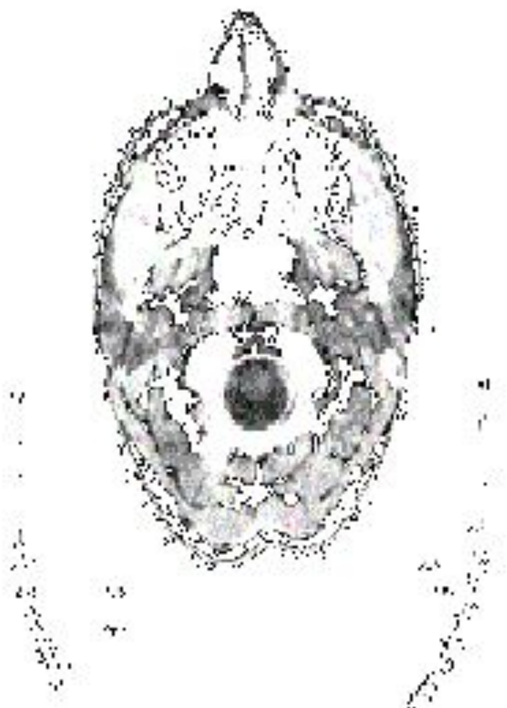
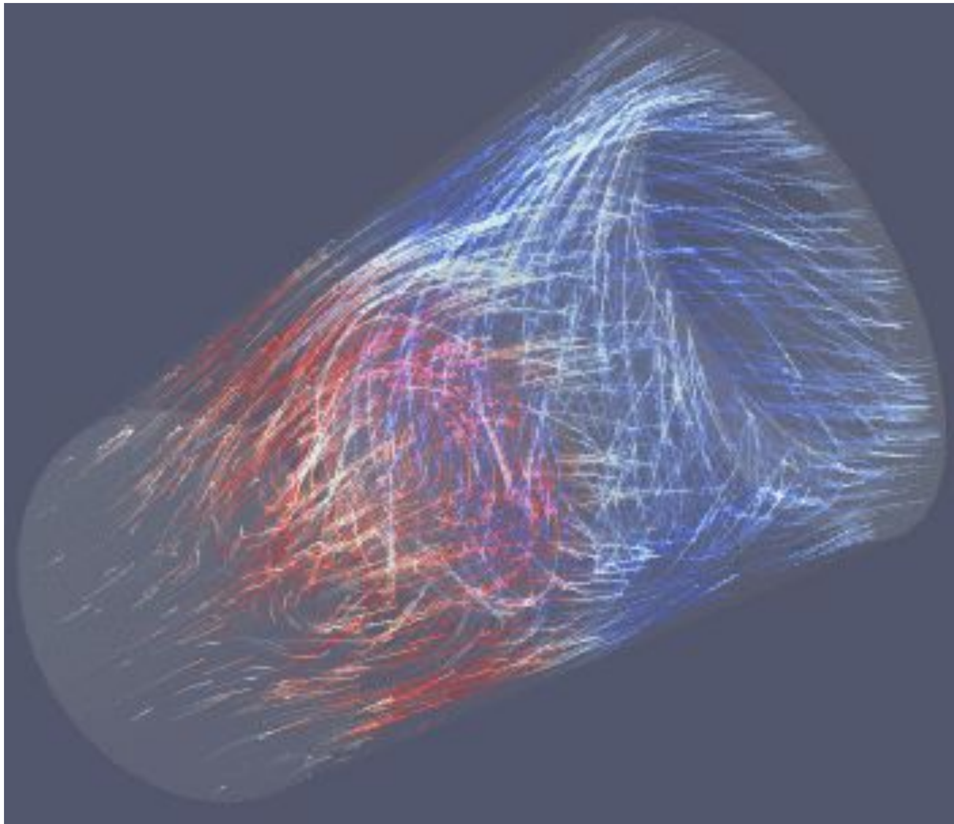
Hands-on demo on high performance 3D Data Visualization using ParaView

Martín Huarte-Espinosa
martin.huartee@gmail.com





Get started!

Download data samples from:

https://tinyurl.com/shpcp-vis

Make sure you save data into the Downloads folder so you can easily find them.

If you have not done so yet, download and install ParaView from:

https://www.paraview.org/download/

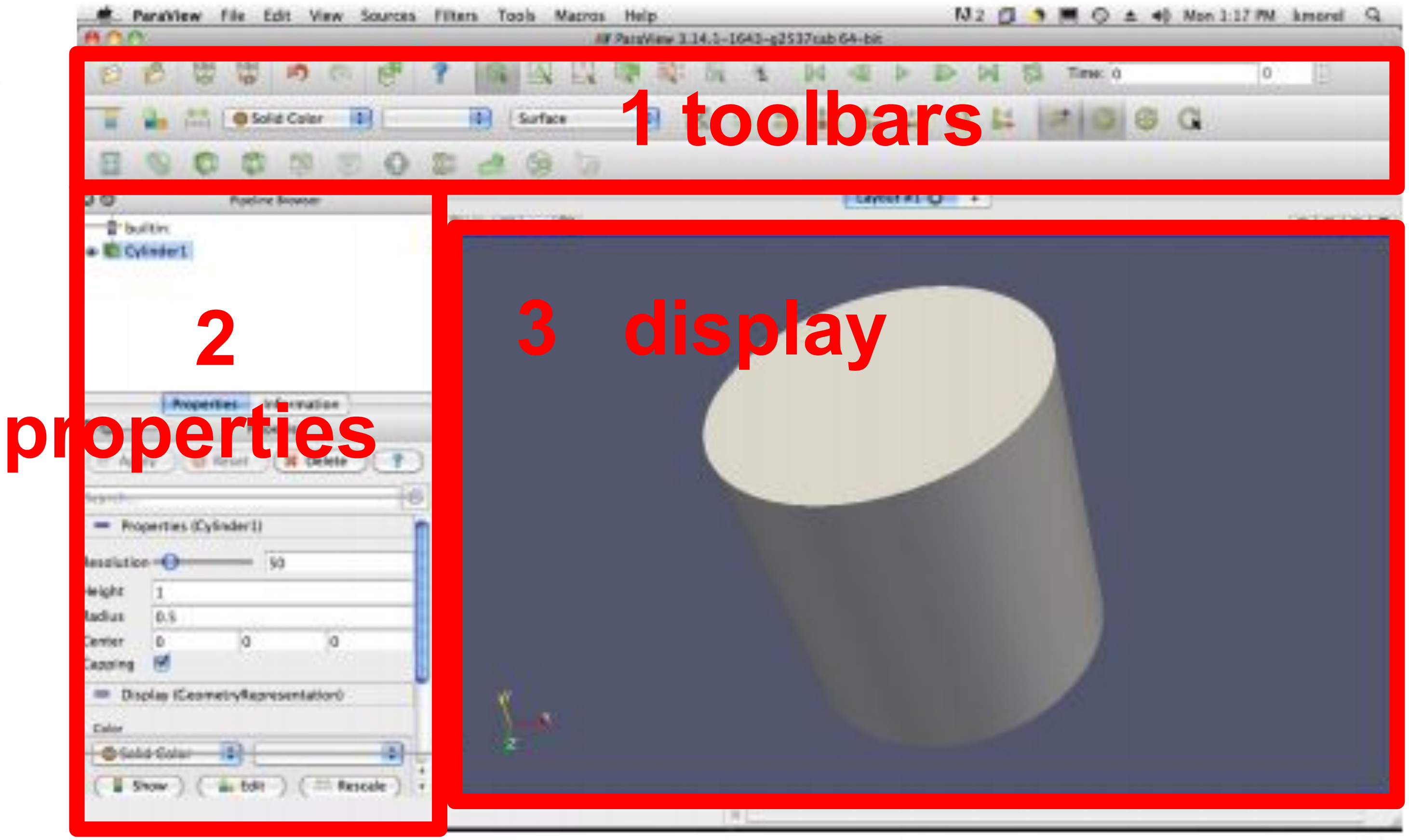
Version v5.6

ParaView — Sources Windows Linux **macOS**

ParaView-5.6.0-MPI-OSX10.8-64bit.unsigned.dmg	Mar 5 13:28	153.6M
ParaView-5.6.0-MPI-OSX10.8-64bit.pkg	Mar 5 13:28	155.0M
ParaView-5.6.0-MPI-OSX10.8-64bit.dmg	Mar 5 13:27	155.5M

Paraview's GUI

3 sections



Important buttons (for now)

open data

undo   *redo*

camera & rotation controls

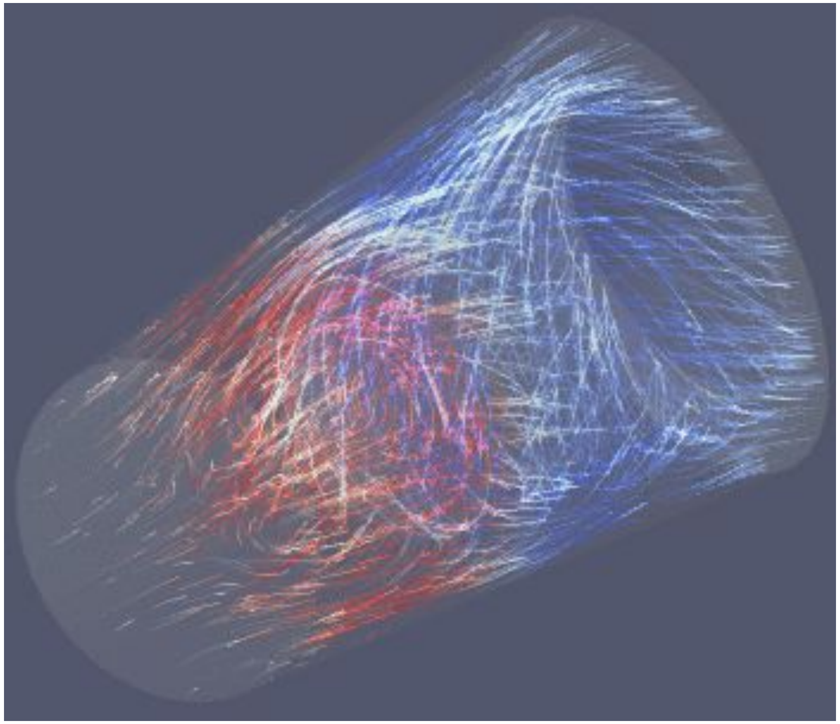
Let’s load some data

Paraview provides some examples to help you learn. Let’s use *disk_out_ref.ex2:*

- Output from a simulation of air flowing around a hot, spinning disk

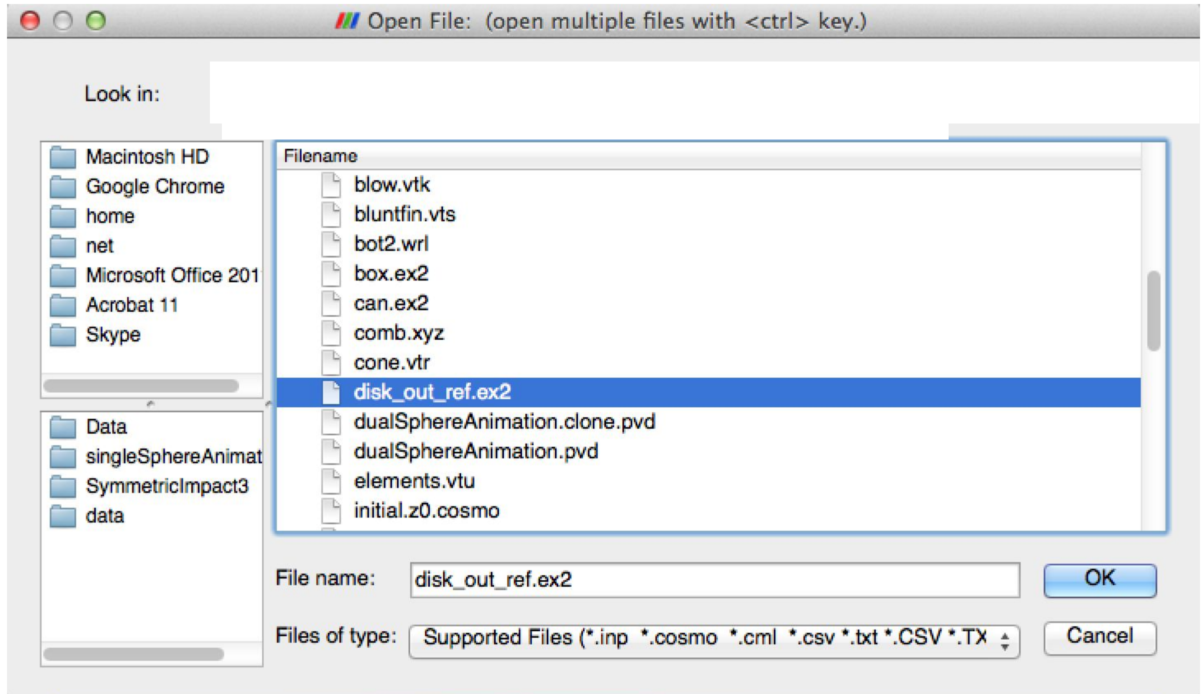
- Has a cylindrical simulation domain/grid

