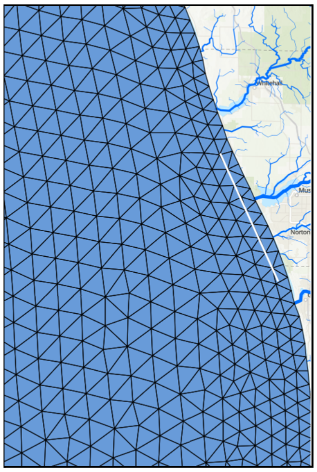
Visualizing a Transect using VisIt

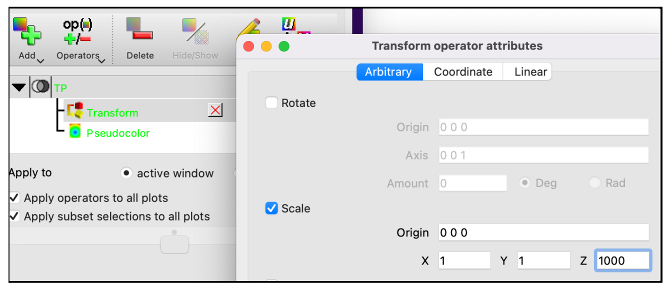


Example is from
A: x = 561,848.5, y = 4,756,940.5   (lon= -86.24, lat = 42.96)
to point
B: x = 548,466.63, y = 4,793,653.0   (lon= -86.40, lat = 43.29)

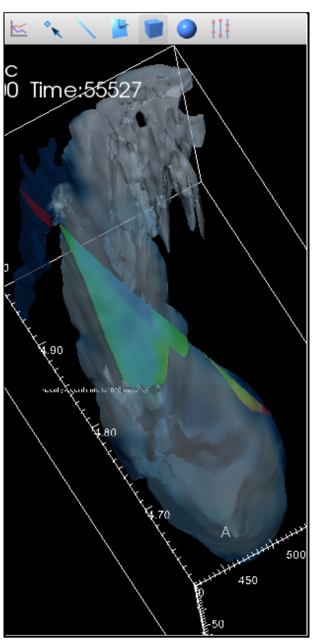
Open file. These plots are from mi_0013.nc (EPA's TP)
File:Open File:mi_0013.nc

Show plot
Add:Pseudocolor:TP
Draw

To see depth coordinate, scale the Z axis
Operators:Transforms:Transform
Click 'Transform'. A box will pop up.
Click the box next to Scale, and set Z to 1000.
Click Apply, then Dismiss
Draw



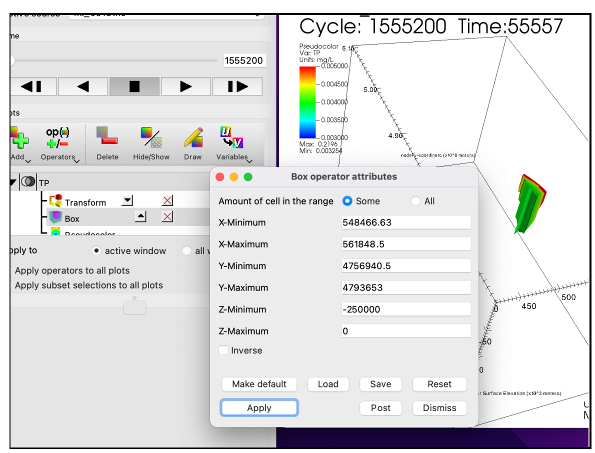
Change the min/max for the color map
Click 'Pseudocolor'. A box will pop up.
Check the boxes for Minimum and Maximum and choose values.
This example uses 0.003 and 0.005
Click Apply, then Dismiss
Draw



By default, VisIt will slice across the whole domain, as shown on the left.
To limit the domain, use the Box operator
Operators:Selection:Box
Click 'Box'. A box will pop up.
Put the min/max of the transect coordinates.

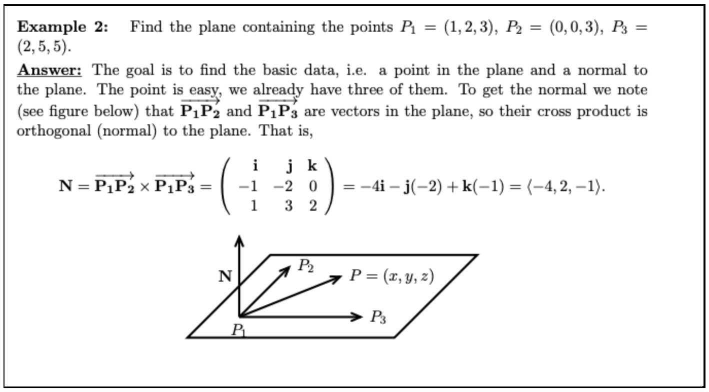
```
548466.63
561848.5
4756940.5
4793653.0
-250000
0
```

Click Apply, then Dismiss
Draw



Calculate then Create the slice

Previously, we showed a slice (plane) through a constant X, Y, Z. Now, the plane of our transect is 'diagonal'. A plane is defined by three points, and can also be specified by a point origin and a vector normal to the plane. VisIt requires that arbitrary slices be specified in point-normal form. Use the two points given for the transect, and use a third point that is identical to one of those with a different Z. Use the vector cross product to find the normal vector to the plane. Here is a reference, shamelessly copied from [MIT open courseware](#).



