grid into an unstructured grid. This can balloon the size of the data that is being operated on.



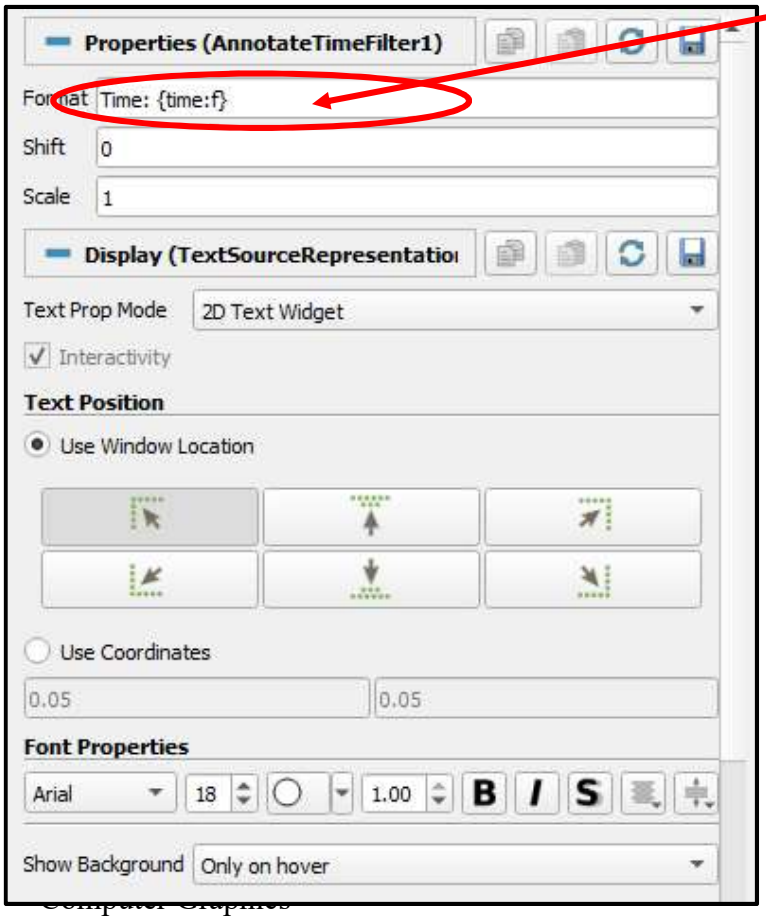
Annotating

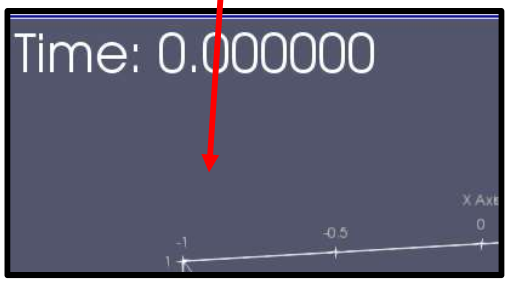
Adding Titles



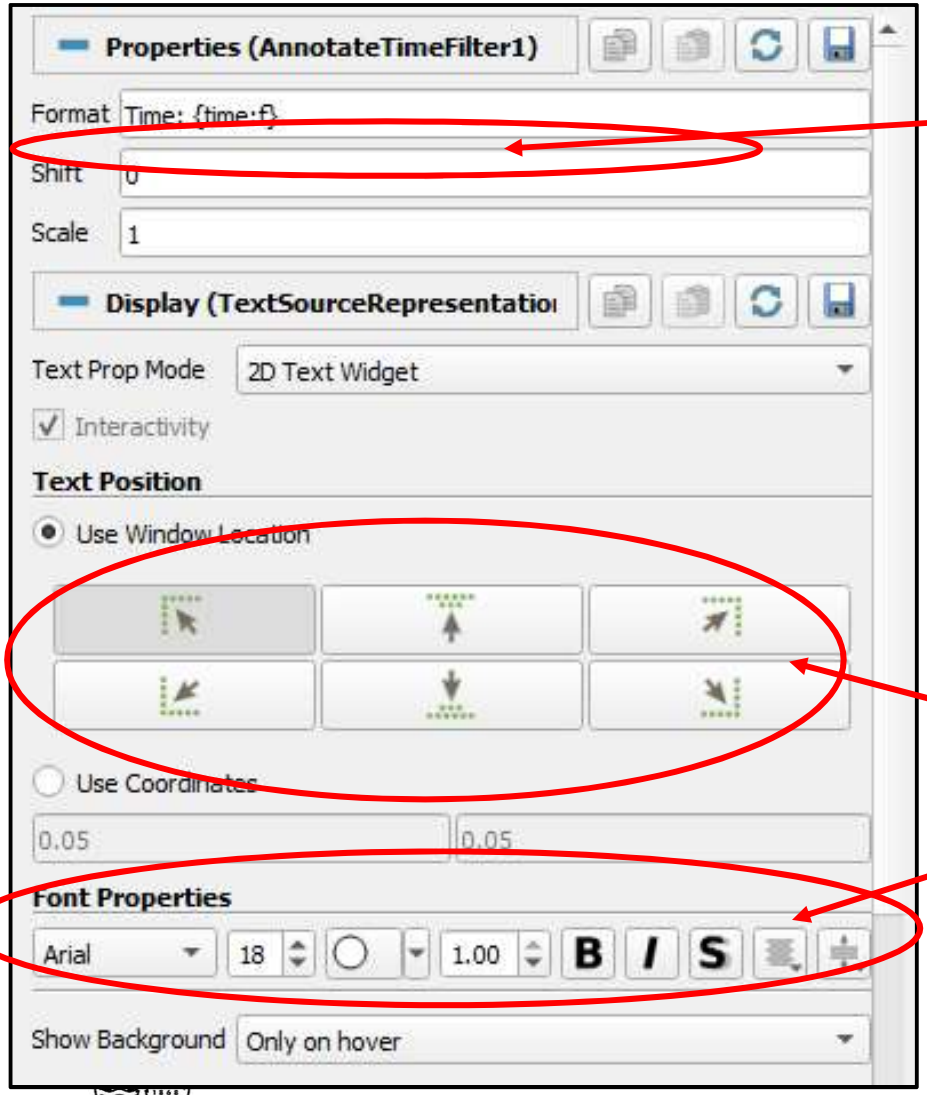
Add an **Annotate Time Filter** to the pipeline

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





Adding Titles

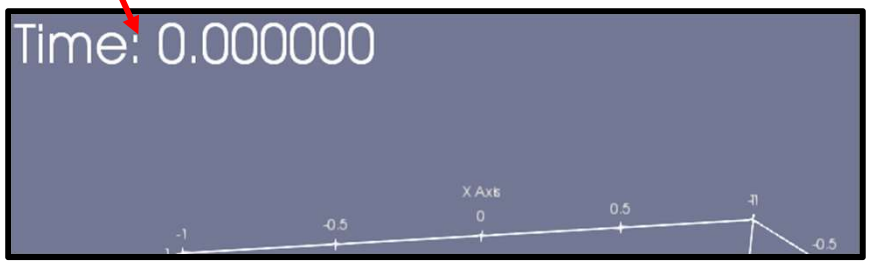


The label to use (the printf-notation is to format the Time – get rid of this if you just want a title)

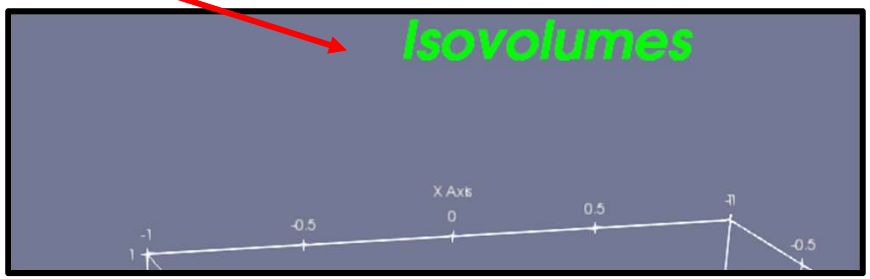
The position for the title

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

From this:



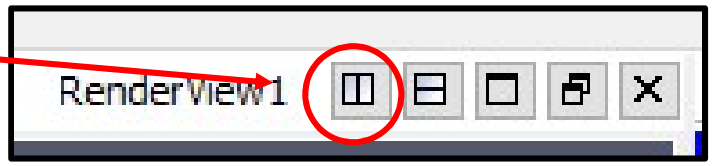
to this:



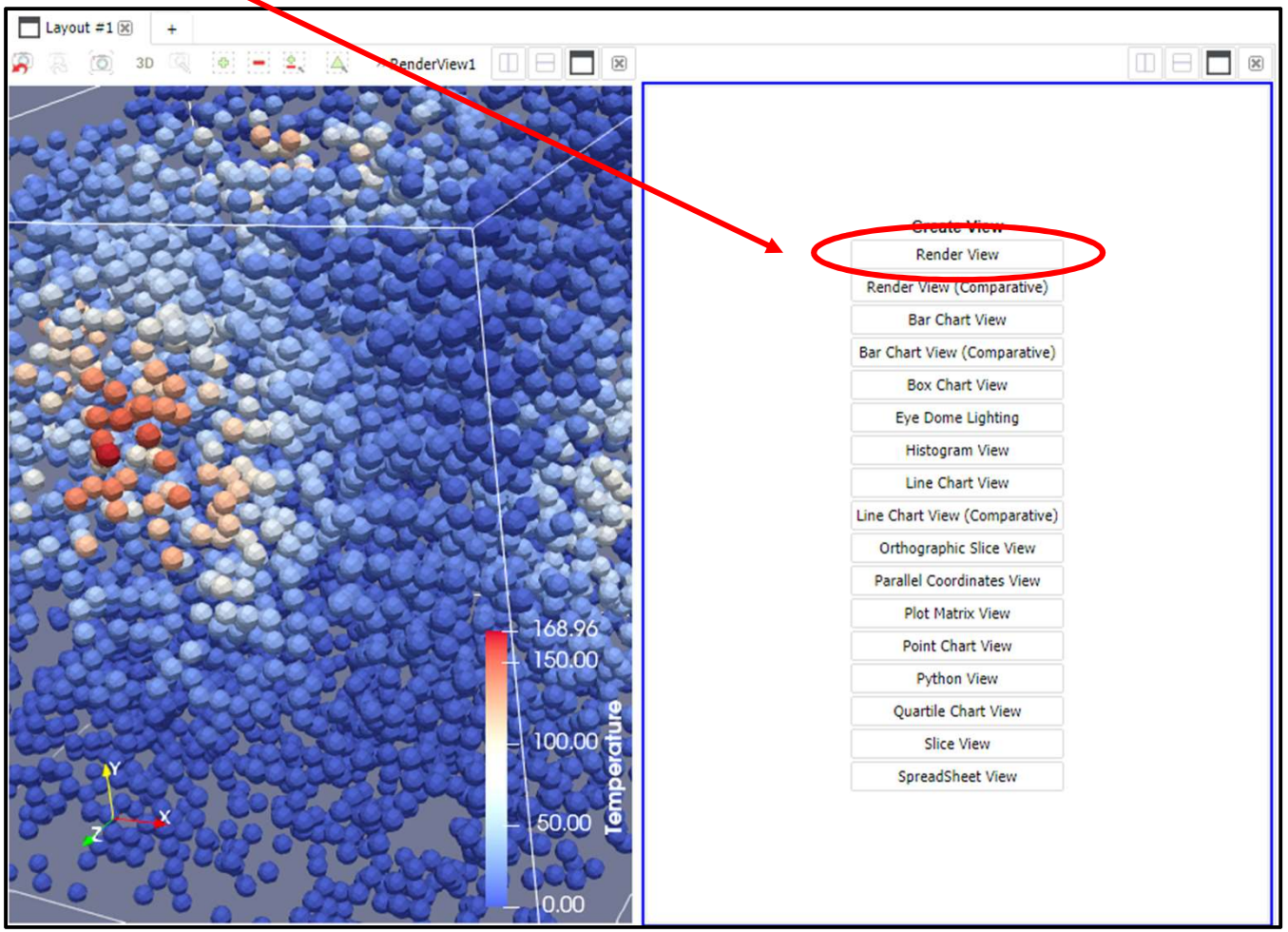
Multiple Views

Multiple Views

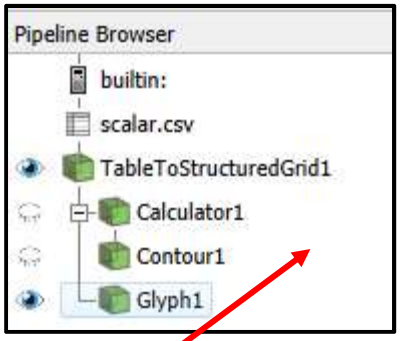
Step #1: Split the Window

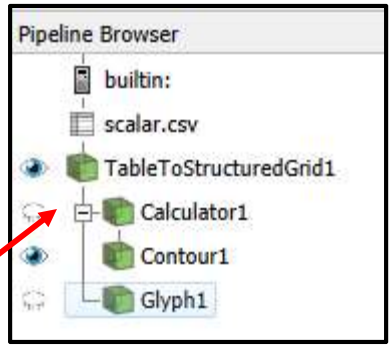


Step #2: Click on **Render View**



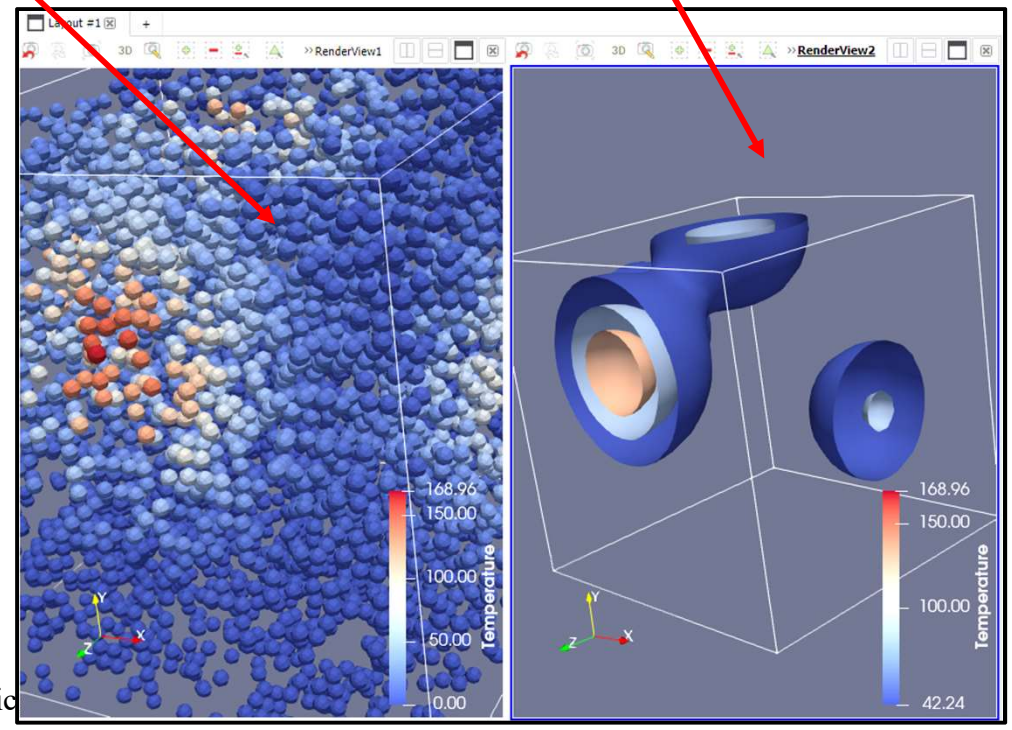
Multiple Views





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

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

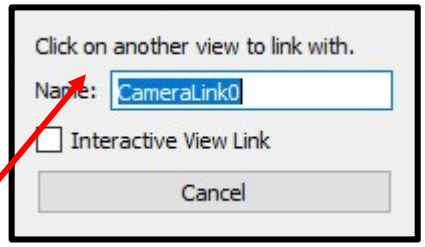


… and, you get this – with each Window being allowed its own viewing transformation

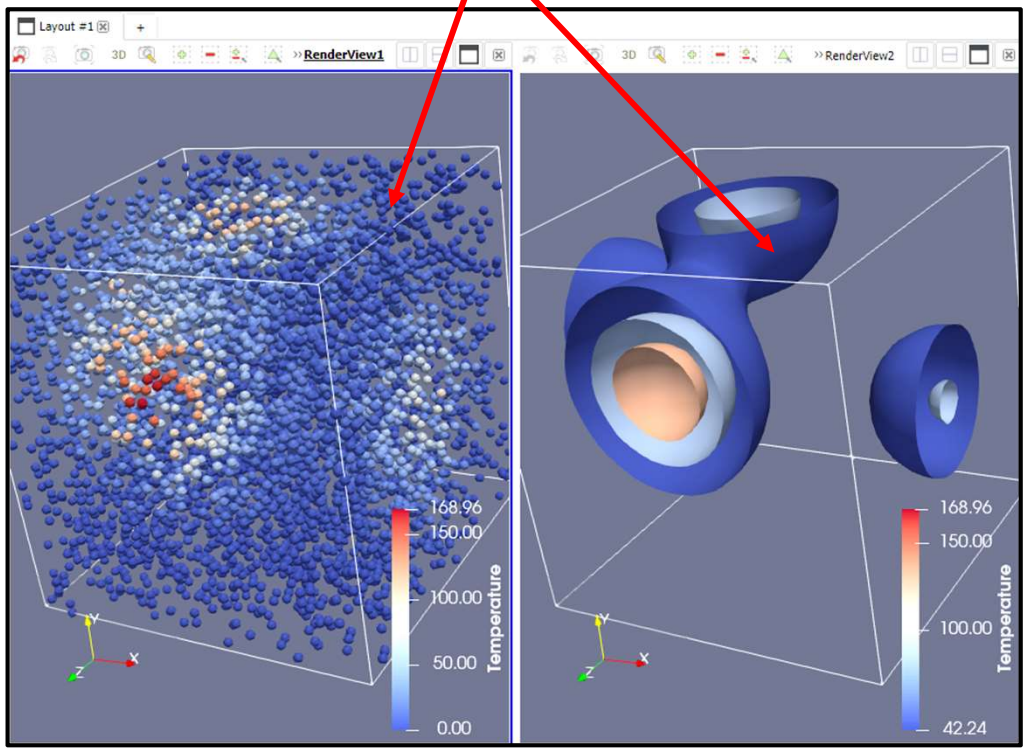
Multiple Views with Linked Viewing Transformations

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



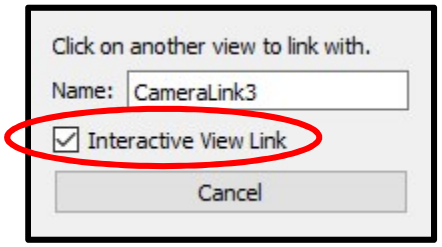
Your Windows now share a single transformation



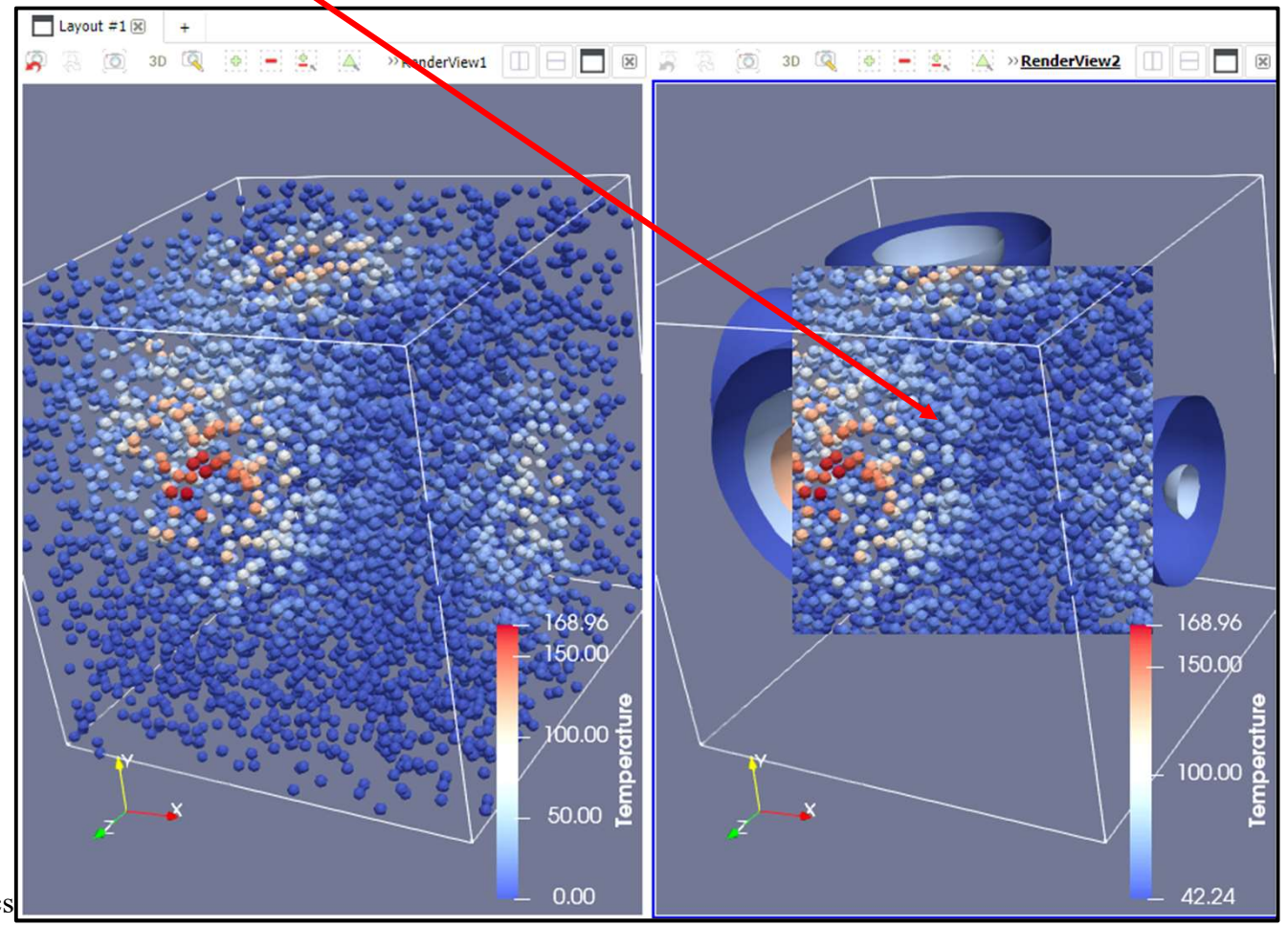
Oregon State University
Computer Graphics

Multiple Views with Linked Viewing Transformations

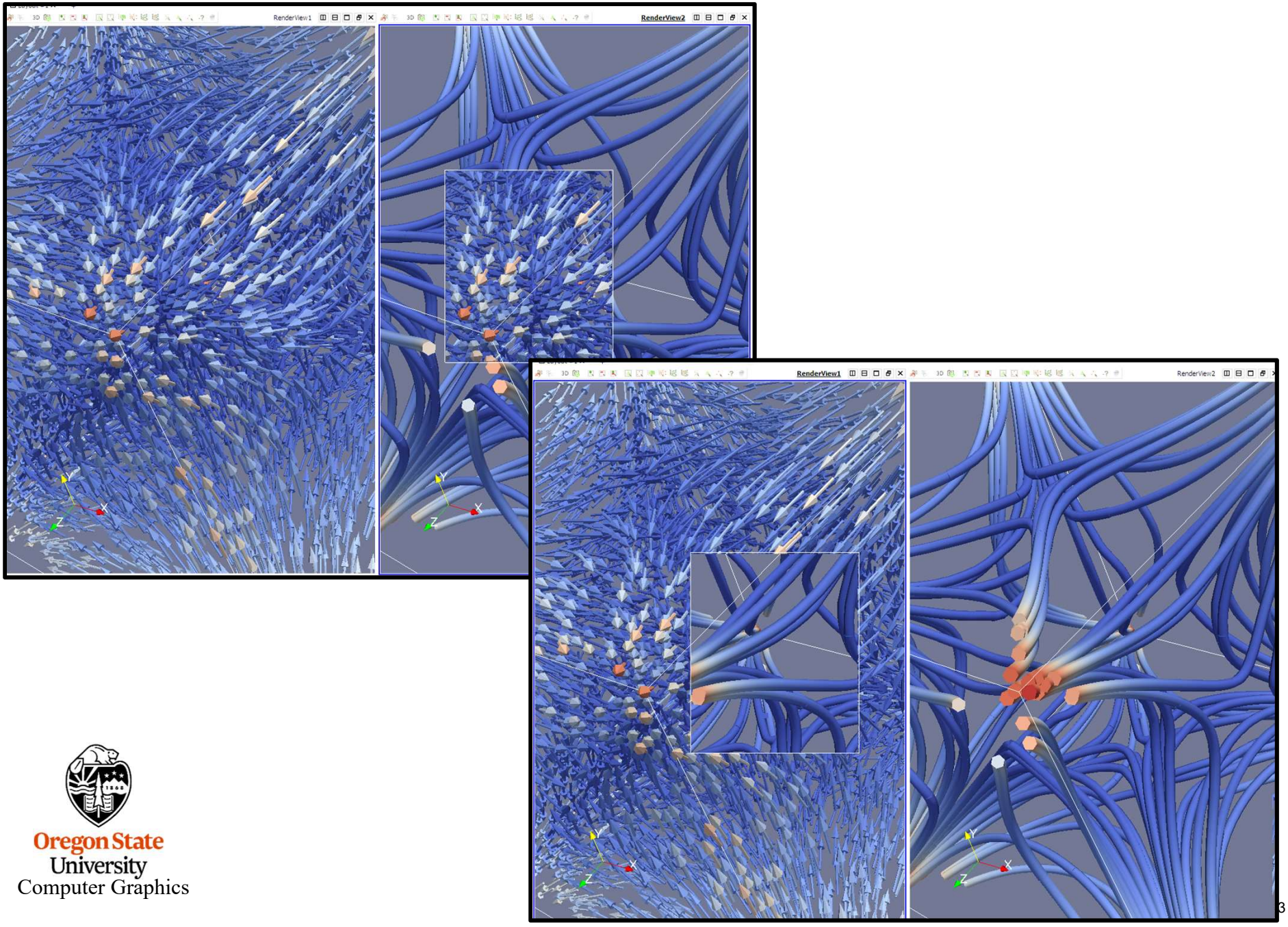
If you click on this checkbox and then click in another Window …



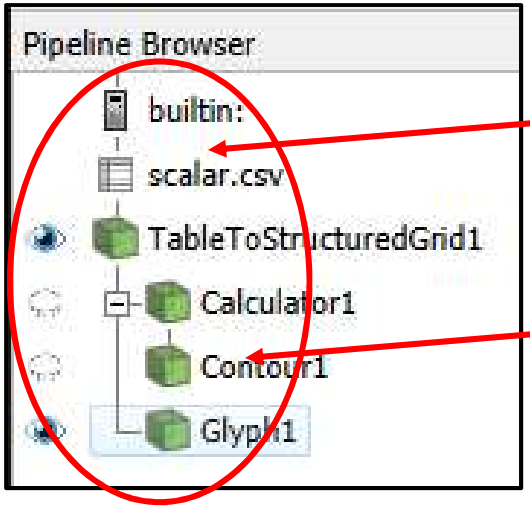
… you get a Magic Lens

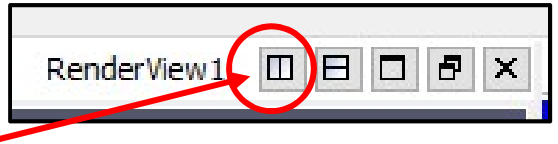


Order Matters!

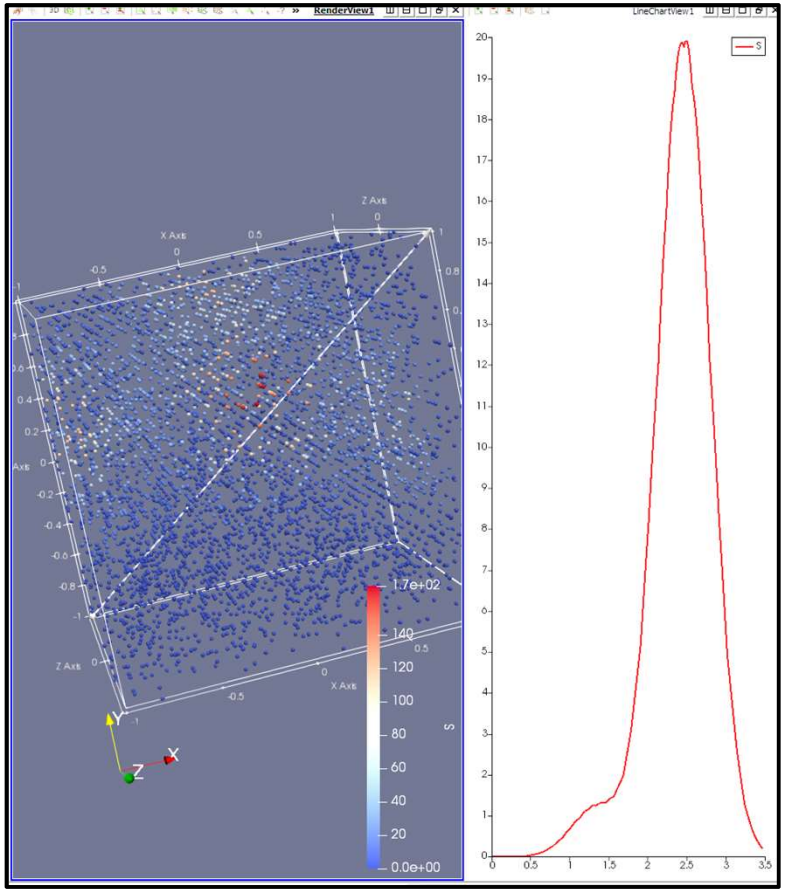


Using Plot Over Line



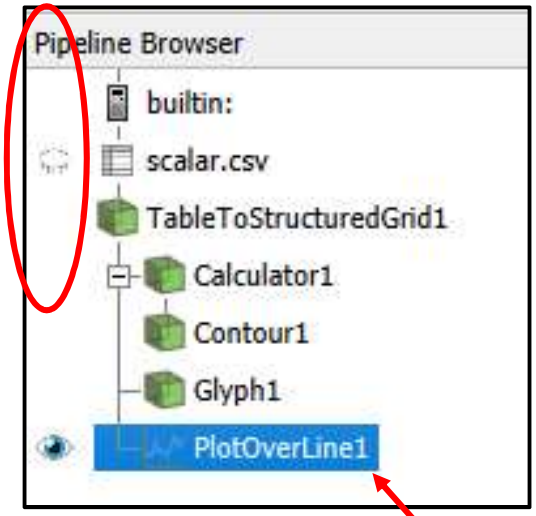


1. Create this Pipeline
2. Split the Render window and ask for a **Line Chart View**
3. When you click in the Render window, make the eyeballs look like this, with the **TableToStructuredGrid** representation set to **Outline** and the **Glyph** representation set to **Surface**



Oregon State University
Computer Graphics

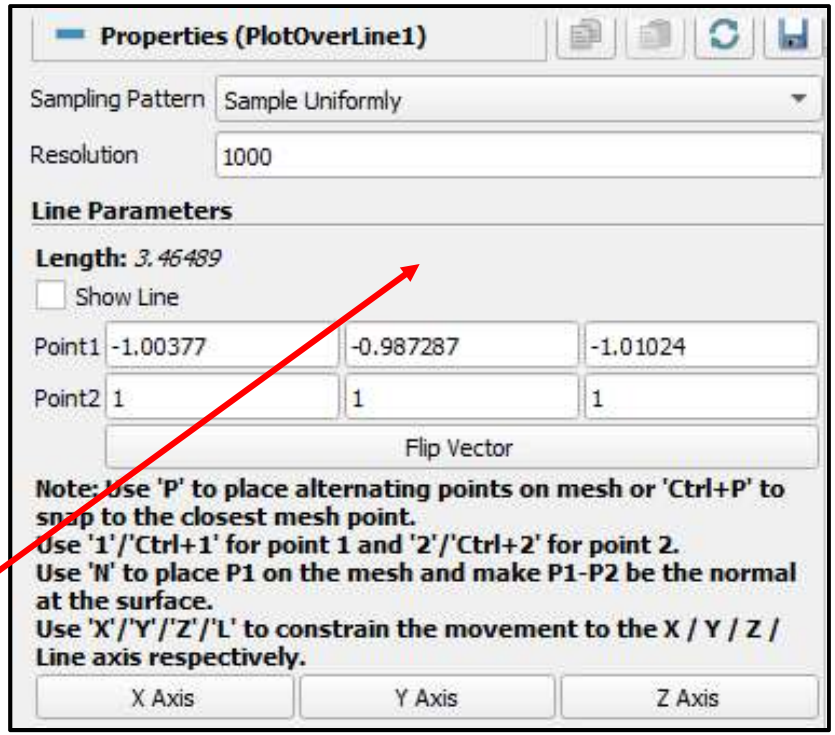
Using Plot Over Line



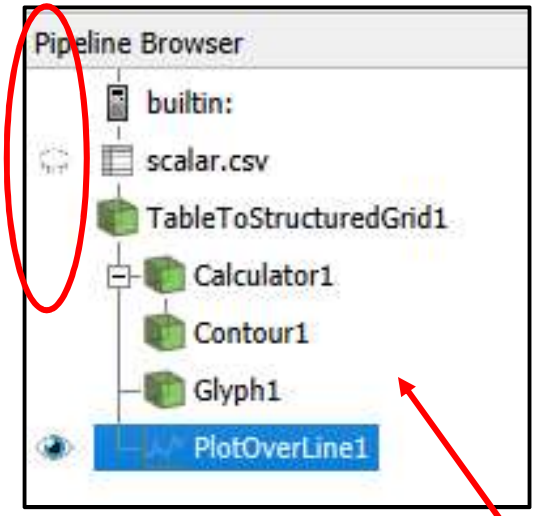
1. Click in the **Line Chart View** window. Add a **PlotOverLine** filter that is parented to the **TableToStructuredGrid**