- Contains the data of the grid & the air



Load disk_out_ref.ex2

1. Open 

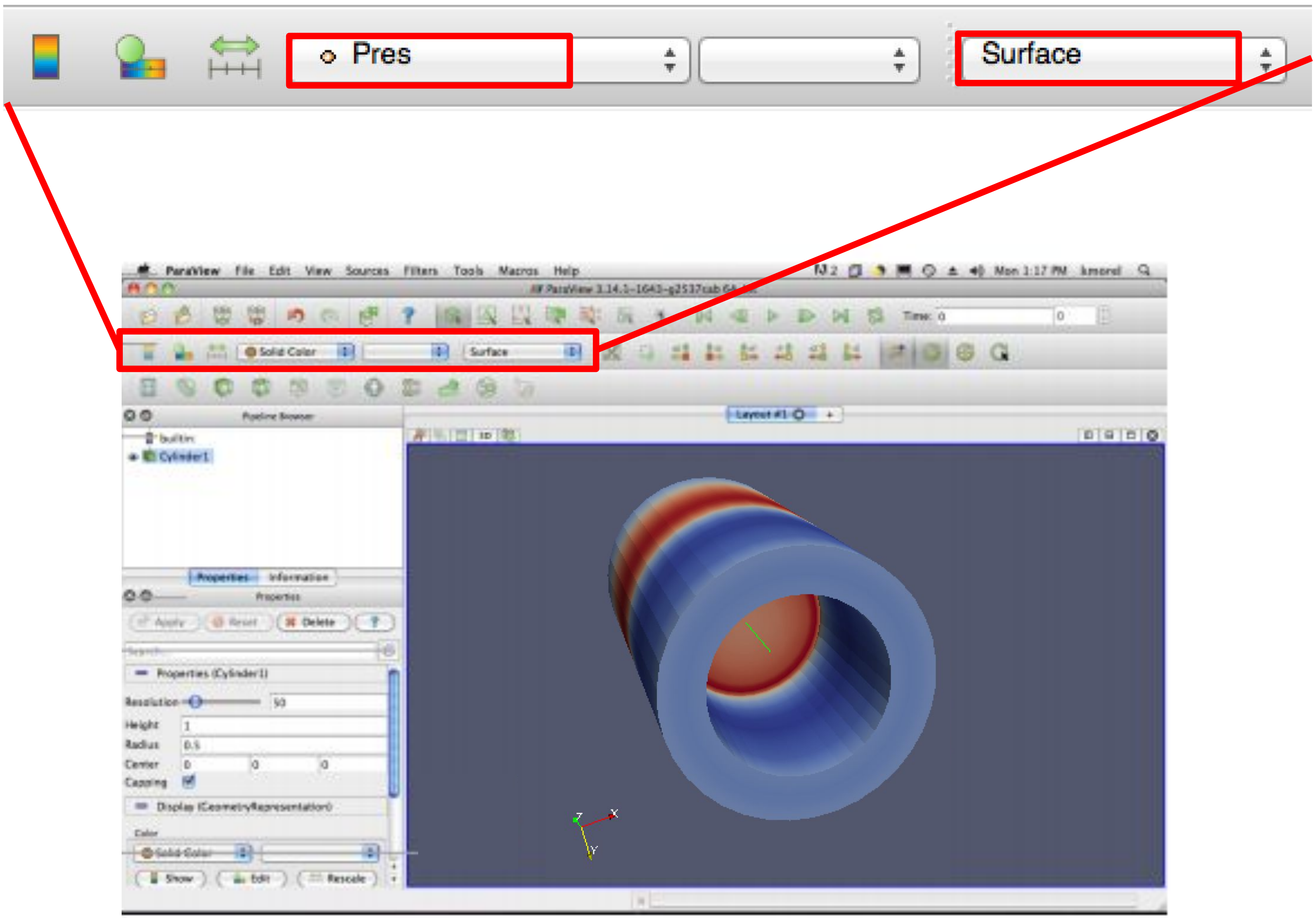


2. Select data

3. Select 

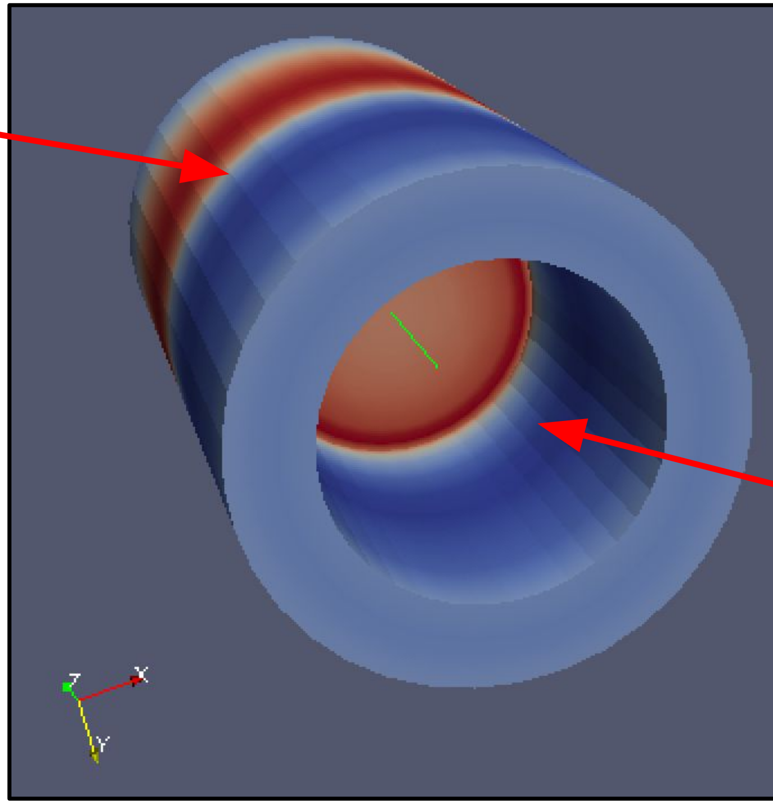
4. 

5. Choose this representation:



Use the mouse to move the data around.

Air



Disk (not shown) would be in this volume.

Air flow pressure. This is a 3D surface color map. Clearly shows changes as a function of position on the surface only.

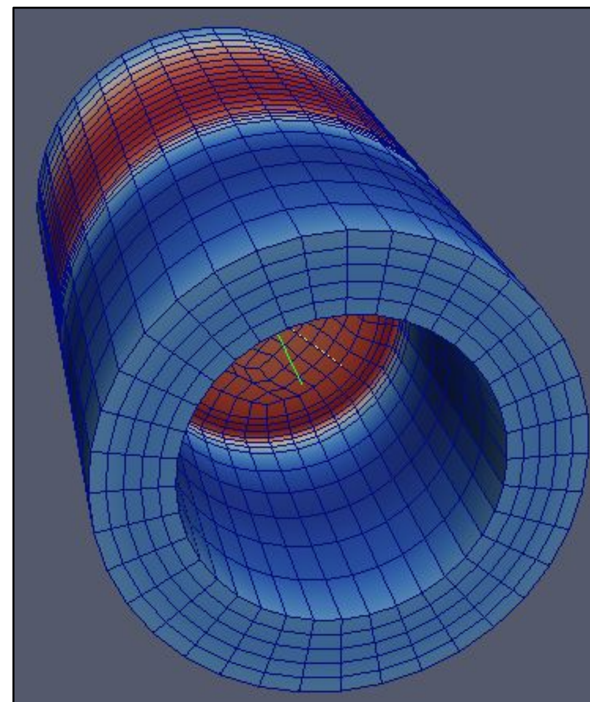
Loads ALL the data so it is available for visualization

disk_out_ref.ex2 is small

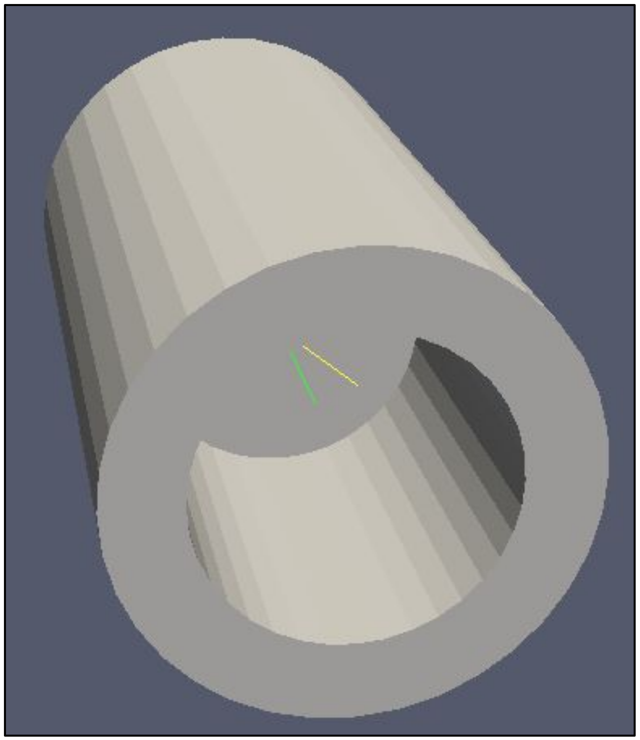
Not a good idea when using large data; the system may slow down or crash.

Explore other **representations**.

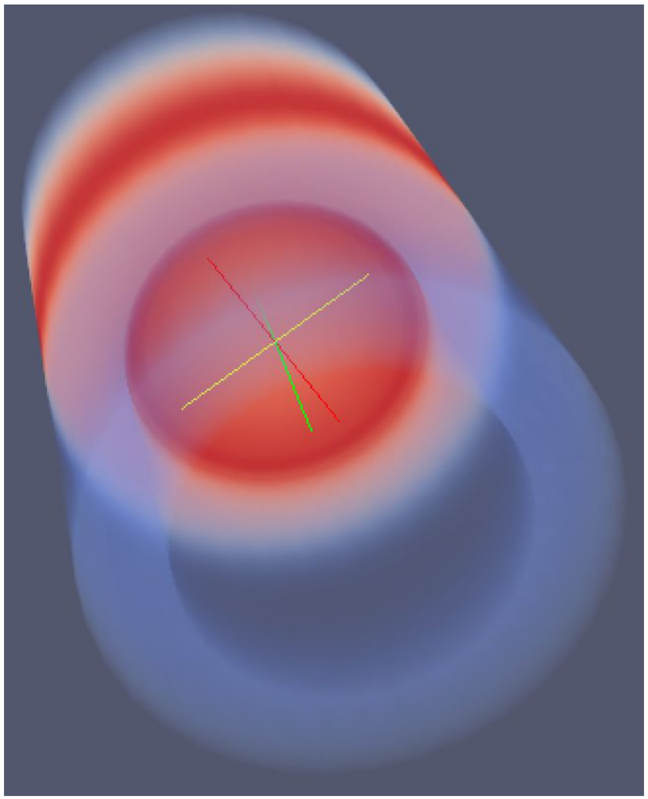
Pressure surface with edges. Shows pressure vs. position along the geometry of domain’s cells at the surface.



Solid color surface. Shows the surface of the computational domain and its geometry.

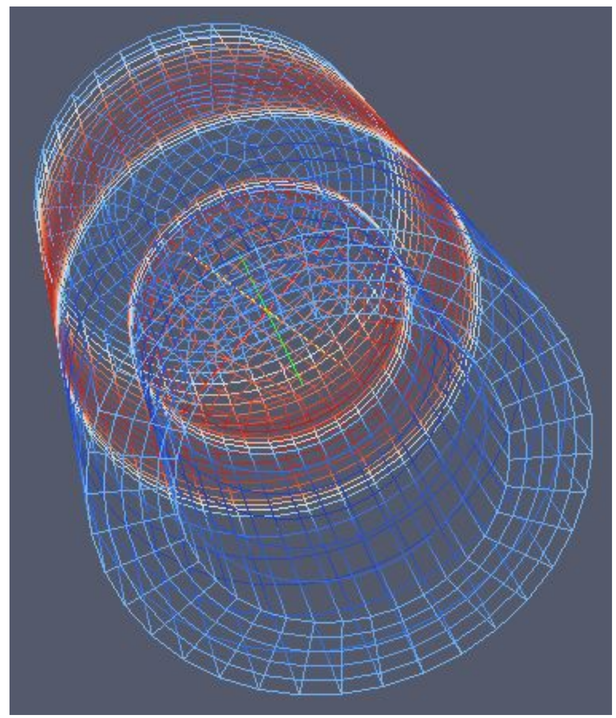


Representations show different data components within the same space. Depends in what YOU want to see.

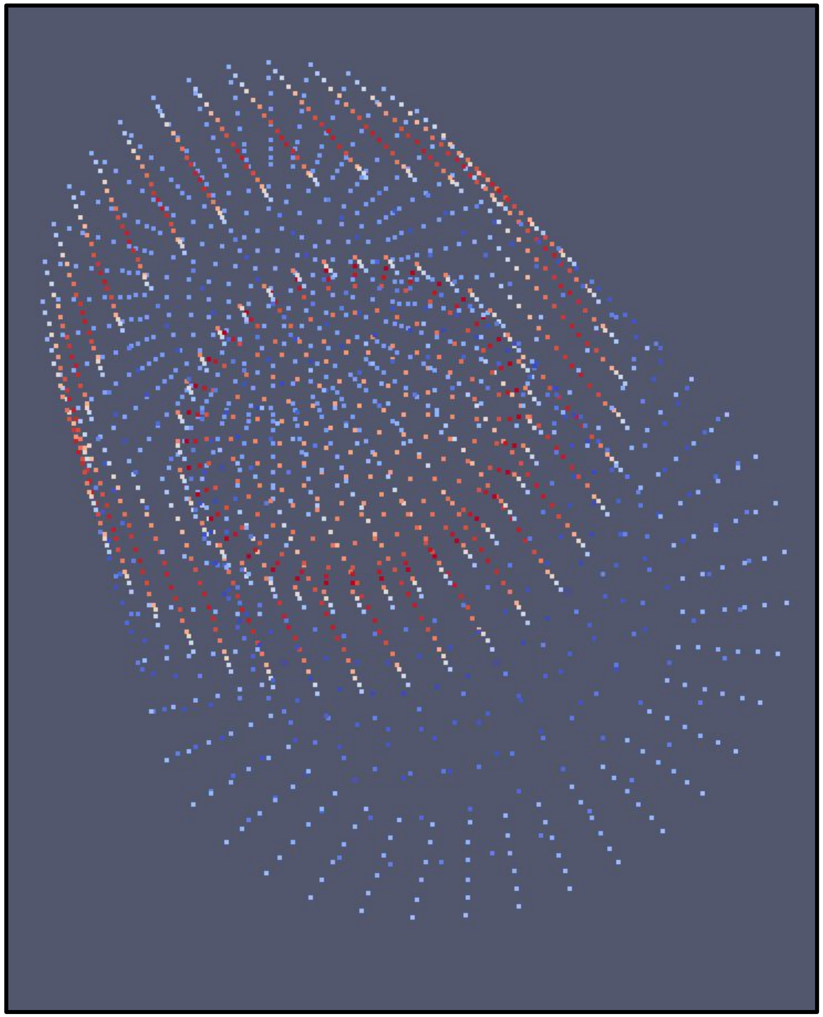


Volume rendering of pressure. Shows the field all the way through the volume.

