RDAV Tutorial: Hands-on with VisIt on Nautilus

`http://rdav.nics.tennessee.edu/`

If you want to work hands-on, you will need to install VisIt and register a password token.

The data that we are using today comes from:
`http://iridl.ldeo.columbia.edu/SOURCES/.CARTON-GIESE/.SODA/.v1p2/`

You don't need to download the data for today's session on Nautilus, as I already have it saved on Nautilus as `/gpfs/medusa/szczepa/tutorial/ssh_global.cdf`. However, if you wish to work through these exercises at home, you can download the data set and work with it on your local machine. As long as you have VisIt installed on your computer, you can do *all* these exercises locally. To work locally on your own computer instead of remotely on Nautilus, from VisIt's **Open** dialog box, choose **localhost** instead of a remote computer and navigate to where you've saved the data on your computer.

Installing VisIt on your laptop varies depending on your operating system.

Windows: Copy **visit2.1.0.exe** to your desktop, double-click, and follow the instructions. It should install a double-clickable application and should add VisIt to **Start** → **All Programs**.

Linux: Run the provided VisIt install script:

```
visit-install 2.1.0 platform directory
```

Add *directory*`/bin` to your path.

Macintosh: If you know how to follow the Linux instructions, you can follow those. Otherwise, here are detailed steps that should work for *most* Mac users.

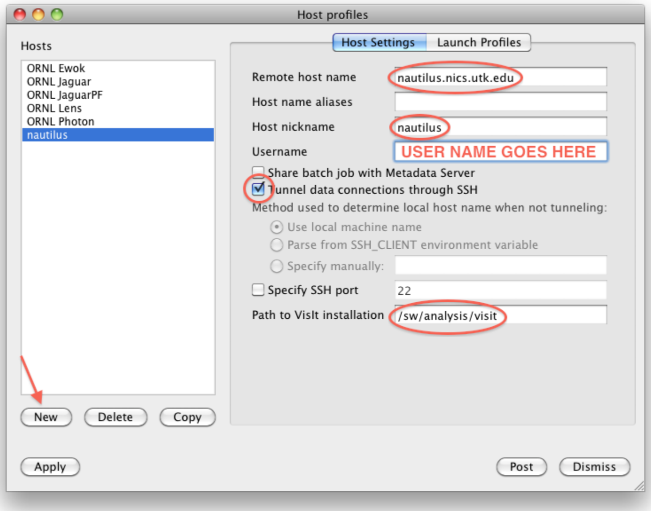
1. Double-click the **VisIt-2-1.1.0-x86_64-installer.dmg** file. Note: the exact file name will depend on which version of OS X you are running.
2. Launch the Terminal application found in **/Applications/Utilities**
3. Type: `cd '/Volumes/VisIt 2.1.0 installer'`
4. Type: `./visit-install 2.1.0 darwin-x86_64 /Applications/visit`

   Note: If you are running an older version of Mac OS X, you would need to replace **darwin-x86_64** with **darwin-i386**.
5. The script will ask you some questions. The answers don't really matter. You can answer things like "none" and "no" to all of them.
6. Type: `echo "export PATH=/Applications/visit/bin:$PATH" >> ~/.bash_profile`
7. Type: `source ~/.bash_profile`
8. To launch VisIt, type: `visit`

Next, launch VisIt. This is done by double-clicking on Windows and from launching from a shell (Terminal) window on Linux and Mac. Before we can use VisIt on Nautilus, we need to set some things up. These are settings that allow your local copy of VisIt (the client) to connect to and communicate with the copy of VisIt installed on Nautilus and to launch jobs remotely on your behalf. These settings are only necessary when working remotely on a system like Nautilus (and not for working locally just on your own computer).

