A Basic Overview of Visualization Tools

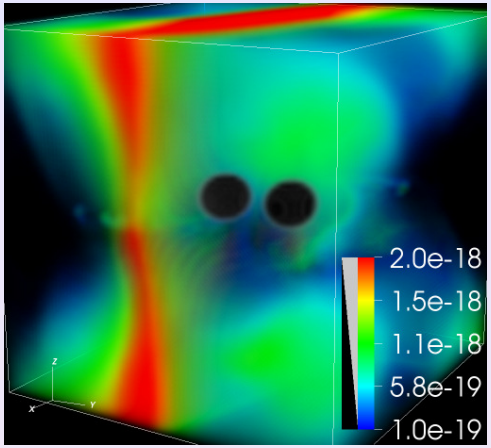
Marcelo Ponce

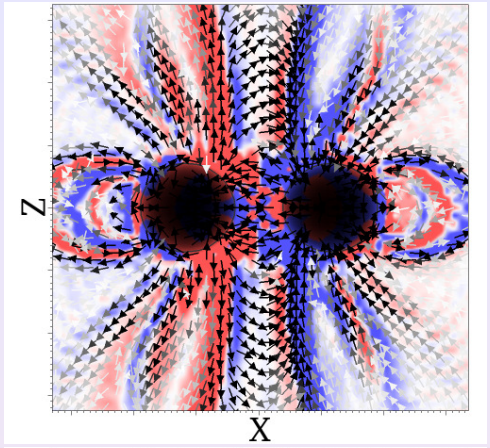
SNUG: SciNet User Group meeting

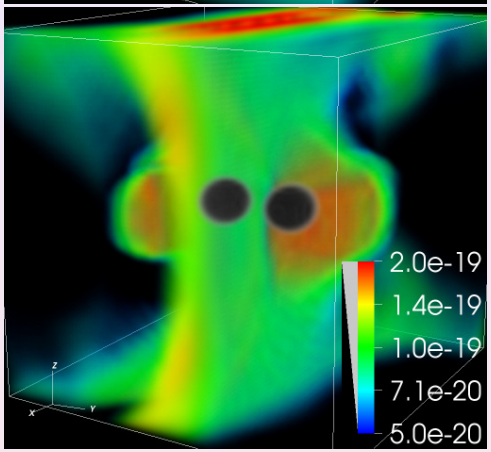
September 14th, 2016

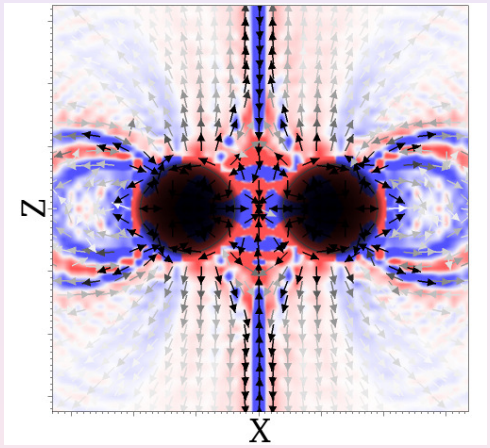
Outline

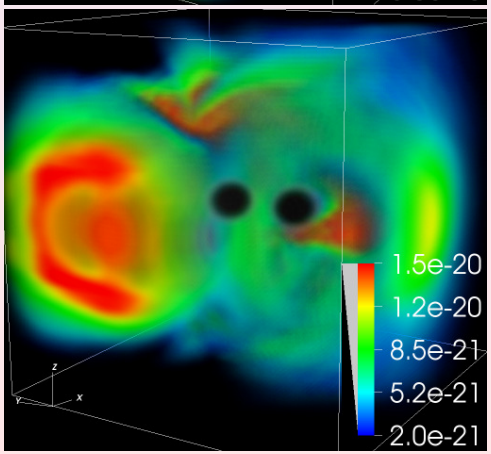
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