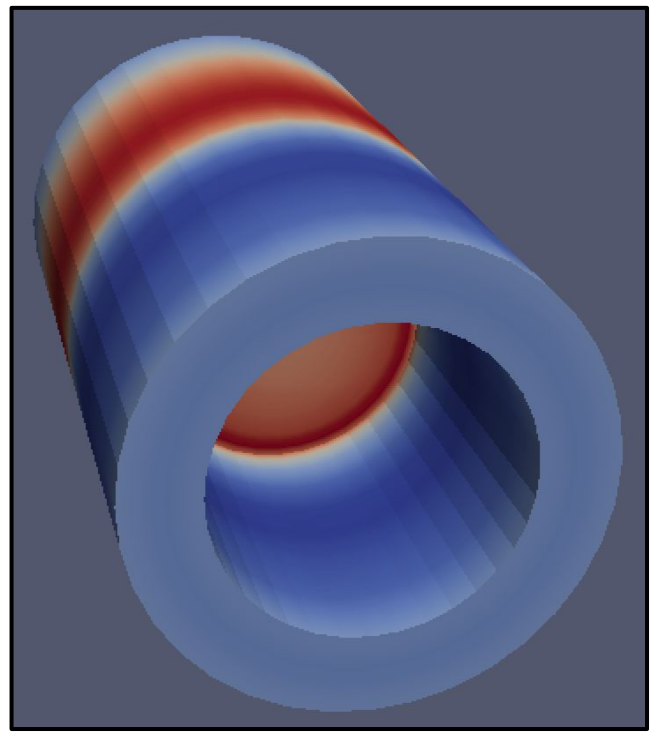
Wireframe of pressure. Shows pressure vs. position on the domain's grid.



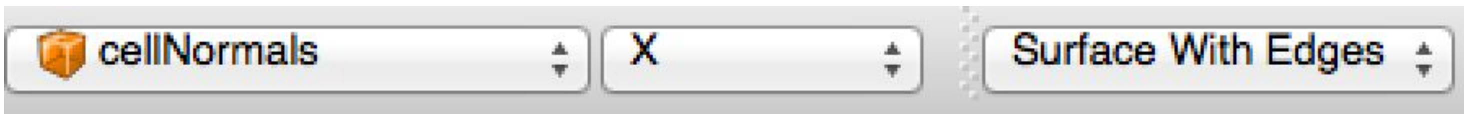
Note that the actual data consists of values of each variable (pressure, temperature, etc.) at a set of **points** located in a cylindrical structure.



A surface/volume map (below) is the result of **interpolating** the data.

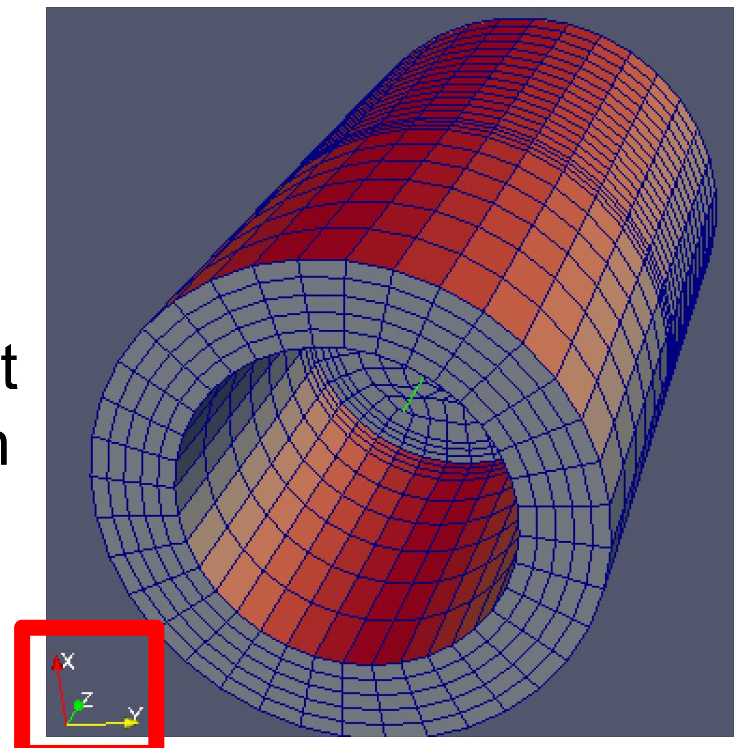


Explore other variables.

E.g. 

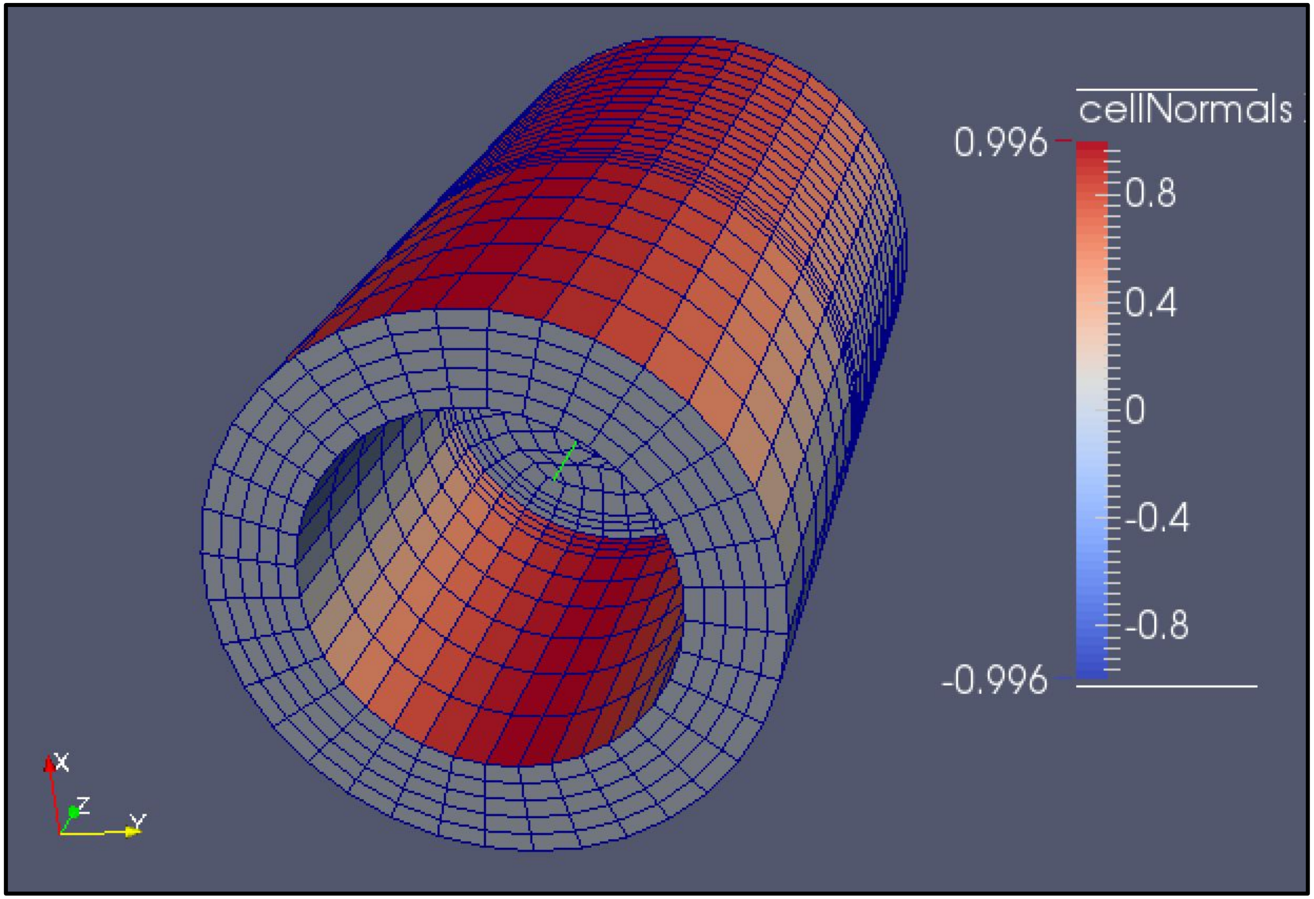
Magnitude of unitary vectors parallel to the x axis.

Useful to compute air flows in/out the domain or pressure acting on the air, along the x axis.
May be used in the design of the disk.



Let’s make our previous example clearer by adding a color legend.

Click: 



Note you can move the bar using the mouse.

Modify the legend using 

Color map editor

- Colors & their distribution

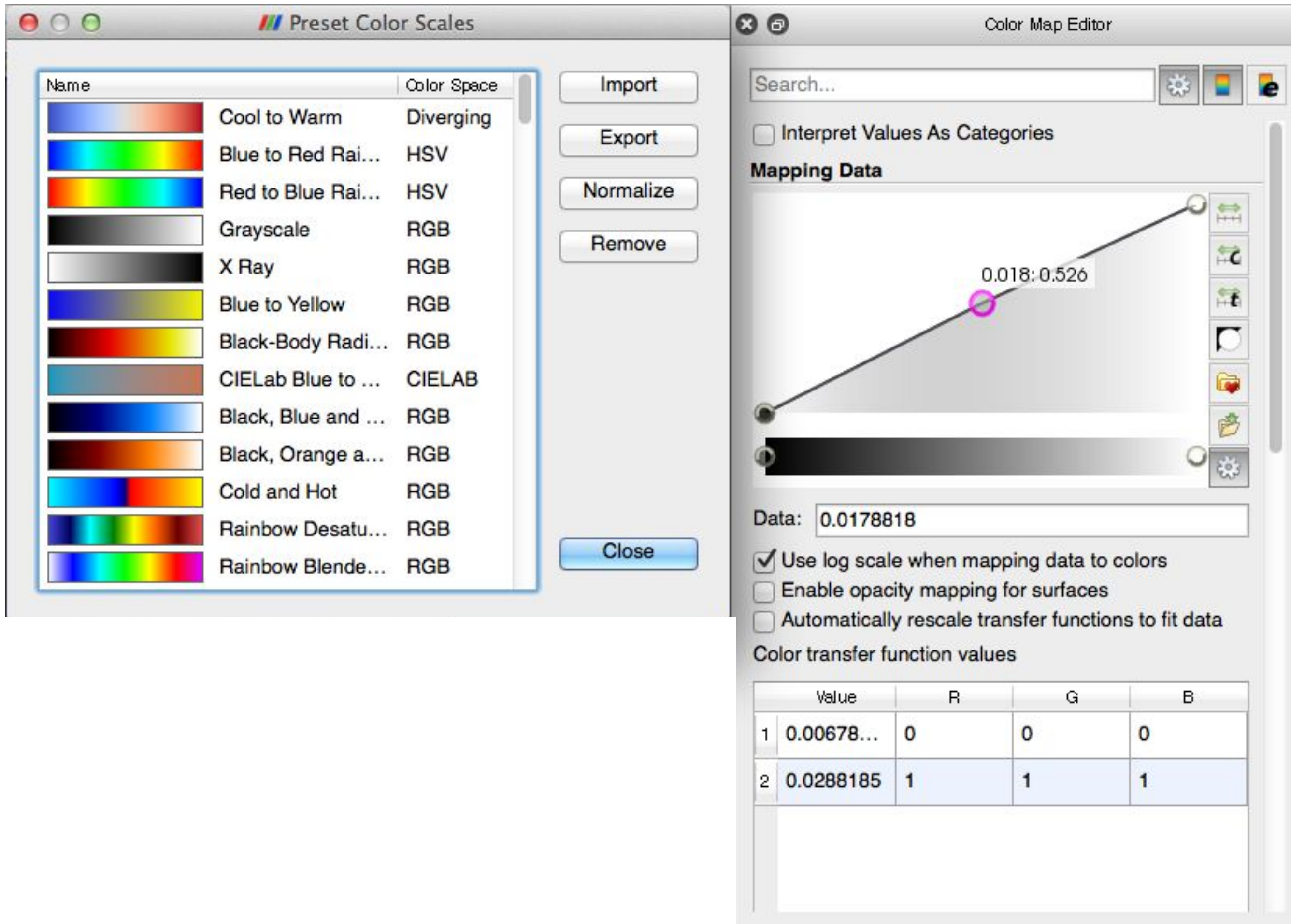
- Labels

- Linear/log scales

- Scale limits

- Functions

…



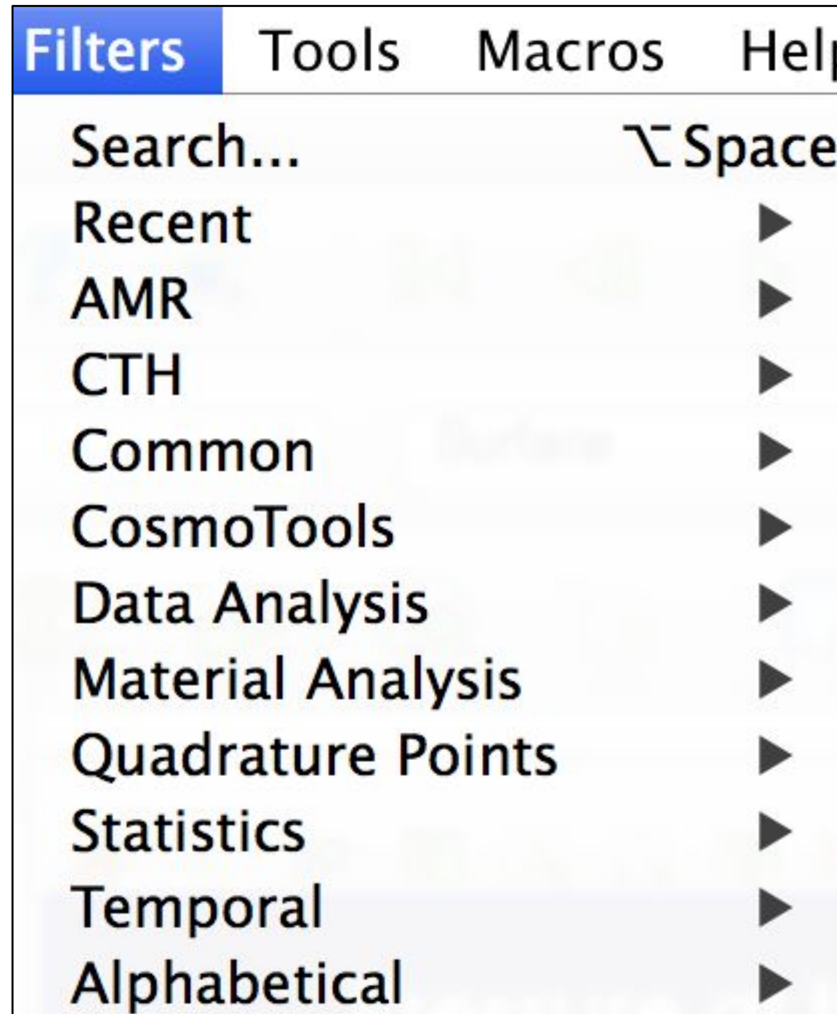
More details at: http://www.kitware.com/blog/home/post/573

Visualization algorithms or filters

To show exactly what you want, and to perform analyses

- Isosurfaces
- Cutting planes
- Streamlines
- Glyphs
- Clipping
- Height maps

...