2. Setup the PlotOverLine Properties like this

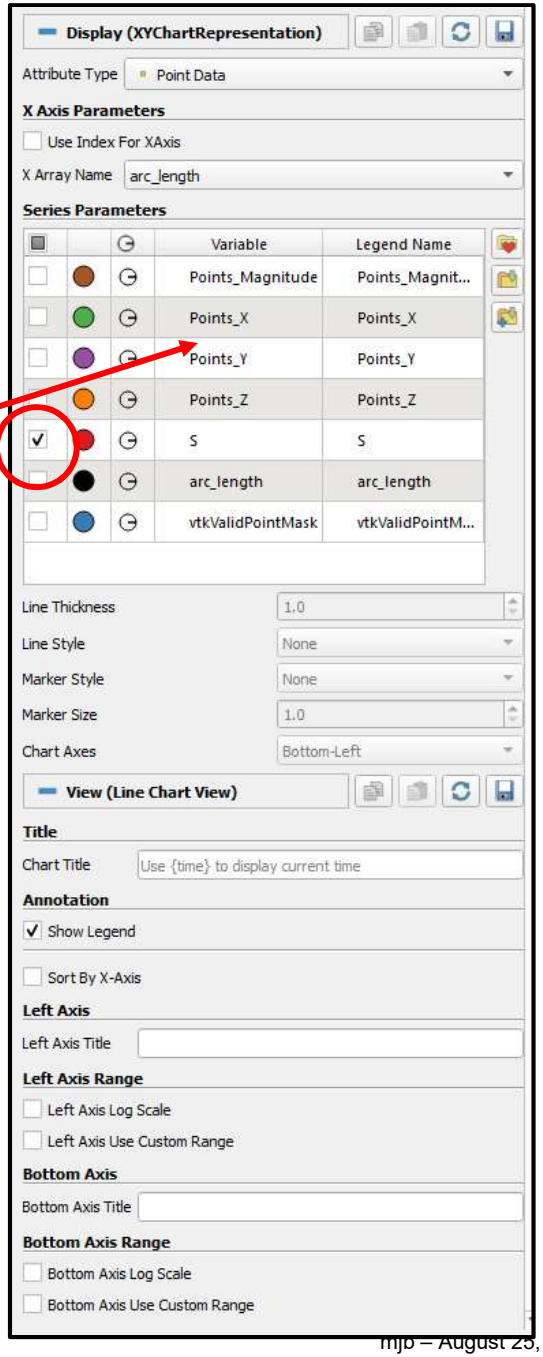
3. Be sure **Auto-Apply** is turned on



Using Plot Over Line

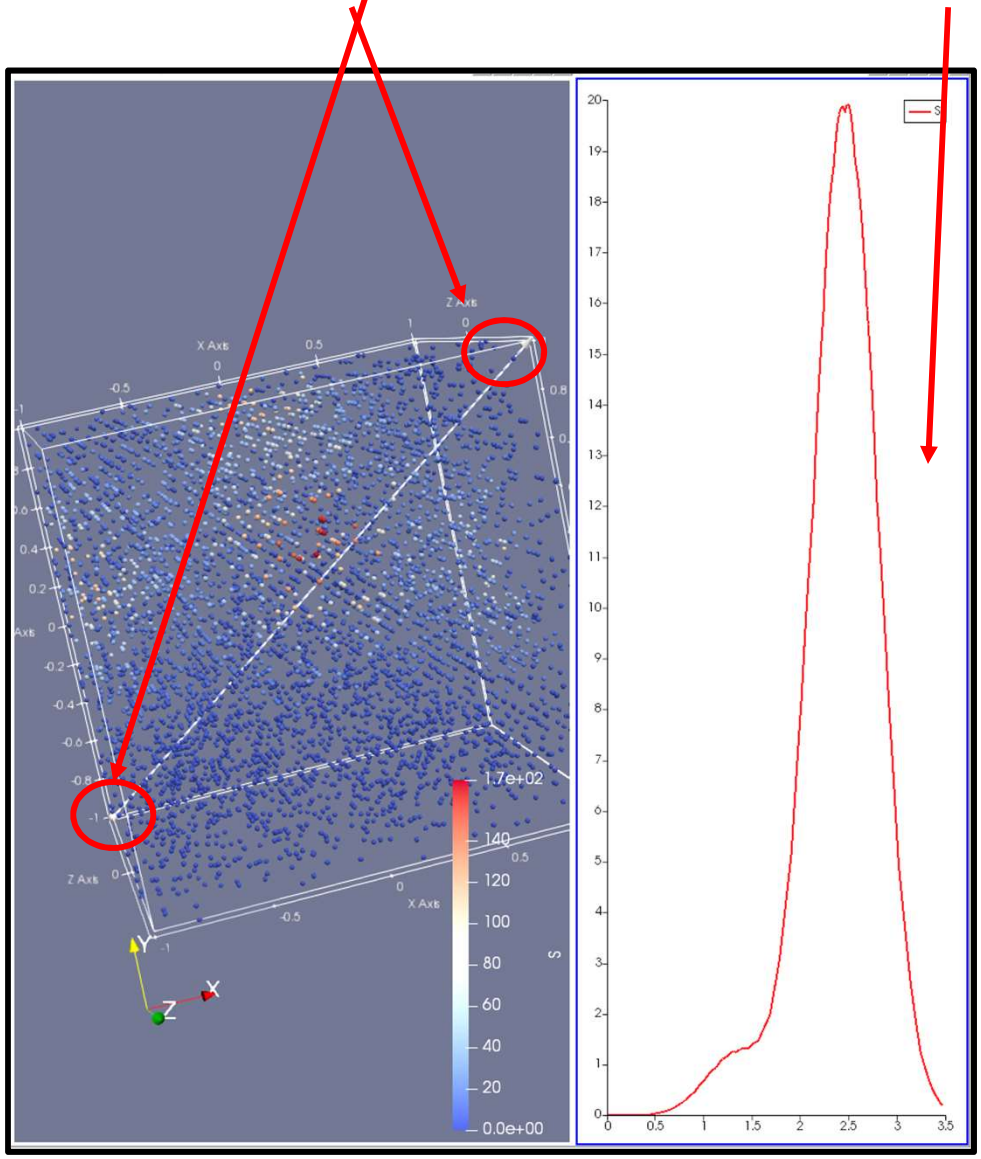


Setup the **Display (XYChartRepresentation)** Properties like this



Using Plot Over Line

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



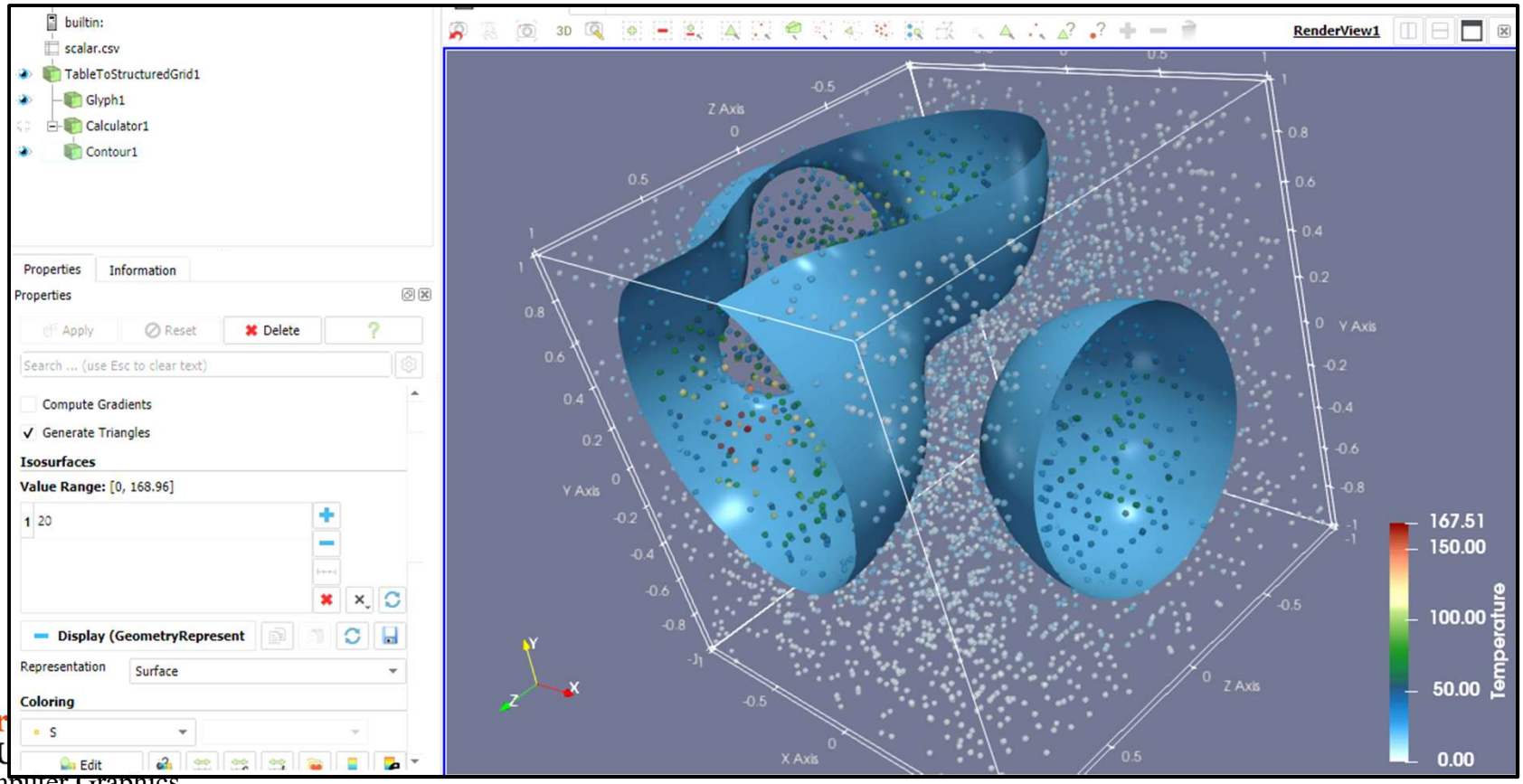
Comparative Visualization

scalarcompare.pvsm

Comparative Visualization

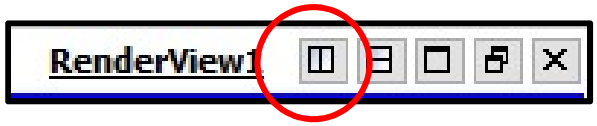
ParaView can setup a side-by-side visualization comparison with different vis parameters in each view.

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

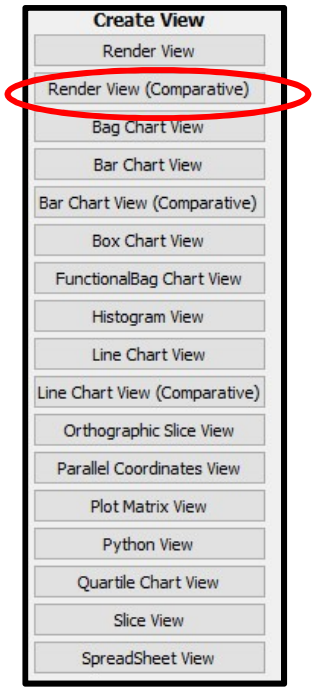


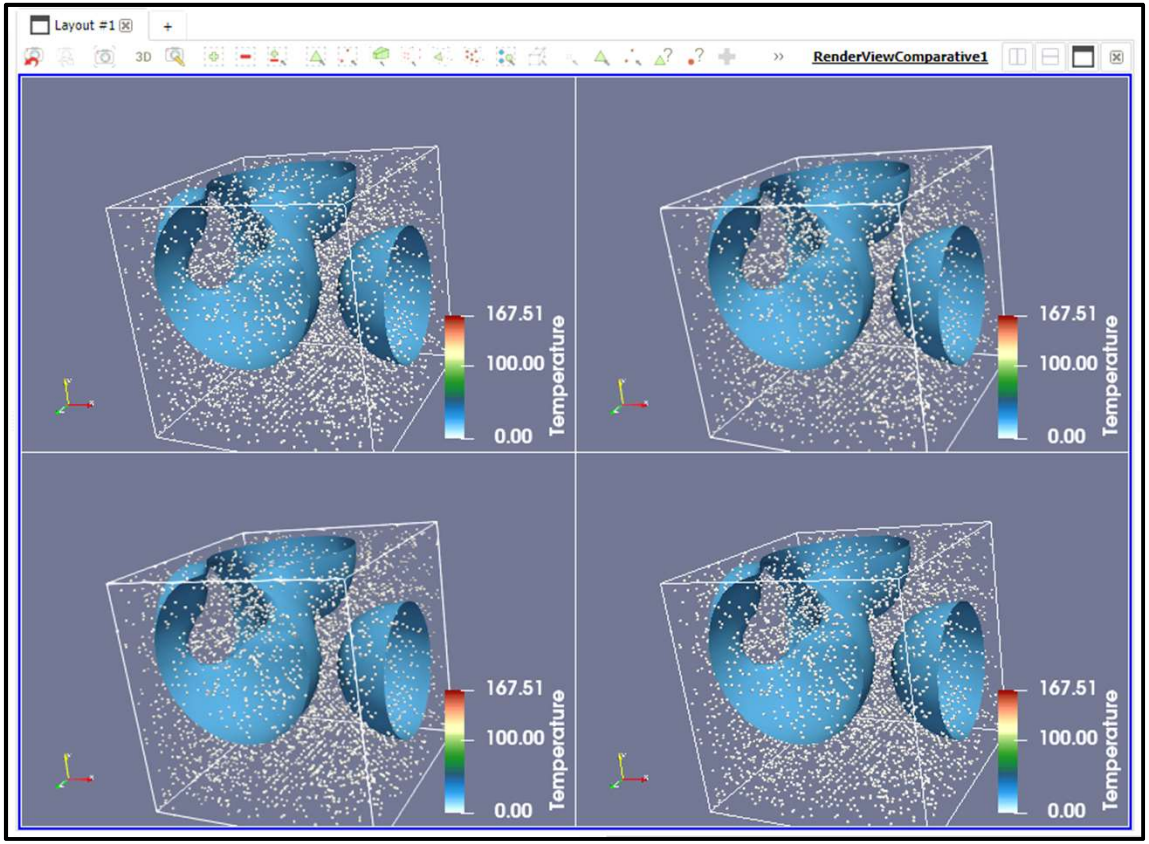
Comparative Visualization

Now, split the window

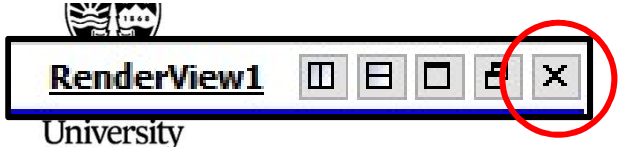


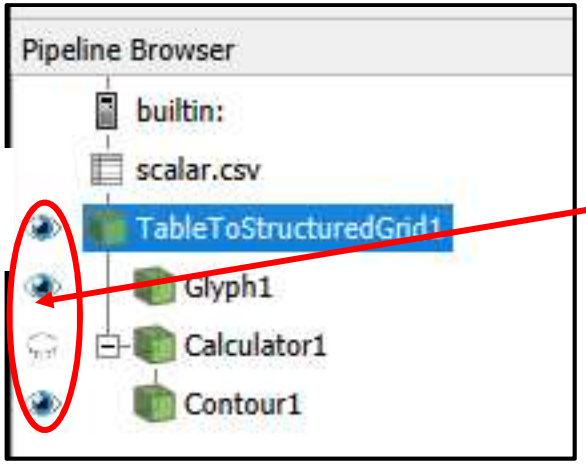
and select:
Render View (Comparative).





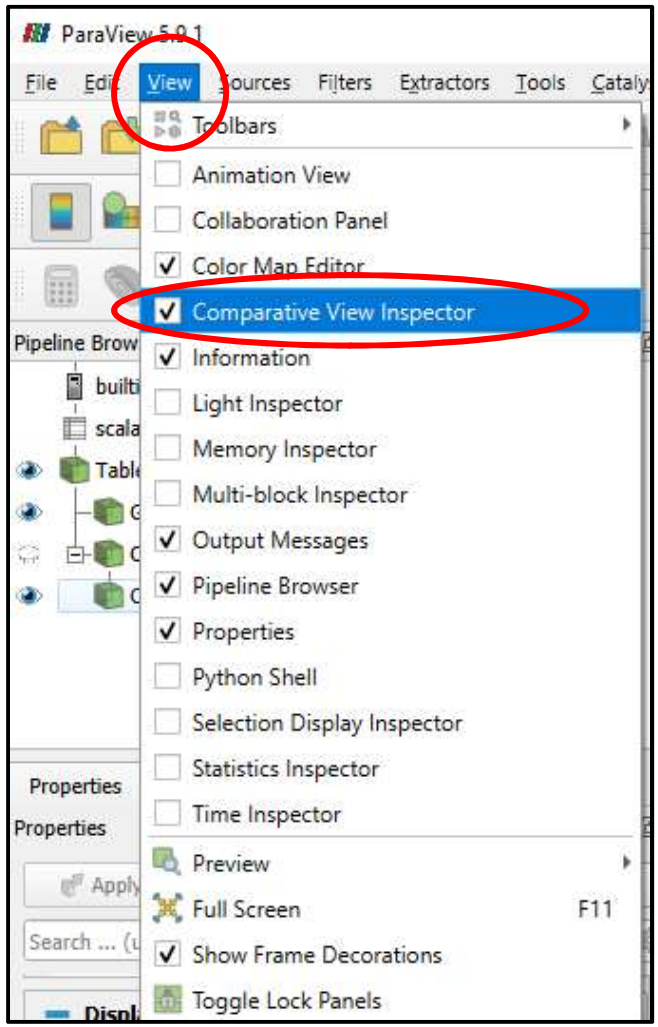
You can now eliminate the left-hand window if you want.

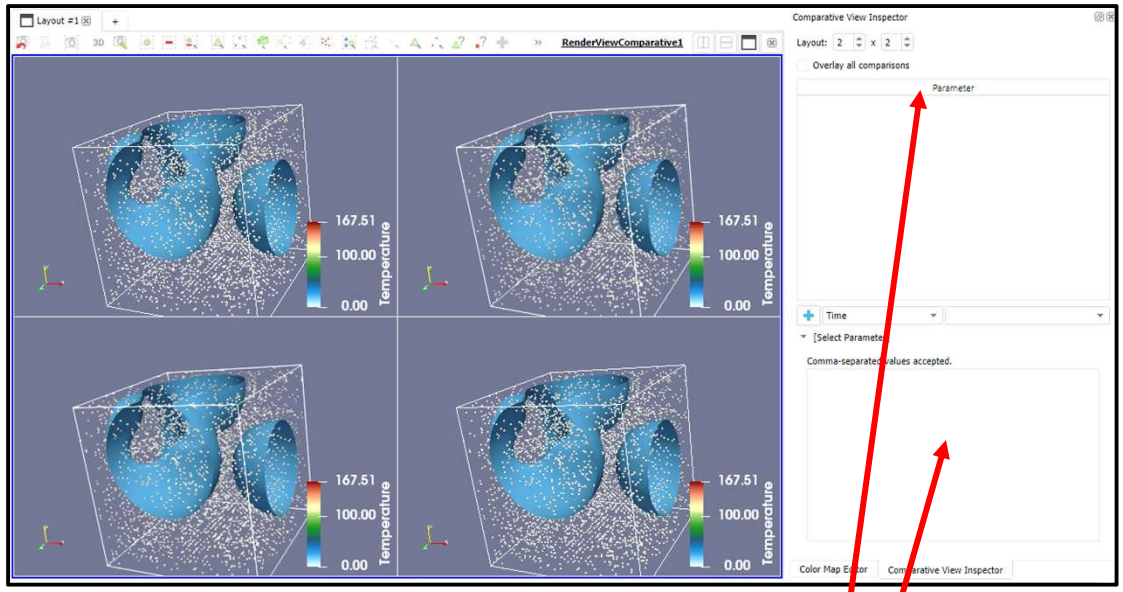




Click all the eyeballs on for the visualization features you want to see.

Comparative Visualization





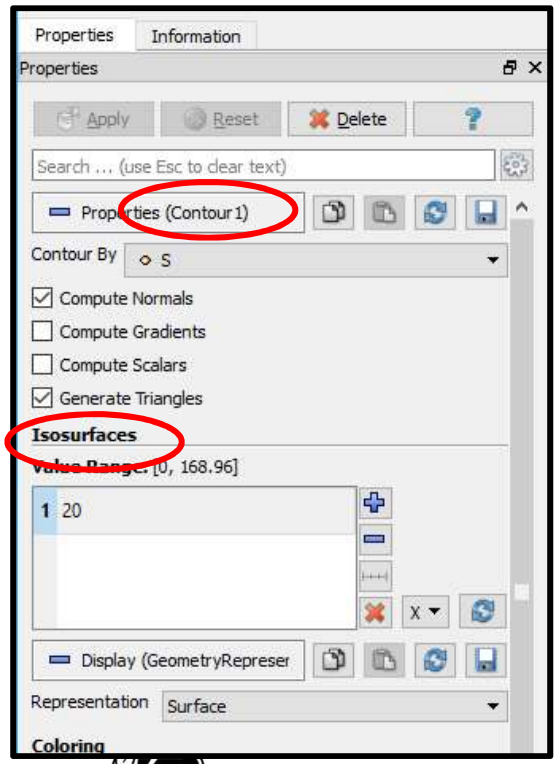
Select **View** → **Comparative View Inspector**

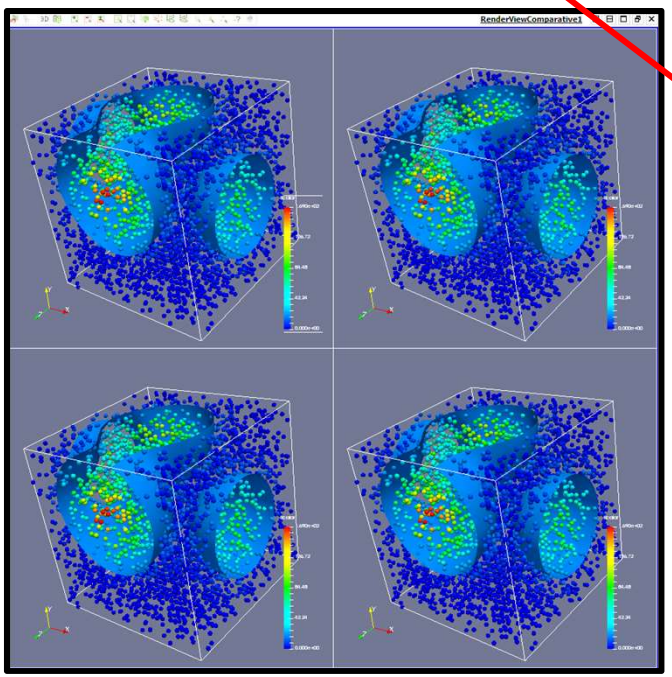
These two areas are created

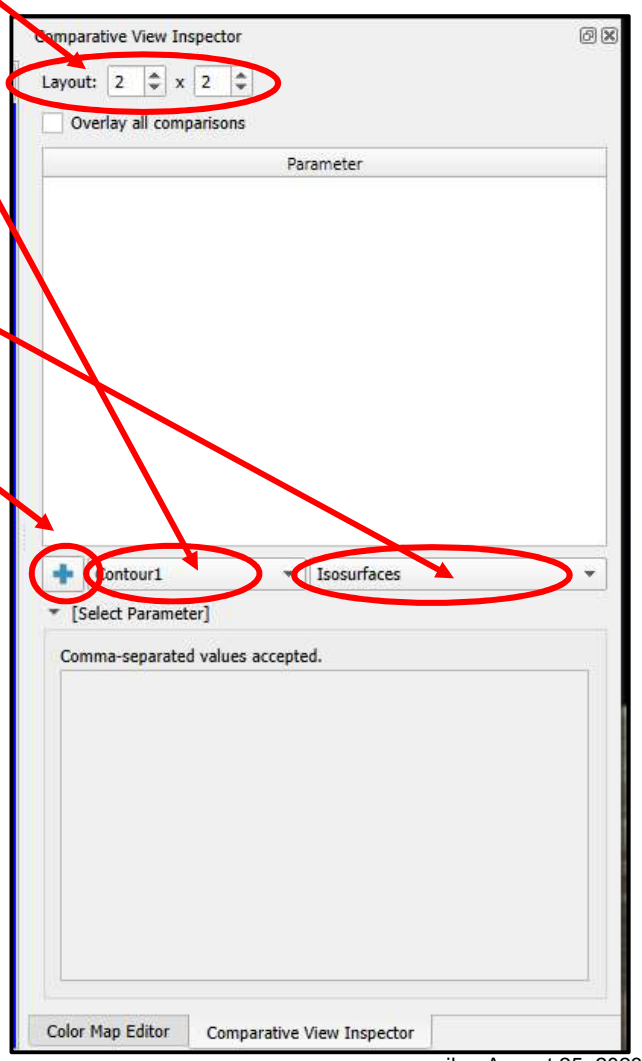
Comparative Visualization

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

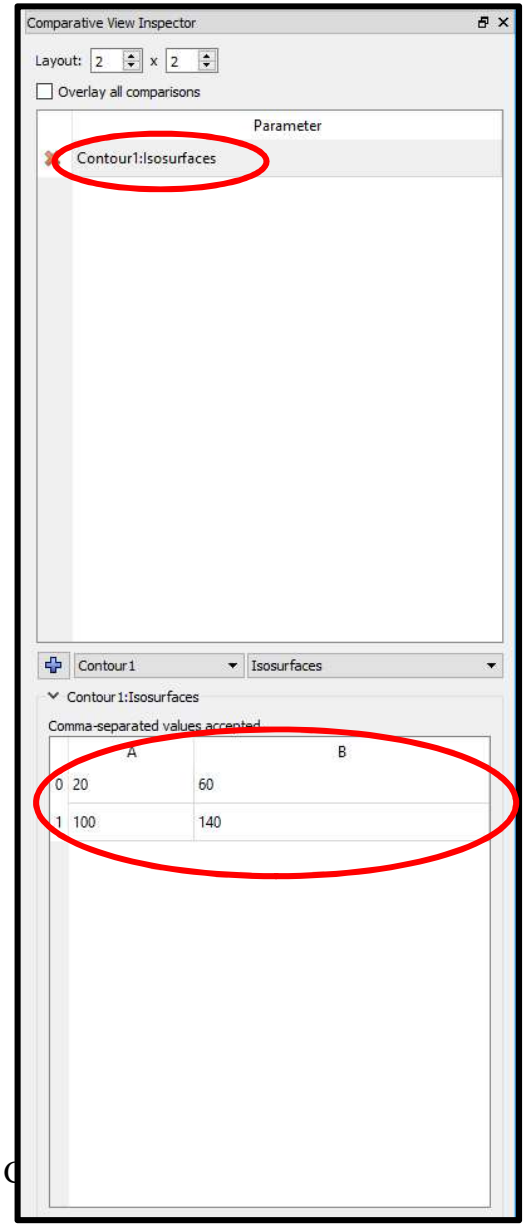
1. Select the layout dimensions of the comparative window grid
2. Select the pipeline module that owns the parameter
3. Select the parameter
4. Hit the Big Plus Sign







Comparative Visualization

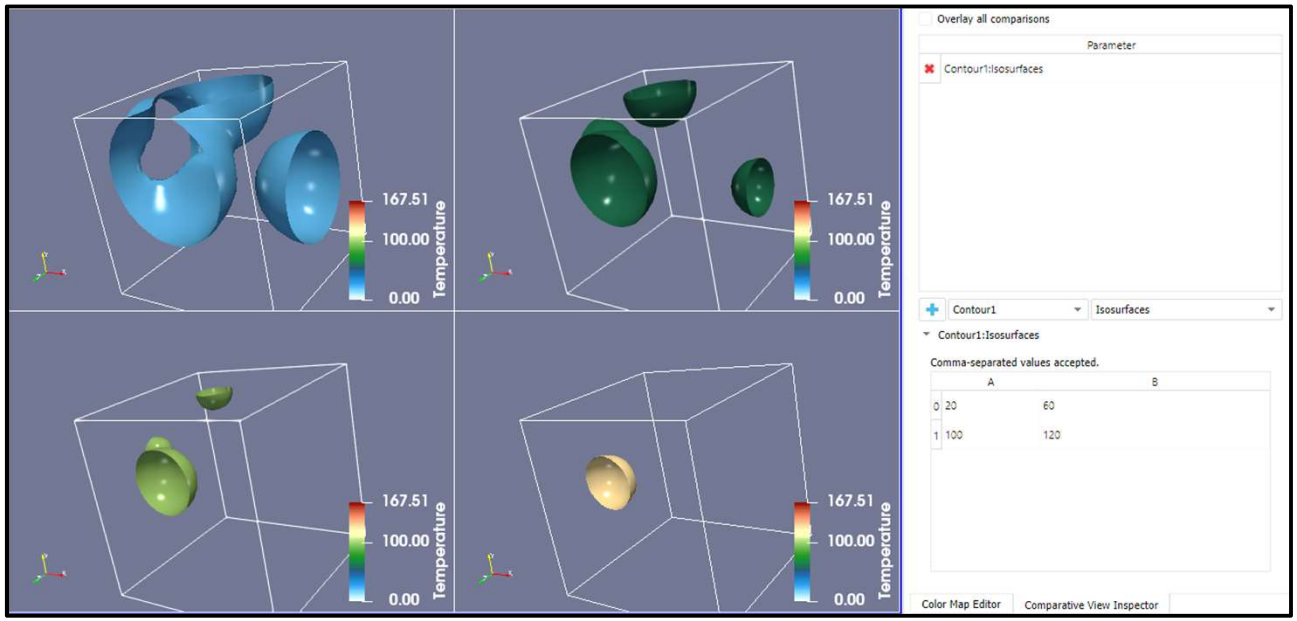


ParaView stocks the number grid with evenly-spaced values and applies them to each visualization, respectively.