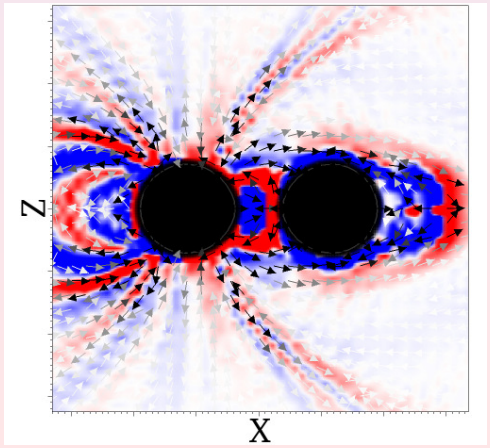


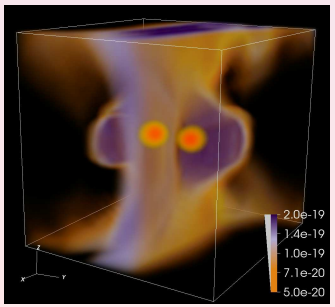


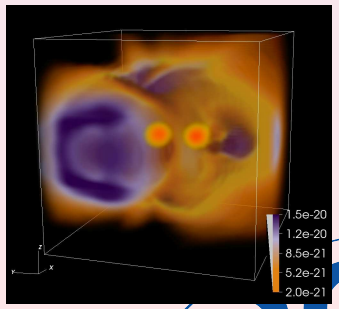