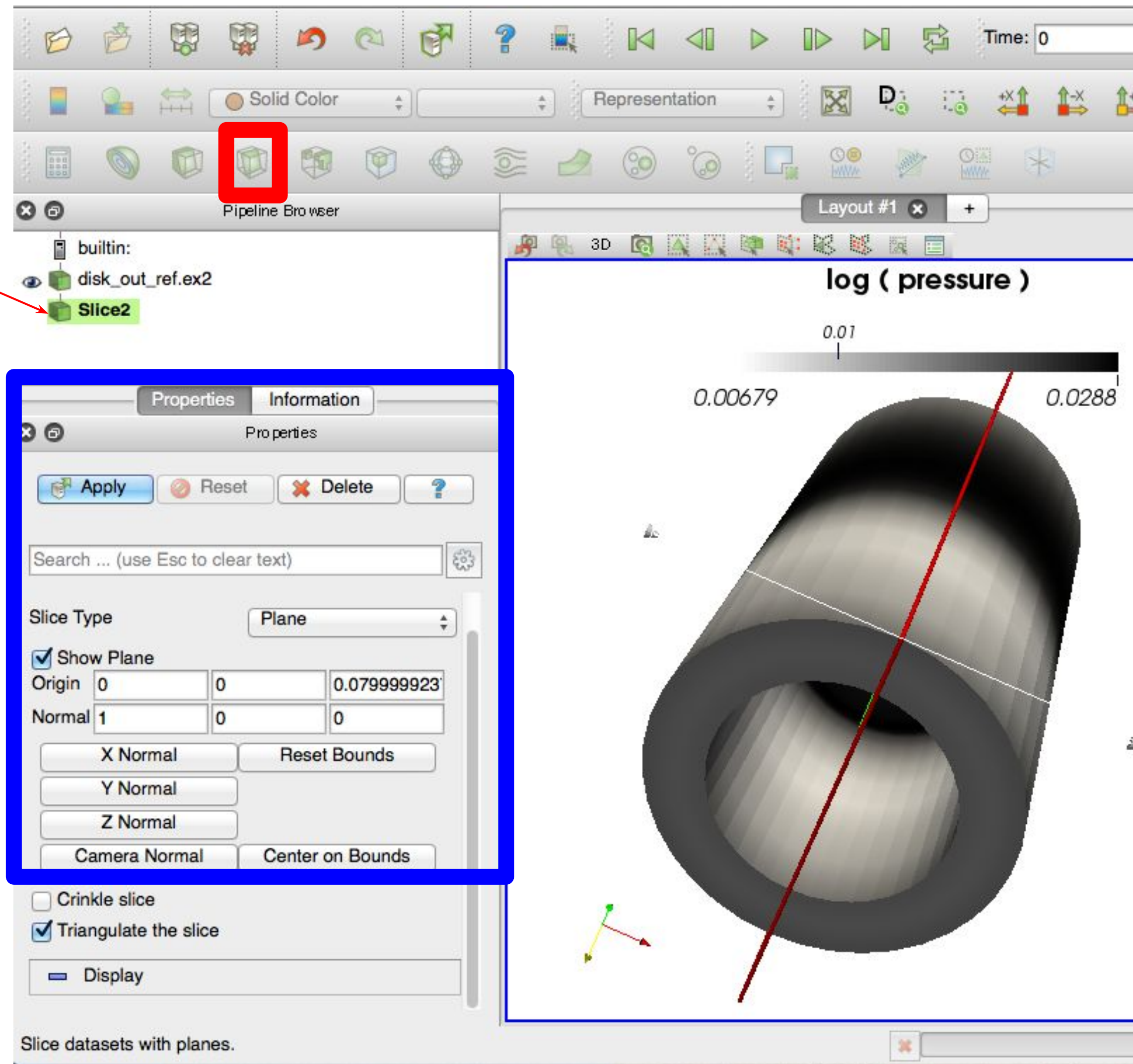
Slice

Note it adds onto the pipeline

First, set properties.

Use the mouse too.

Then 

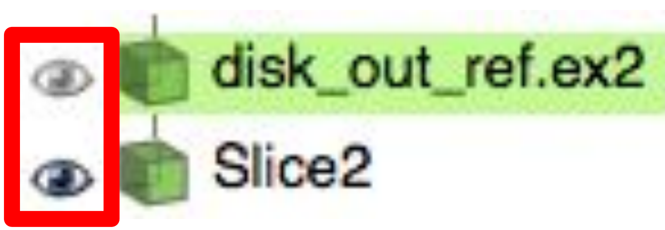


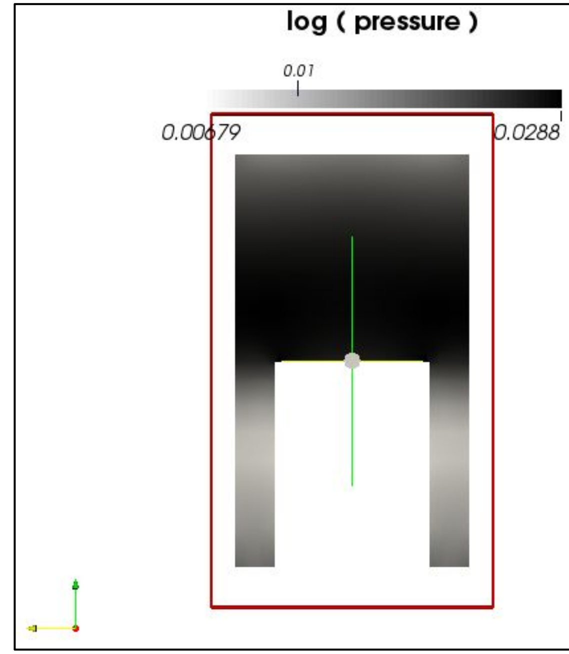
These buttons are handy now

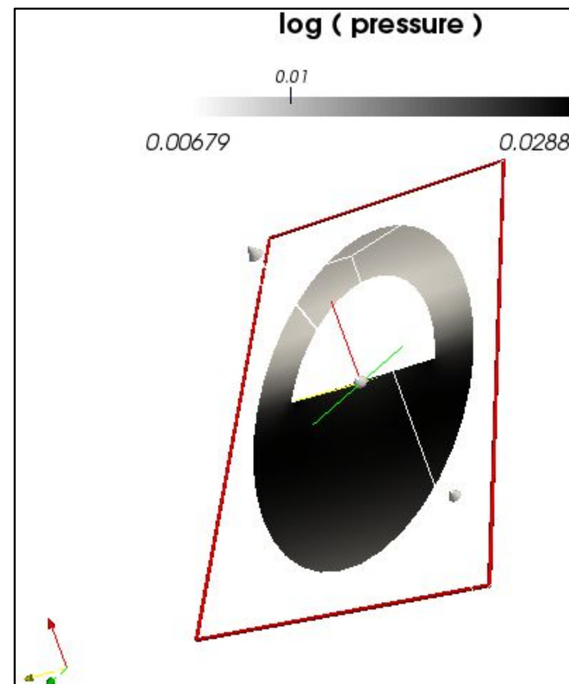
do/undo

change view

Note you can activate pipeline objects using their eyes





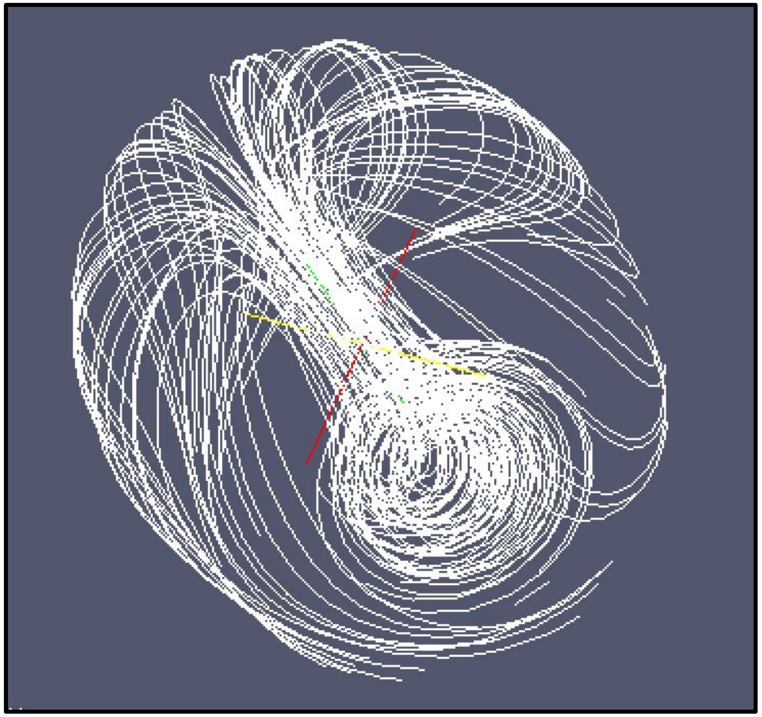


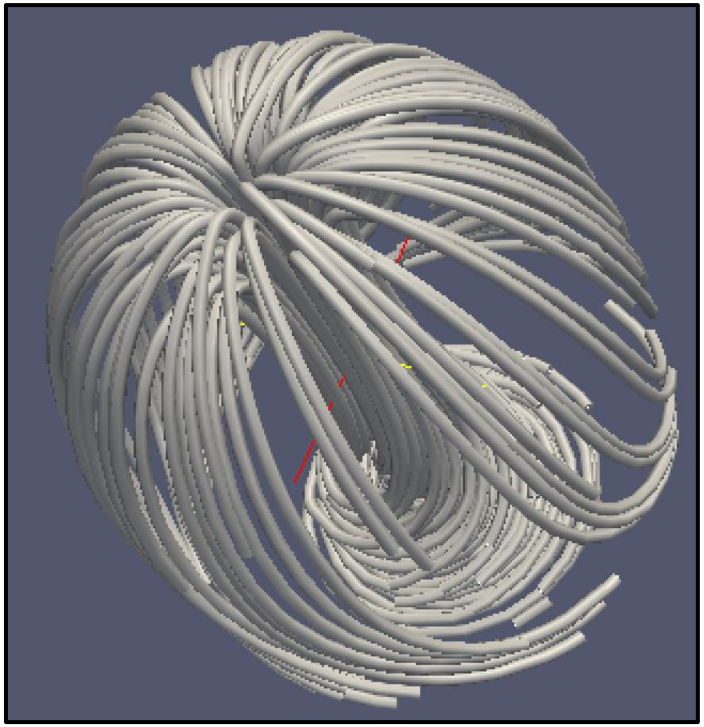
Vectors and Streamline plots

Add the streamline filter

to the disk_out_ref.ex2 file.