1. From the **Options** menu, choose **Host profiles**. Click on **New** in the lower left of the Host Profile window. Enter the following **Host Settings**:

Remote host name:
  `nautilus.nics.utk.edu`
Host nickname: `nautilus`
Username: Type your username here. Remember, this is the username that was written on the envelope that the password token came in.
Tunnel data connections through SSH:
  Click this check box
Path to VisIt installation:
  `/sw/analysis/visit`



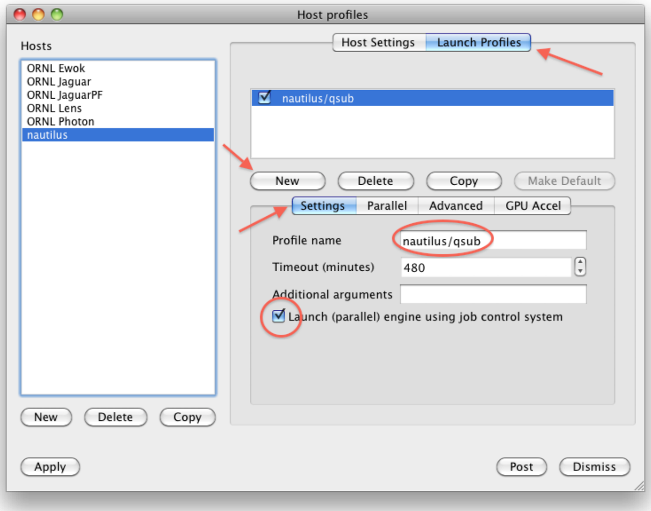
ORNL Ewok
ORNL Jaguar
ORNL JaguarPF
ORNL Lens
ORNL Photon
nautilus

Remote host name: nautilus.nics.utk.edu
Host name aliases:
Host nickname: nautilus
Username: USER NAME GOES HERE
☐ Share batch job with Metadata Server
☑ Tunnel data connections through SSH
Method used to determine local host name when not tunneling:
● Use local machine name
○ Parse from SSH_CLIENT environment variable
○ Specify manually:
☐ Specify SSH port: 22
Path to VisIt installation: /sw/analysis/visit

Move to the **Launch Profiles** tab. Click on **New** in the middle of the window. Enter the following Settings:

Profile name: `nautilus/qsub`

Launch (parallel) engine using job control system: Click this check box.



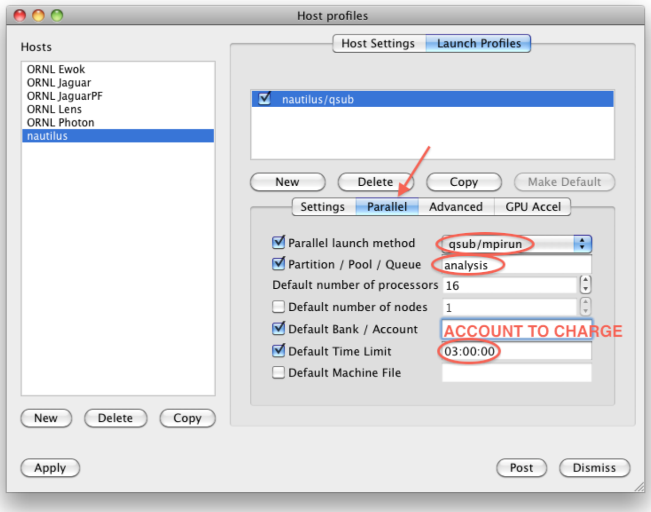
ORNL Ewok
ORNL Jaguar
ORNL JaguarPF
ORNL Lens
ORNL Photon
nautilus

☑ nautilus/qsub

Profile name: nautilus/qsub
Timeout (minutes): 480
Additional arguments:
☑ Launch (parallel) engine using job control system

From the **Parallel** tab:

Parallel launch method: qsub/mpirun
Partition / Pool / Queue: `analysis`
Default Bank / Account: `UT-NTNLEDU`
Default Time Limit: `03:00:00`