Example 2: Find the plane containing the points $P_1 = (1,2,3)$, $P_2 = (0,0,3)$, $P_3 = (2,5,5)$.

Answer: The goal is to find the basic data, i.e. a point in the plane and a normal to the plane. The point is easy, we already have three of them. To get the normal we note (see figure below) that $\overrightarrow{\mathbf{P_1P_2}}$ and $\overrightarrow{\mathbf{P_1P_3}}$ are vectors in the plane, so their cross product is orthogonal (normal) to the plane. That is,

$$\mathbf{N} = \overrightarrow{\mathbf{P_1P_2}} \times \overrightarrow{\mathbf{P_1P_3}} = \begin{pmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -1 & -2 & 0 \\ 1 & 3 & 2 \end{pmatrix} = -4\mathbf{i} - \mathbf{j}(-2) + \mathbf{k}(-1) = \langle -4, 2, -1 \rangle.$$

For our transect

P1 **= (**561848.5, 4756940.5, 0)

P2 = (548466.63, 4793653.0, 0)

P3 = (548466.63, 4793653.0, -1000)

For now I'm using R and putting in that value by hand. I need to look up the Python syntax in order to add it to the visit scripts. R commands listed way below, but the highlight is:

```
X <- p2-p1
Y <- p3-p1
myvec <- cross(X,Y)
norm_myvec <- myvec / sqrt(sum(myvec^2))
```

For these points, the normal vector is:

```
-0.9395312 -0.3424633  0.0000000
```

Create the slice using P1 and the normal vector

Operators:Slicing:Slice

Normal: Arbitrary: `-0.9395312 -0.3424633  0`

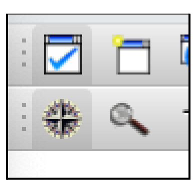
Origin:Point: `561848.5 4756940.5 0`

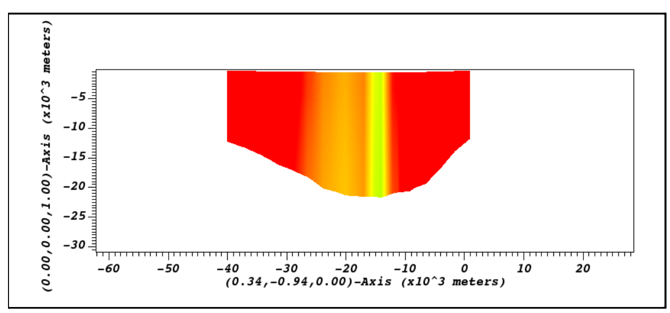
Project to 2D should be clicked.

Click Apply, then Dismiss.

Draw.

Zoom in with the magnifying glass.



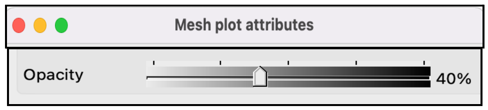


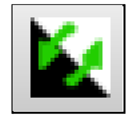
Check that it makes sense

To check the slice makes sense, use additional windows with a mesh, transparency, and lighting features. Here is a short summary of how I made the image below.

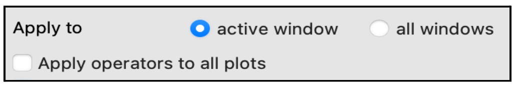
Starting a new session in VisIt, open the file, add a Pseudocolor plot of TP, change the color map, and scale the z axis.

Add a semi transparent mesh





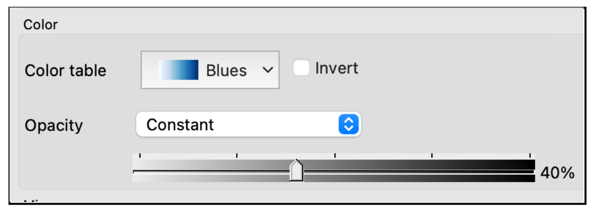
Add:Mesh:SigmaLayer_Mesh
Opacity: 40%
In the toolbars at the top of the screen, clicking the icon shown above for a black background.

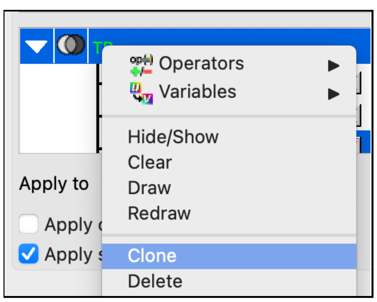


Unclick 'Apply operators to all plots'.