Apply (below)



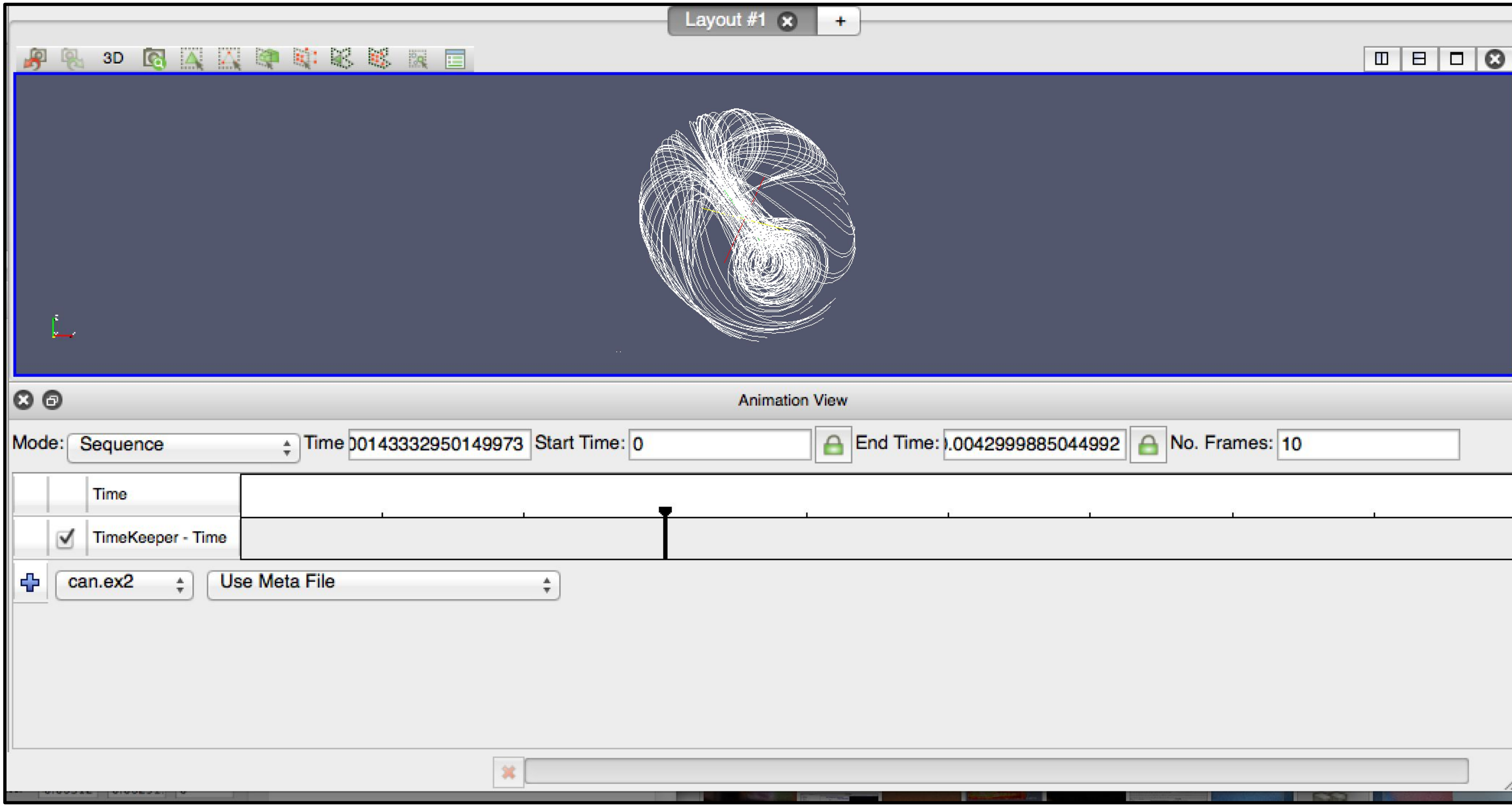


Then add a *tube filter*

Filters>alphabetic>tube

Animations -- more controls

View > Animation view



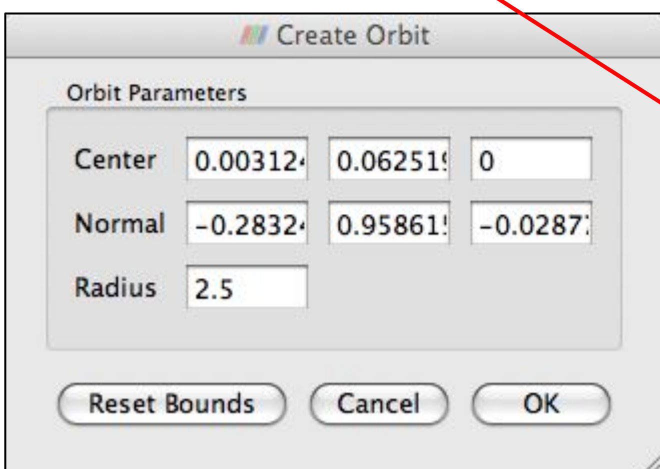
Animations: Camera

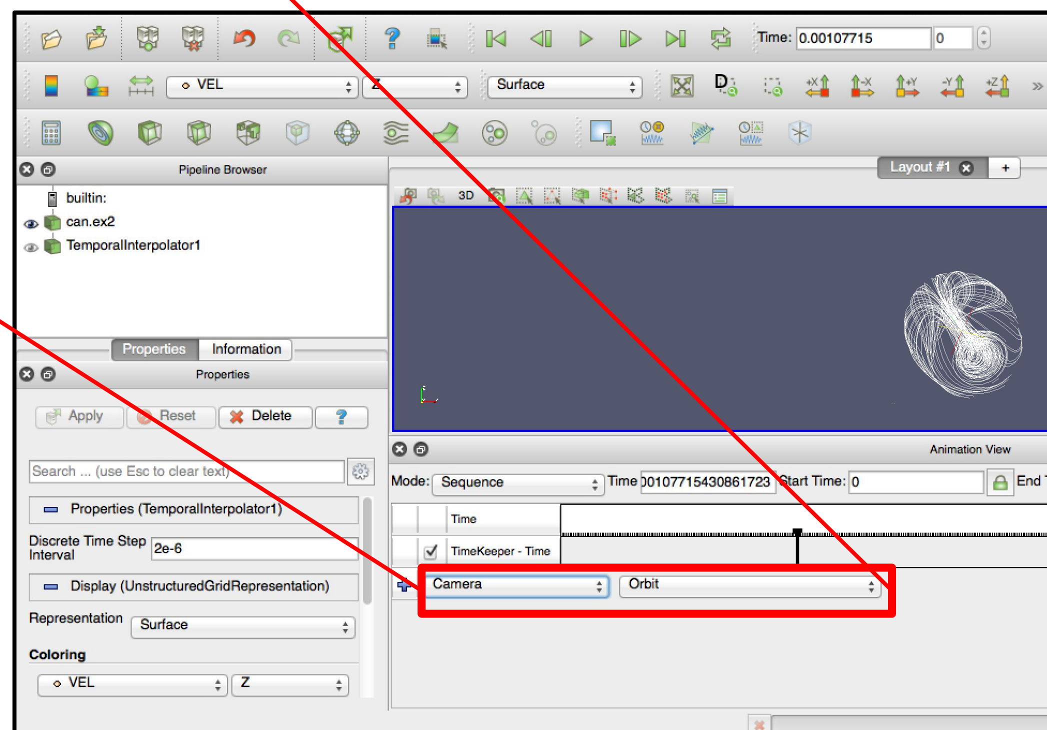
1. Place the camera where you want the orbit to start.

2. In the property selection widgets, select Camera in the first box and Orbit in the second box.

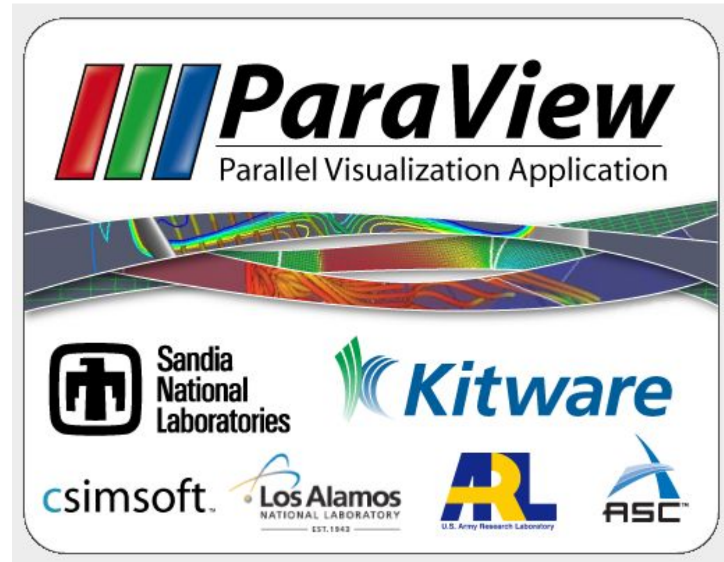
3. 

4. Adjust orbit:





About Paraview

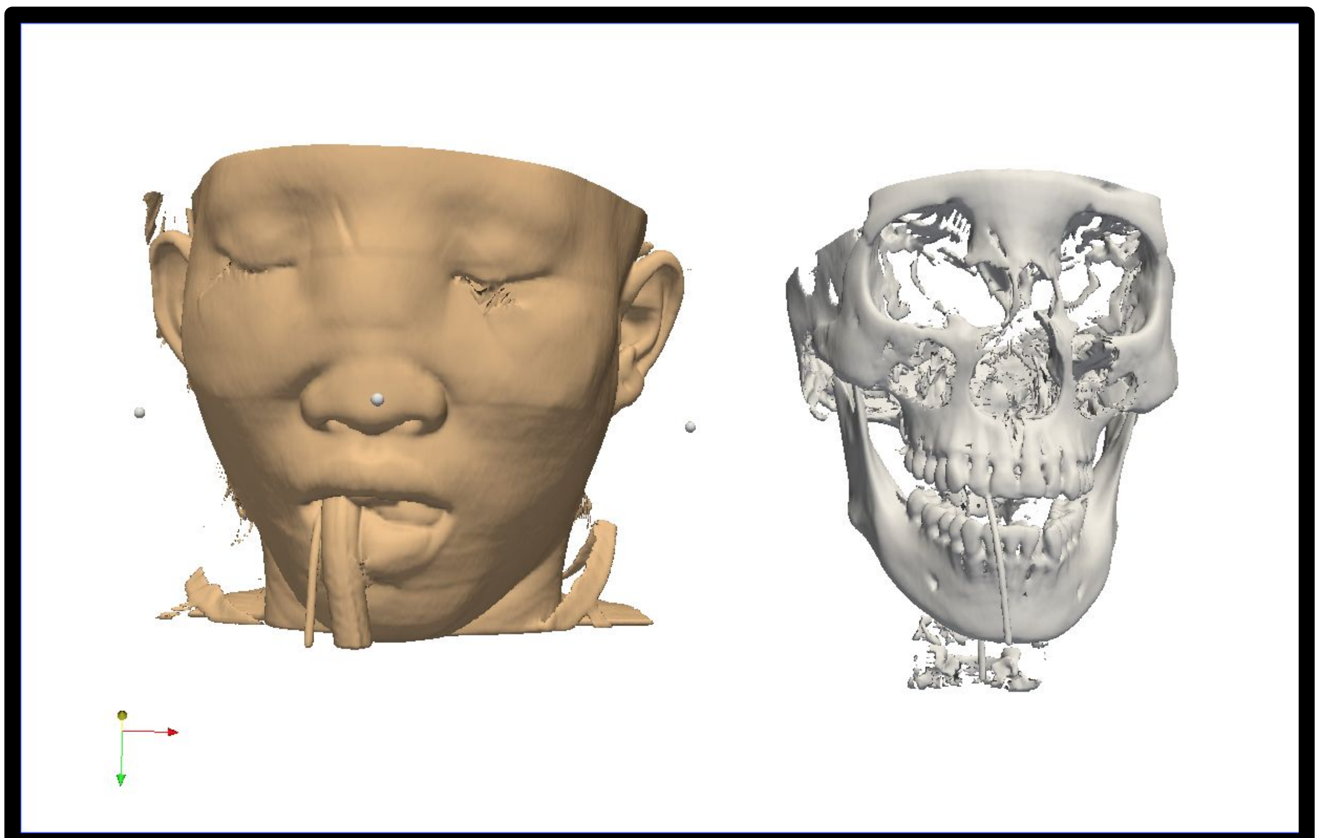


- Interactive visualization and analysis platform for 2D and 3D scientific & engineering datasets
- Funded by Department of Energy
- Open Source
- Scalable: from single-processor workstations to multi-processor distributed memory supercomputers.
- Multi-platform: Windows, OS X, Linux
- Extendible modular architecture

Special features

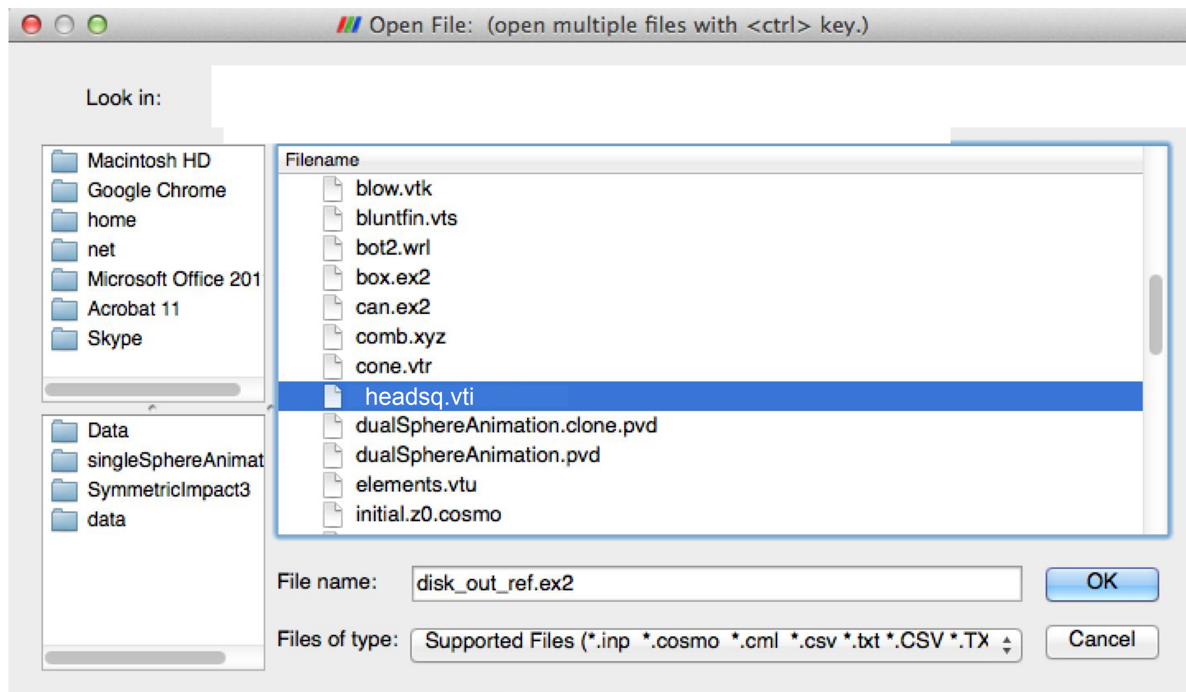
- Supports derived variables
- Scriptable via python
- Animations
- Run in parallel/ distributed mode for big data