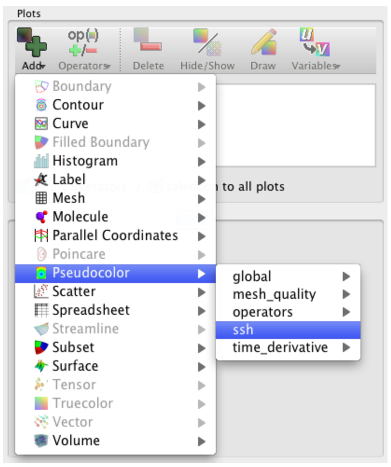
Hosts
ORNL Ewok
ORNL Jaguar
ORNL JaguarPF
ORNL Lens
ORNL Photon
nautilus

nautilus/qsub

Parallel launch method: qsub/mpirun
Partition / Pool / Queue: analysis
Default number of processors: 16
Default number of nodes: 1
Default Bank / Account: ACCOUNT TO CHARGE
Default Time Limit: 03:00:00
Default Machine File

Click **Apply** then **Dismiss**. From the **Options** menu, choose **Save Settings**. If you forget to do this, your host profile will not be saved!

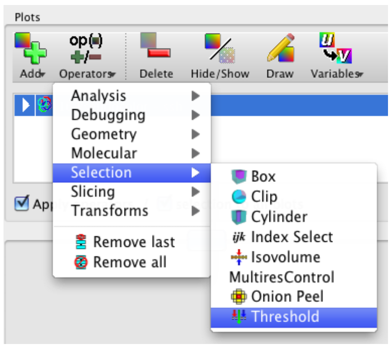
2. Next we will open a file on Nautilus. Choose **Open**. For **Host** choose **nautilus**. When prompted, enter your *PASSCODE*. Next, for the **Path** enter `/gpfs/medusa/szczepa/tutorial/`. Notice at the bottom of this window there is a pop-up list of all the file types that VisIt can read. One of VisIt's strengths is that it has built-in readers for many, many, many types of scientific data.

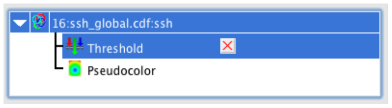


3. From the **Plots** area, choose **Add**. We will add a **Pseudocolor** plot of **ssh**. Once you have selected this, click on the **Draw** icon. The graph should clearly be a map of the world. However, you probably don't see much detail. This is because this NetCDF file uses very negative numbers as fill values. In this case, there is no sea surface height on land, so all the values on land are very negative numbers. VisIt tries to plot these as if they were real data.

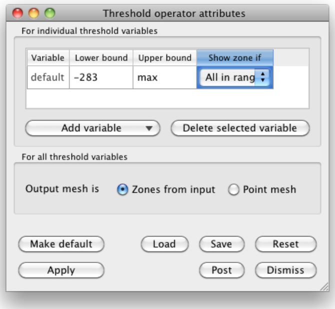
4. The icons at the top of the viewer area can adjust the view.

5. So that the filler values in the data aren't plotted, we'll set a threshold. From the **Operators** icon, go to **Selection** then **Threshold**. Click on the **Draw** icon again. Click on the triangle to the left of **ssh_global.cdf:ssh** so that we can adjust the settings. Double-click on **Threshold**.




16:ssh_global.cdf:ssh
Threshold
Pseudocolor

6. In the window that comes up, set the **Lower bound** to −283. Set **Show zone if** to **All in range**. Click **Apply**.


Threshold operator attributes

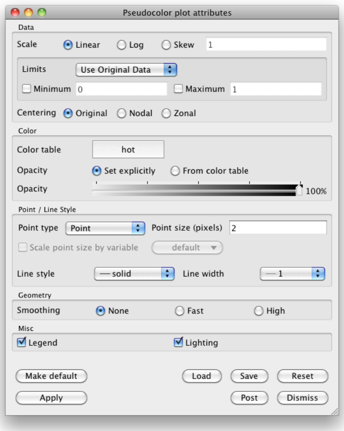
For individual threshold variables

Variable	Lower bound	Upper bound	Show zone if
default	-283	max	All in rang

For all threshold variables

Output mesh is [✓ Zones from input] Point mesh

7. We still haven't finished fixing the plot. We've filtered out the fill values, but the plotted values are still colored based on the original plot. Next, double-click on **Pseudocolor**. Change the **Limits** from **Use Original Data** to **Use Current Plot** and click **Apply**. We can make other changes to the plot from this window. We can change the color table. Choose a color table that you find aesthetically pleasing. We will be back in this window later to change the **Limits**.