Usually, these are not what you wanted to see. But you can type your own numbers in each cell

(I eliminated the Glyphs to better see the isosurfaces)

The windows are all transform-linked

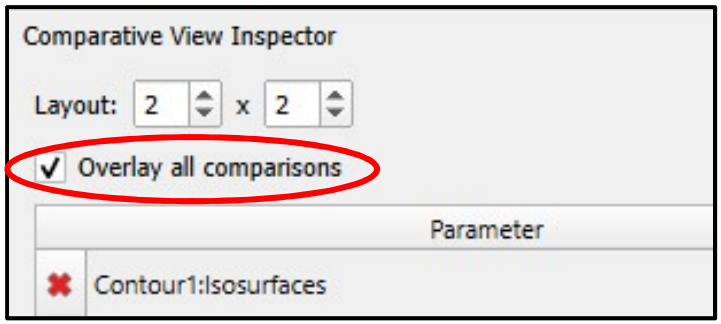


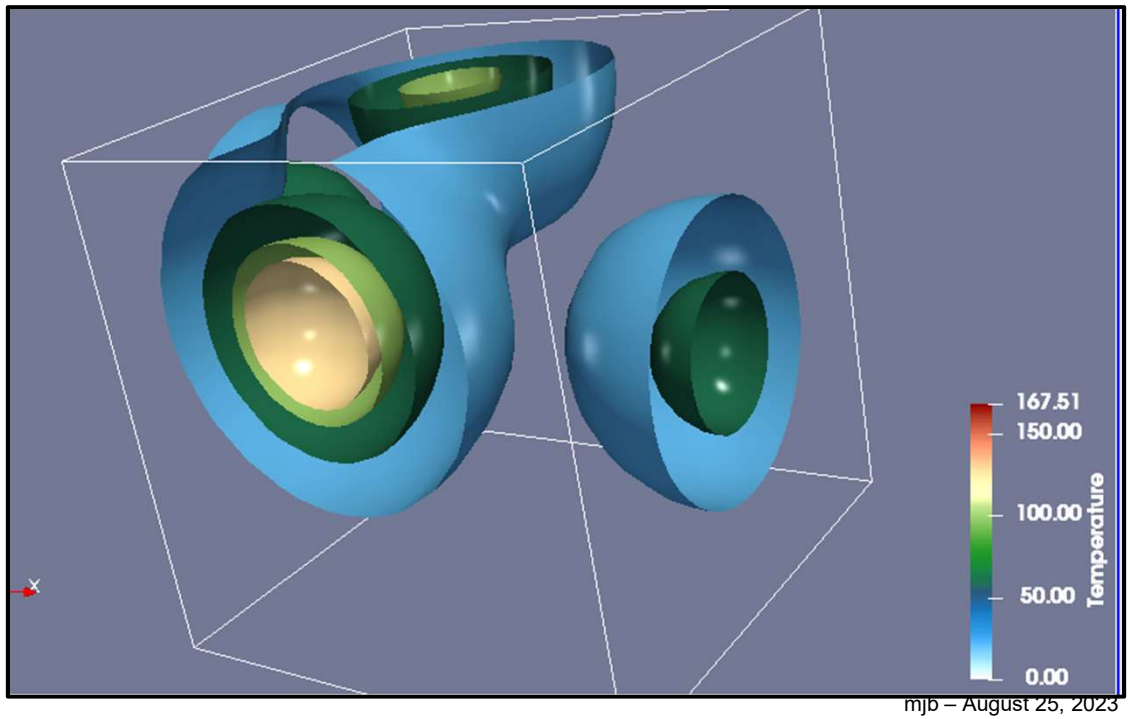
Comparative Visualization

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

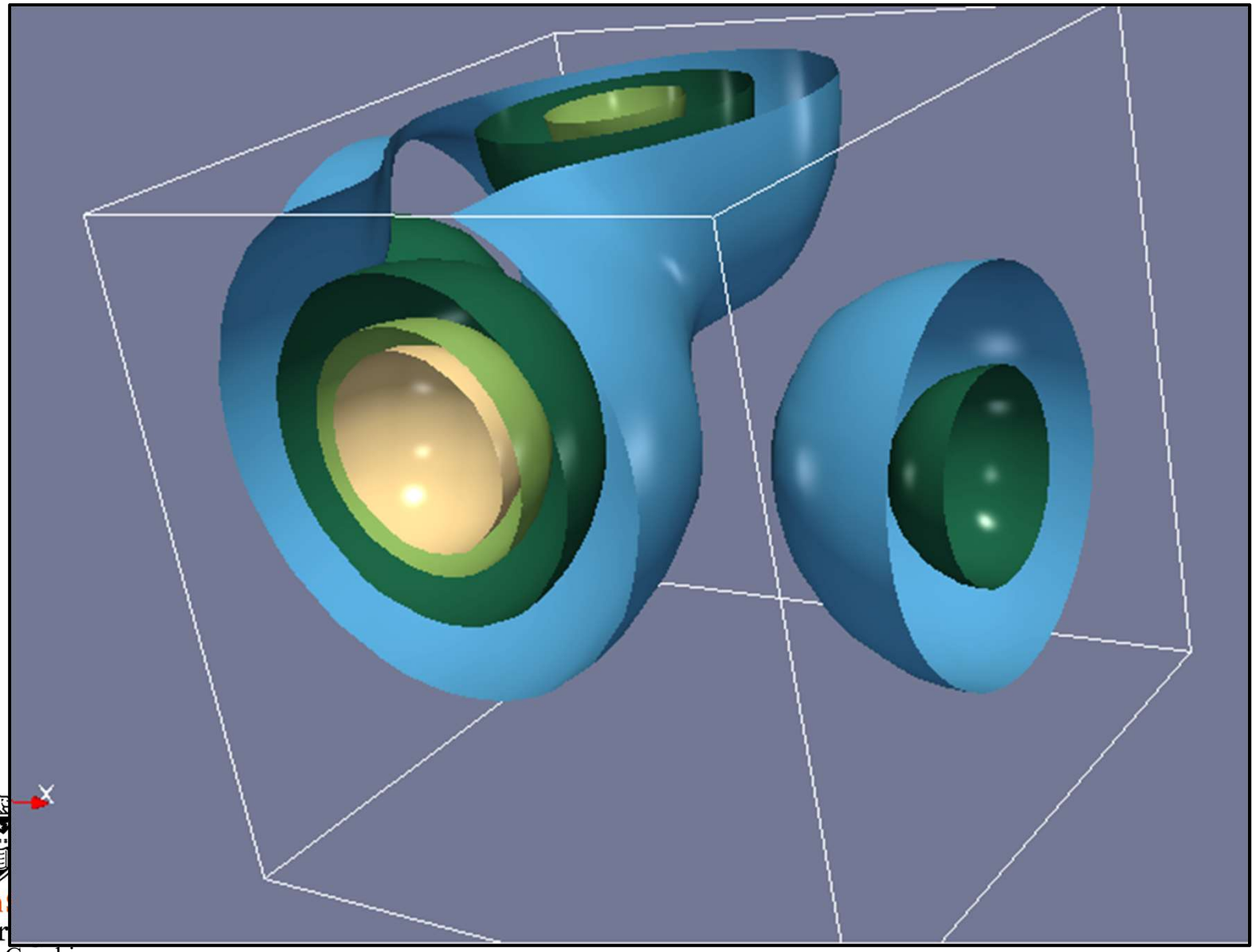
You can vary multiple parameters – just setup multiple pipeline elements and parameters and put numbers separated by commas in the cells