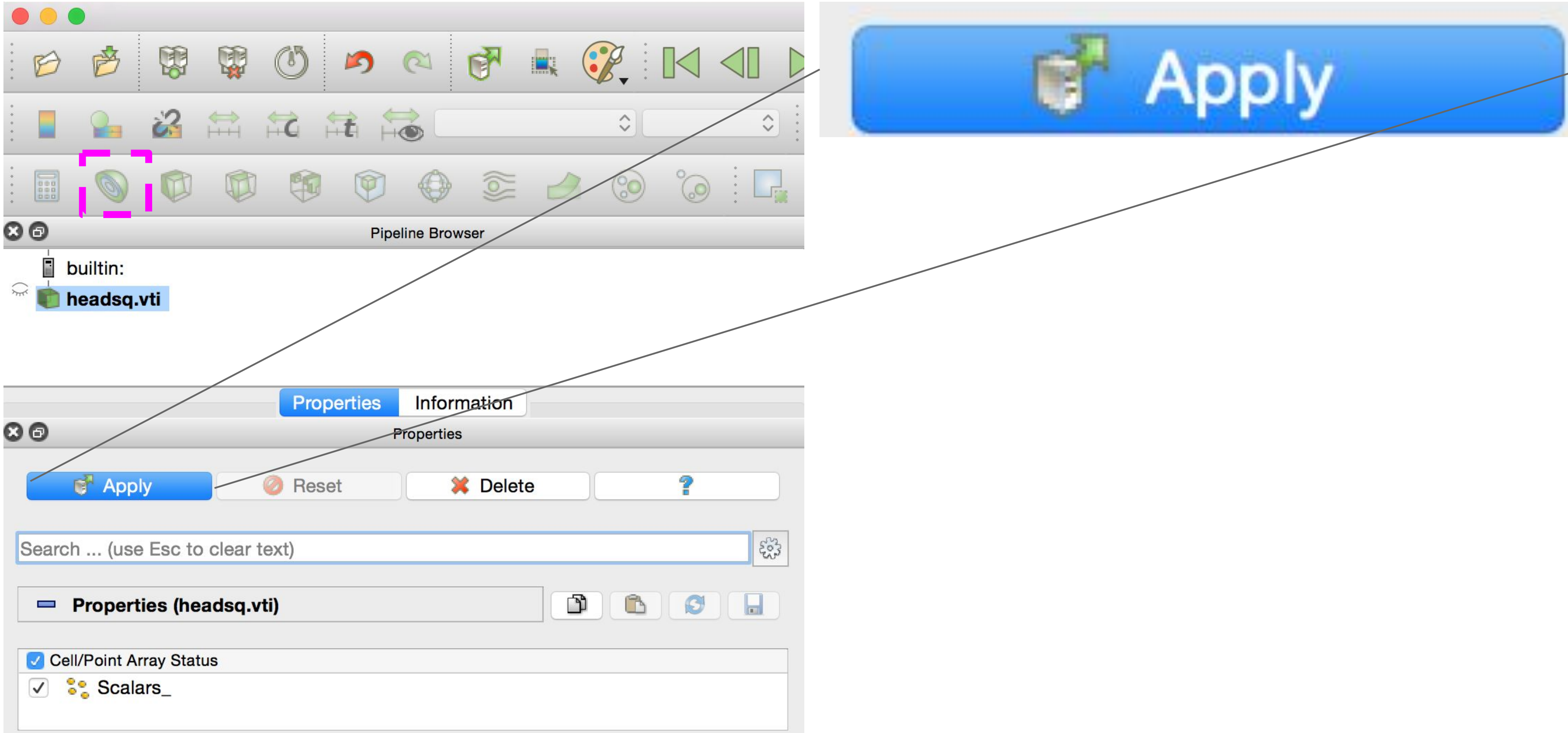
Now, let's explore a medical CT scan, *headsq.vti*. Let's produce the plot below (note all objects are in the same window). Finally, be creative and produce an interesting plot using ParaViews filters of your choice. Next, ask yourself, can you do this better?