Apply the Box operation on *only* the Pseudocolor plot (do not highlight the Mesh plot).

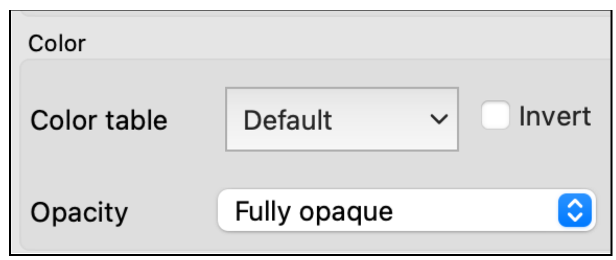
Change the color map of the Pseudocolor plot and use Constant Opacity of 40%.





Right click the variable TP and drop down to Clone.
Now there are two copies of the TP variable.

For the copy, change back to the original settings.
Color table:Default
Opacity:Fully opaque.



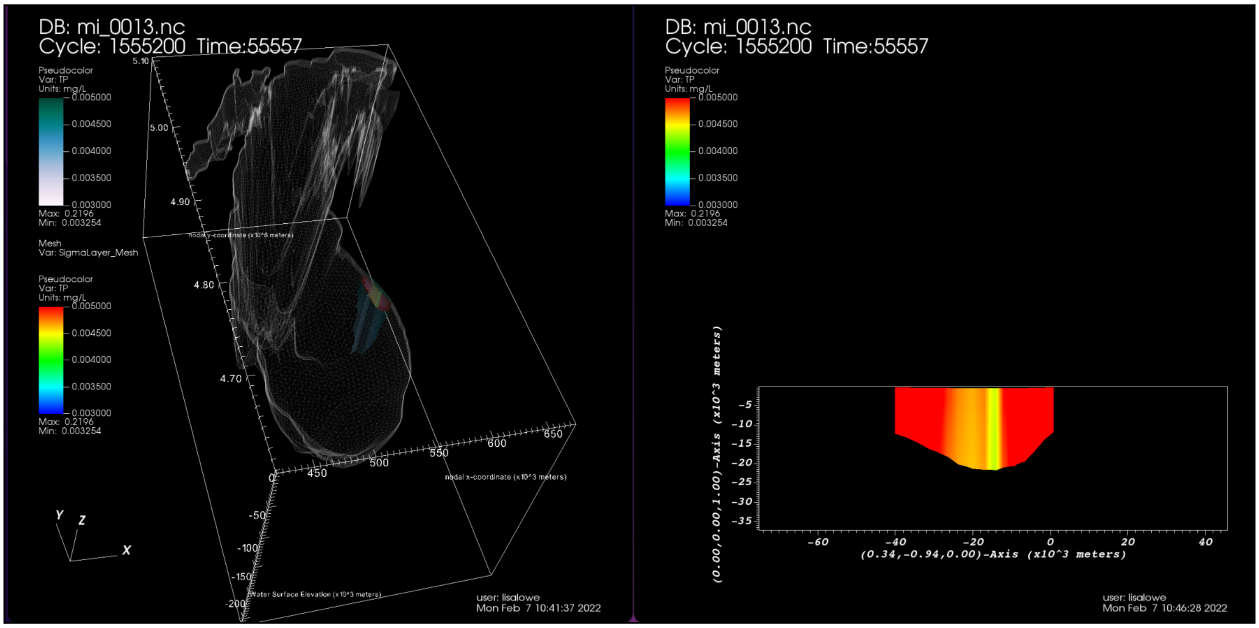
Apply the slicing procedure, but this time, *unclick* Project to 2D. Apply, Dismiss, and Draw. Change the opacity, colormaps, and lighting (Controls:Lighting) until you are convinced it is doing the correct thing.

Reduce the size of the window. Right click on the window and do Window:Clone, then Draw.

With Active Window=2, make sure to use Apply to active window, remove both the Mesh and the Pseudocolor of TP that does not have the slicing operator. Click Draw.

Click on Slice, then check Project to 2D, hit Apply, then Dismiss. Zoom in.

Final figure for '**Check that it makes sense**'



R commands used in interactive session

```
~$ R
library(pracma)
#Define your 2 points, plus third as a copy of one with different Z
p1 <- c(561848.5, 4756940.5, 0)
p2 <- c(548466.63, 4793653.0, 0)
p3 <- c(548466.63, 4793653.0, -1000)
#define vectors:
X <- p2-p1
Y <- p3-p1
#cross product
myvec <- cross(X,Y)
#normalize and print out
norm_myvec <- myvec / sqrt(sum(myvec^2))
norm_myvec
#I got:
#[1] -0.9395312 -0.3424633  0.0000000
```