Pseudocolor plot attributes

Data
Scale [✓ Linear] Log Skew 1
Limits Use Original Data
Minimum 0 Maximum 1
Centering [✓ Original] Nodal Zonal

Color
Color table hot
Opacity [✓ Set explicitly] From color table
Opacity 100%

Point / Line Style
Point type Point Point size (pixels) 2
Scale point size by variable default
Line style solid Line width 1

Geometry
Smoothing [✓ None] Fast High

Misc
[✓ Legend] [✓ Lighting]

8. We can save the image locally. If you choose **Save window...** from the **File** menu, VisIt will automatically save the current plot in whatever directory you launched VisIt from—and it will pick a name for it on your behalf. If you want to have some say in how things are saved, you first need to **Set Save Options....** The other option for saving is **Save Session....** This will save the steps that VisIt took to reach this point. If you want to recreate your plots in the future, you can **Save Session...** and **Restore Session....**

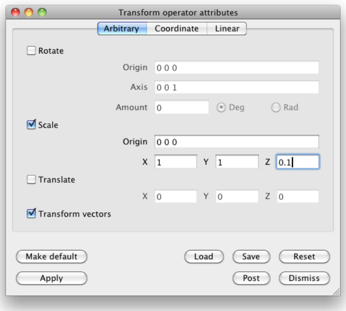
9. We can also animate this plot. Press **play** from the VCR buttons in the middle of the VisIt window. We can watch this movie in VisIt or we can export the movie to our own laptops. To save the movie, from the **File** menu, choose **Save movie...** and follow the directions from the movie wizard. *Don't do this now because it can be really slow to do this in client-server mode over the wireless network.*

10. There is a problem with the collection of frames that we've created. If you look, the minimum and maximum value are set to the largest and smallest in each time step, so it's hard to compare

timesteps in an absolute sense. Double-click on **Pseudocolor** again. We'll set the **Limits** to minimum and maximum values that will apply to all time steps. We'll set the minimum value to -283 and the maximum value to 194. Click **Apply**. Where did the numbers -283 and 194 come from? I found them with R.

11. Since this is a plot of *height*, it might be nice to render the height with a more 3D look. From the **Plots** section of the VisIt window, choose the **Operators** icon, then go to **Transform**, and then **Elevate**. Click the **Draw** icon. Once the plot renders, you can grab it with your mouse and rotate it to get a better look. One thing that you may find is that the elevation level is too extreme. We can fix that with yet another operator.

12. From the **Operators** icon, choose **Transforms** then **Transform**. Double click on **Transform** from the list of operators applied to this plot. Select the **Scale** check-box and set the z-scale to 0.1. Click **Apply**.



Transform operator attributes

Arbitrary | Coordinate | Linear

☐ Rotate
Origin 0 0 0
Axis 0 0 1
Amount 0 (● Deg) ○ Rad
☑ Scale
Origin 0 0 0
X 1 Y 1 Z 0.1
☐ Translate
X 0 Y 0 Z 0
☑ Transform vectors

13. Let's go back to our original plot. We'll click the Xs for our **Elevate** and **Transform** operators to remove them from our plot. We'll keep the pseudocolor plot and the threshold operator.

14. We'll add a new plot. We'll **Add** a **mesh** plot and plot the variable **mesh720x330**. Click **Draw**. Don't worry about the error message. Use the magnifying glass near the top of the viewer window to zoom in on the graph. To find values at particular points, use the **node pick** ($+_N$) and **zone pick** ($+_Z$) tools from the tool bar at the top of the viewer window. The node pick tool finds the value where the mesh lines intersect; the zone pick finds the value in the middle of the zone.