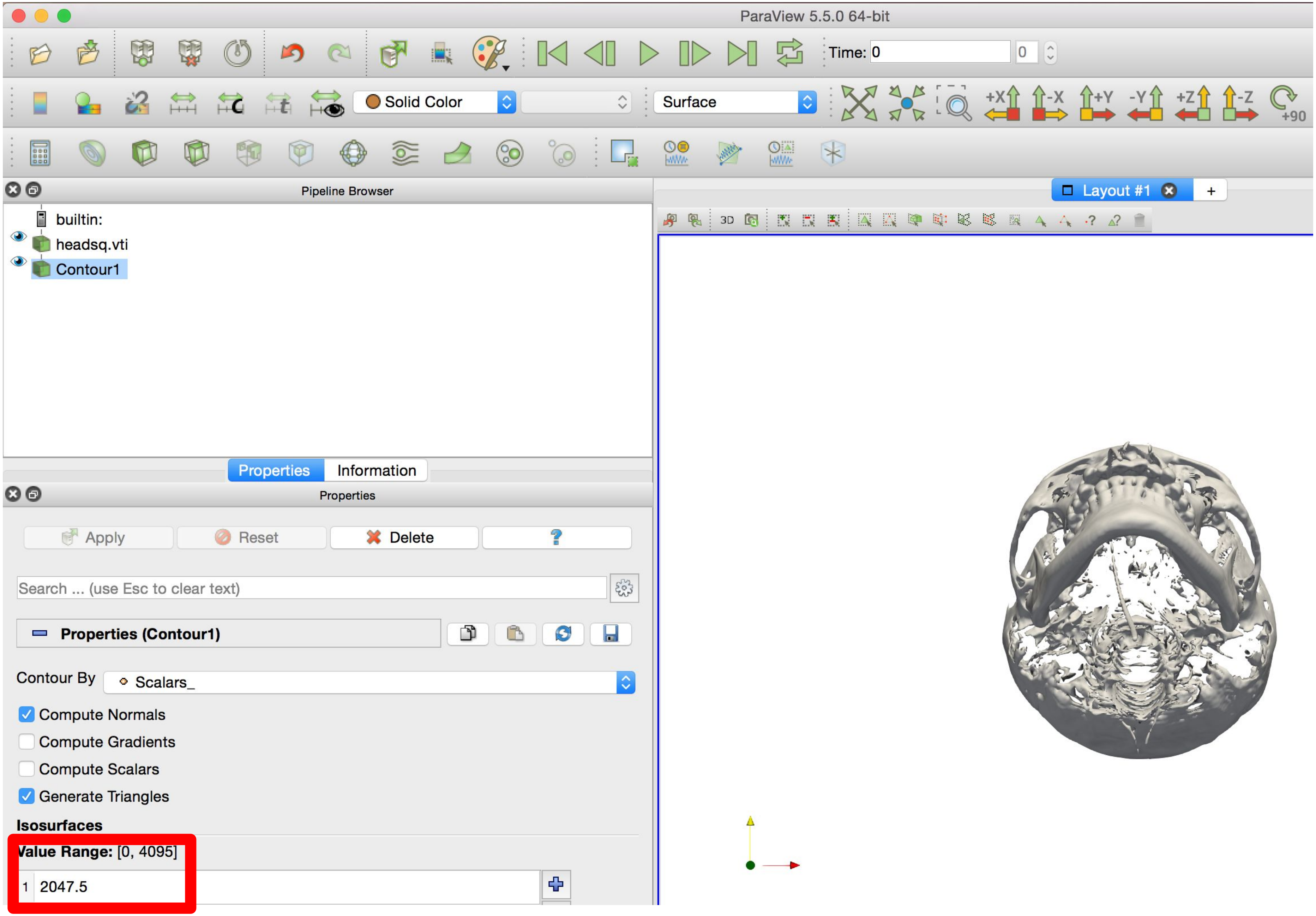
Load headsq.vti

1. Start a new session  (top left)
2. Open  (top left)
3. Select data

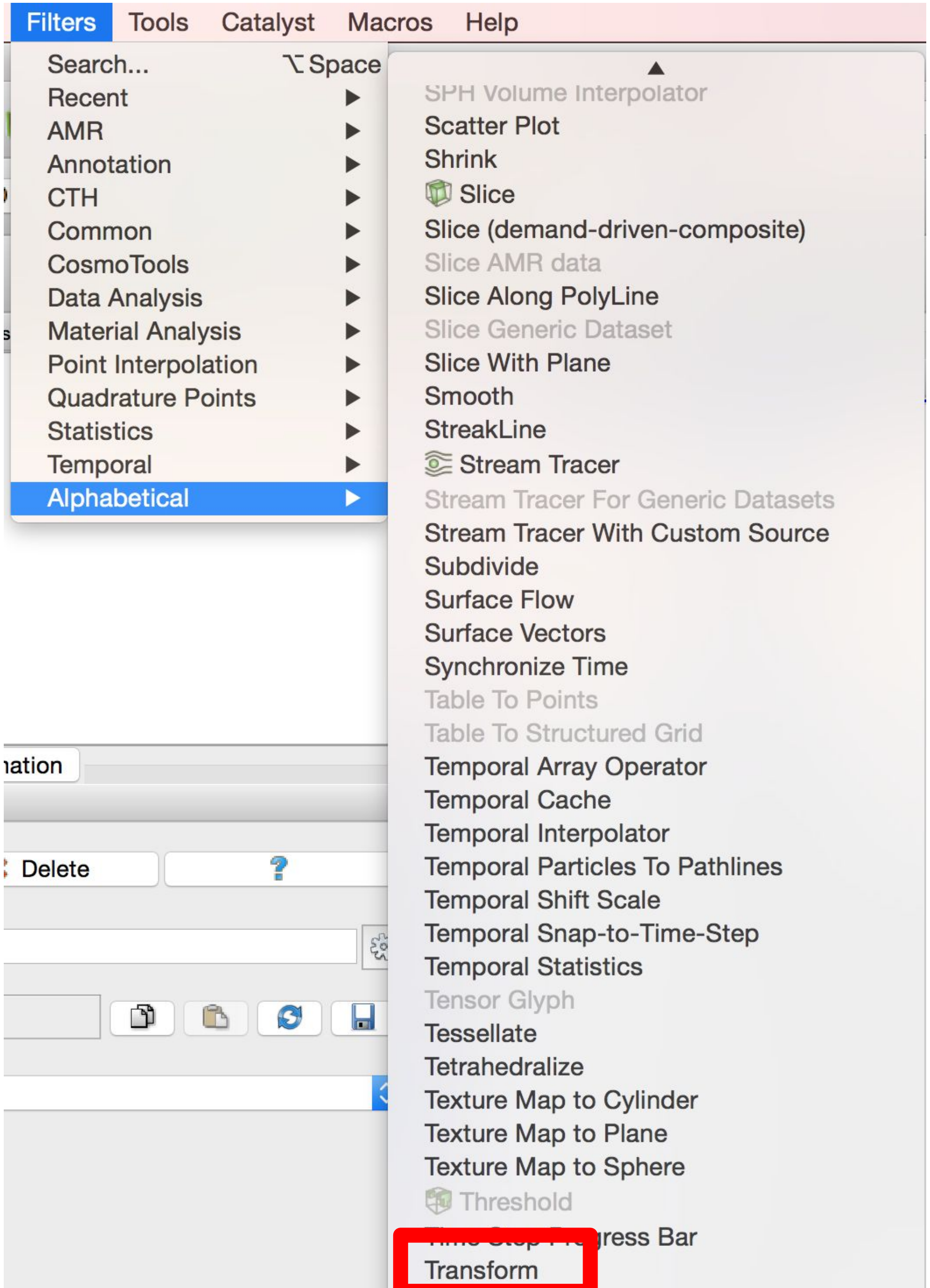


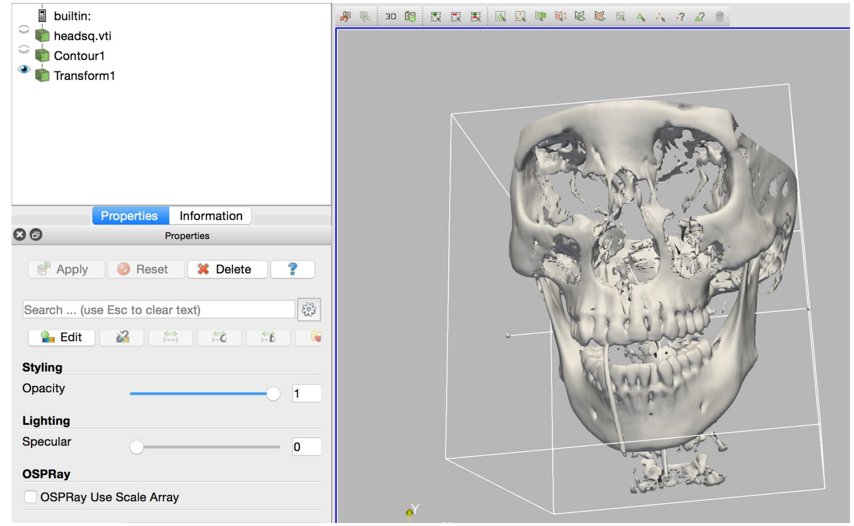


Click the contour filter



Scan frequencies, change and apply

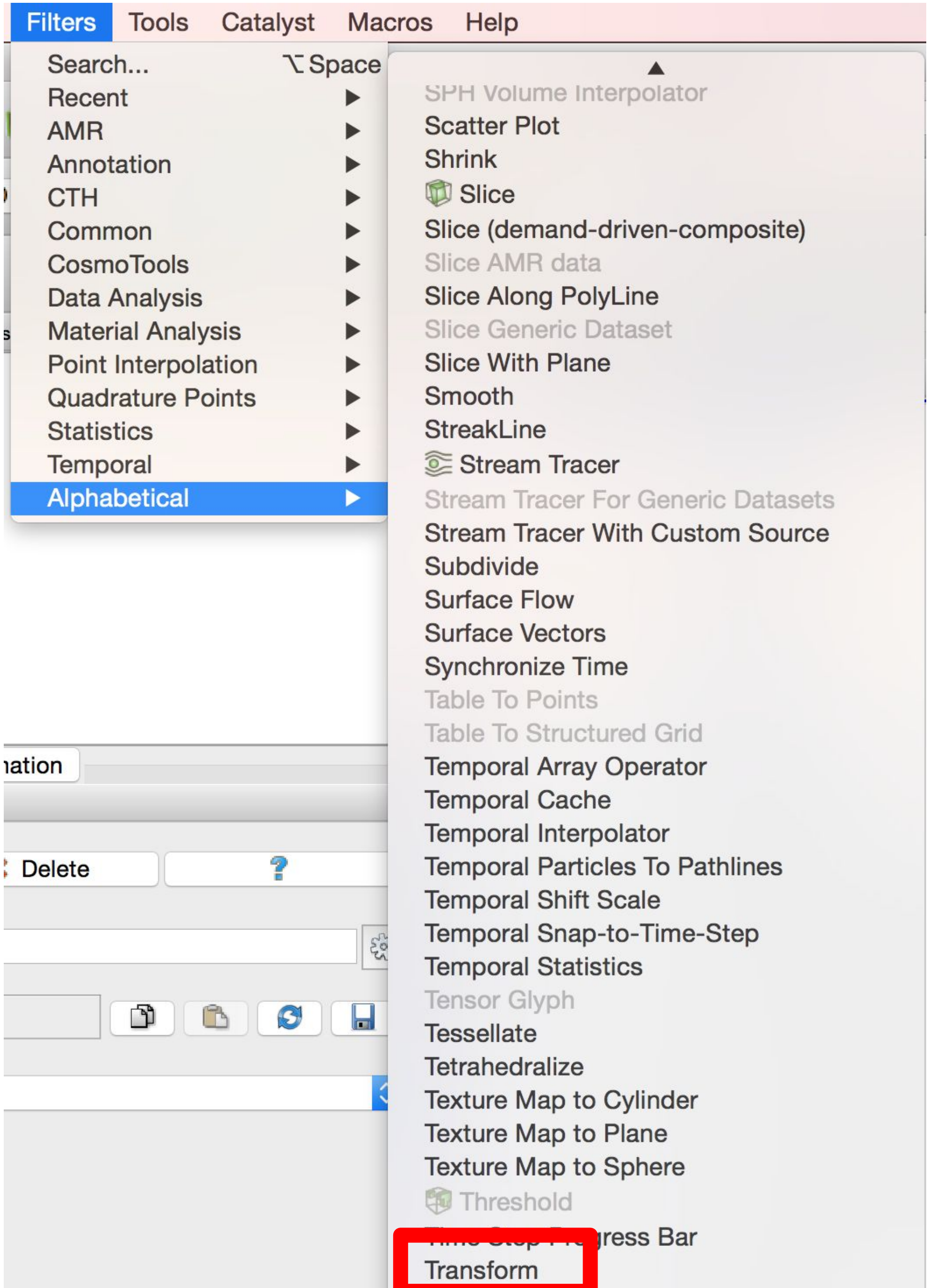


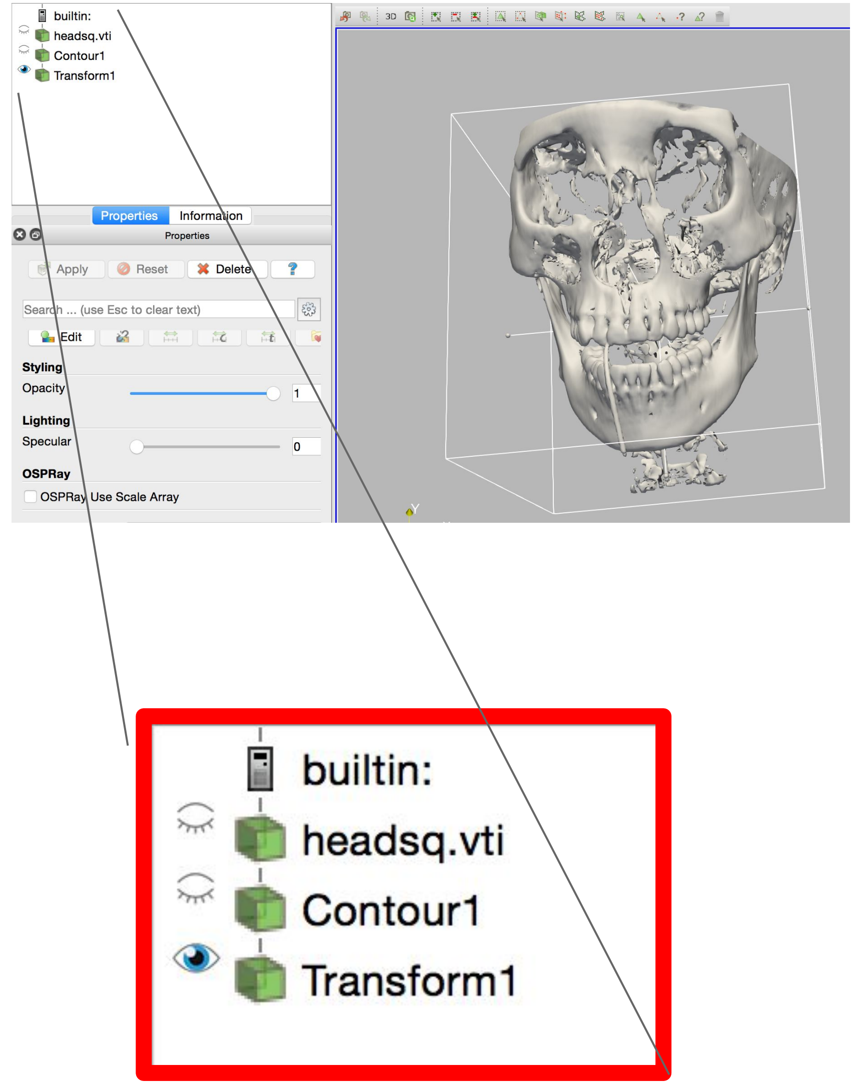


Use the box containing the head to move data around

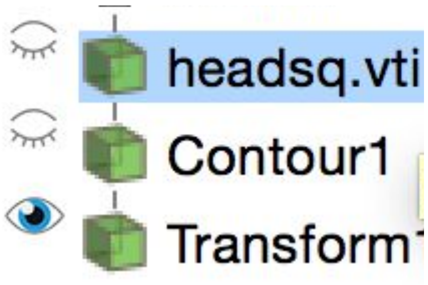
Combine w/ shift key to move but not rotate

Note the box and axes respond to the mouse pointer



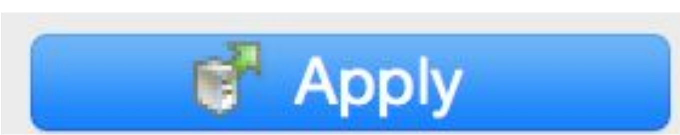


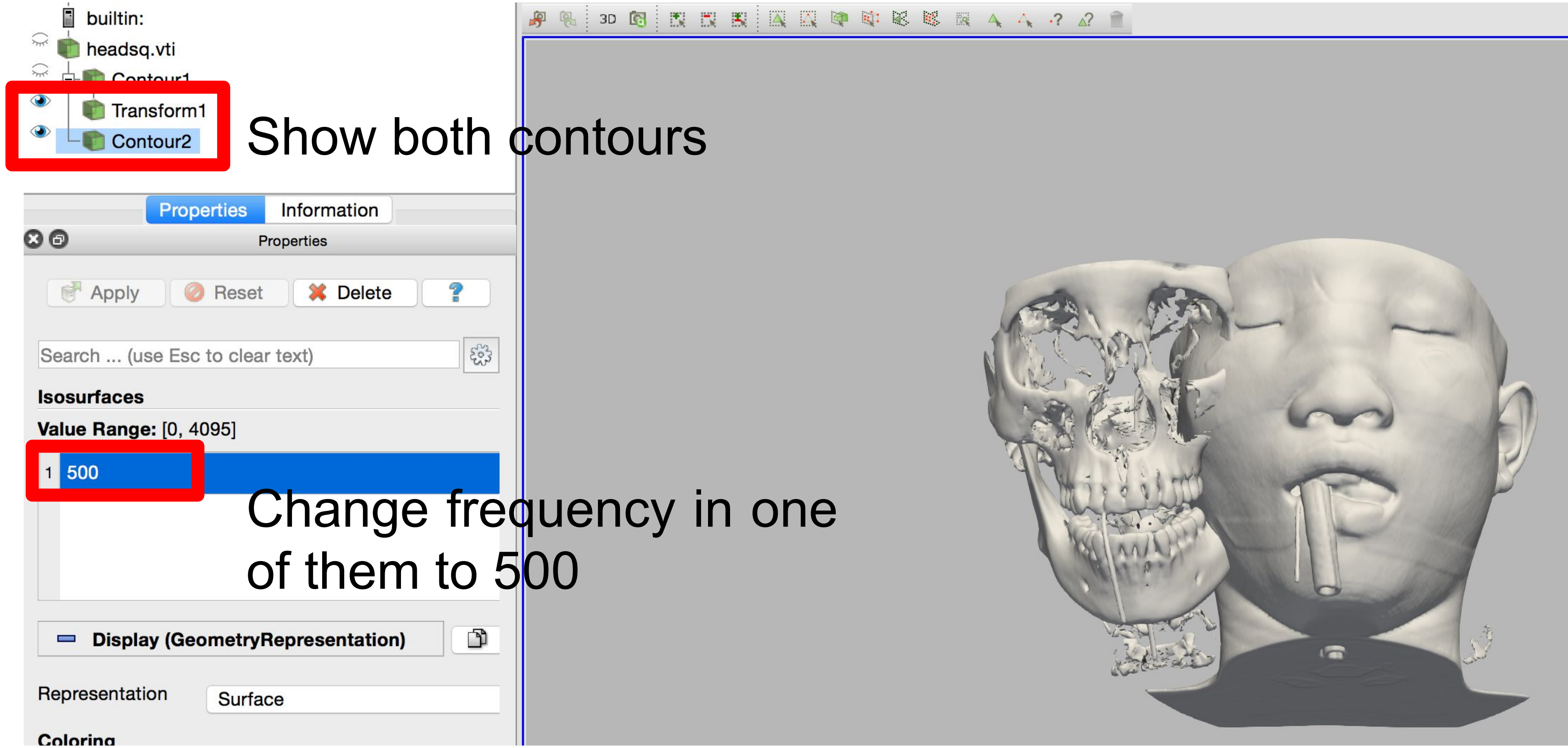
Notice the pipeline



Click on the data

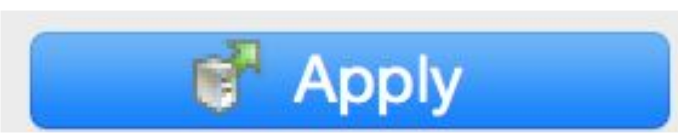
Apply another contour





Show both contours

Change frequency in one of them to 500



PARAVIEW -- Python scripting

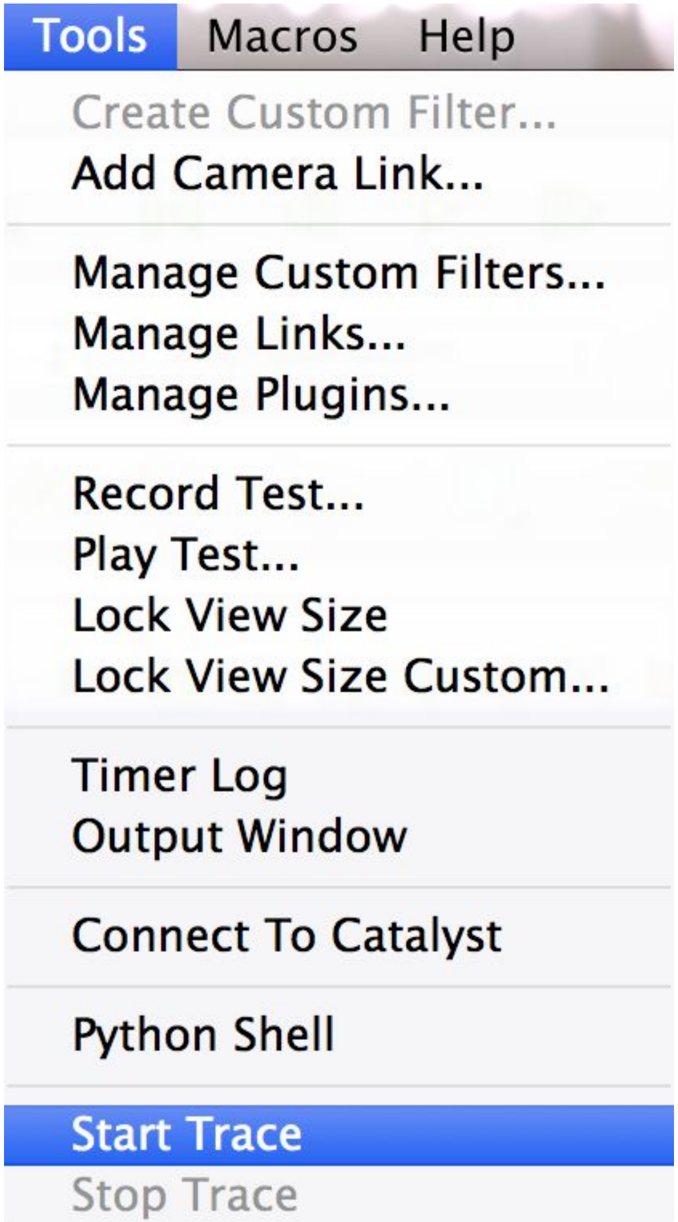
- http://www.paraview.org/Wiki/ParaView/Python_Scripting

- Scripts can **automate** visualization algorithms by performing actions as a user would at the GUI

- Scripts run inside pipeline objects, performing parallel visualization algorithms

- Useful command line environment to create analysis programs

Creating a Python Script Trace



1. Reset 

2. Tools > Start Trace

3. Build a simple pipeline in the main ParaView GUI.

4. Stop Trace

A python script window will open with commands that replicate your actions.