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

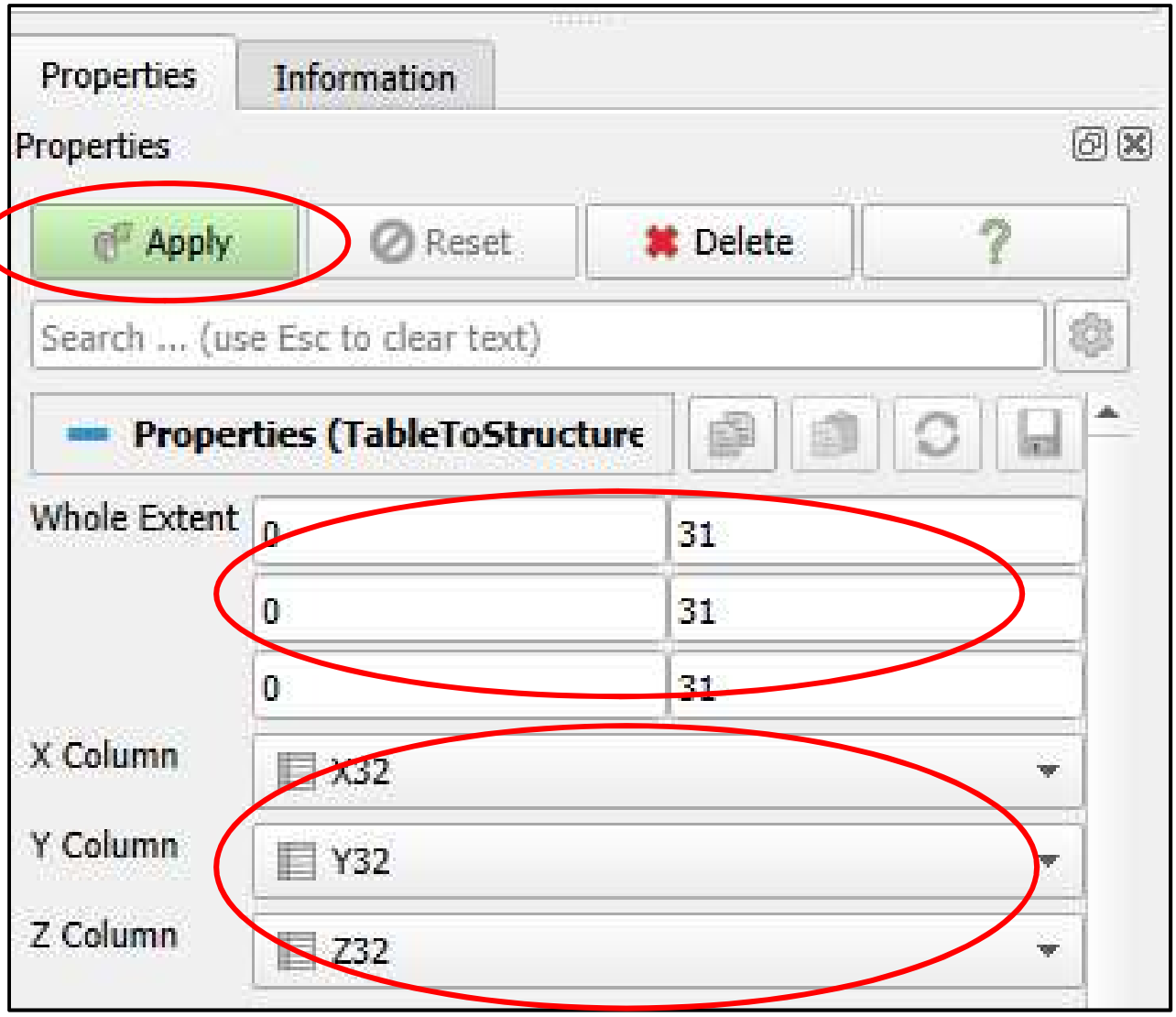
```
X32,Y32,Z32,Vx,Vy,Vz
-1.00,-1.00,-1.00,2.00,2.00,2.00
-1.00,-1.00,-0.94,1.75,1.75,2.00
-1.00,-1.00,-0.87,1.53,1.53,2.00
-1.00,-1.00,-0.81,1.33,1.33,2.00
-1.00,-1.00,-0.74,1.15,1.15,2.00
-1.00,-1.00,-0.68,0.99,0.99,2.00
-1.00,-1.00,-0.61,0.84,0.84,2.00
-1.00,-1.00,-0.55,0.71,0.71,2.00
-1.00,-1.00,-0.48,0.60,0.60,2.00
-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.

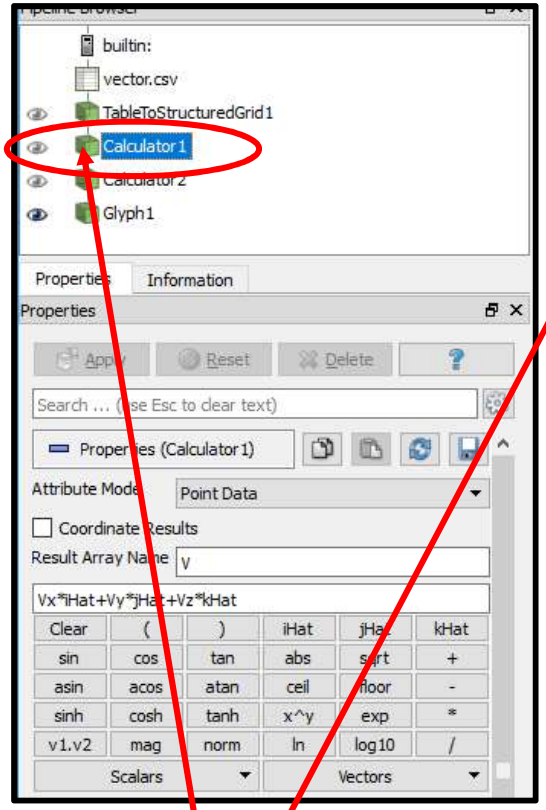
vector.csv

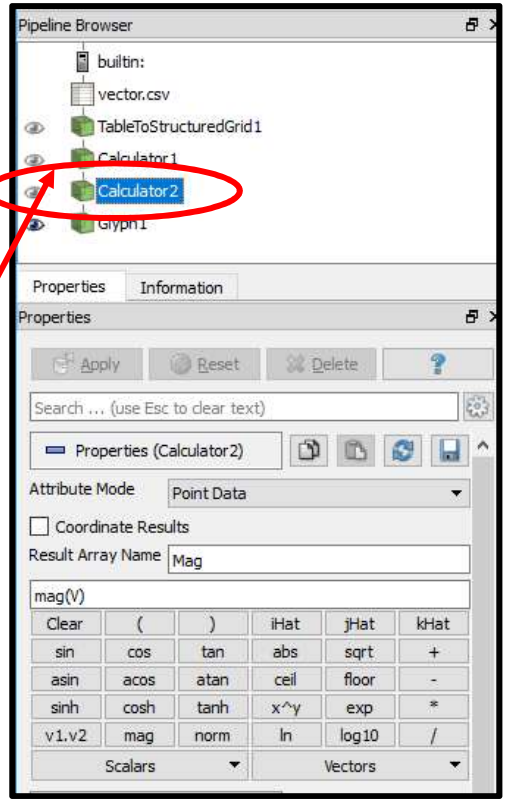
How to Read the Vector Data in the CSV File

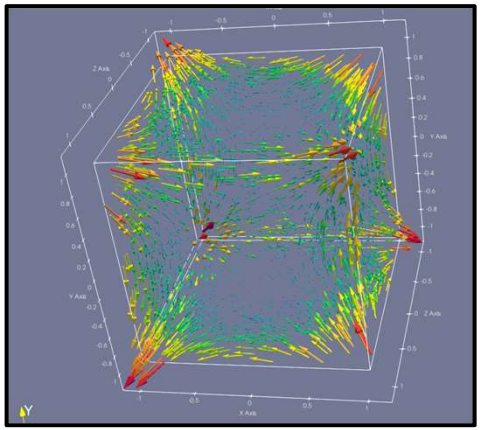
vector.csv

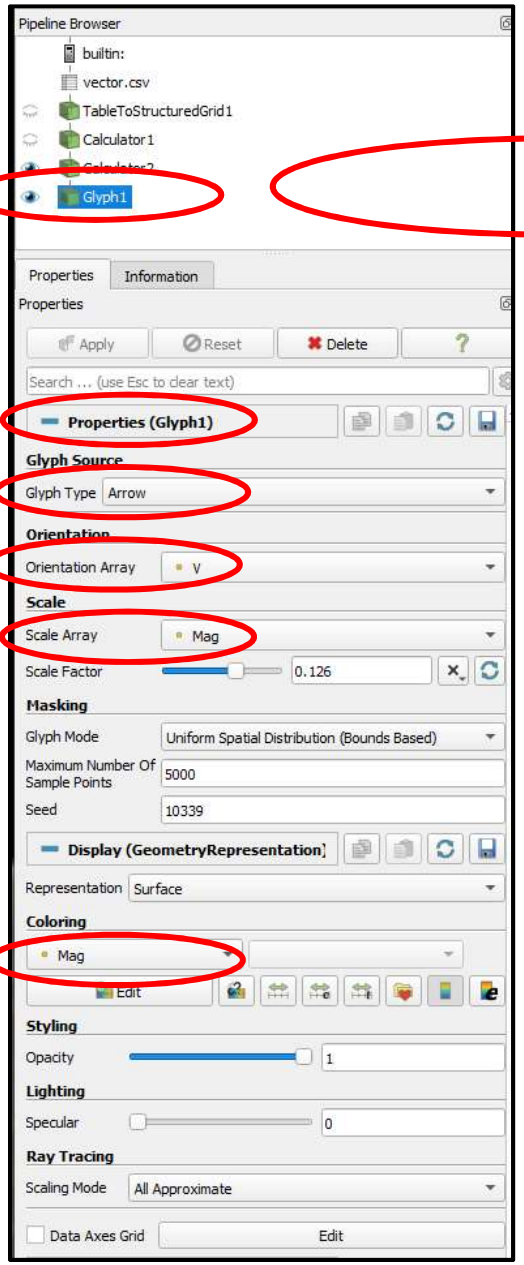


Vector Visualization As Glyphs



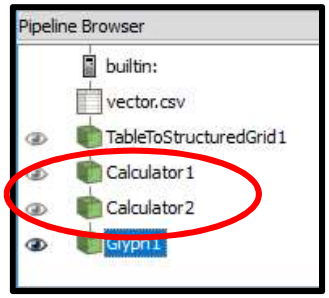


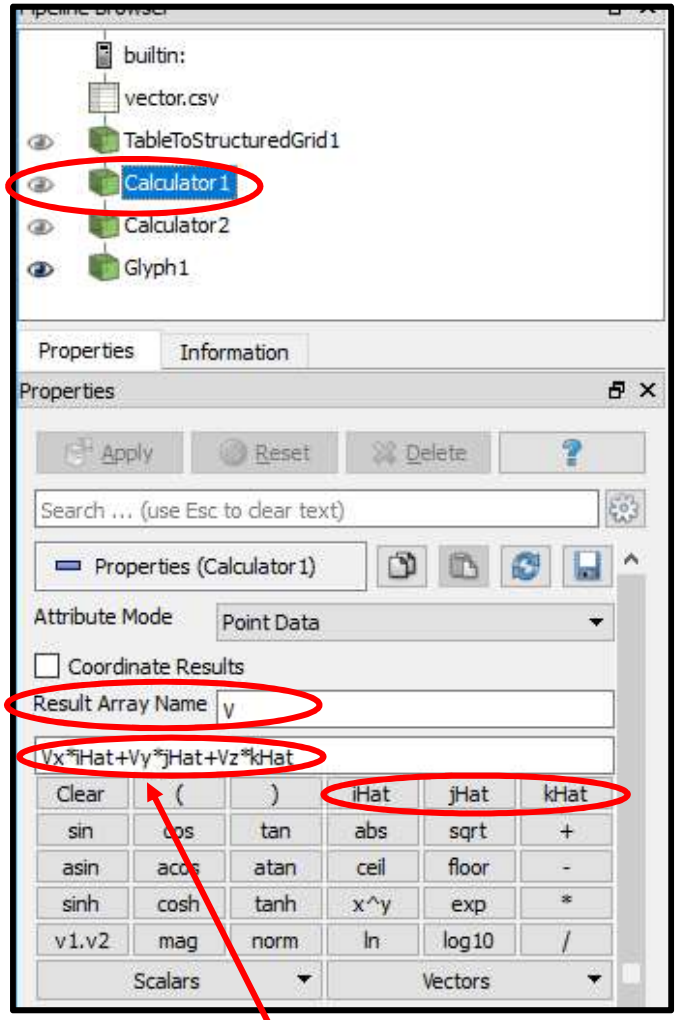


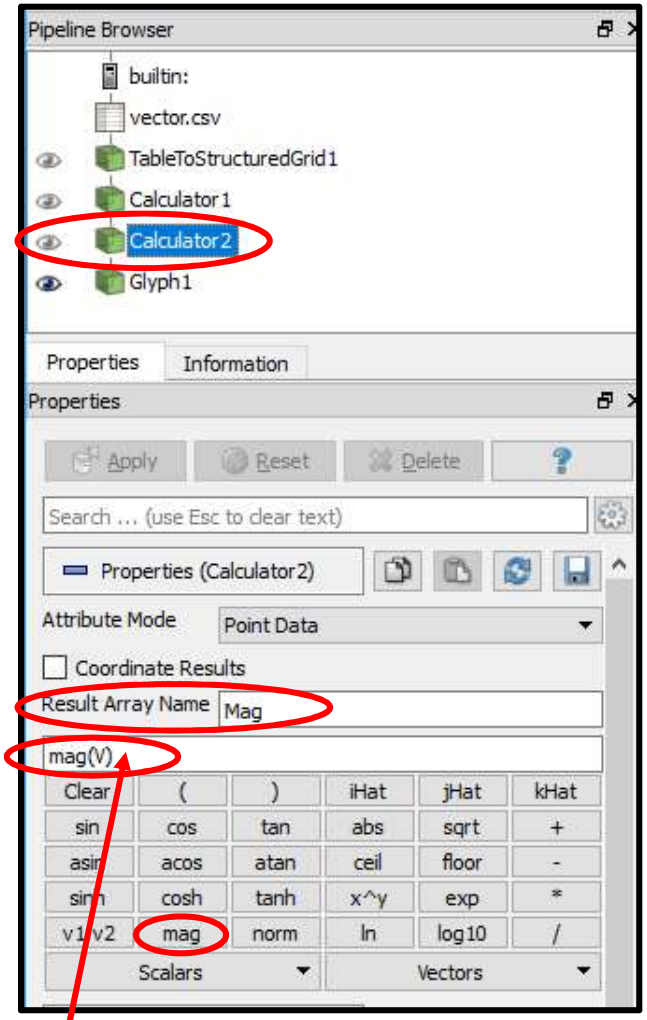


Add the 2 **Calculator** filters for now. The reason will be explained in the next slide.

Why Are the Two Calculator Filters There?







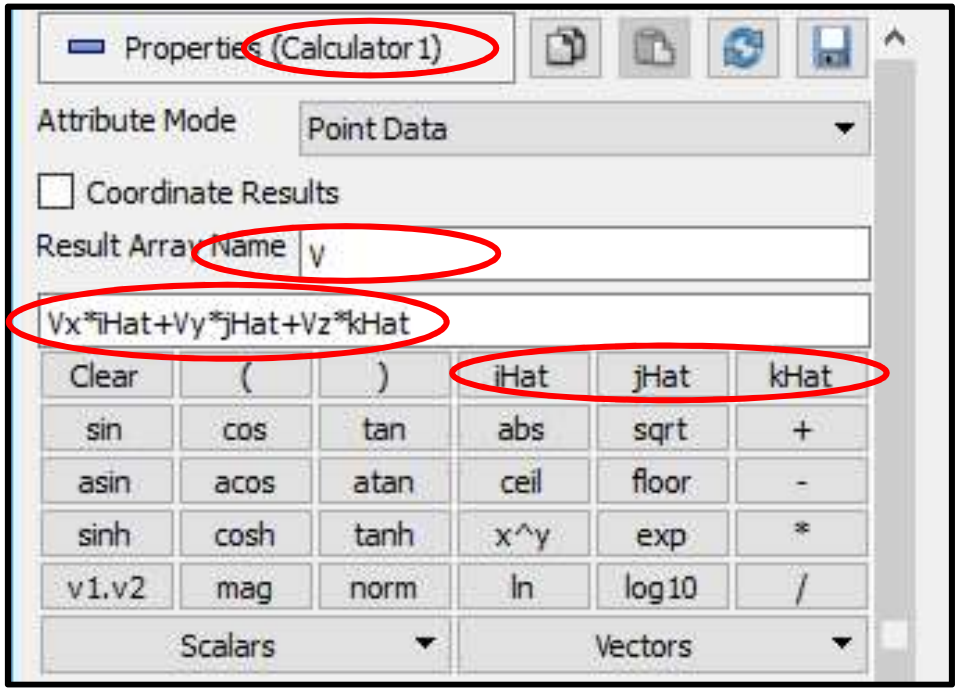
The **vector.csv** file brought in the three vector components **Vx**, **Vy**, and **Vz**. ParaView's vector vis filters want a 3-element vector instead. **Calculator1** is used to create that 3-element vector using the **iHat**, **jHat**, and **kHat** buttons (unit vectors in x, y, and z) :

$$\boldsymbol{V = V_x\hat{\imath} + V_y\hat{\jmath} + V_z\hat{k}}$$

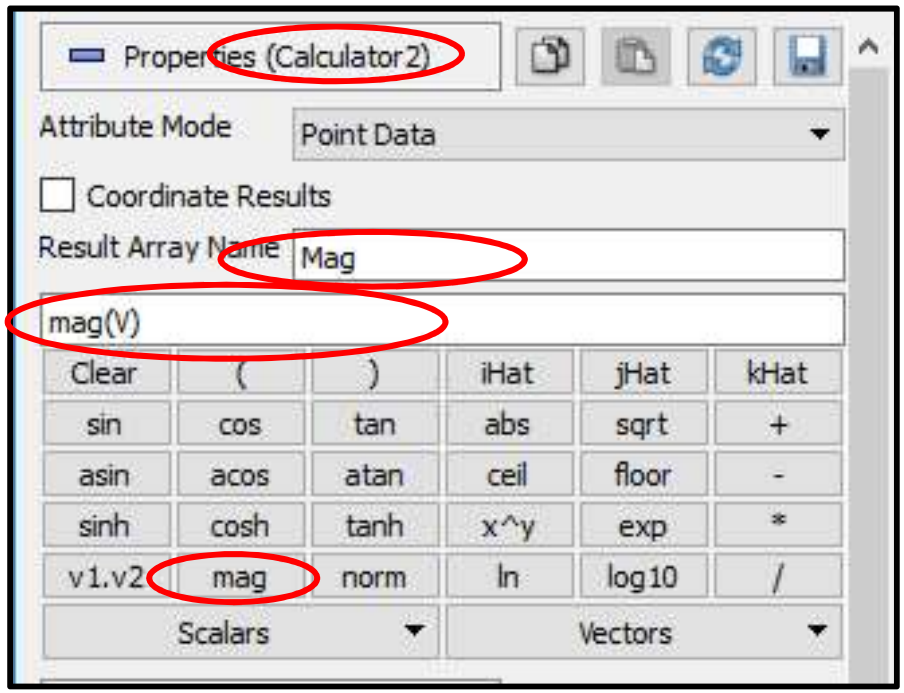
We want to color the vector visualizations by the magnitude of the vector. **Calculator2** computes that magnitude using the **mag** button:

$$\boldsymbol{Mag = \|V\|}$$

Why Are the Two Calculator Filters There?

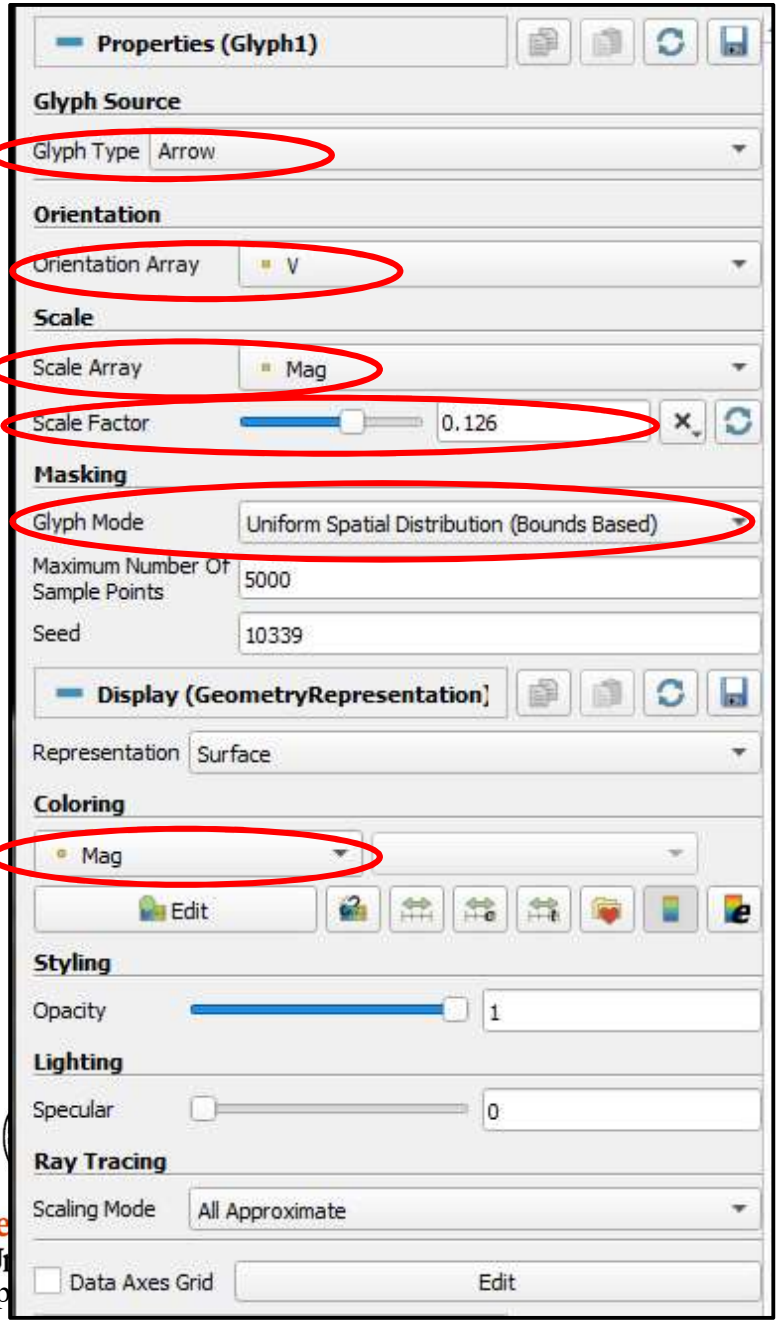


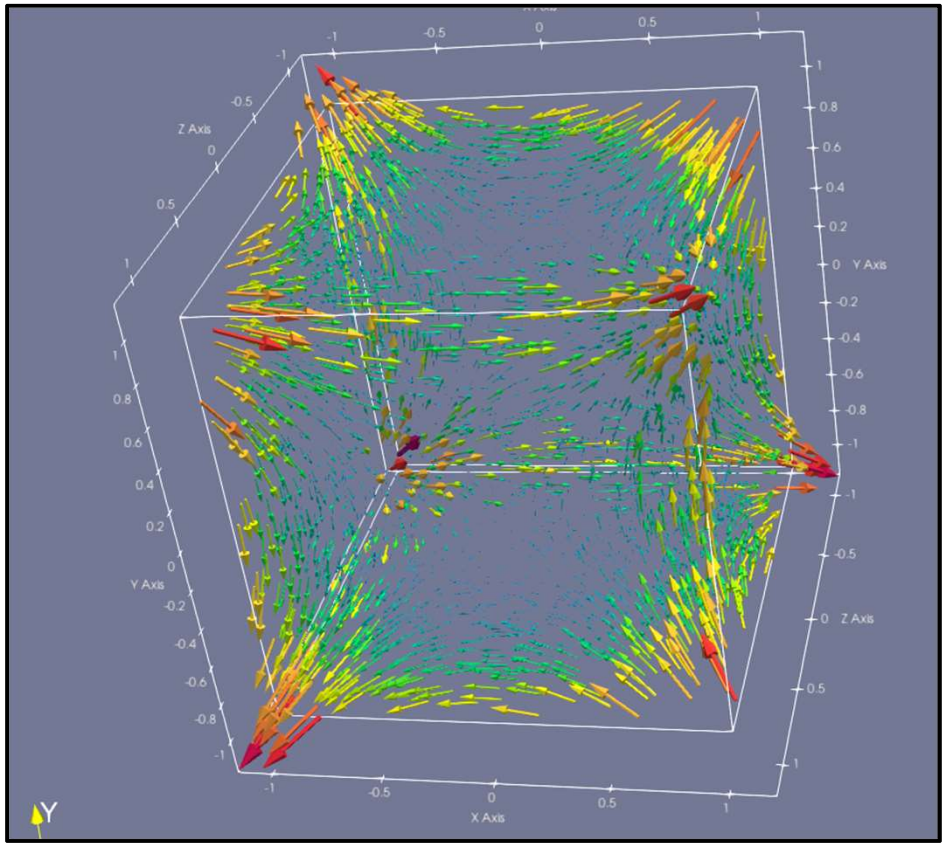
$$\boldsymbol{V} = \boldsymbol{V_x}\hat{\boldsymbol{\imath}} + \boldsymbol{V_y}\hat{\boldsymbol{\jmath}} + \boldsymbol{V_z}\hat{\boldsymbol{k}}$$

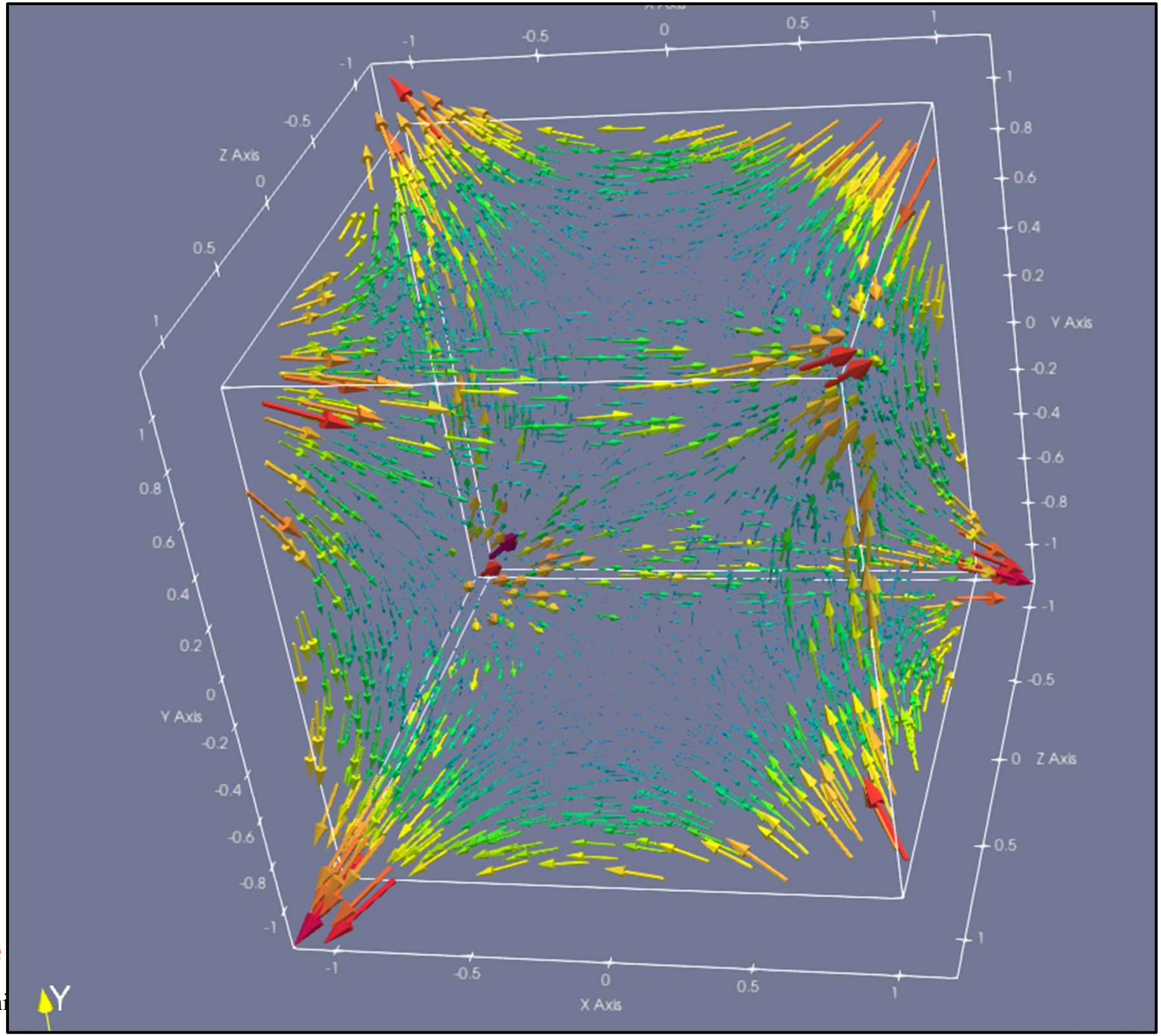


$$\boldsymbol{Mag} = \|\boldsymbol{V}\|$$

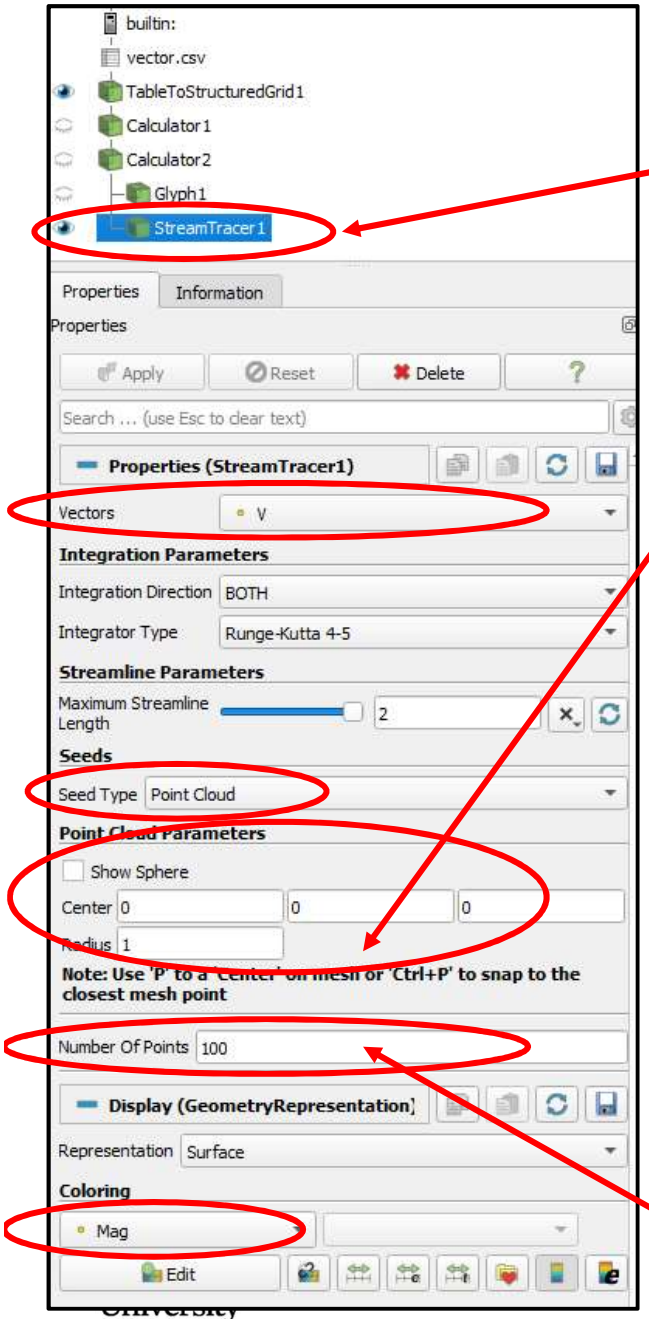
Setting Up the Glyph and its Coloring





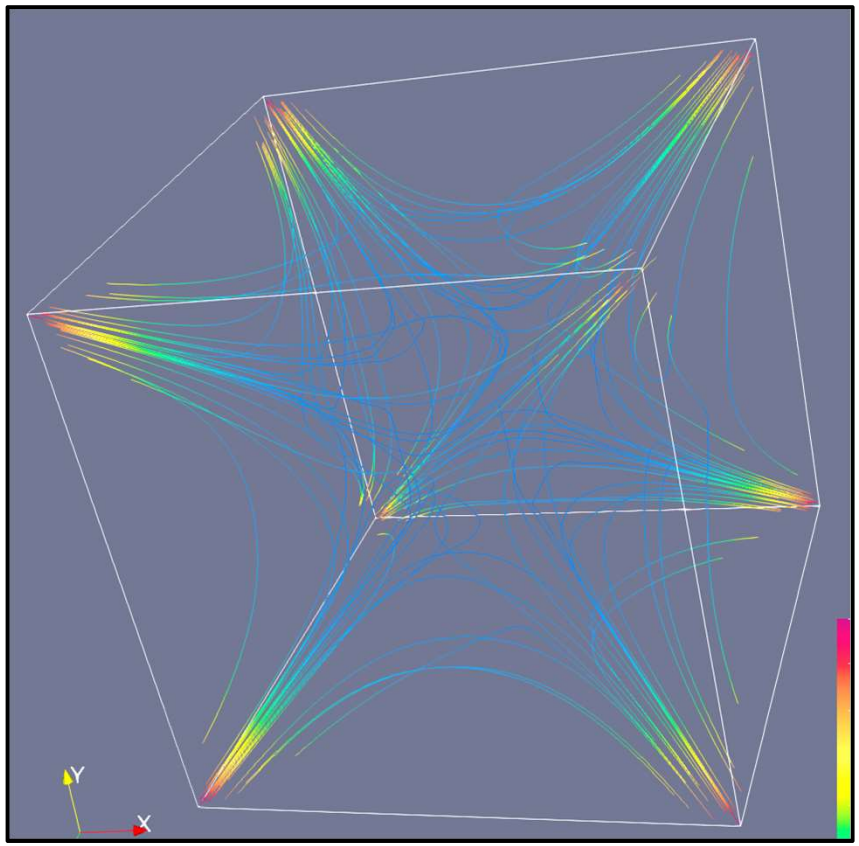

Z Axis
Y Axis
X Axis
Y Axis
Z Axis
Y

As Streamlines

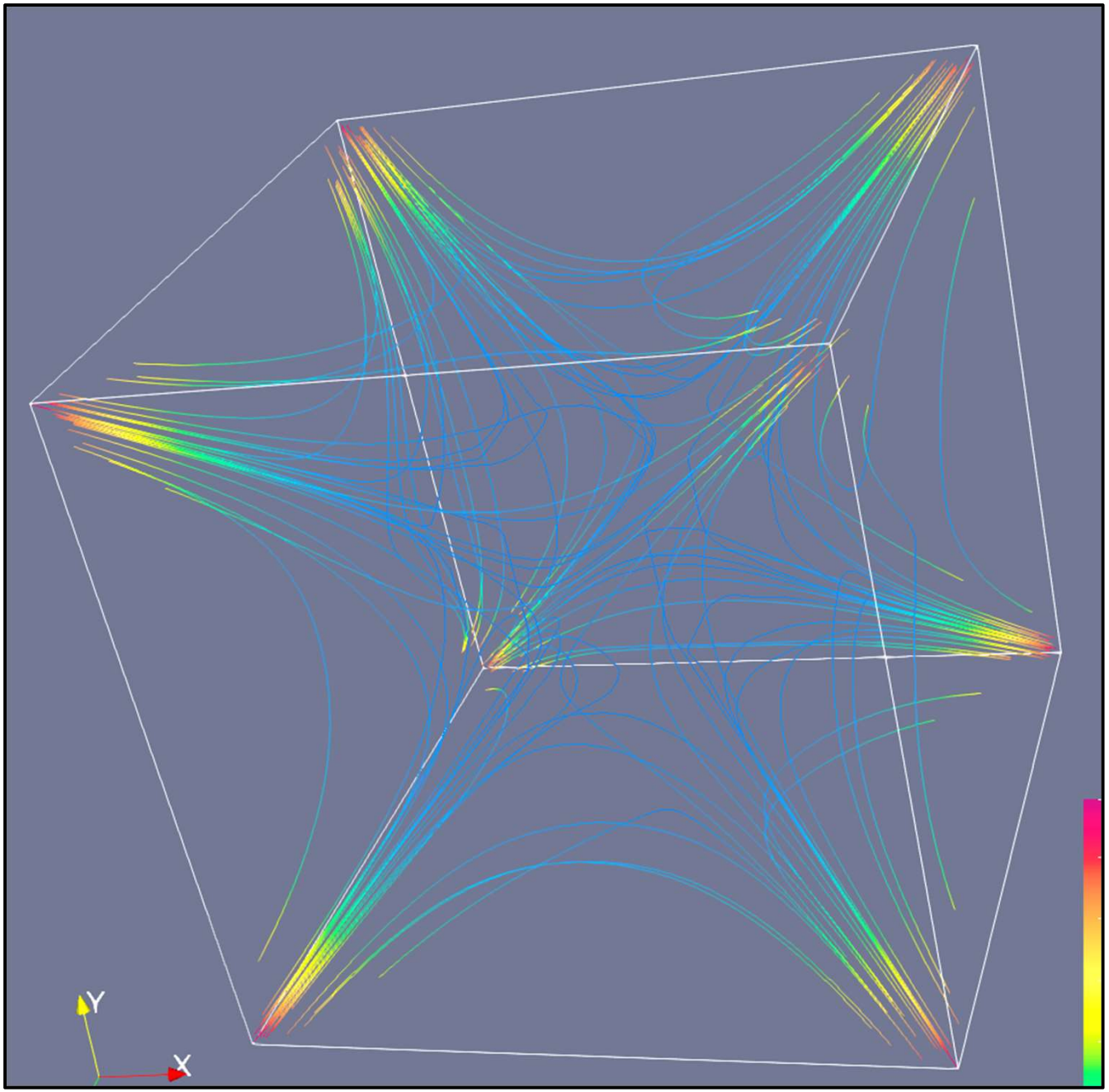


StreamTracer filter, *parented from the second* ***Calculator***

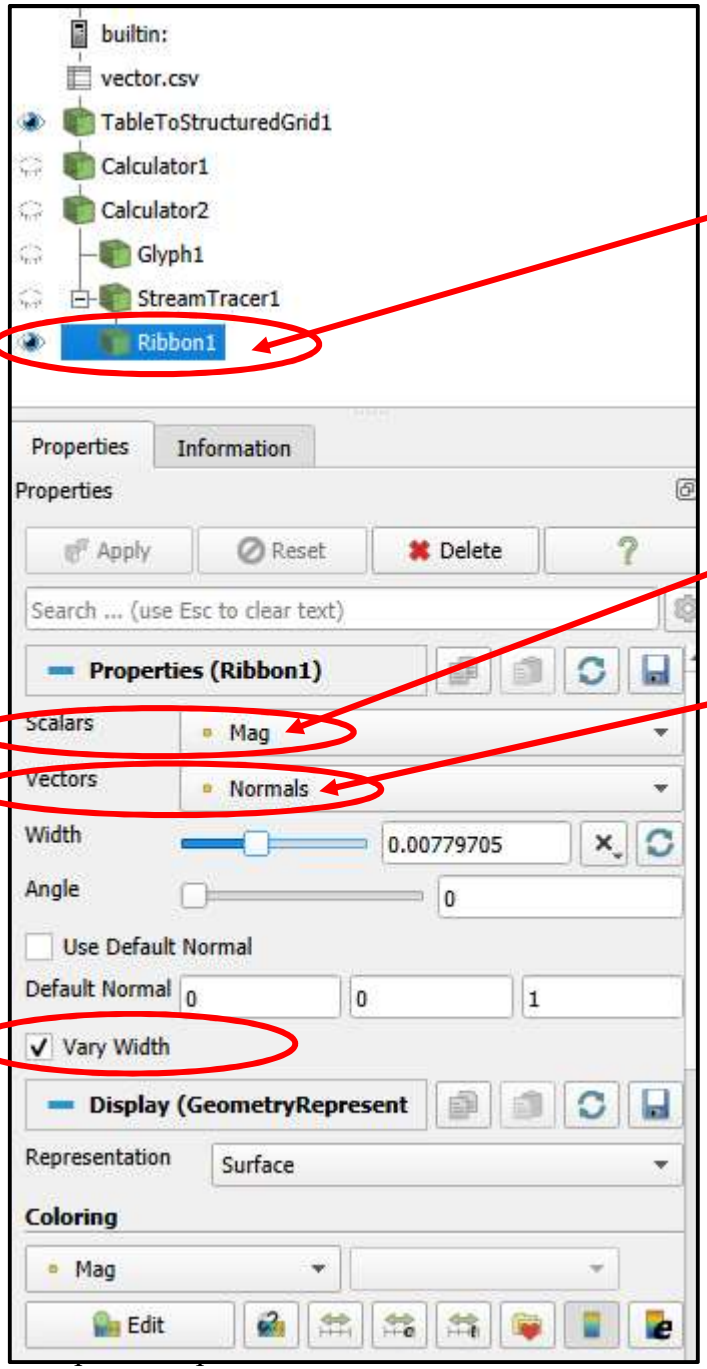
Will start the streamlines from within this sphere. You can move it and resize it.



Number of points to start from


Y
X

As Ribbon Traces

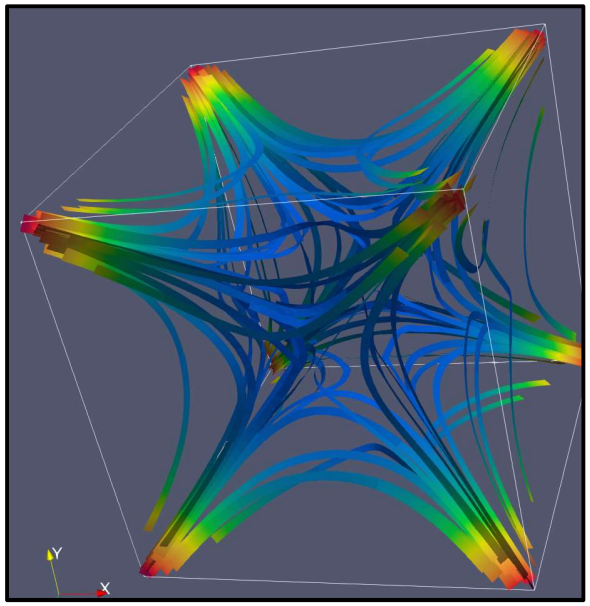


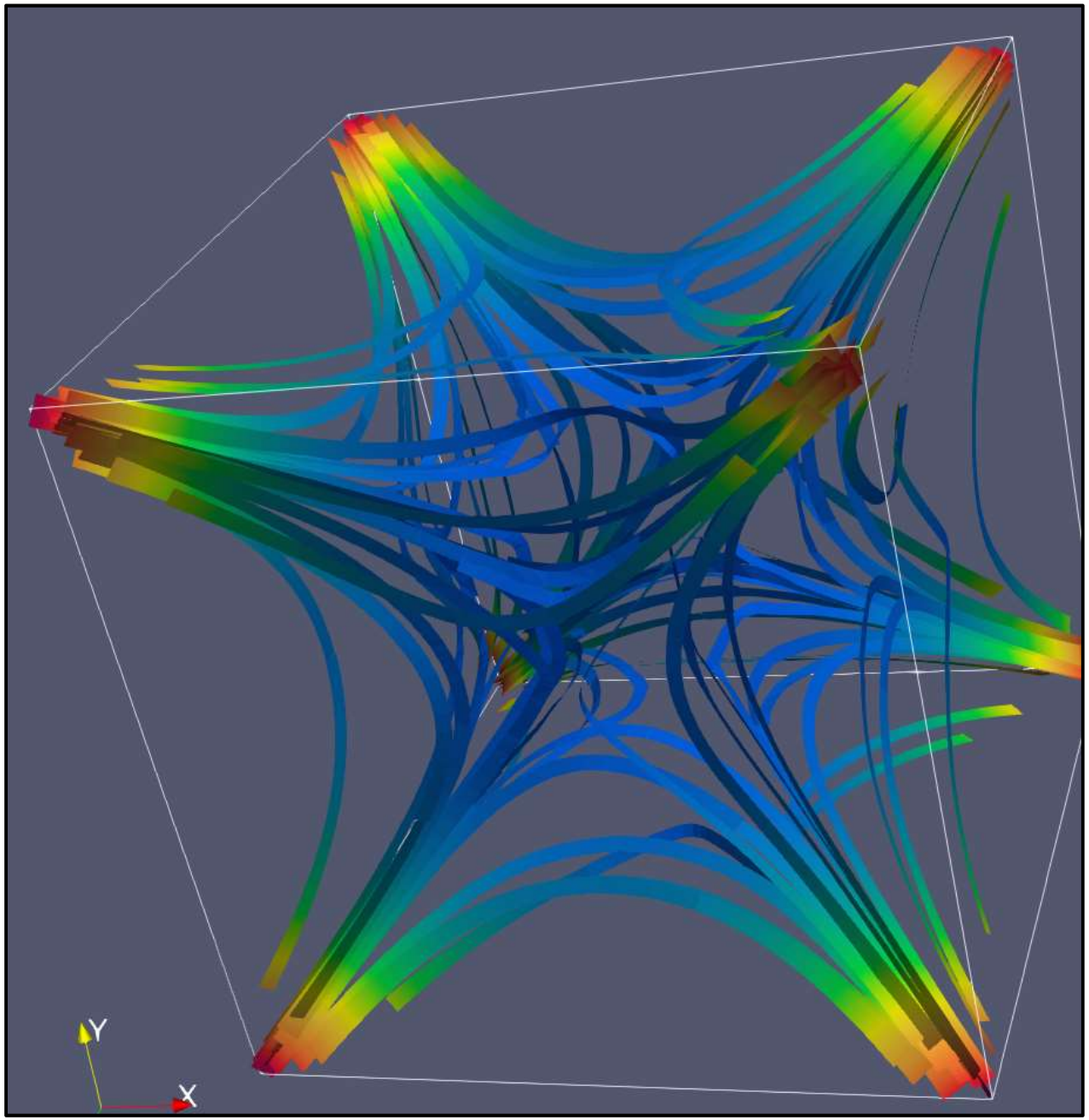
Note – **Ribbon** is *parented from* ***StreamTracer*****.**

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

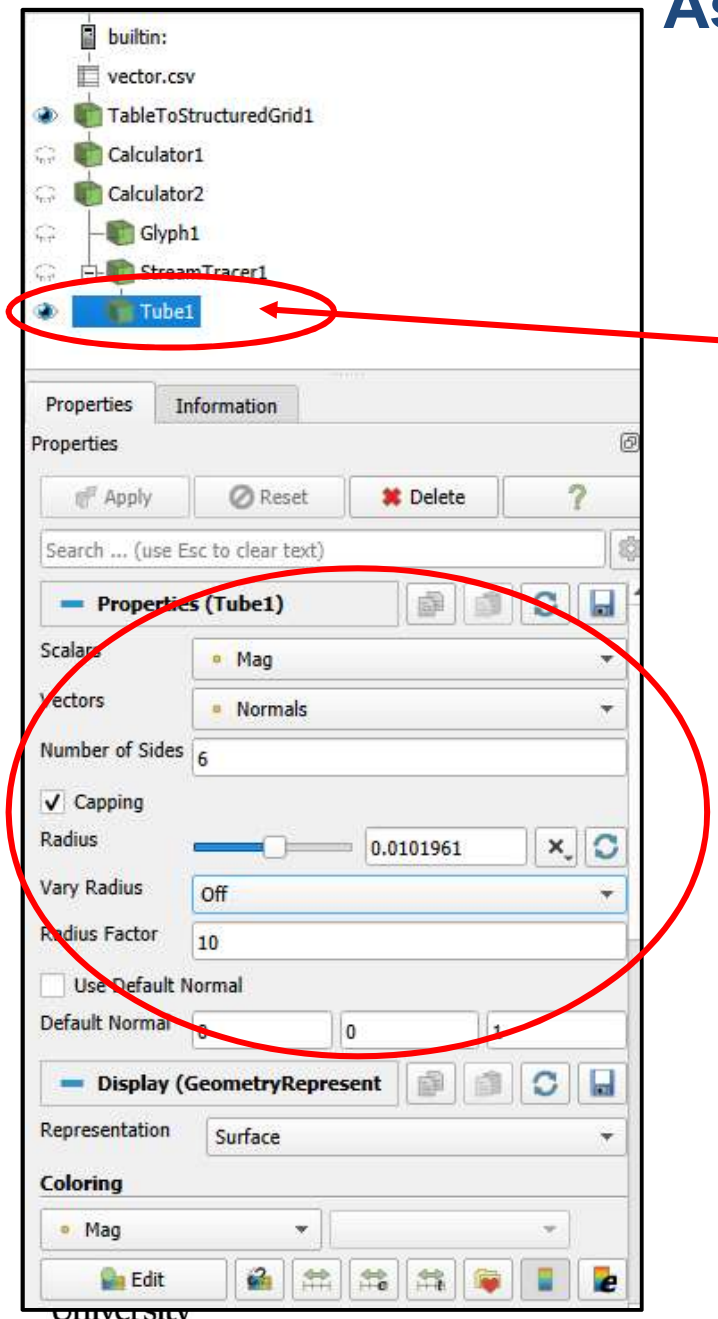
The **Scalar** setting tells what will be used to size the width of the ribbons.

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

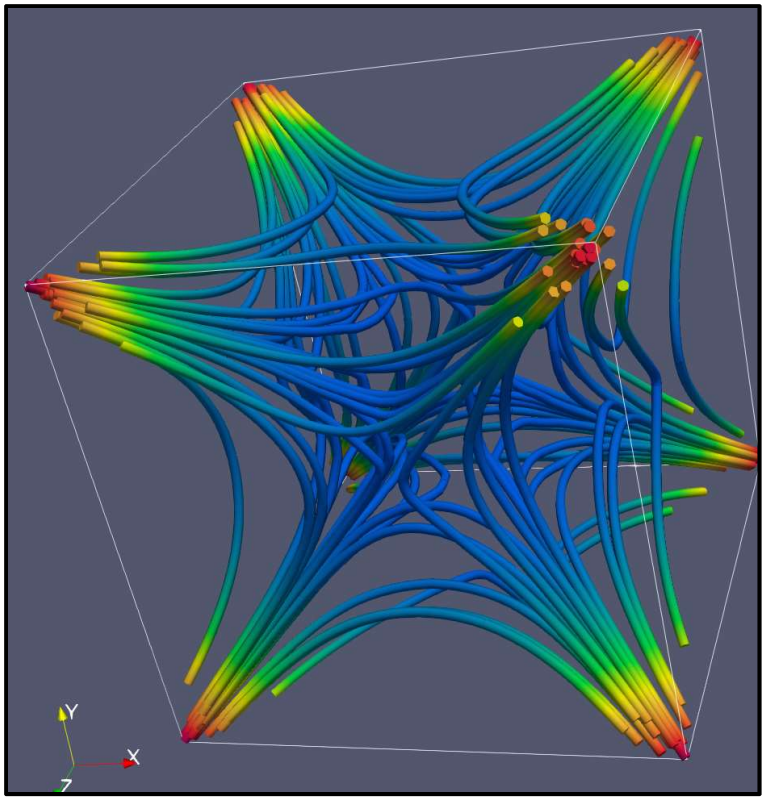


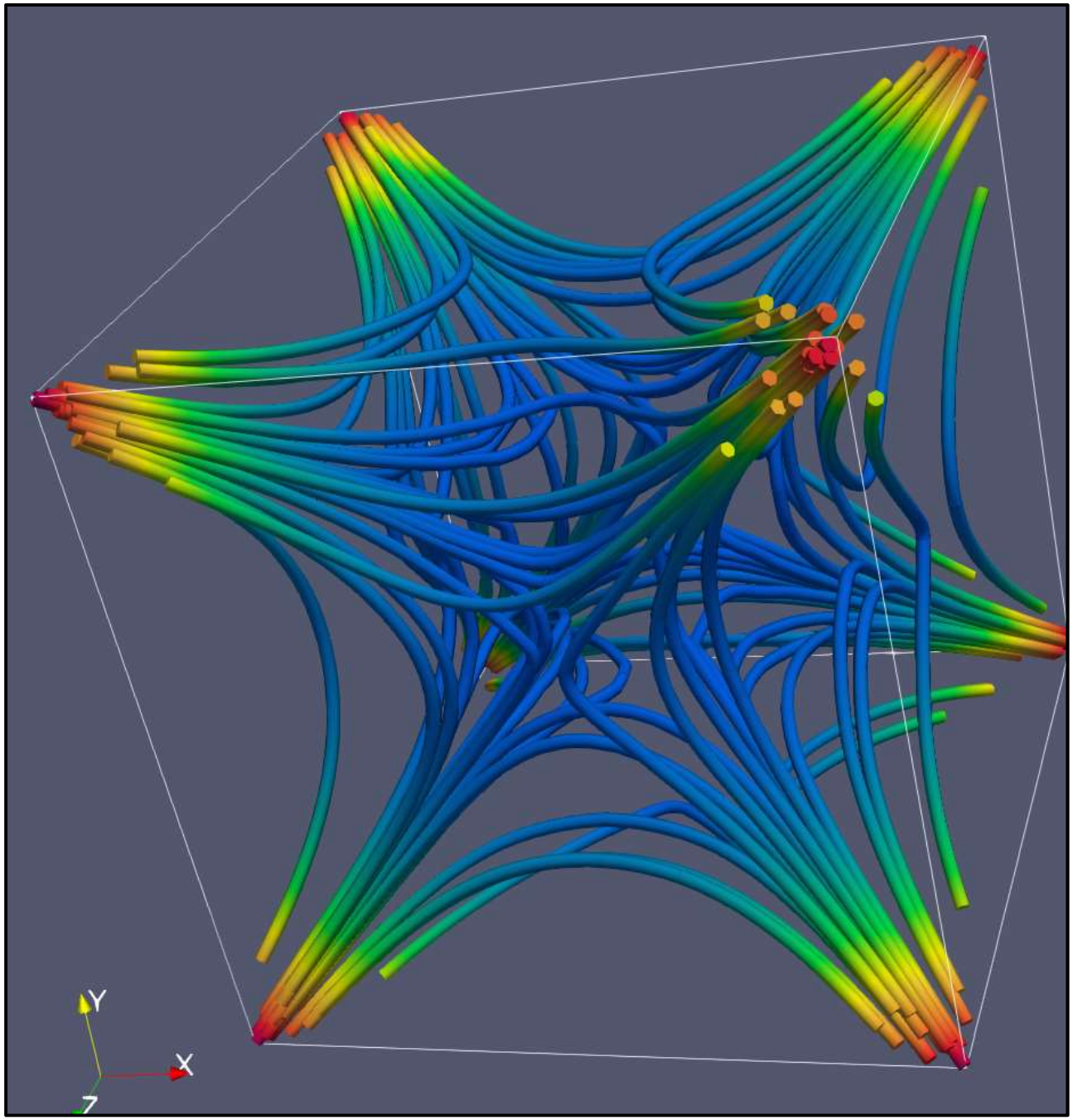

Y
X

As Streamtubes



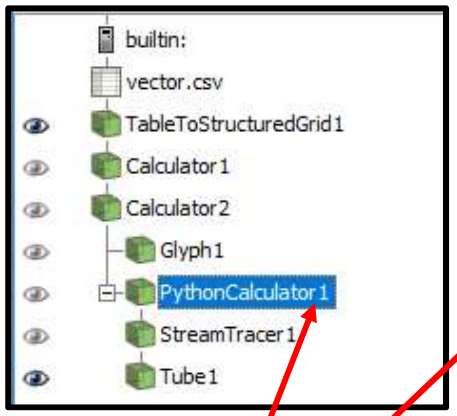
Note – **Tube** is *parented from* ***StreamTracer*.**

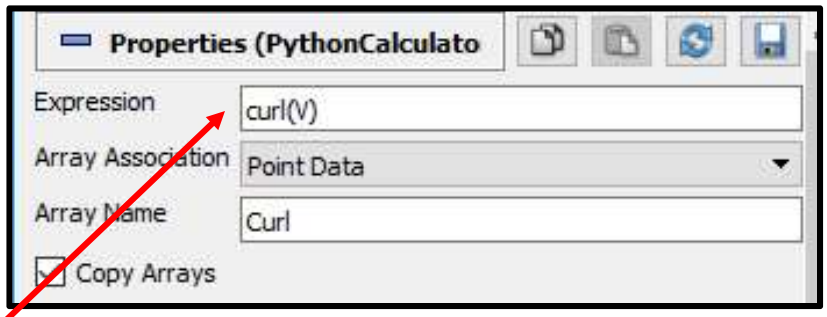


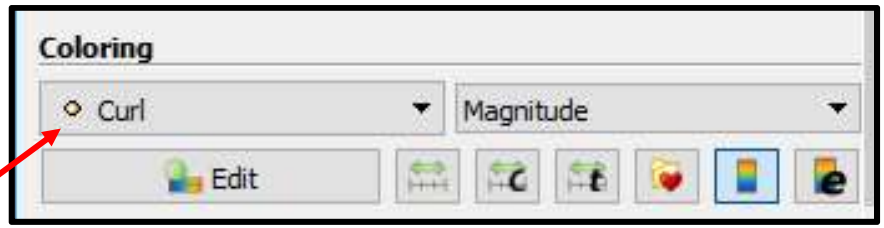

Y
X
Z

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

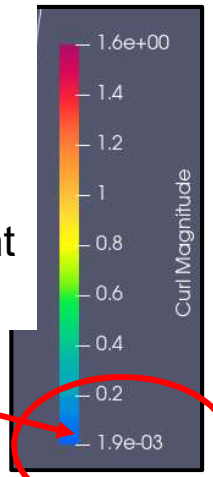
In this case, we will map curvature (defined by the curl of the vector field)

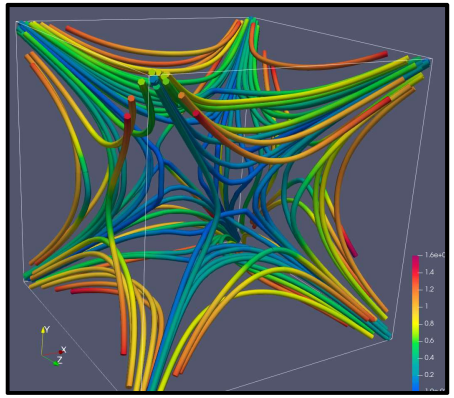




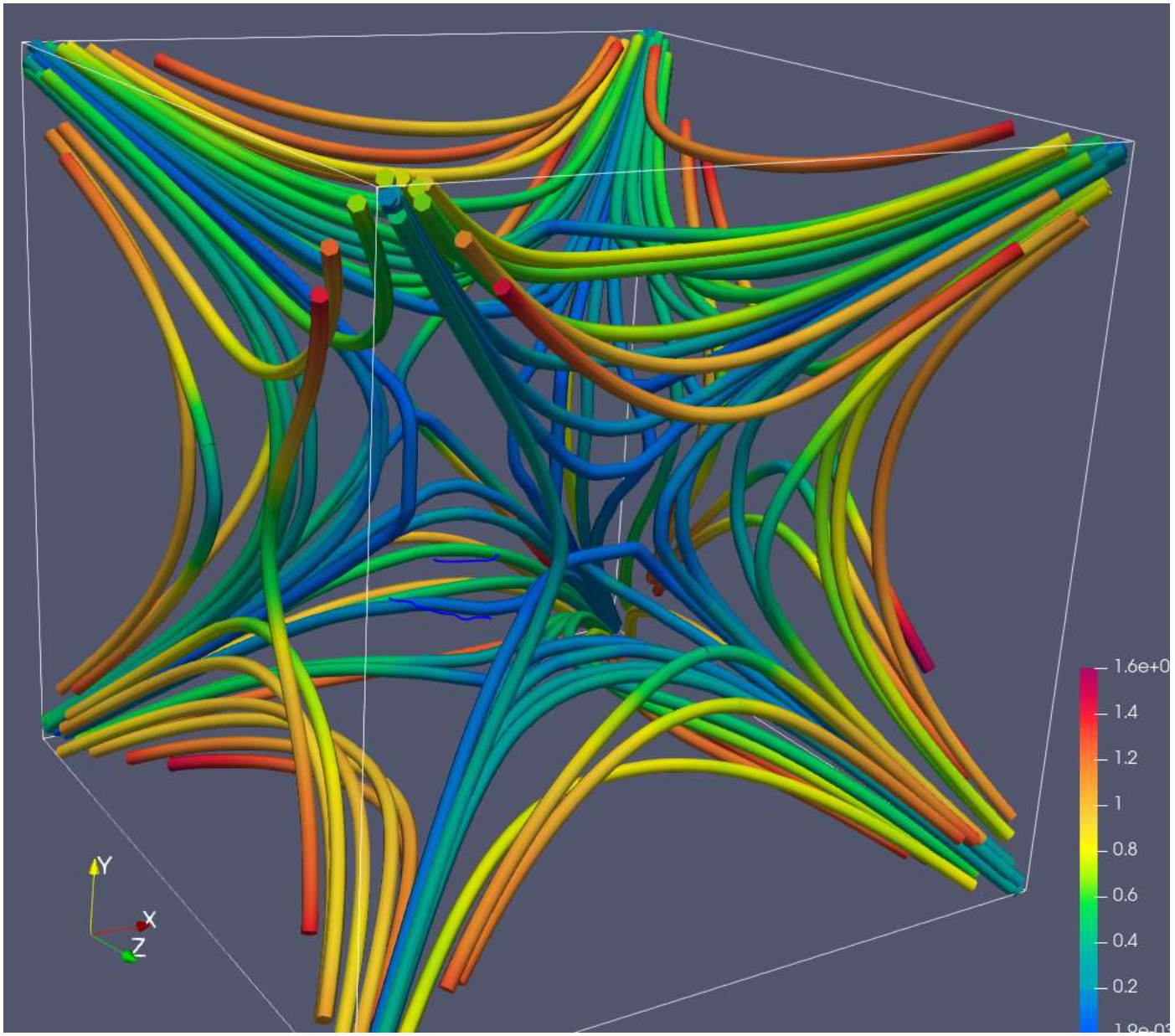


- The **Python Calculator** filter was used to produce the **Curl** of the vector field (it has a built-in **curl( )** function – the Calculator does not)
- The StreamTube's coloring was changed from **Mag** to **Curl**
- The color mapping was changed to cut down on the amount of blue (lots of low curl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





Oregon State University
Computer Graphics


1.6e+0
1.4
1.2
1
0.8
0.6
0.4
0.2
Y
X
Z

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

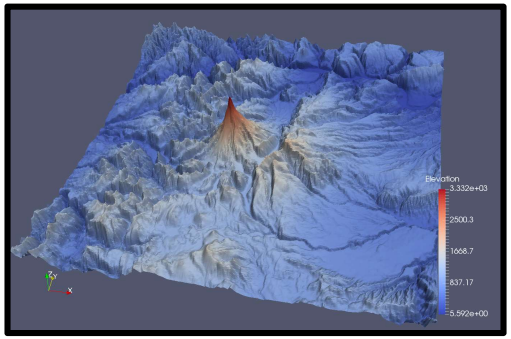
terrain.csv

Creating Terrain Data in a CSV File

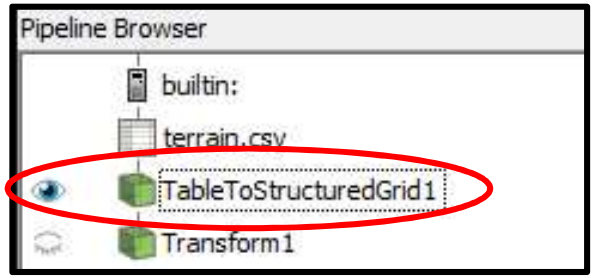
```
UTMx512,UTMy361,Z,Longitude,Latitude,Elevation
-6909.865,-6870.170,1174.991,-122.200,45.010,1174.991
-6882.896,-6870.356,1268.436,-122.198,45.010,1268.436
-6855.759,-6870.542,1308.478,-122.196, 5.010,1308.478
-6828.789,-6870.728,1266.755,-122.193,45.010,1266.755
-6801.820,-6870.911,1203.239,-122.191,45.010,1203.239
-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.