15. We'll now look at a **line out**. This graphs the z-values along a chosen line. Click on the **Reset View** icon, select the mesh plot and choose **Hide/Show plot** to hide the mesh plot, select the pseudocolor plot, and then choose the **line out** tool. Draw a line *that does not cross land*.

16. Many, many other useful tools are in the **Controls** menu, including **Queries** and **Expressions**.

17. We have been using VisIt interactively through the GUI. You can control VisIt through a scripting interface that can be run from the command line. One of the easiest ways to create a script is to record VisIt doing some action and then copy-and-paste that into a file (and edit if necessary). Delete the plot that we've made by using the **Delete** icon from the plot area of the VisIt window.

18. From the **Controls** menu, choose **Command...**. Click on **Record**. Make a pseudocolor plot of **global** $\rightarrow$ **ssh**. Add a threshold operator (minimum of -283) and limits in the pseudocolor plot (minimum of -283 and maximum of 194, use current plot). Click on **Stop** to stop the recording.

19. You can copy-and-paste the text from this box and then edit it, if necessary. Suppose you saved the script as *myvisitscript.py*. You could then run it from the command line on your local computer as:

```
visit -cli -nowin -s myvisitscript.py
```

20. Here is the listing (next page) of a script runs on Nautilus.

```
import sys
OpenDatabase("/gpfs/medusa/szczepa/tutorial/ssh_global.cdf", 0)
AddPlot("Pseudocolor", "ssh", 1, 1)
AddOperator("Threshold", 1)
ThresholdAtts = ThresholdAttributes()
ThresholdAtts.outputMeshType = 0
ThresholdAtts.listedVarNames = ("default")
ThresholdAtts.zonePortions = (1)
ThresholdAtts.lowerBounds = (-198)
ThresholdAtts.upperBounds = (125)
ThresholdAtts.defaultVarName = "ssh"
ThresholdAtts.defaultVarIsScalar = 1
SetOperatorOptions(ThresholdAtts, 1)
PseudocolorAtts = PseudocolorAttributes()
PseudocolorAtts.legendFlag = 1
PseudocolorAtts.lightingFlag = 1
PseudocolorAtts.minFlag = 1
PseudocolorAtts.maxFlag = 1
PseudocolorAtts.centering = PseudocolorAtts.Natural  # Natural, Nodal, Zonal
PseudocolorAtts.scaling = PseudocolorAtts.Linear  # Linear, Log, Skew
PseudocolorAtts.limitsMode = PseudocolorAtts.OriginalData  # OriginalData, CurrentPlot
PseudocolorAtts.min = -198
PseudocolorAtts.max = 125
PseudocolorAtts.pointSize = 0.05
PseudocolorAtts.pointType = PseudocolorAtts.Point  # Box, Axis, Icosahedron, Point, Sphere
PseudocolorAtts.skewFactor = 1
PseudocolorAtts.opacity = 1
PseudocolorAtts.colorTableName = "hot_desaturated"
PseudocolorAtts.smoothingLevel = 0
PseudocolorAtts.pointSizeVarEnabled = 0
PseudocolorAtts.pointSizeVar = "default"
PseudocolorAtts.pointSizePixels = 2
SetPlotOptions(PseudocolorAtts)
DrawPlots()
s = SaveWindowAttributes()
s.progressive = 1
s.fileName = "VISITIMAGE"
SetSaveWindowAttributes(s)
name = SaveWindow()
sys.exit()
```

21. If you had this script saved on Nautilus as *myvisitscript.py*, you could submit it to the queue with the following PBS script.

```
#PBS -A UT-NTNLEDU
#PBS -q analysis
#PBS -l walltime=0:05:00,ncpus=4,mem=8GB

cd $PBS_O_WORKDIR
module load visit
visit -nowin -cli -s myvisitscript.py -np 4
```

You would then submit it to the queue by using the **qsub** command and the name of the PBS script. If you had named the PBS script *myvisit.pbs*, you would submit it with: `qsub myvisit.pbs` . This script assumes that *myvisitscript.py* and *myvisit.pbs* are in the same directory and that you issue the **qsub** command from that directory.