Saving scripts

File > Save as > NAME.py

or

File > Save state

or

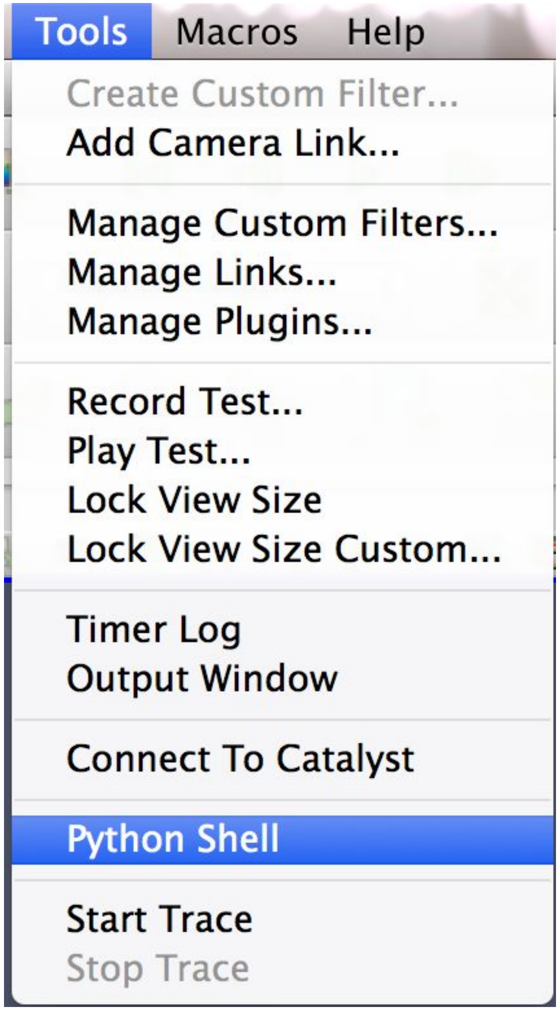
File > Save as Macro

it should appear in the GUI's *Macros* menu then

untitled.py – Script Editor

```
try: paraview.simple
except: from paraview.simple import *
paraview.simple._DisableFirstRenderCameraReset()

can_ex2 = ExodusIIReader( FileName=['/Users/martinhuarte-espinosa/Downloads/ParaViewData-v4.1/Data/c

AnimationScene4 = GetAnimationScene()
AnimationScene4.EndTime = 0.004299988504499197
AnimationScene4.PlayMode = 'Snap To TimeSteps'

can_ex2.FileRange = [0, 0]
can_ex2.XMLFileName = '/Users/martinhuarte-espinosa/Downloads/ParaViewData-v4.1/Data/artifact.dta'
can_ex2.FilePrefix = '/Users/martinhuarte-espinosa/Downloads/ParaViewData-v4.1/Data/can.ex2'
can_ex2.ModeShape = 21
can_ex2.FilePattern = '%s'

can_ex2.ElementVariables = ['EQPS']
can_ex2.GlobalVariables = ['KE', 'XMOM', 'YMOM', 'ZMOM', 'NSTEPS', 'TMSTEP']
can_ex2.SideSetArrayStatus = []
can_ex2.ElementBlocks = ['Unnamed block ID: 1 Type: HEX', 'Unnamed block ID: 2 Type: HEX']
can_ex2.NodeSetArrayStatus = []
can_ex2.PointVariables = ['DISPL', 'VEL', 'ACCL']

RenderView6 = GetRenderView()
RenderView6.CenterOfRotation = [0.21706008911132812, 4.0, -5.1109471321105957]

DataRepresentation27 = Show()
DataRepresentation27.EdgeColor = [0.0, 0.0, 0.50000762951094835]
DataRepresentation27.SelectionPointFieldDataArrayName = 'ACCL'
DataRepresentation27.SelectionCellFieldDataArrayName = 'EQPS'
DataRepresentation27.ScalarOpacityUnitDistance = 1.3901072164734267
DataRepresentation27.ExtractedBlockIndex = 2
DataRepresentation27.ScaleFactor = 1.9778103828430176

RenderView6.CameraViewUp = [0.0, 0.0, 1.0]
RenderView6.CameraPosition = [0.21706008911132812, -47.740573746856327, -5.1109471321105957]
RenderView6.CameraClippingRange = [38.046358729469091, 65.778821972648018]
RenderView6.CameraFocalPoint = [0.21706008911132812, 4.0, -5.1109471321105957]
RenderView6.CameraParallelScale = 13.391445890217907
```

Python shell



Useful command line environment for analysis.

- Start a new session

- Start a Python Shell

- Line by line, enter:

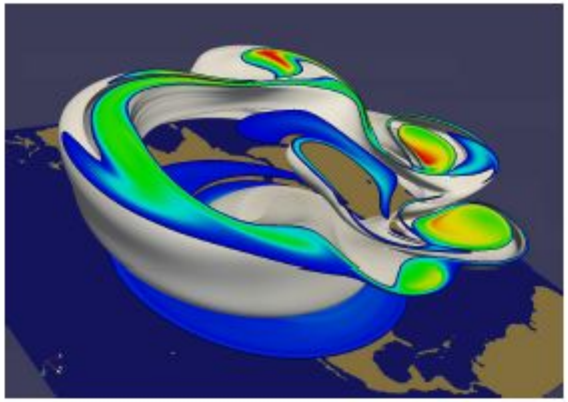
```
reader=OpenDataFile('hpathi/disk_out_ref.ex2')
 Show()
 Render()
```

Get point data:

```
pd = reader.PointData
print pd.keys()
print pd['Pres'].GetNumberOfComponents()
print pd['Pres'].GetRange()
print pd['V'].GetNumberOfComponents()
```

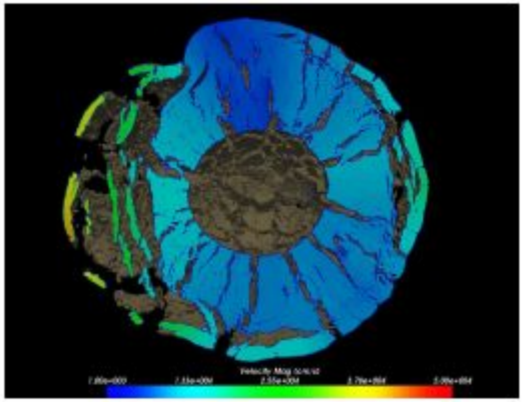
Parallel rendering & remote vis

To set it up see:
-http://www.paraview.org/Wiki/Reverse_connection_and_port_forwarding, option 1.
REMOTEHOST=cusco.hpcc.uh.edu
/REMOTE/PATH/TO/pvserver=/share/apps/paraview-4.3.1/bin/pvserver



SEAM Climate Modeling simulation with 1 billion cells modeling the breakdown of the polar vortex, a circumpolar jet that traps polar air at high latitudes.

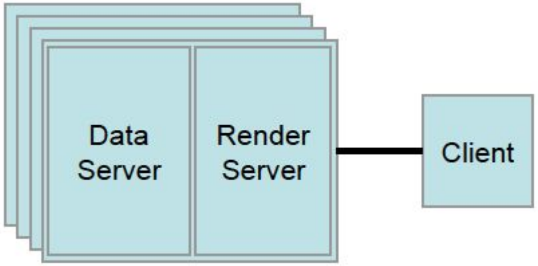
- Paraview is used frequently National Laboratories and other institutions for visualizing data from large-scale simulations run on the world's largest supercomputer (examples, left).



CTH shock physics simulation with over 1 billion cells of a 10 megaton explosion detonated at the center of the Golevka asteroid.

- You can use it in parallel too.

- Architecture:



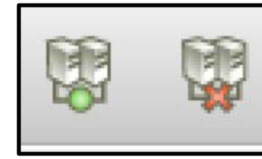
Parallel rendering & remote vis

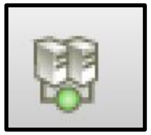
- ParaView must be **compiled** on a parallel machine. See

http://www.paraview.org/Wiki/Setting_up_a_ParaView_Server#Compiling

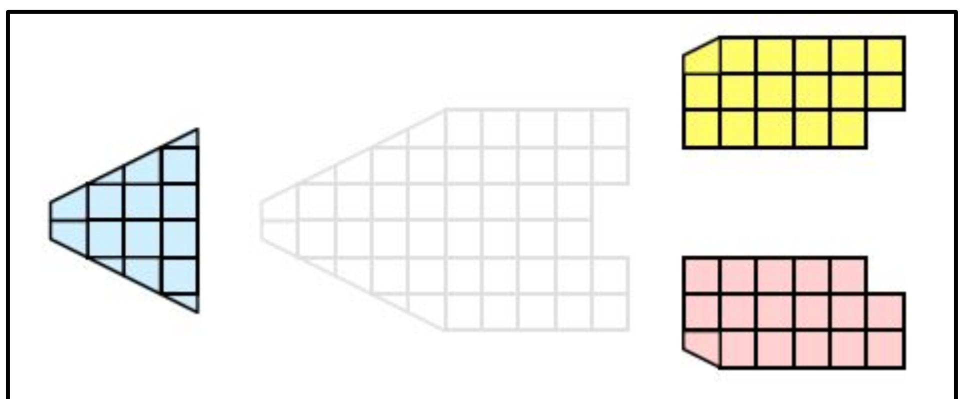
- Its **libraries** have to be in place:
    *CMake cross-platform build setup tool (www.cmake.org)
    *MPI
    *OpenGL (or use Mesa 3D www.mesa3d.org if otherwise unavailable)
    *Qt 4.6 (optional)
    *Python (optional)

Parallel rendering & remote vis

- Connect to servers and disconnect from servers 

-  (would) present you with a dialog box containing a list of known servers you may connect to. This list of servers can be both site- and user-specific.

- Data will automatically be partitioned and distributed amongst processors (see paraview's tutorial documentation for details)



Official documentation for details

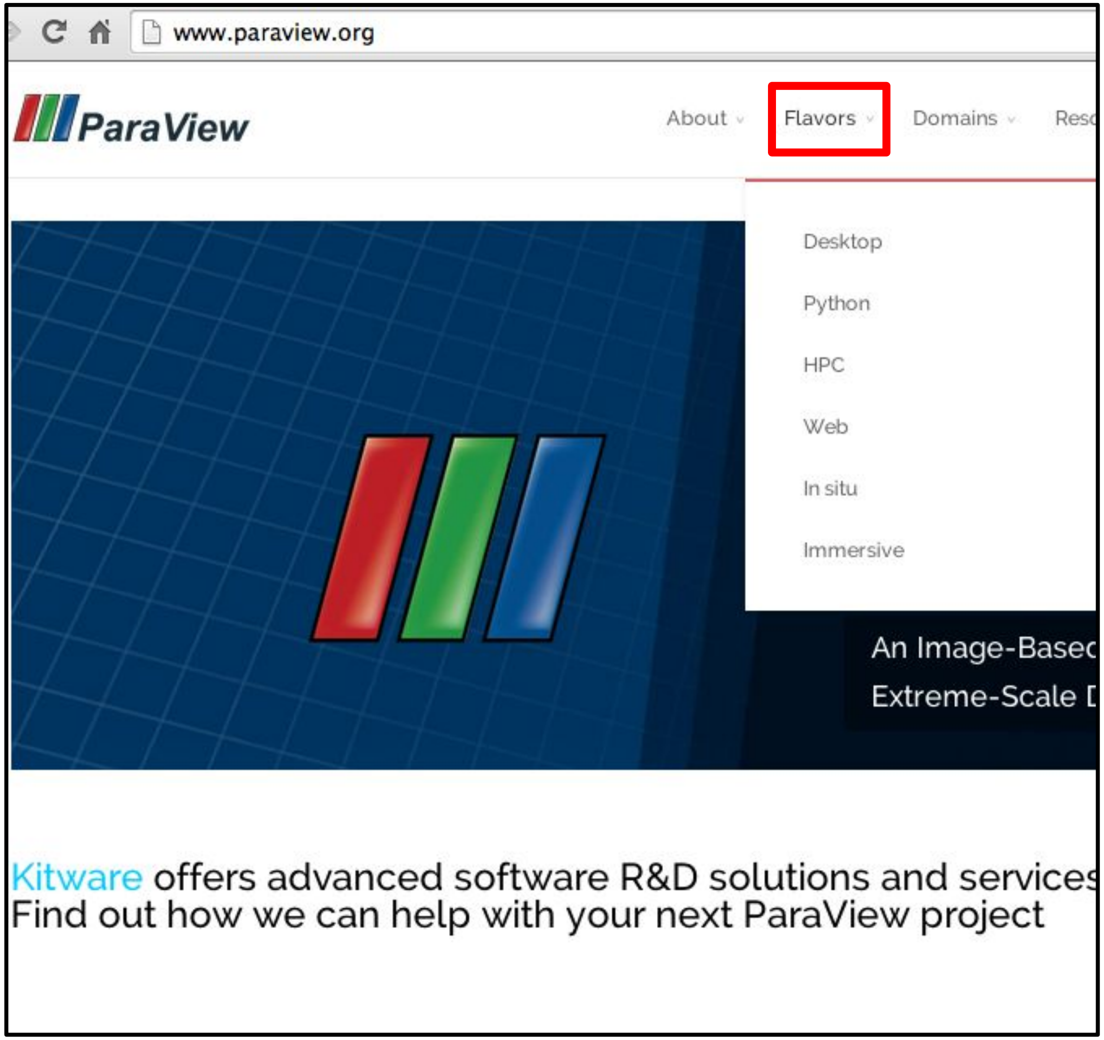
- Set up a visualization server

- Parallel visualization algorithms

- Memory tracking

- Rendering

- and more.



- http://www.paraview.org/Wiki/Main_Page