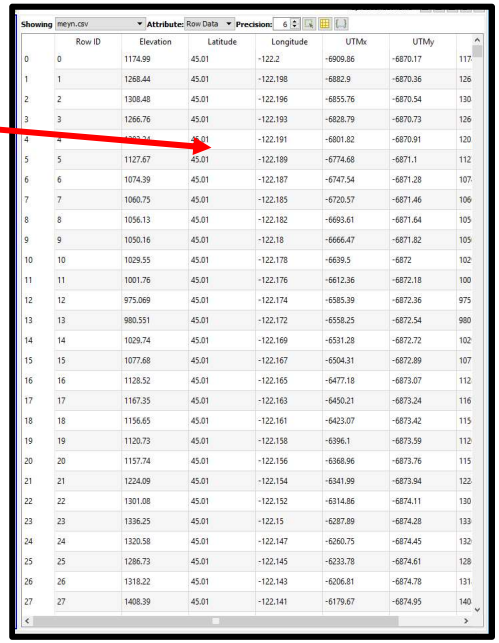
UTM data is in meters, which makes a more reality-looking base than longitude and latitude do. It is good to have both Z and Elevation, even though they are the same number because once you use a variable for a geometric dimension, you can't also use it again for a data value (e.g., to color or contour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

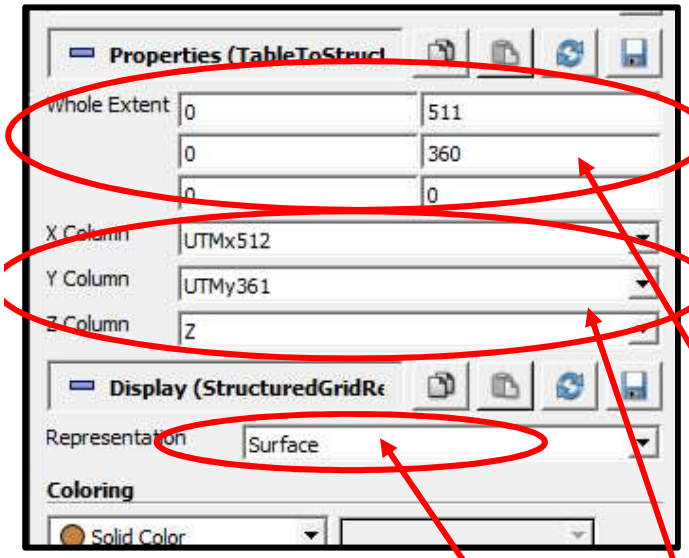


Reading and Converting the CSV File



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





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).

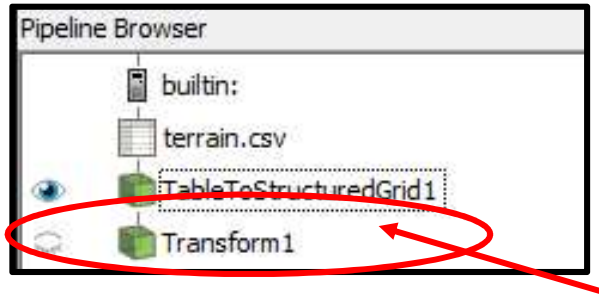
Fill in the **{X,Y,Z} Column** information so ParaView knows how to make your 3D display.

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

Be sure the **Representation** is **Surface**

Oregon State University
Computer Graphics

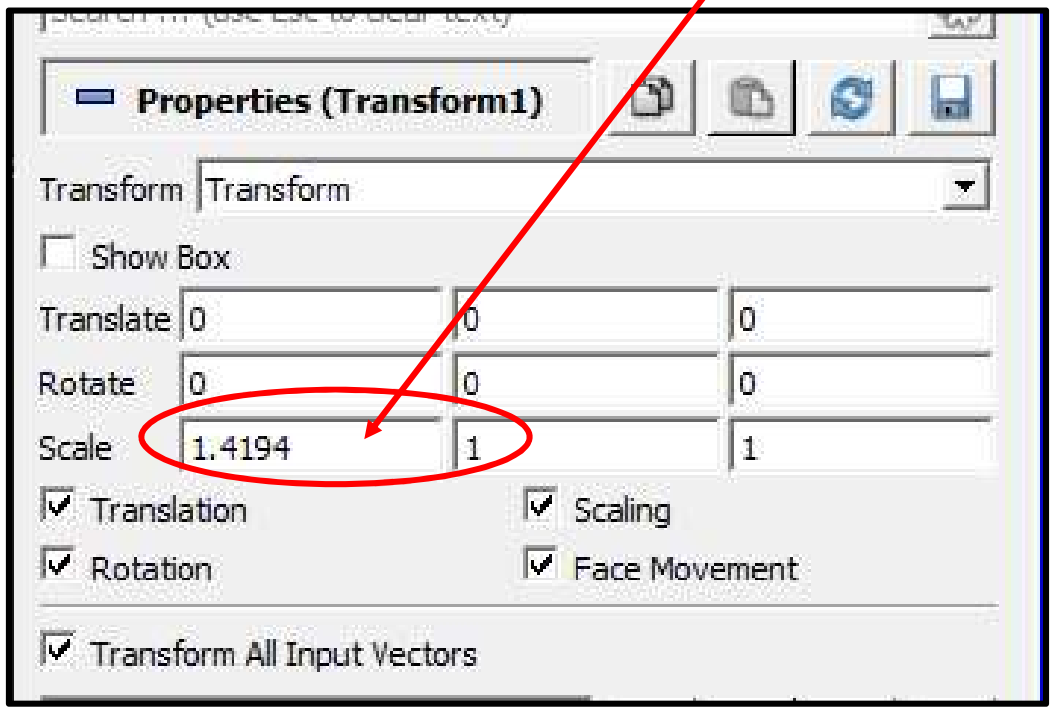
The Correct Scale Factor

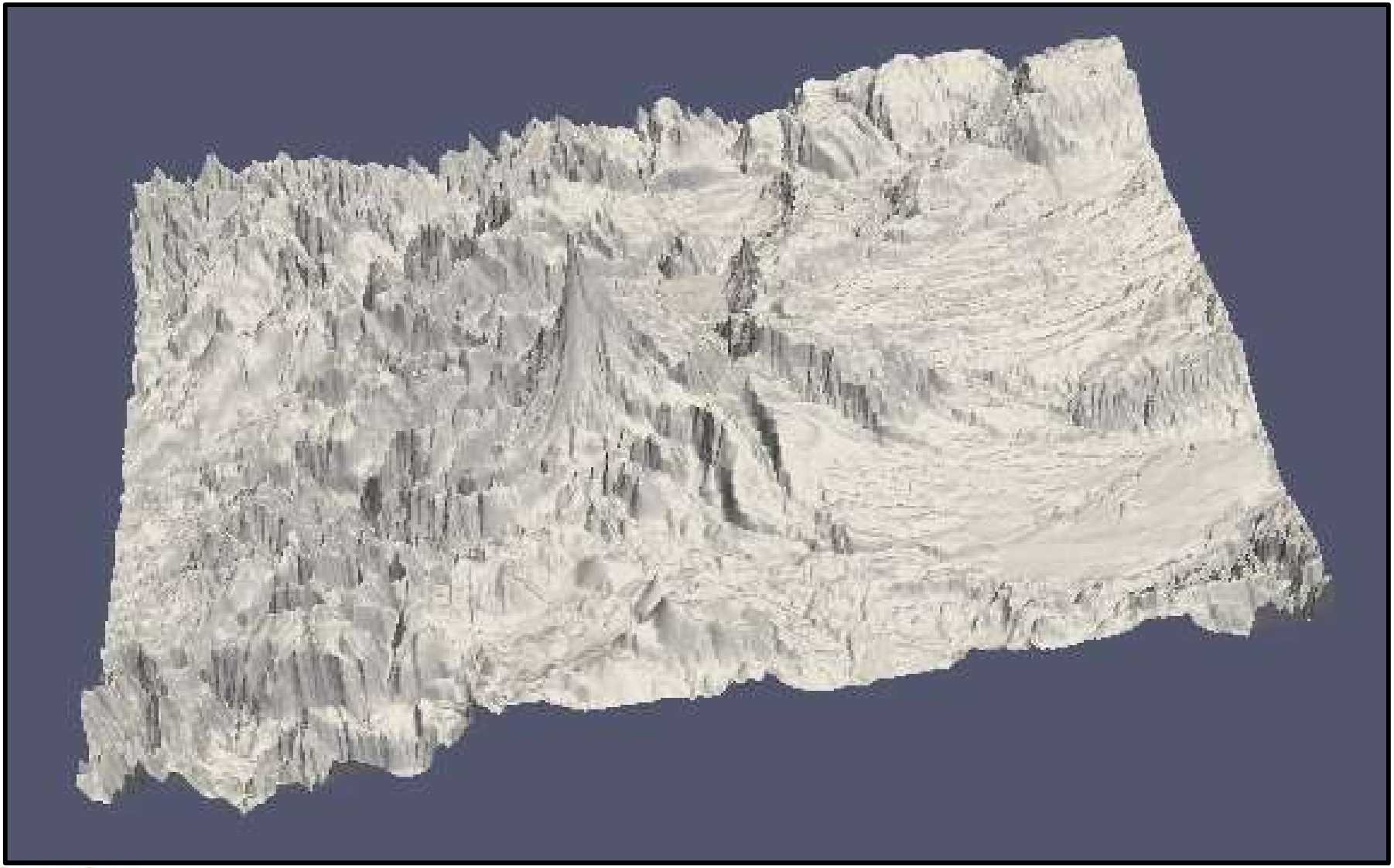


This will bring up a square terrain, which isn't what we want. We notice that the UTM coordinates are 511 and 360, so we really want to scale by $^{511}/_{360}$ = 1.4194 in the X direction.

Now, go to
Filters → Alphabetical → Transform

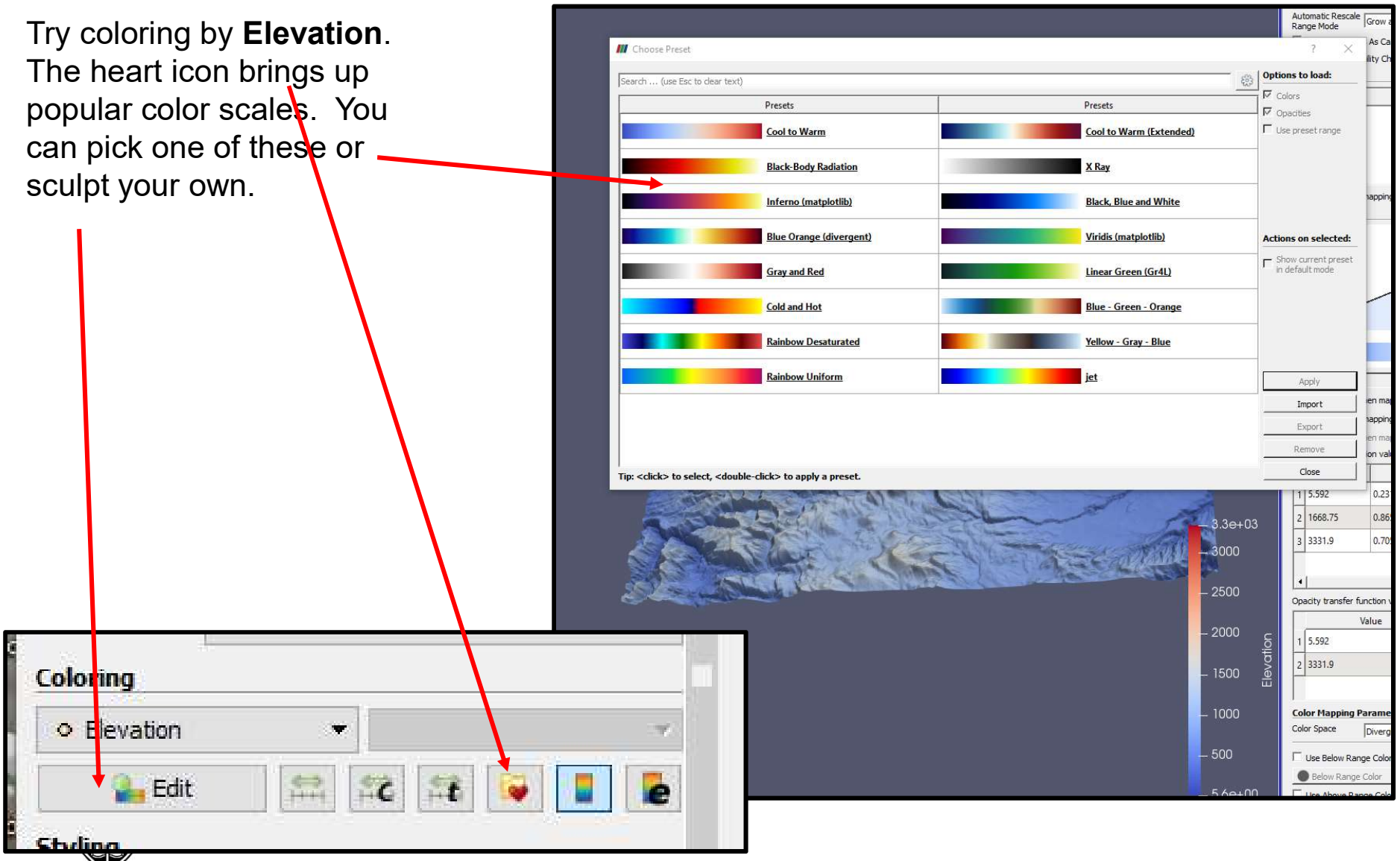
Set the X scale factor to 1.4194

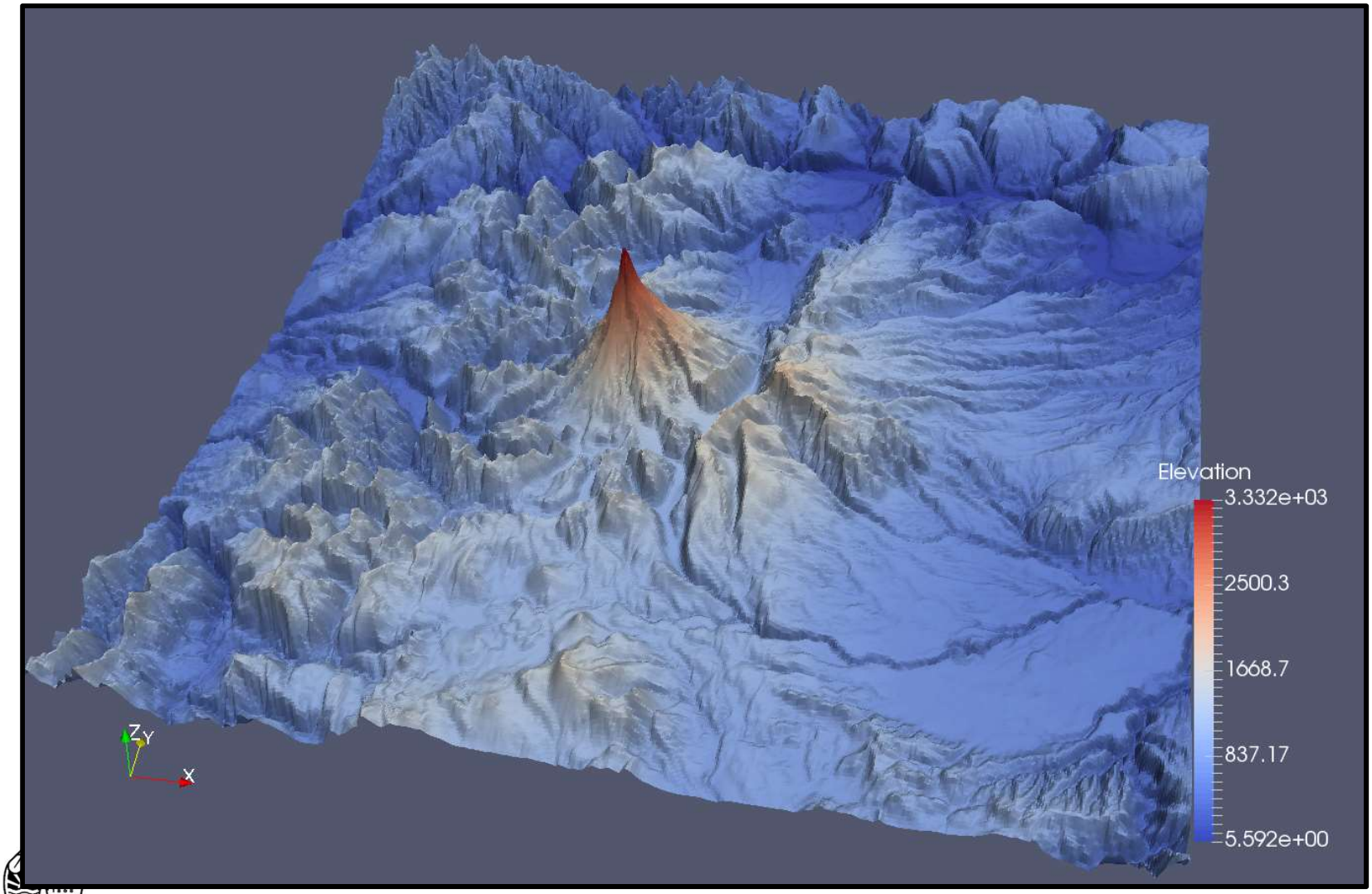




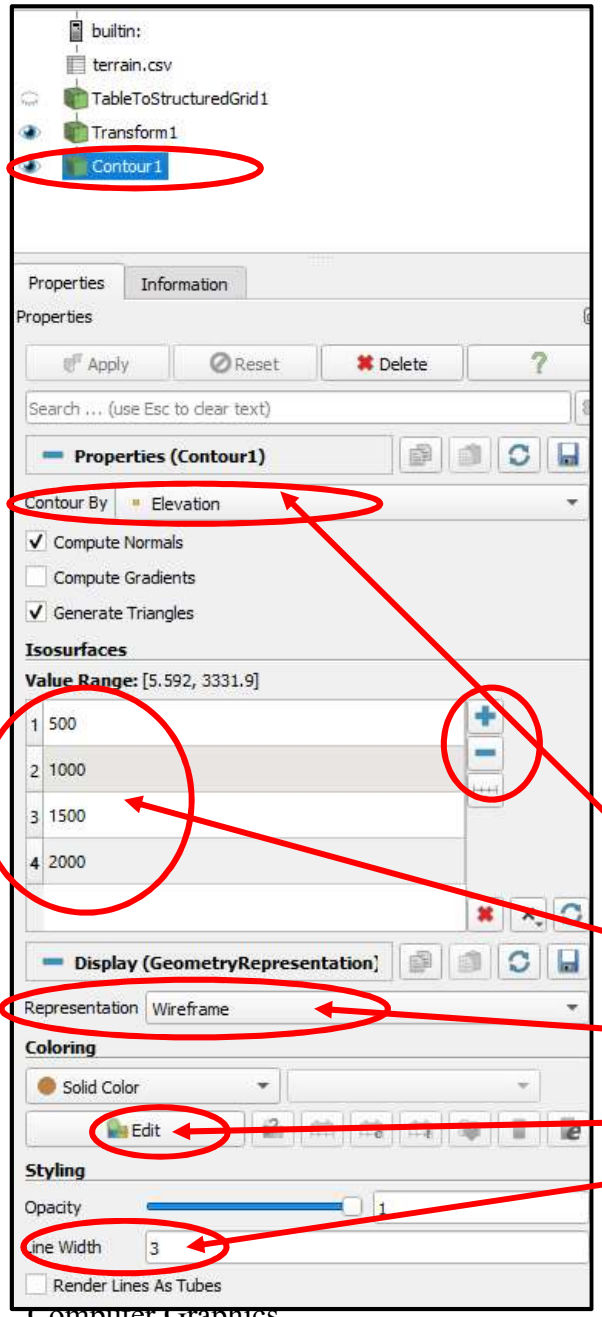
Color by Elevation

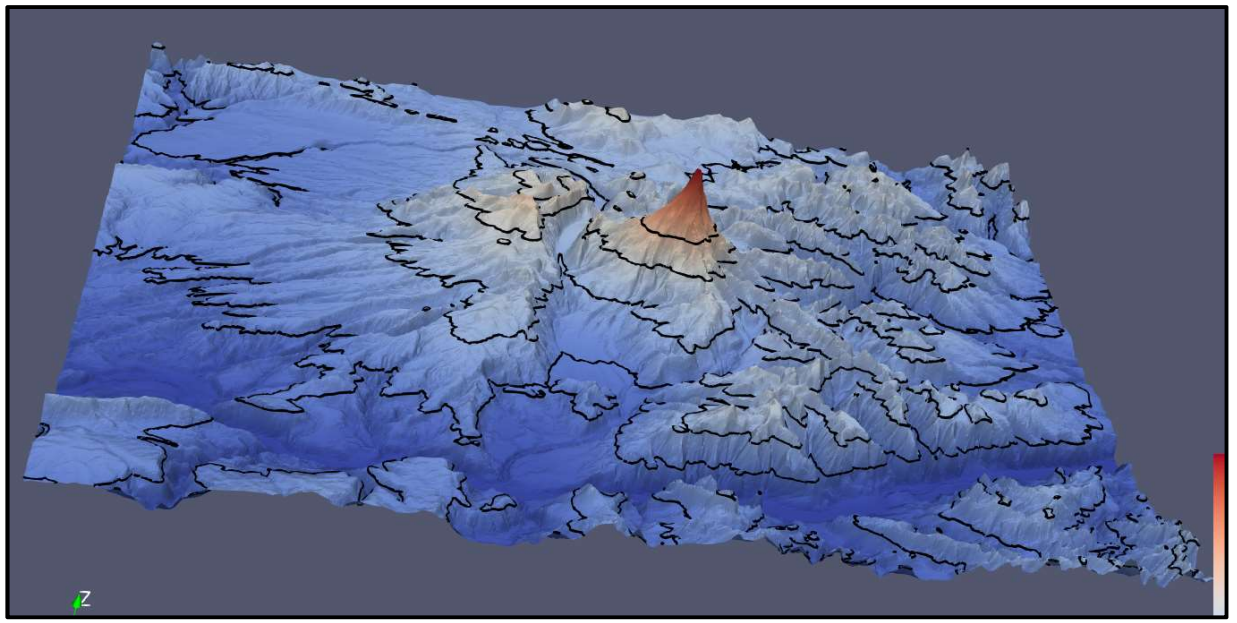
Try coloring by **Elevation**. The heart icon brings up popular color scales. You can pick one of these or sculpt your own.




Elevation
3.332e+03
2500.3
1668.7
837.17
5.592e+00
Z
Y
X

Contouring





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

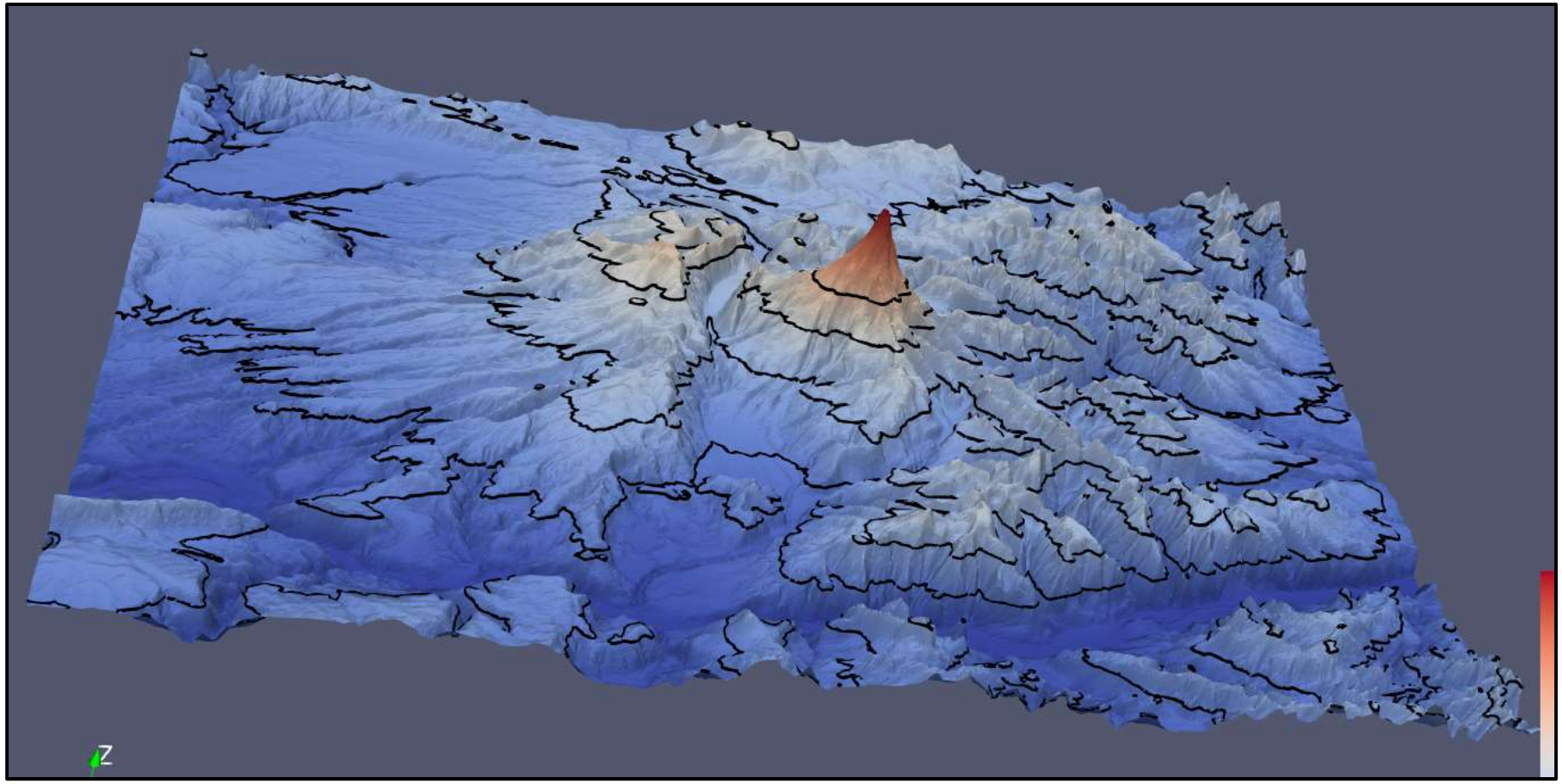
ParaView gives one default contour elevation, but you can add more.

Display as Wireframe.

Edit to select a contour color.

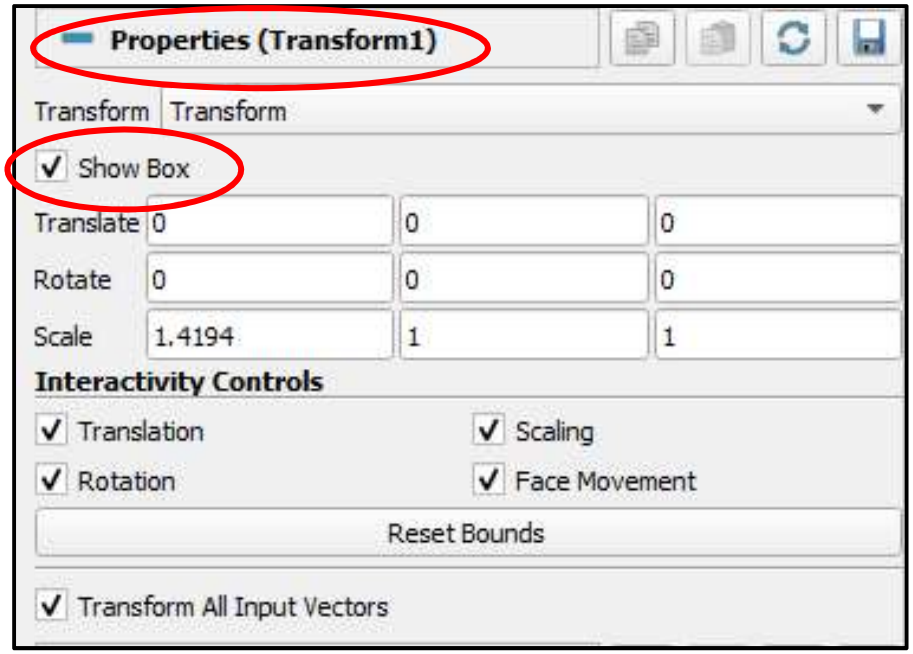
Enter a **Line Width**.

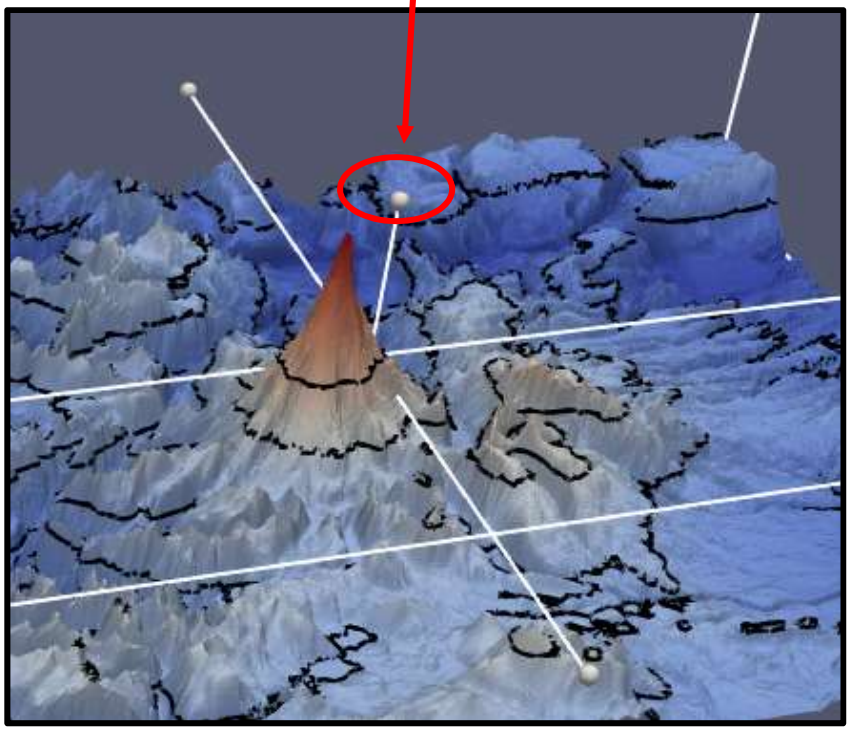
Be sure the eyeballs are turned on.


Z

Changing the Vertical Exaggerations

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

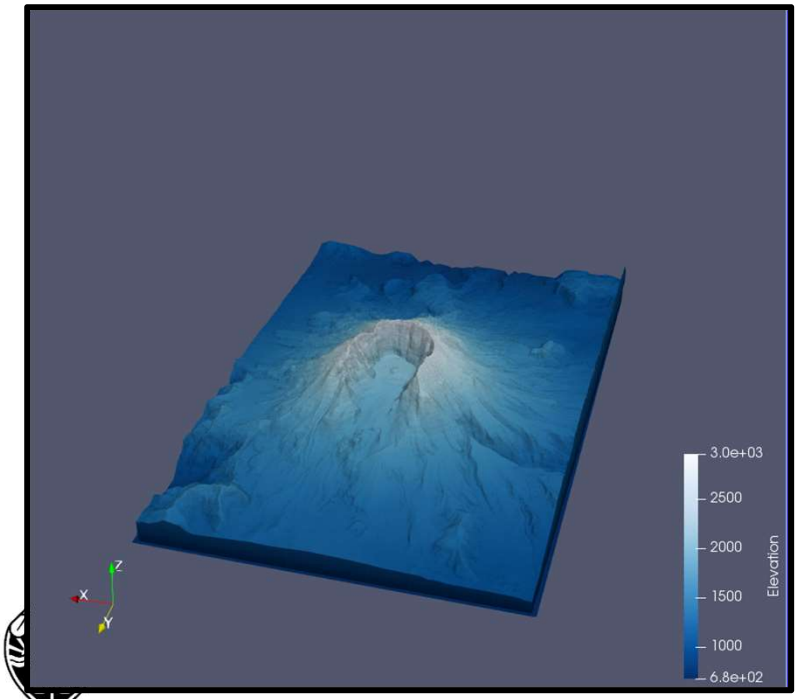


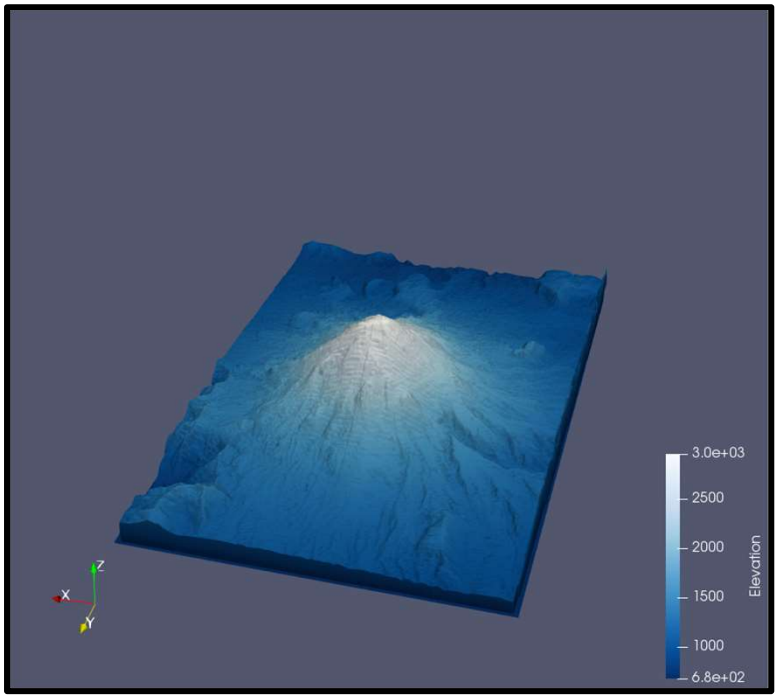


Reading ARCGIS DEM Files

I was able to get to get two DEM files loaded into ParaView, and while not straightforward it's not too hard to do. You need to load in the file, add the **Extract Surface** filter to it, and then the **Warp By Scalar** filter.

Without these filters, ParaView will leave your data as a 2D surface.



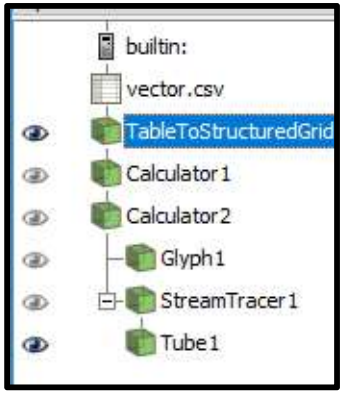


Parallel Coordinates

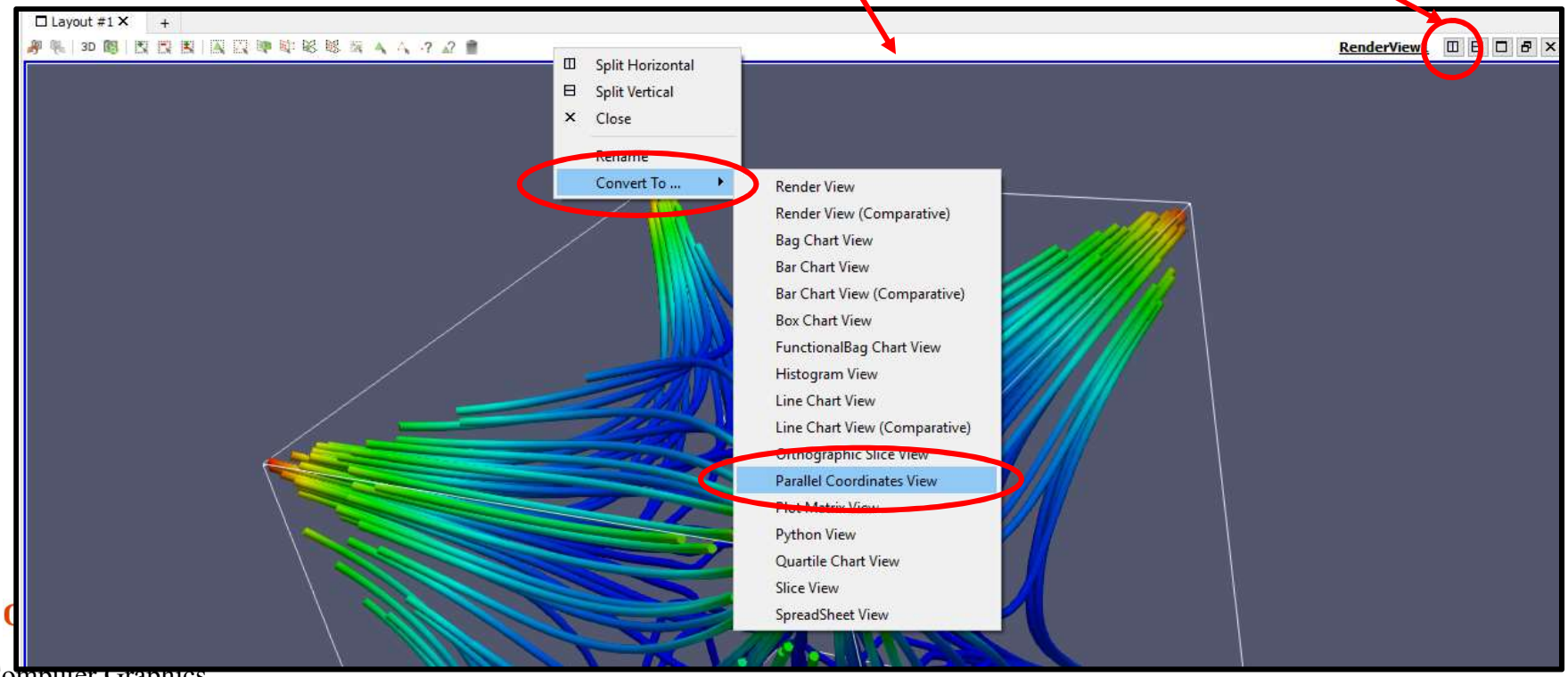
vector.pvsm
parallelcoords.pvsm

Parallel Coordinates – Correlating Fields

Let's say you were to start with this:

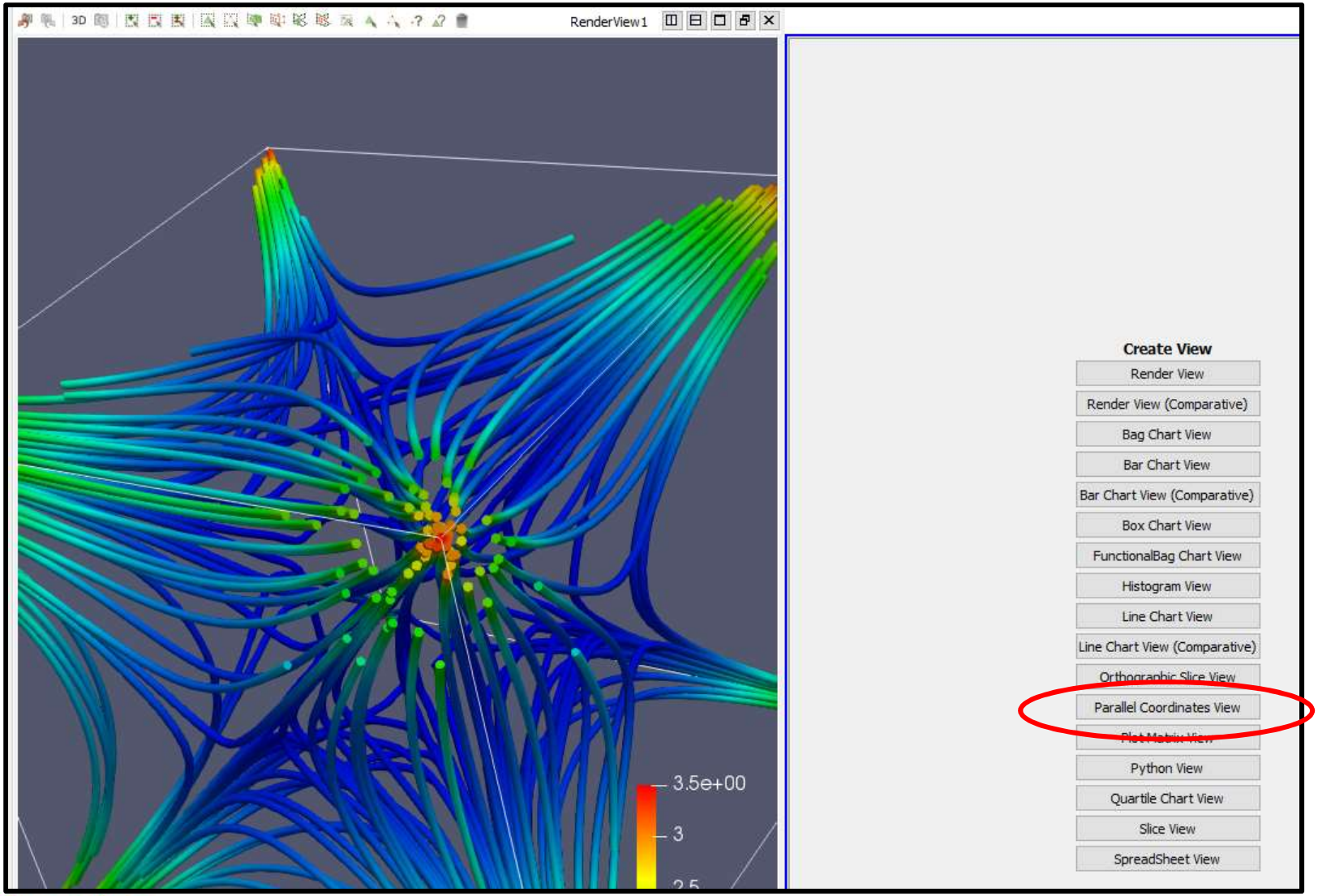


Either convert the **Render View** window to a **Parallel Coordinate View** window by **right-clicking** anywhere in the window header bar, or by splitting the window



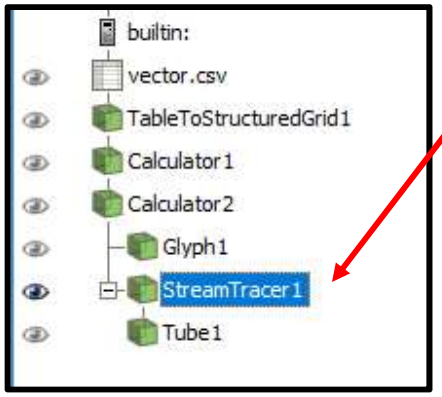
Parallel Coordinates

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

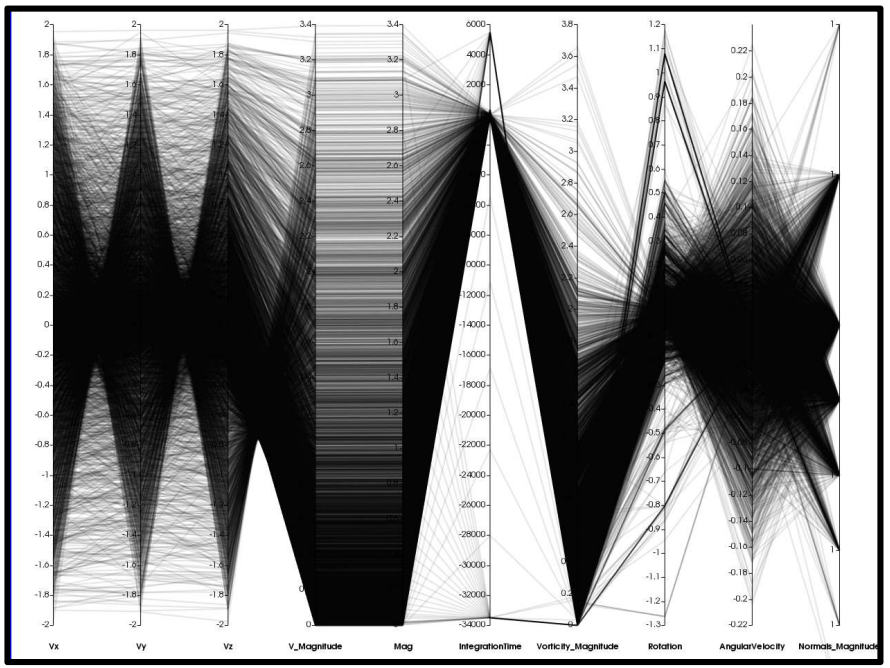


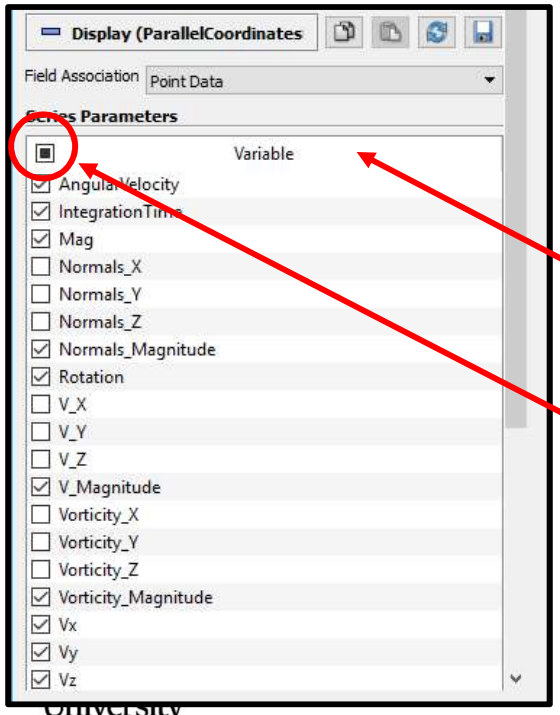
I'm going to do it the first way to give more room for the Parallel Coordinates display.

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.

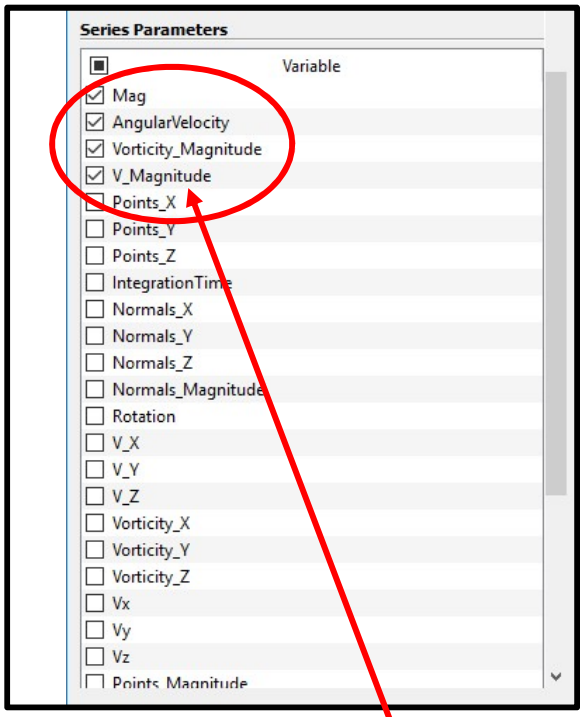


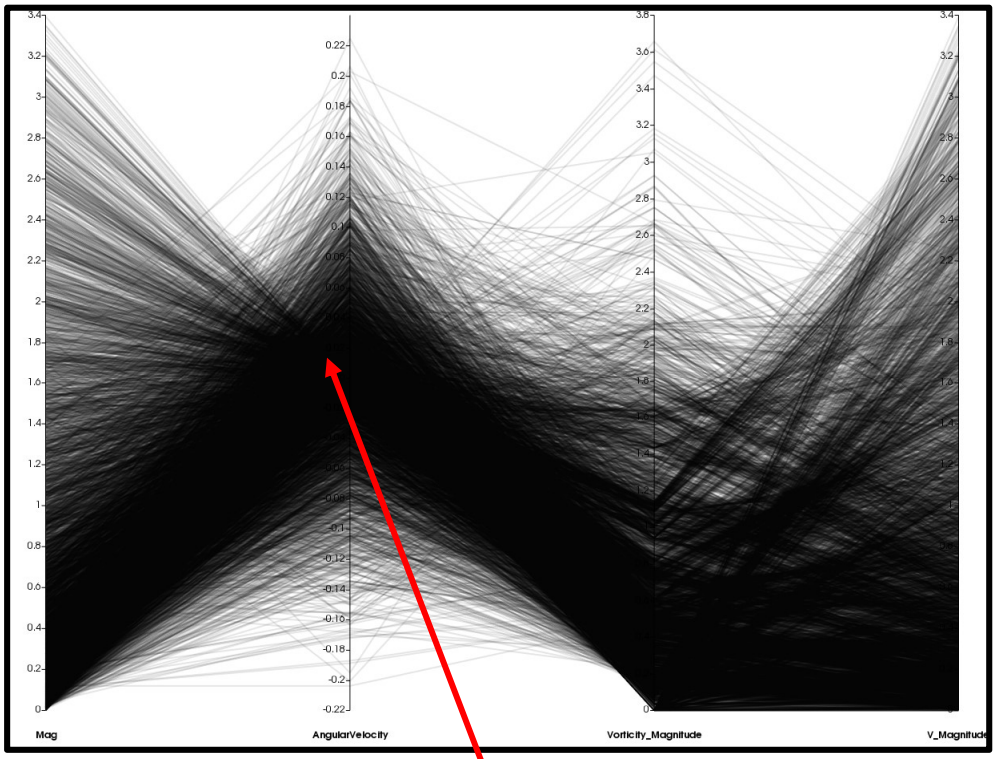


The **Parallel Coordinates Display Properties** shows what variables will be displayed. No matter what, they are probably not exactly the variables you wanted to see and they are not in the desired horizontal order.

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

Parallel Coordinates





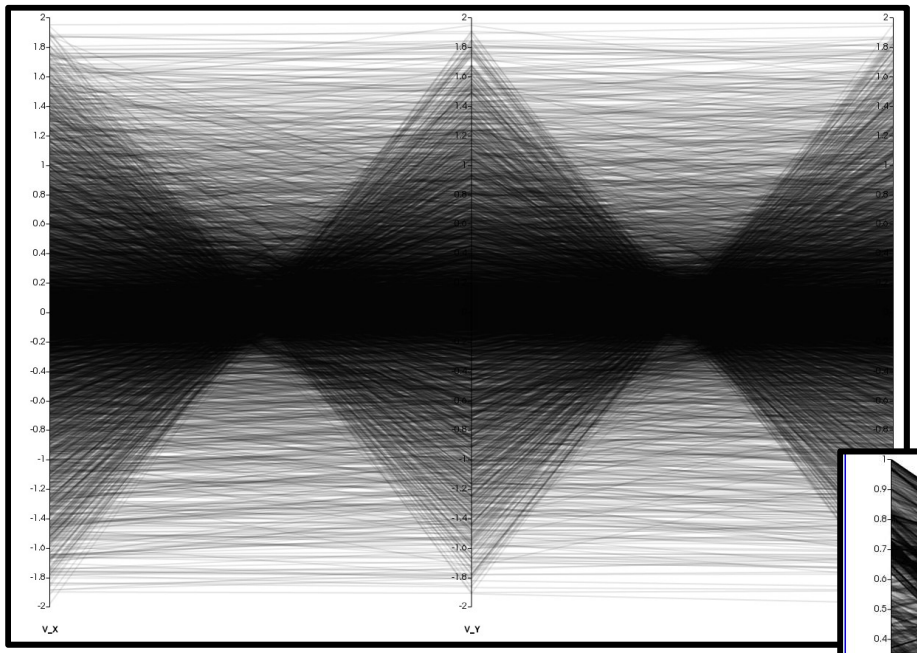
So, click them all off and turn them back on in the horizontal order you want to see them.

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.

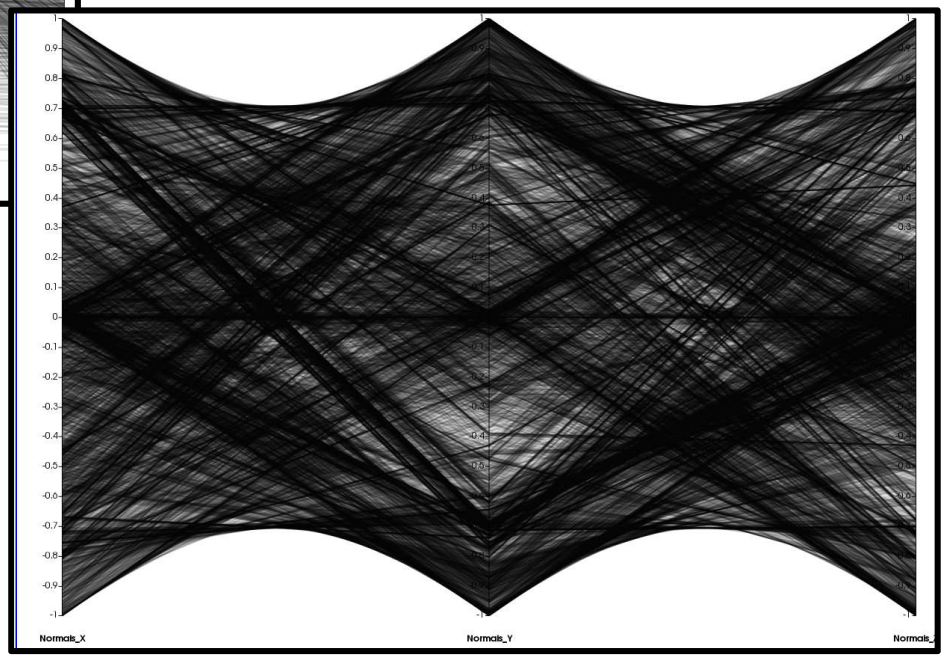
The narrowness of the bundle of lines shows the strength of the positive and negative correlations.

Parallel Coordinates

Lots of (negative) correlation

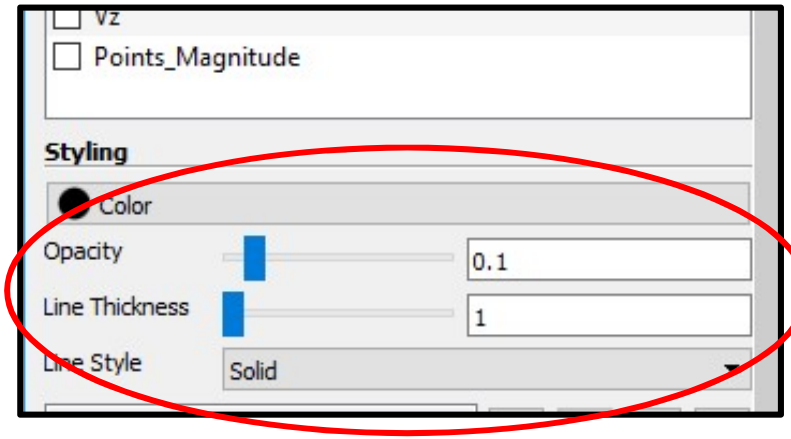


Little correlation

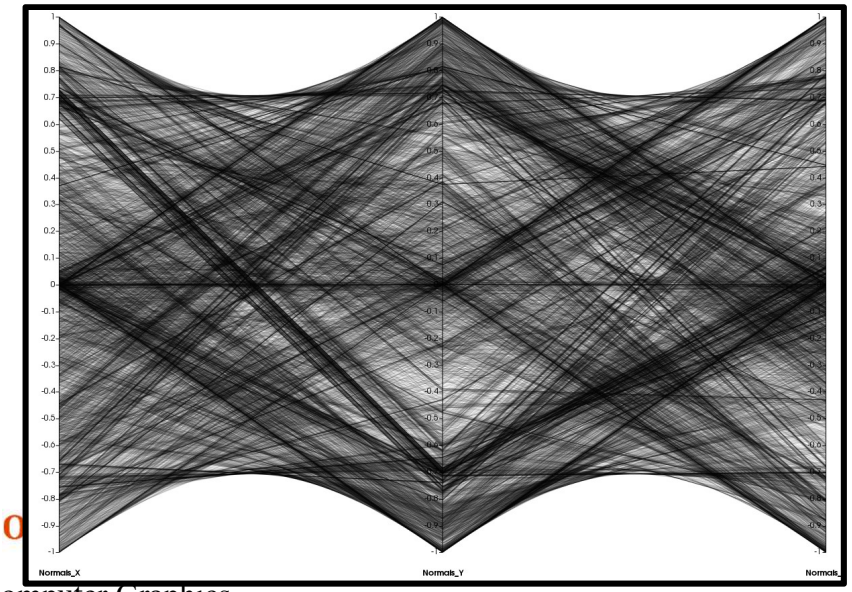


Parallel Coordinates

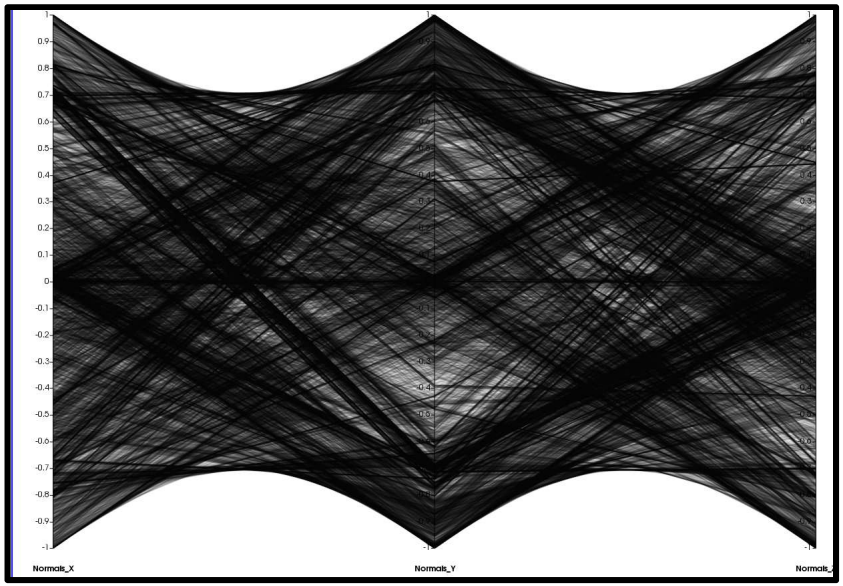
Scroll down a little more in the properties menu and you will find the **Parallel Coordinates Styling** menu:



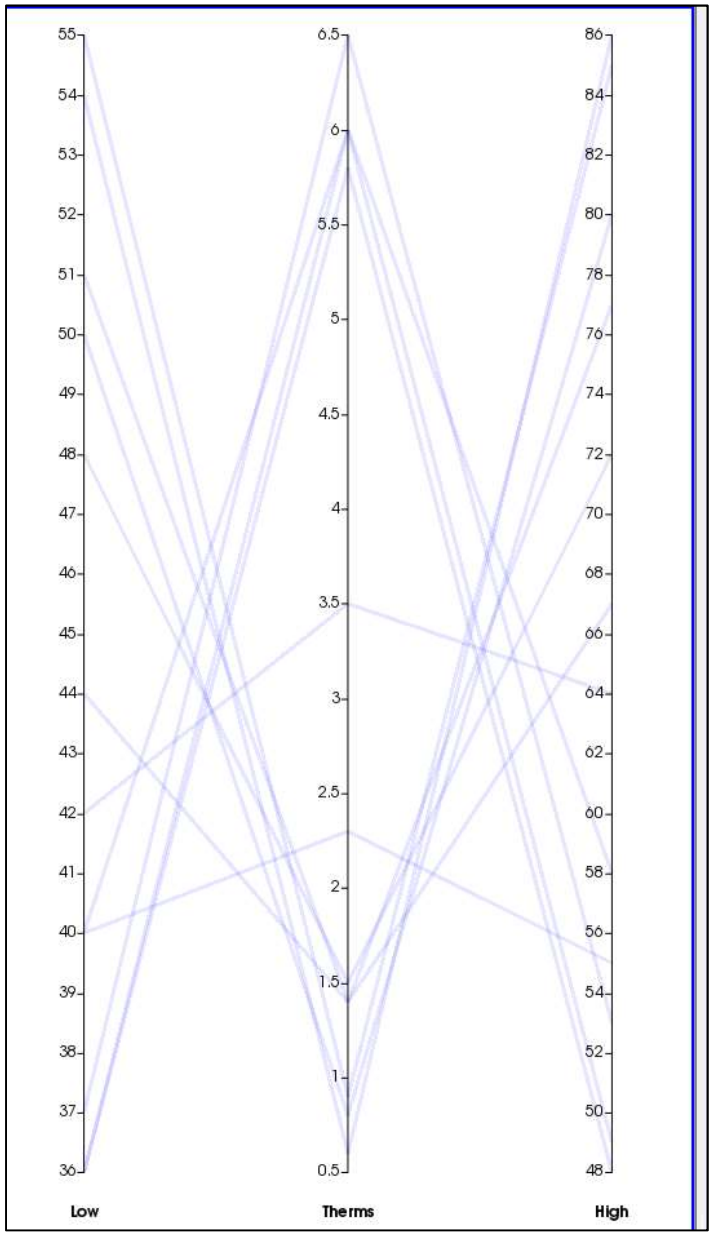
Line Thickness = 1



Line Thickness = 2



Therms on my Natural Gas Bill vs. Average Corvallis Low and High Temperatures

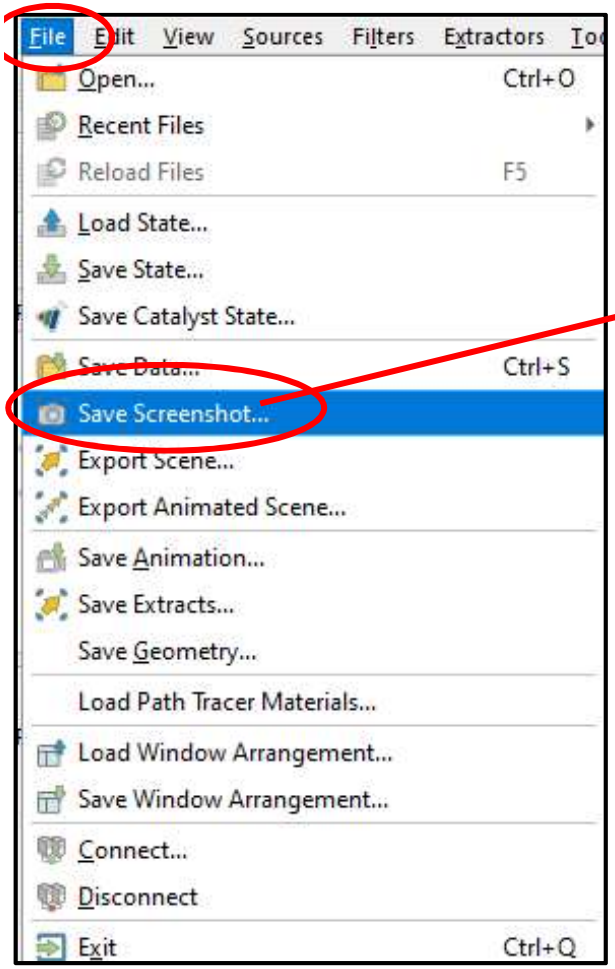


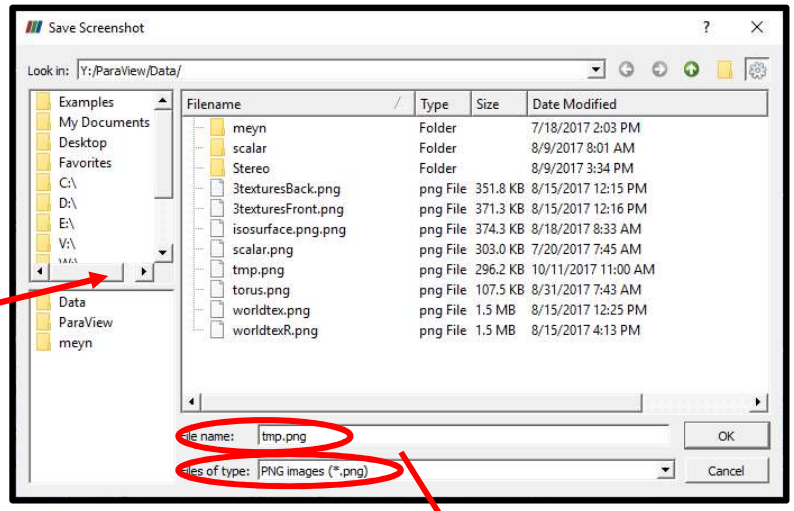
These Parallel Coordinates show that when temperatures are high, natural gas consumption goes down. (Duh, …)

Saving an Image of the Screen

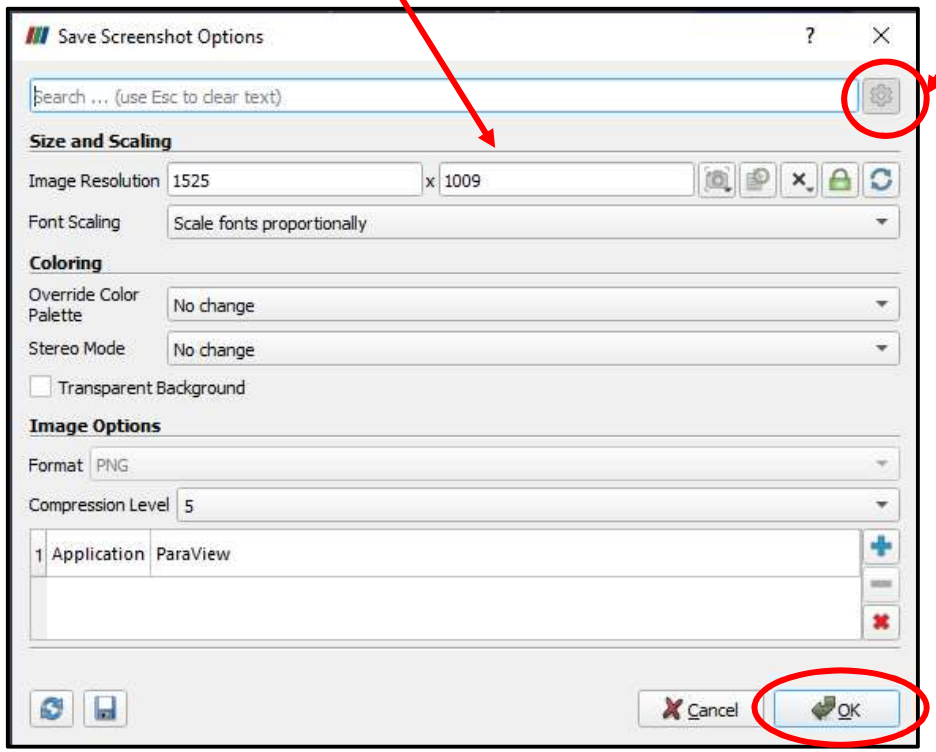
scalar.pvsm

File → Save Screenshot



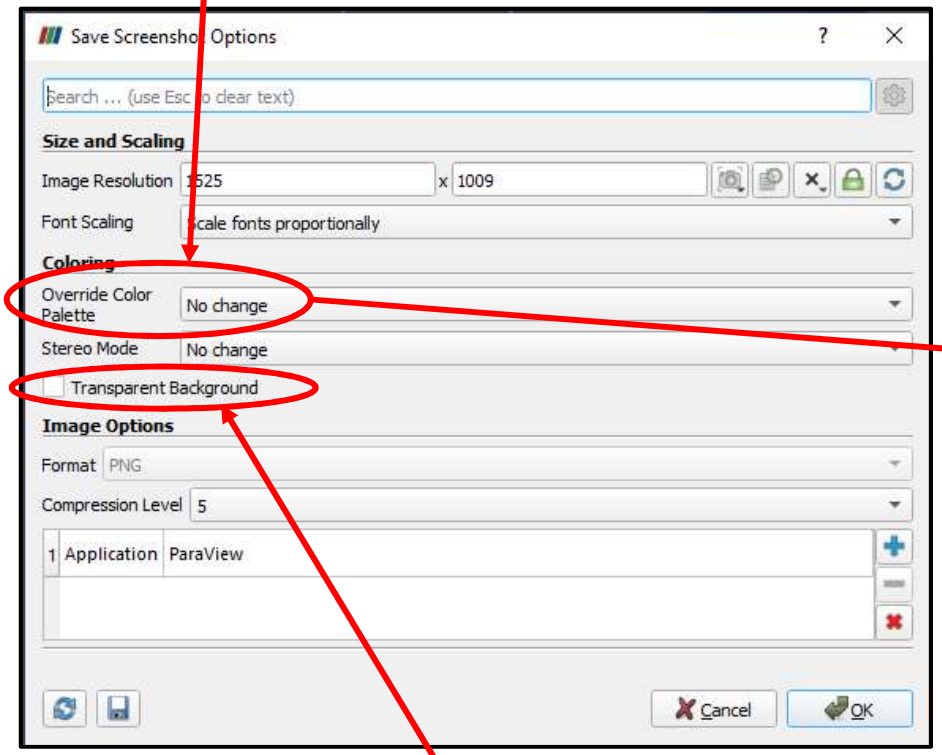


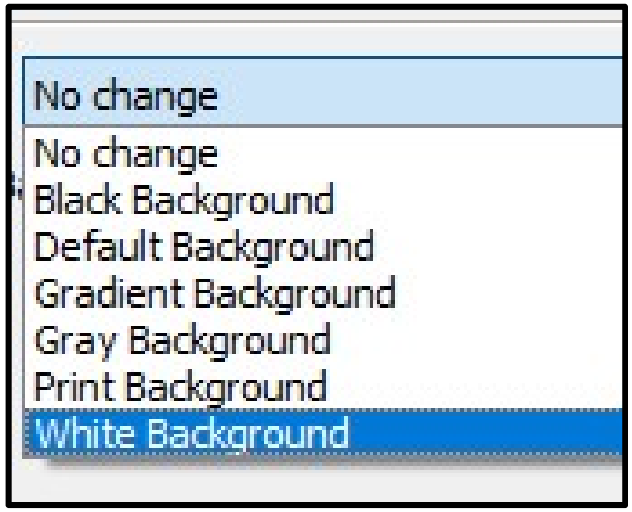
Select the gear to show all options (I recommend this)



Changing the Background Color

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

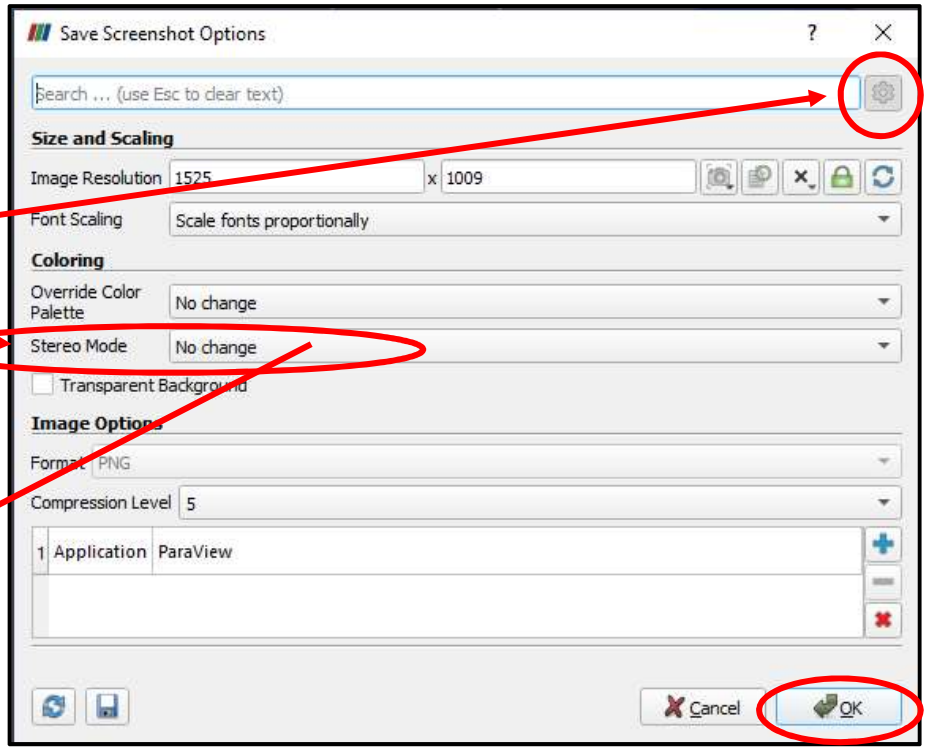


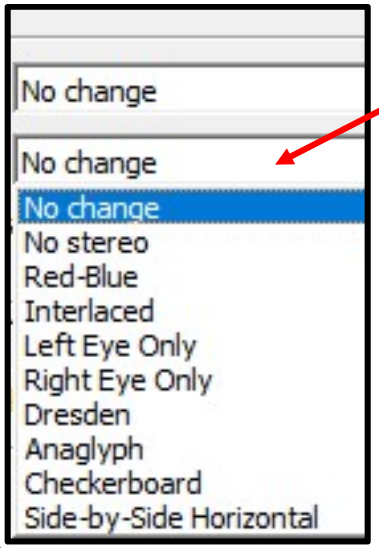


You can also force the image background to be transparent. (This only works on some image file formats, such as PNG.)

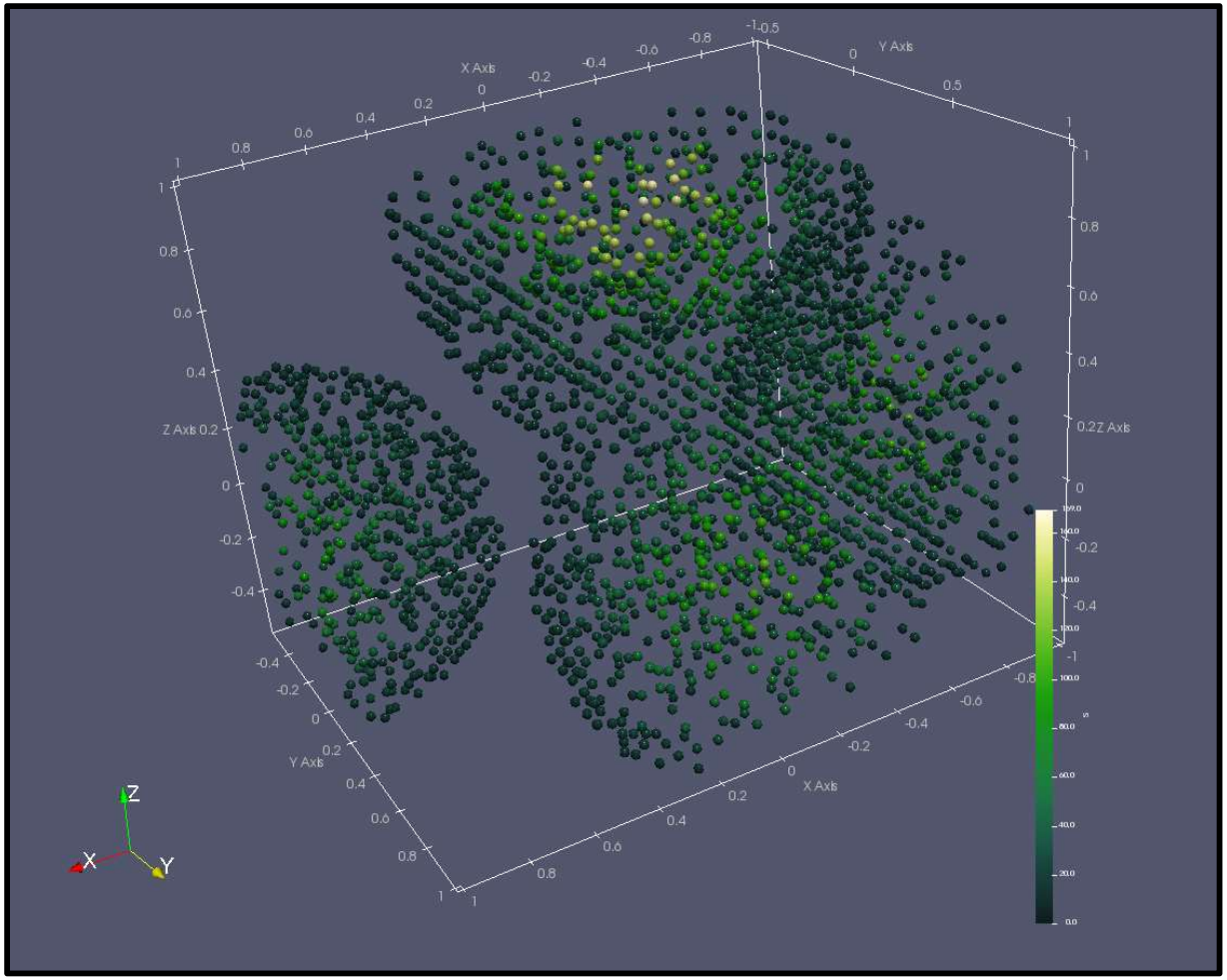
Creating Stereographics Images

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





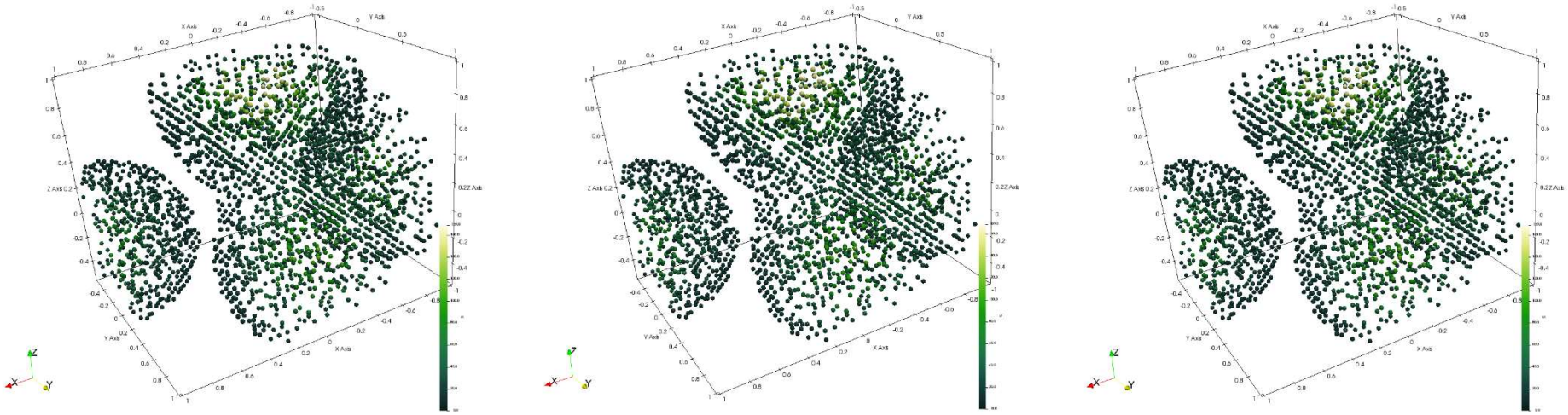
An Original Visualization



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

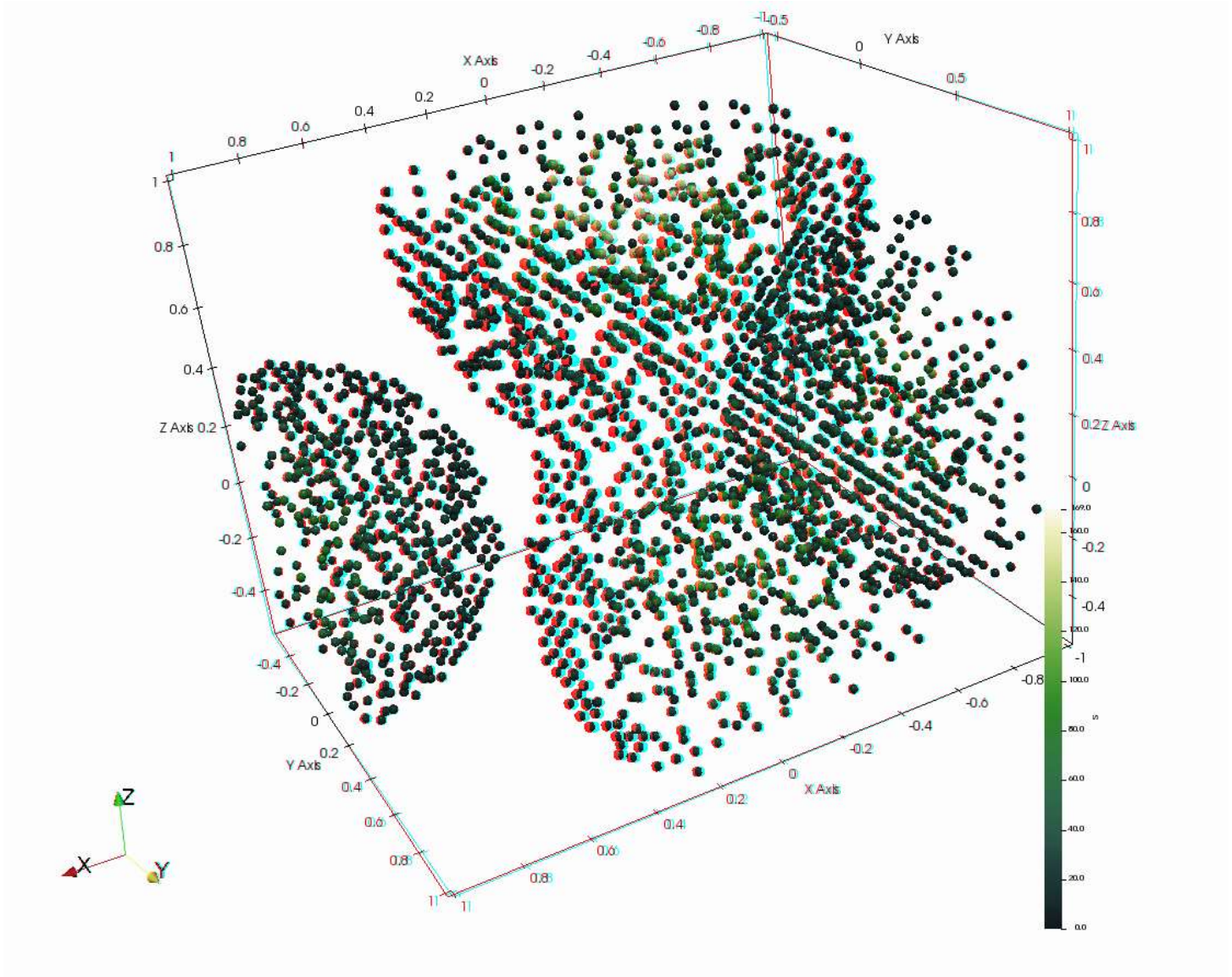
Side-by-Side Stereopairs

L R L



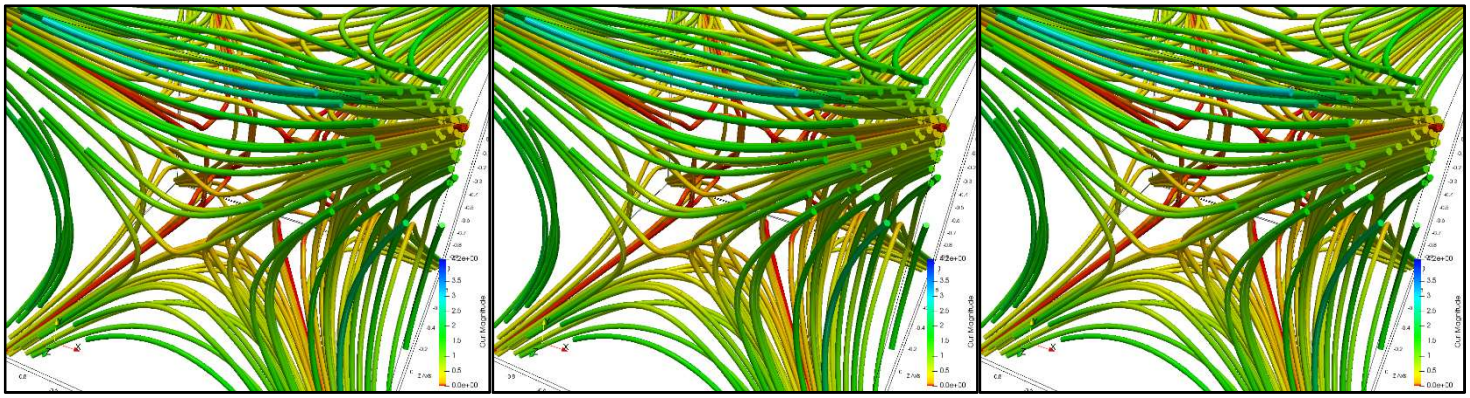
If you can parallel freeview, use the left two images.
If you can cross-eyes freeview, use the right two images
If you can't do either, then never mind

Red-Cyan Anaglyph



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

L R L



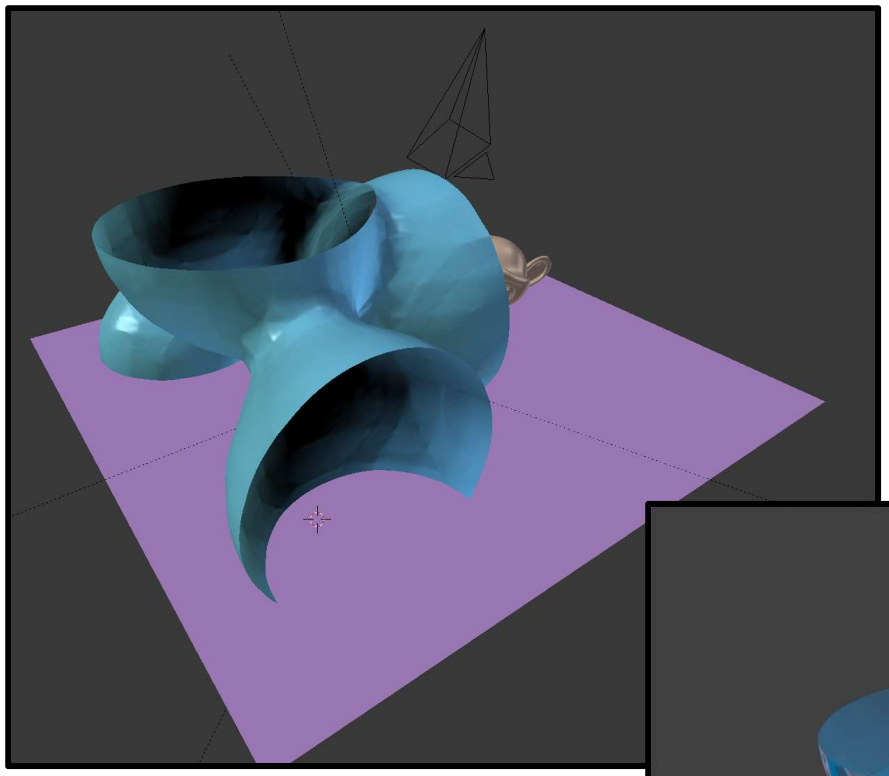
Print this page and cut out the left two images

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.

Exporting the Scene Geometry

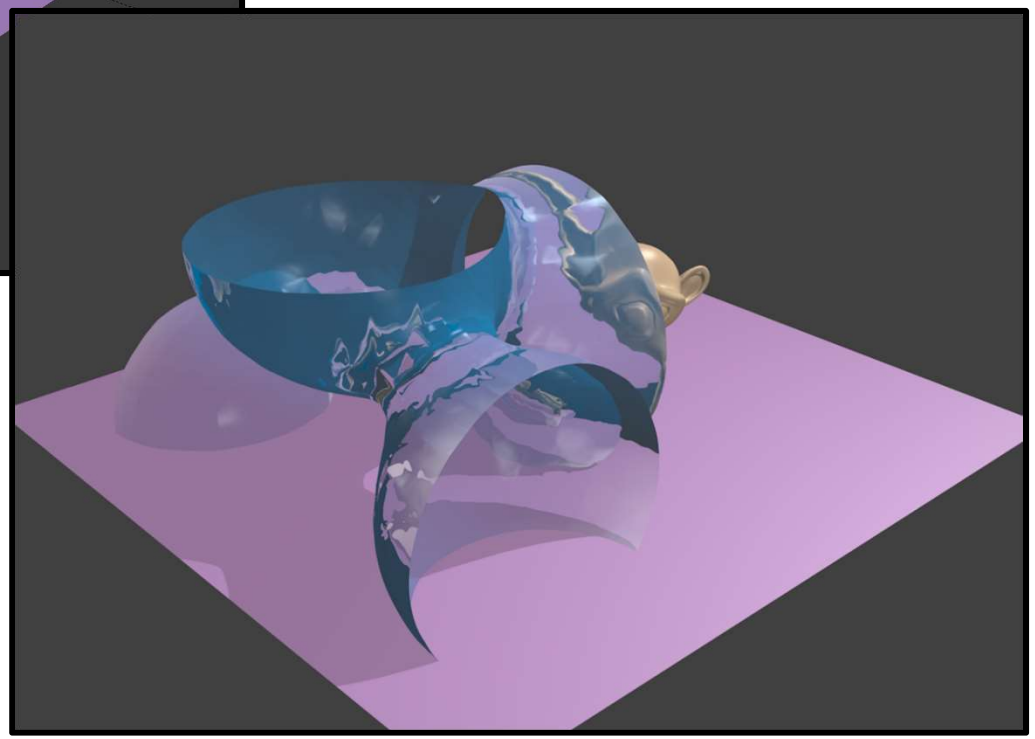
scalar.pvsm

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

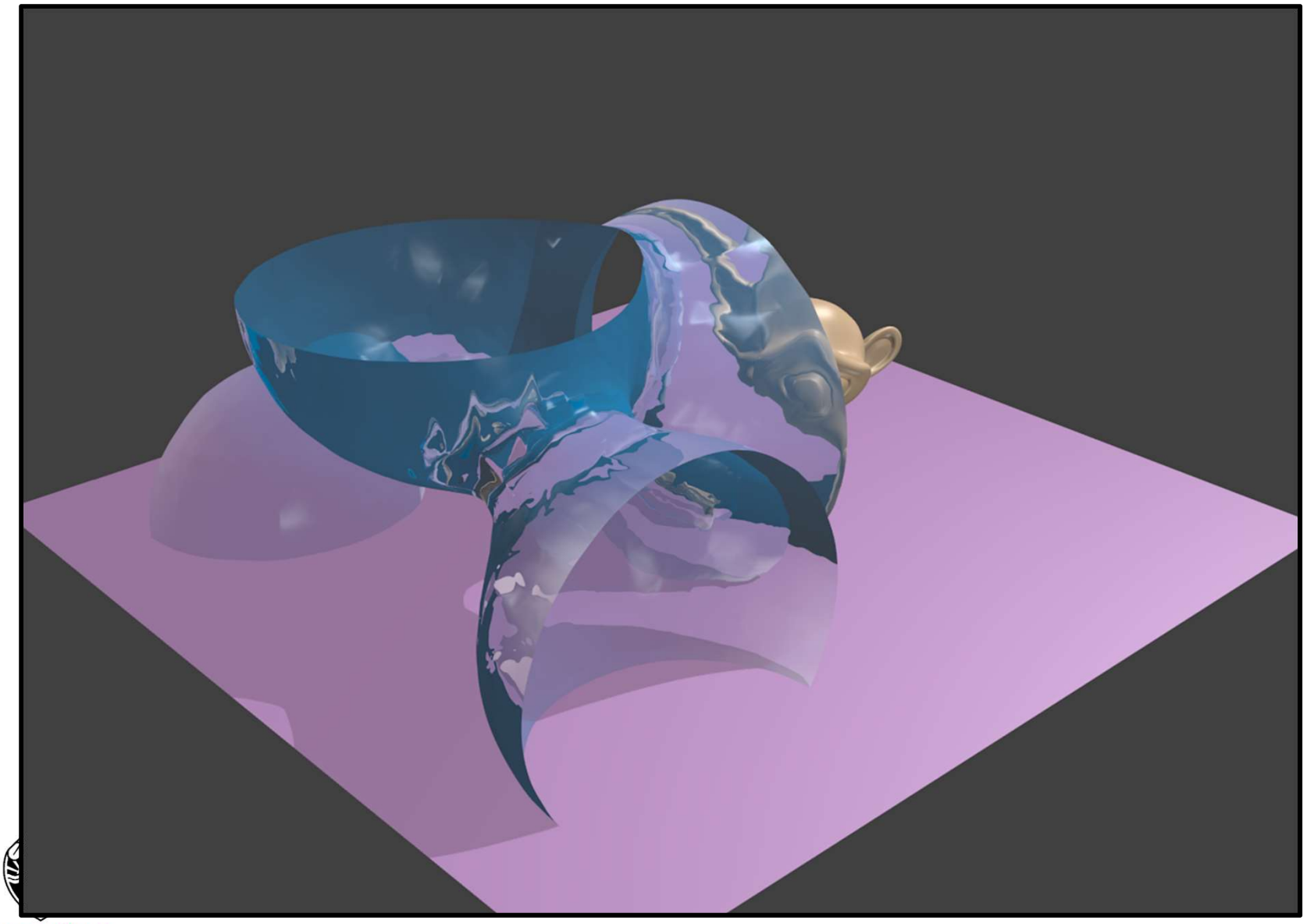


File → Export Scene

You can also export the scene as a GLTF file. I would guess that USD isn't far off.



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

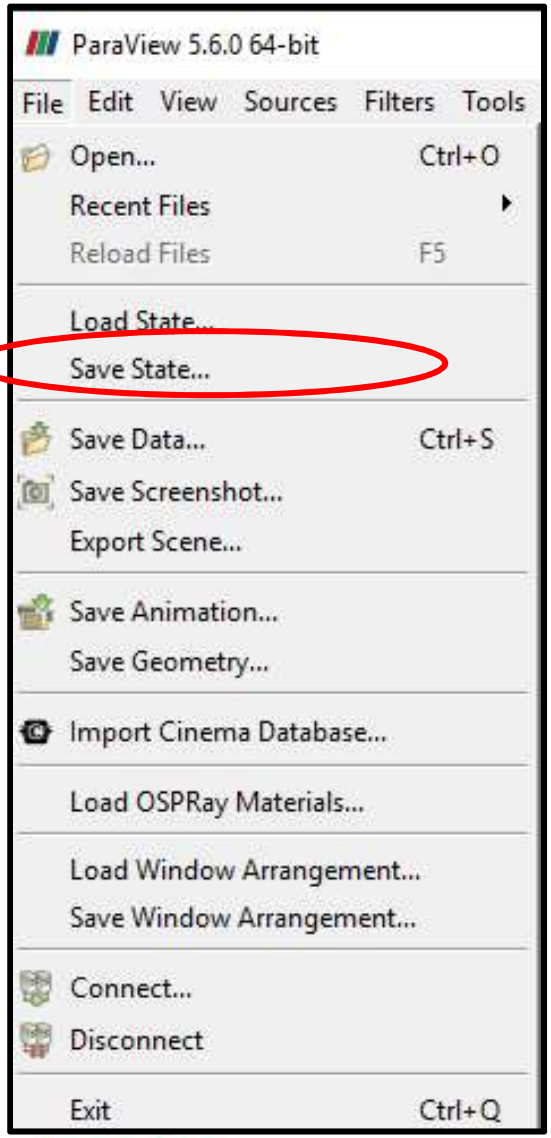


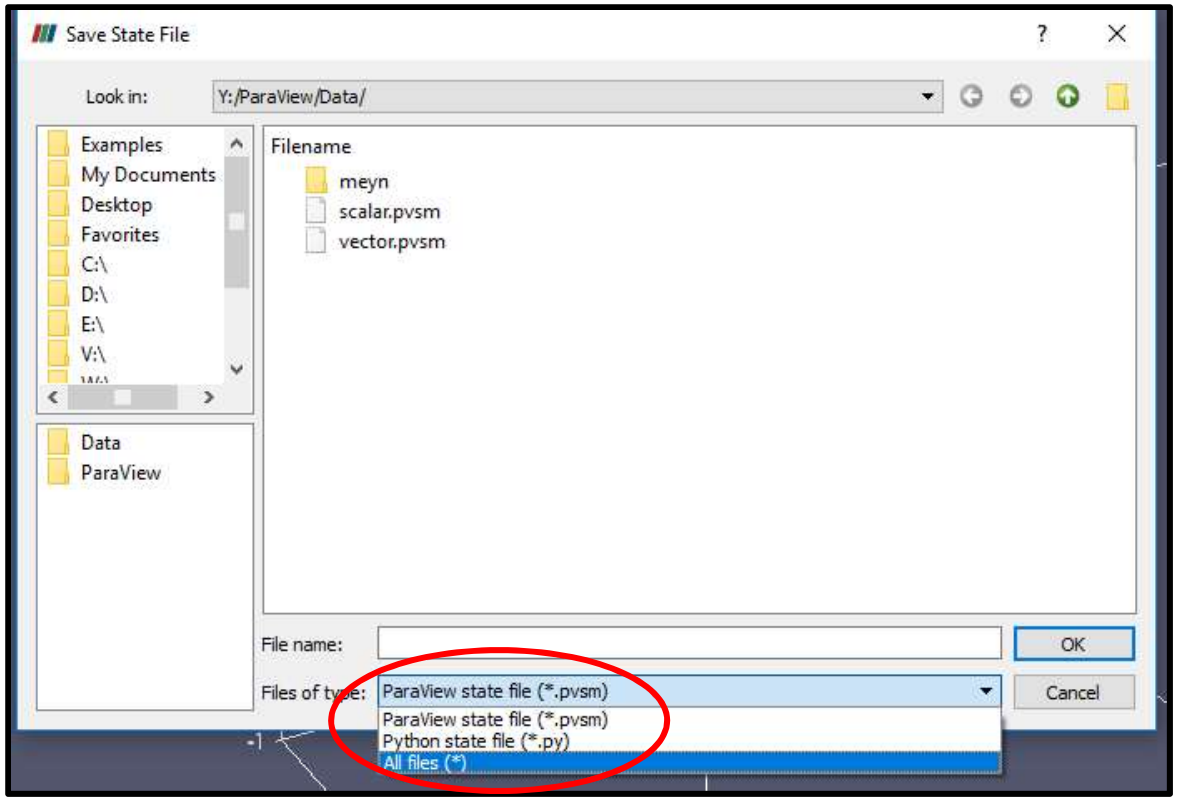
"Should" be able to create STL files from legal solid geometry (e.g., isovolumes) this way, too

Saving the ParaView State

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





"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.

scalar.py

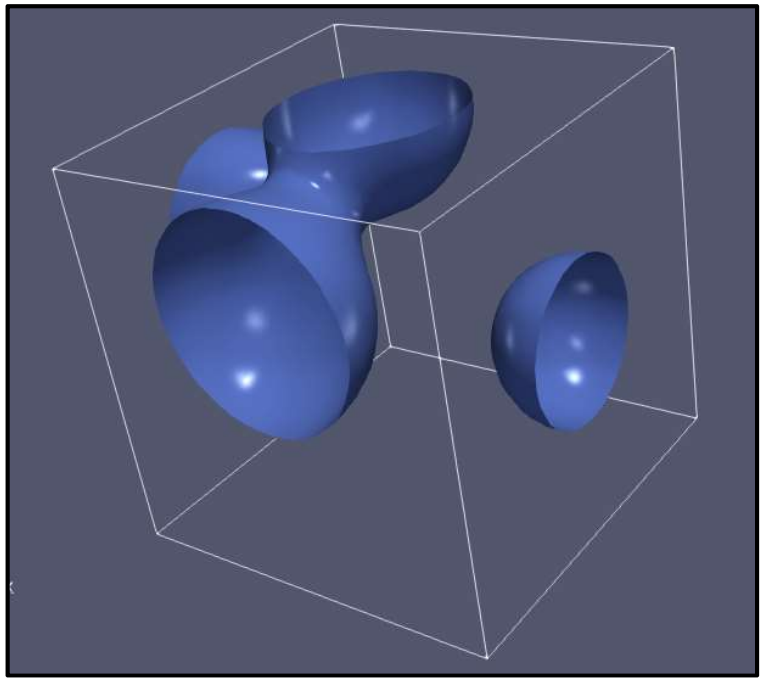

```
# state file generated using paraview version 5.1.2

# ----------------------------------------------------------------
# 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'])
 . . .
```

Animation in ParaView

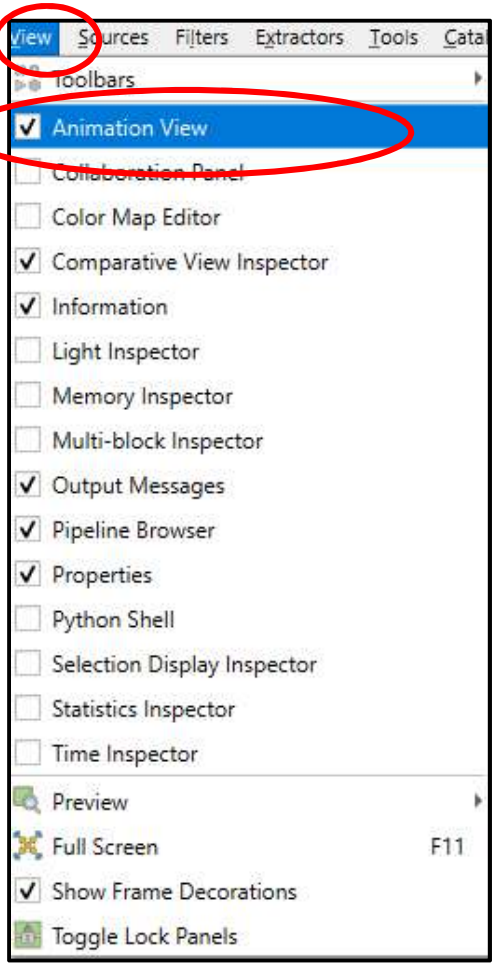
anim.pvsm

Animation in ParaView

Start with this:

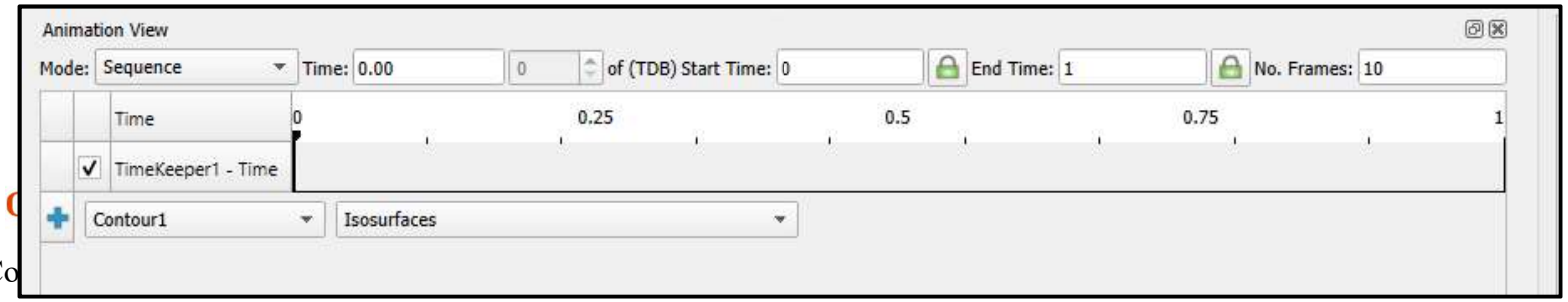


Select this:



anim.pvsm

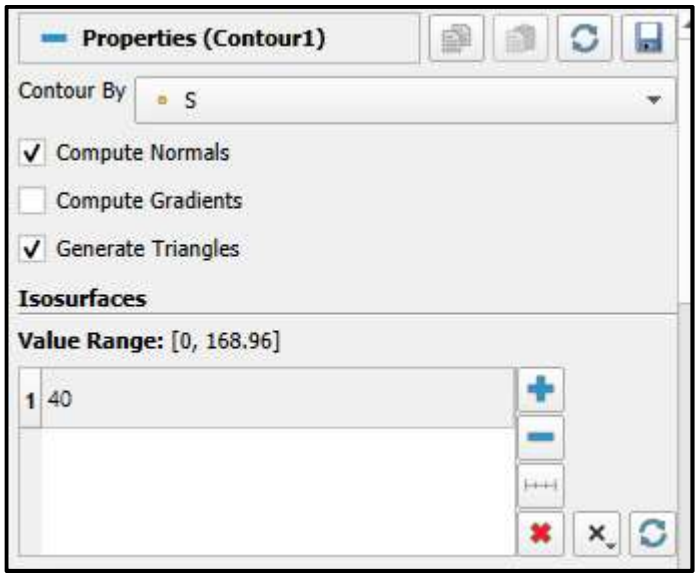
And this appears at the bottom:

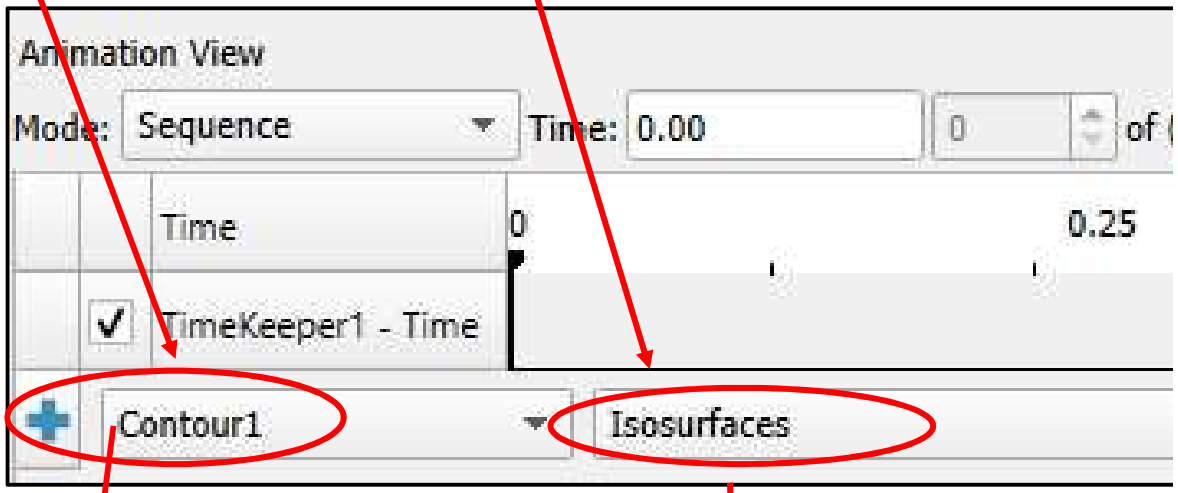


Animation in ParaView – Pick Something to Animate

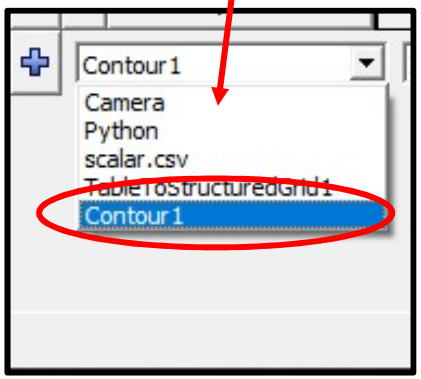
Conveniently, the user interface for animation in ParaView looks a lot like the user interface for Comparative Visualization:

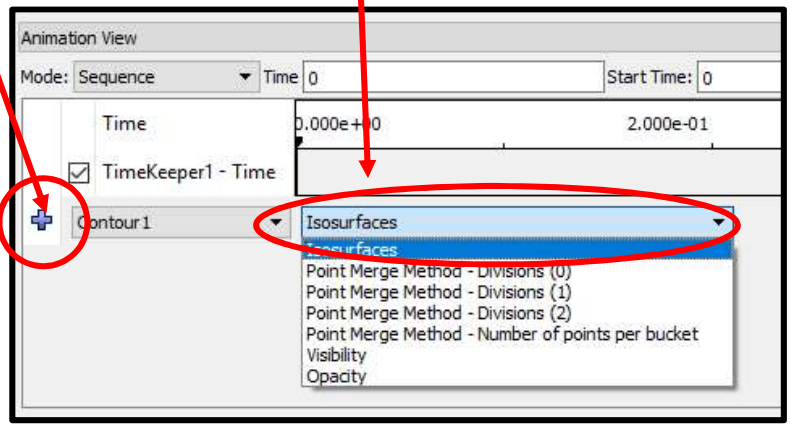
Select a Pipeline Element and a Parameter within that Element





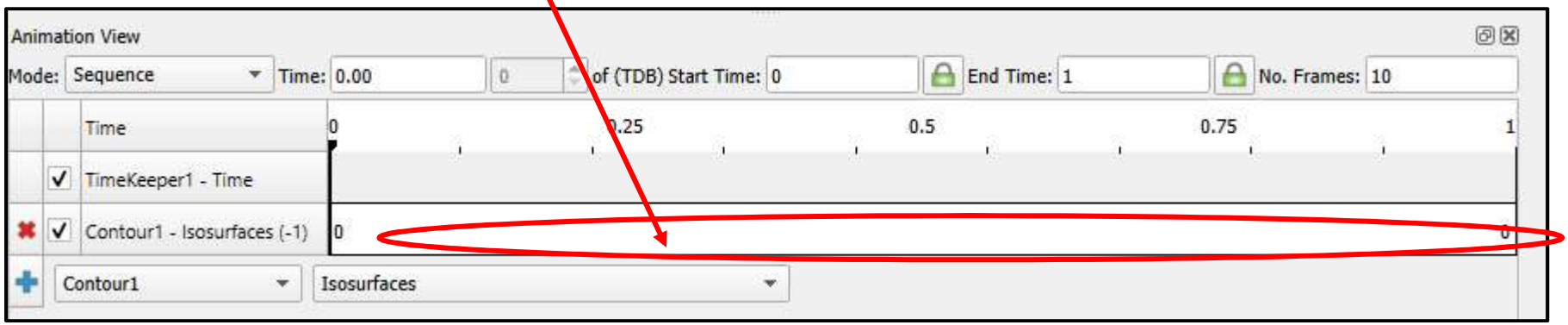
Hit the + when you are done



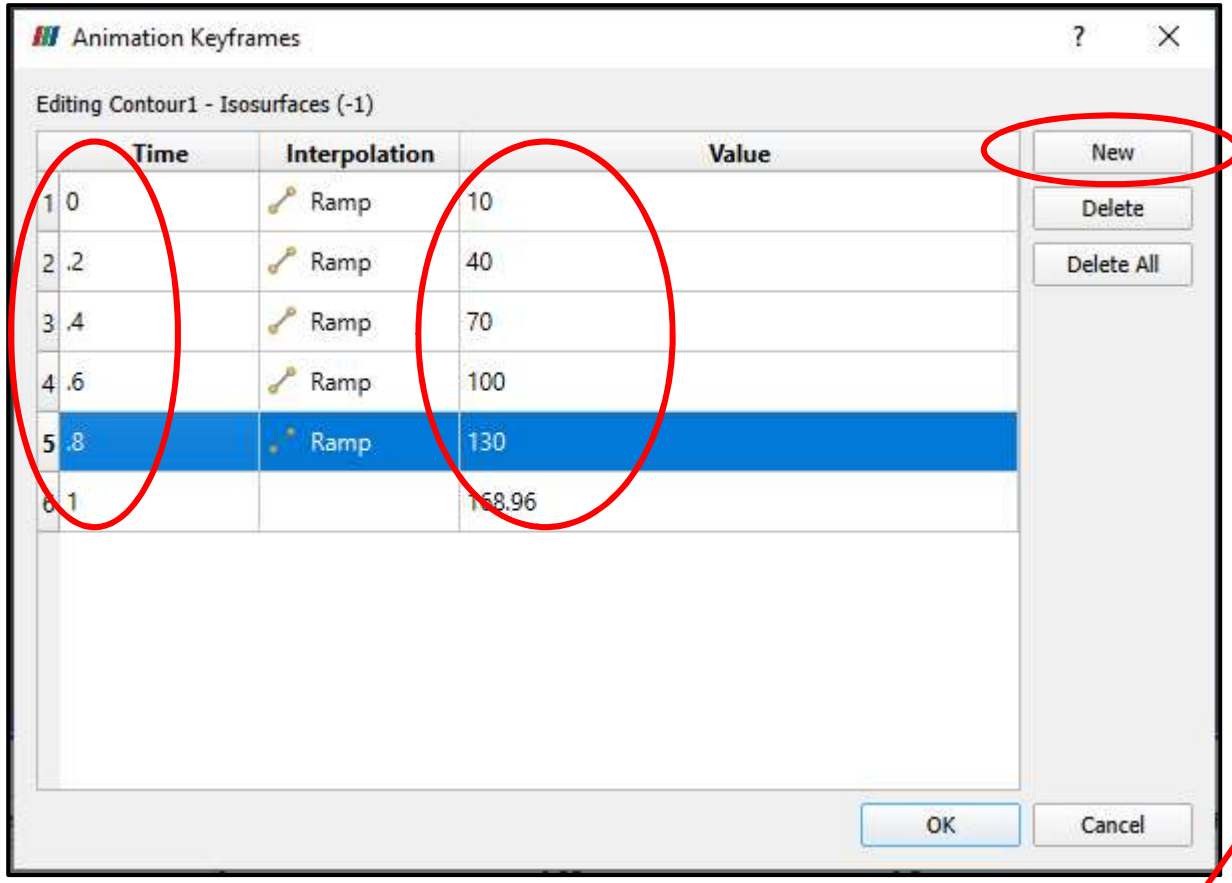


Animation in ParaView – Bring up a Keyframe Menu

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



Animation in ParaView – Setting Parameter Keyframes

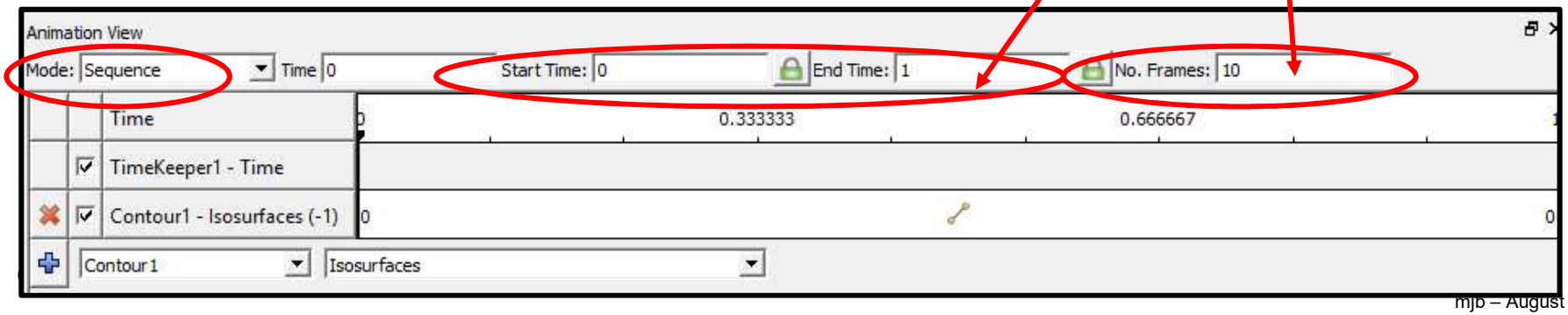


Click **New** to add a new row

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

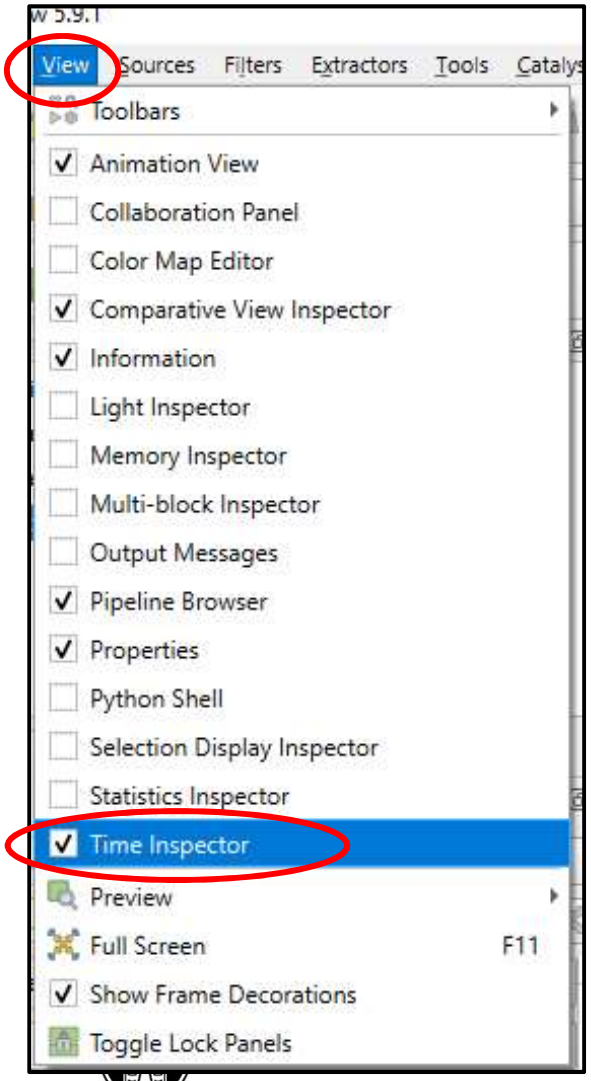
By default, the **Time** starts at **0.** and goes to **1**. – I just left it that way.

I did change the **10** frames to **100** frames, though.



Animation in ParaView – the Time Inspector

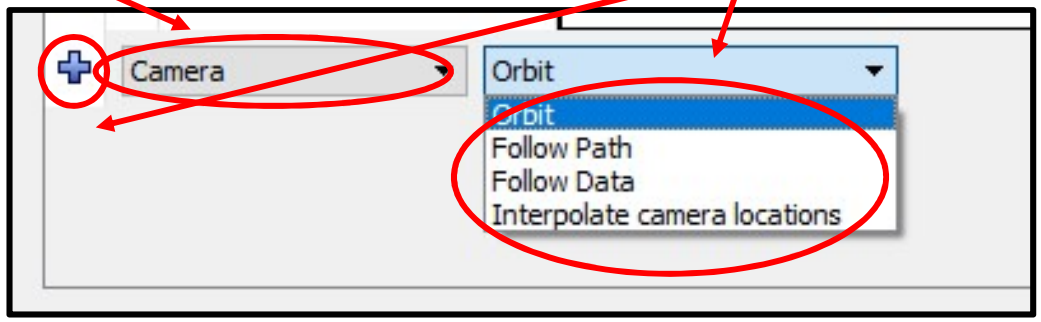
Select this:



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

Animation in ParaView -- Animating the Camera

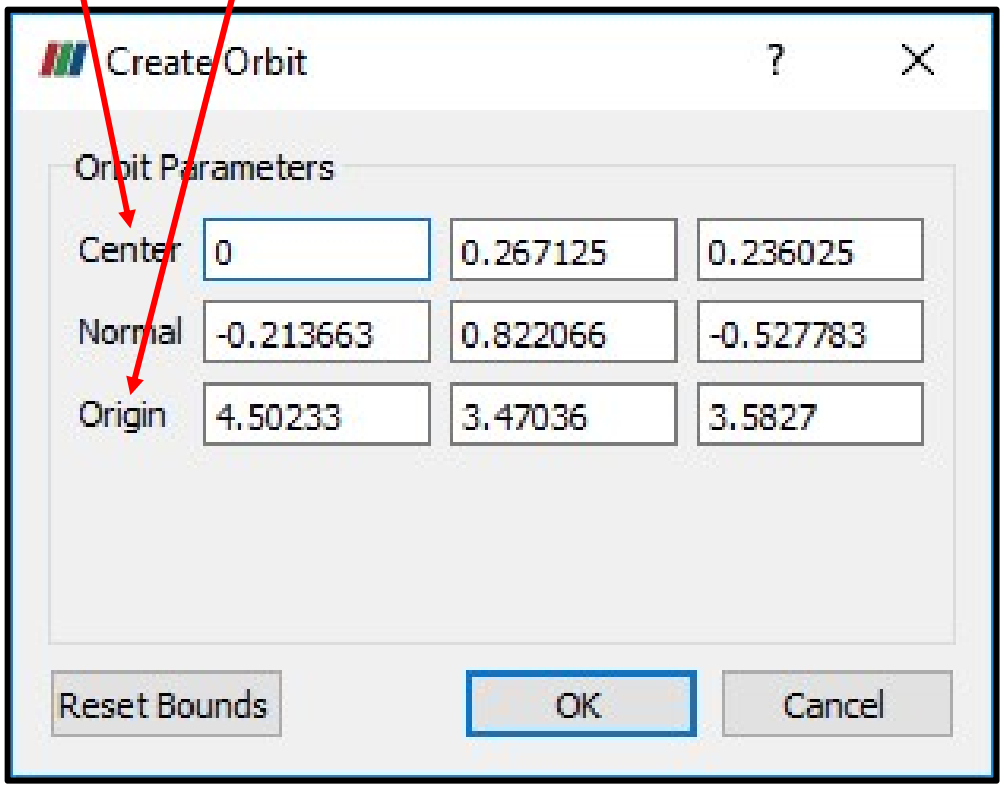
Here's how to animate the **Camera** – select **Camera** from the list of **Properties** and select one of these from the list of **Parameters,** then hit the **+**:



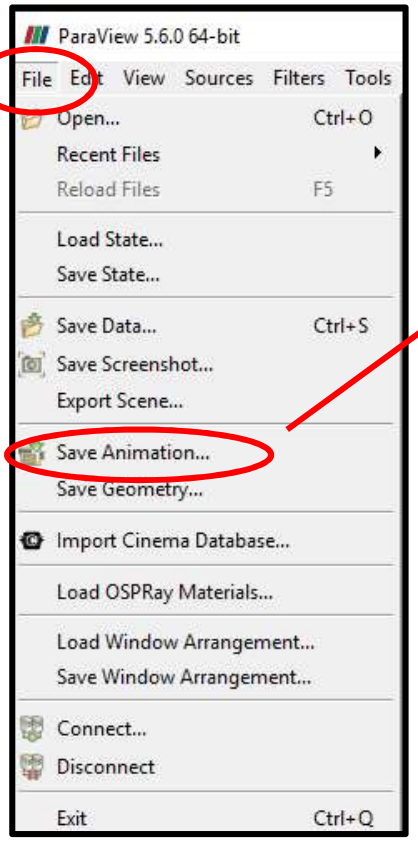
Orbit: animate the camera in a circle around a specific point
Follow Path: set keyframes for the camera position and look-at point
Follow Data: ??
Interpolate camera locations: Manually specify keyframe camera locations

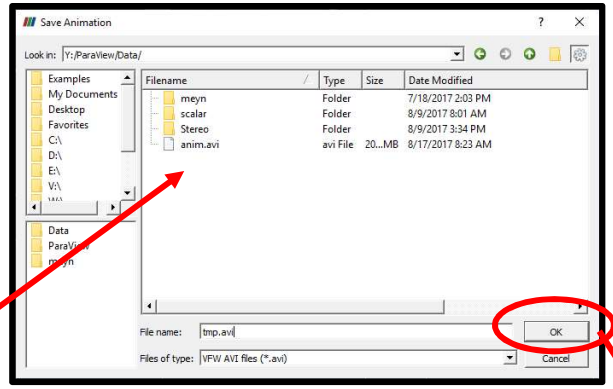
Animation in ParaView -- Orbiting the Camera

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.



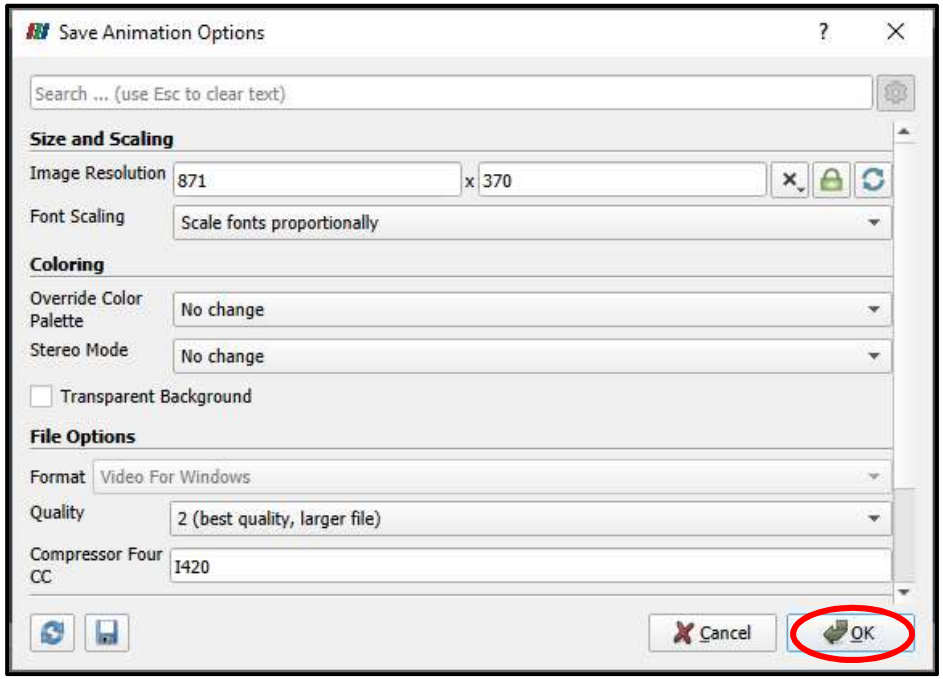
Saving the Animation





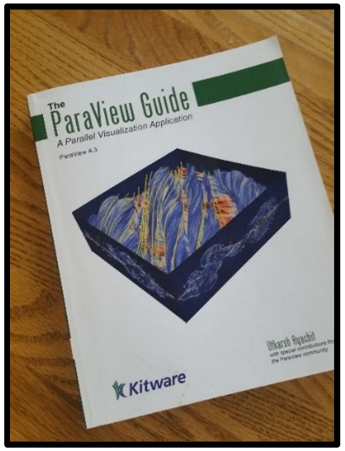
Clicking **Save Animation** brings up a file navigator dialog. You can save the animation in either **AVI** or **OGV** formats.

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.

References



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

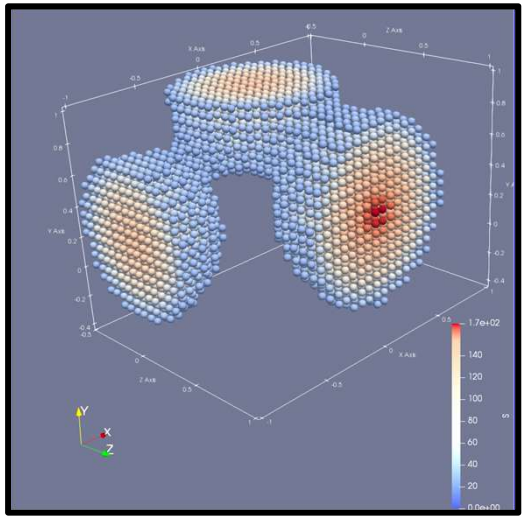
Utkarsh Ayachit. *The ParaView Guide: A Parallel Visualization Application*, Kitware, 2015.

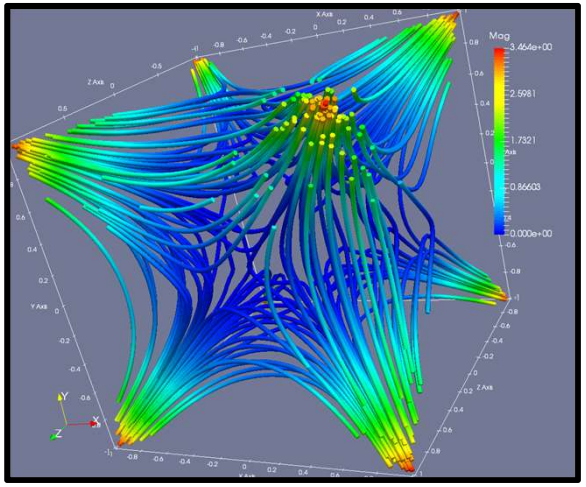
A free PDF of the book can be found here:
https://www.paraview.org/paraview-guide/

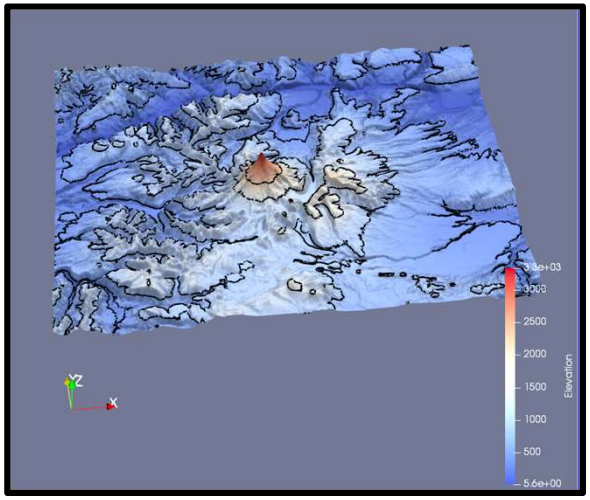
The ParaView tutorial:
https://www.paraview.org/Wiki/The_ParaView_Tutorial

ParaView

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







Oregon State University

Mike Bailey

mjb@cs.oregonstate.edu

Oregon State University
Computer Graphics