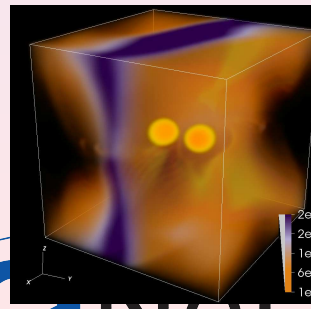






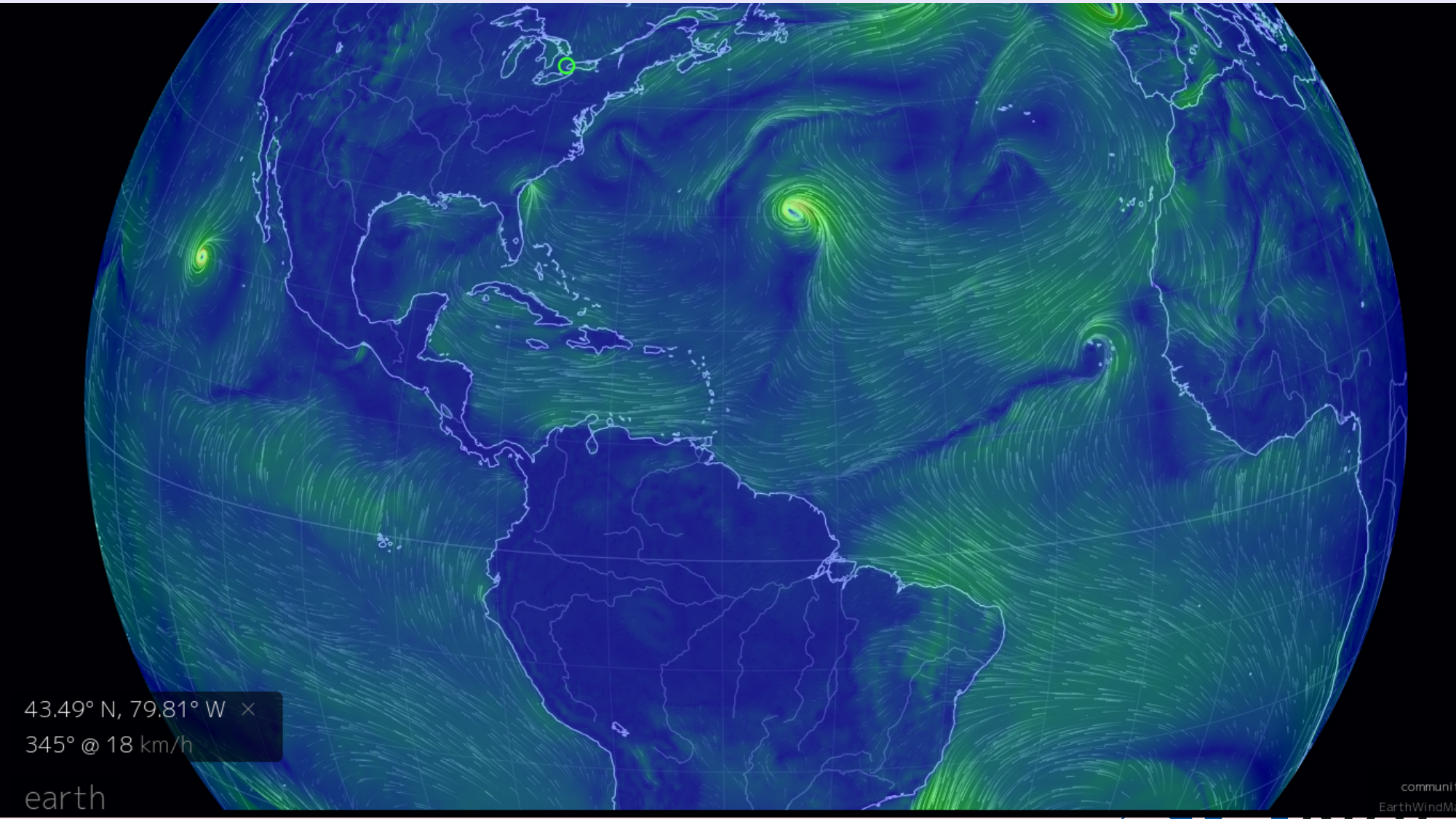




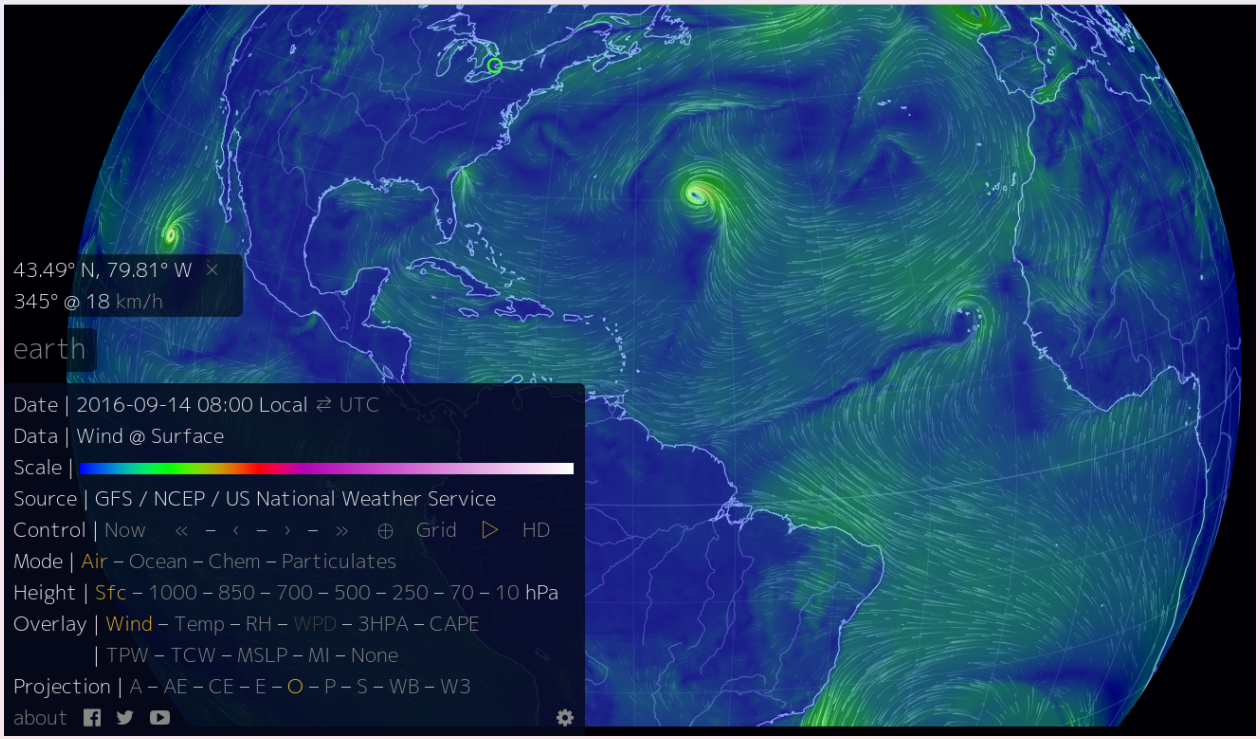


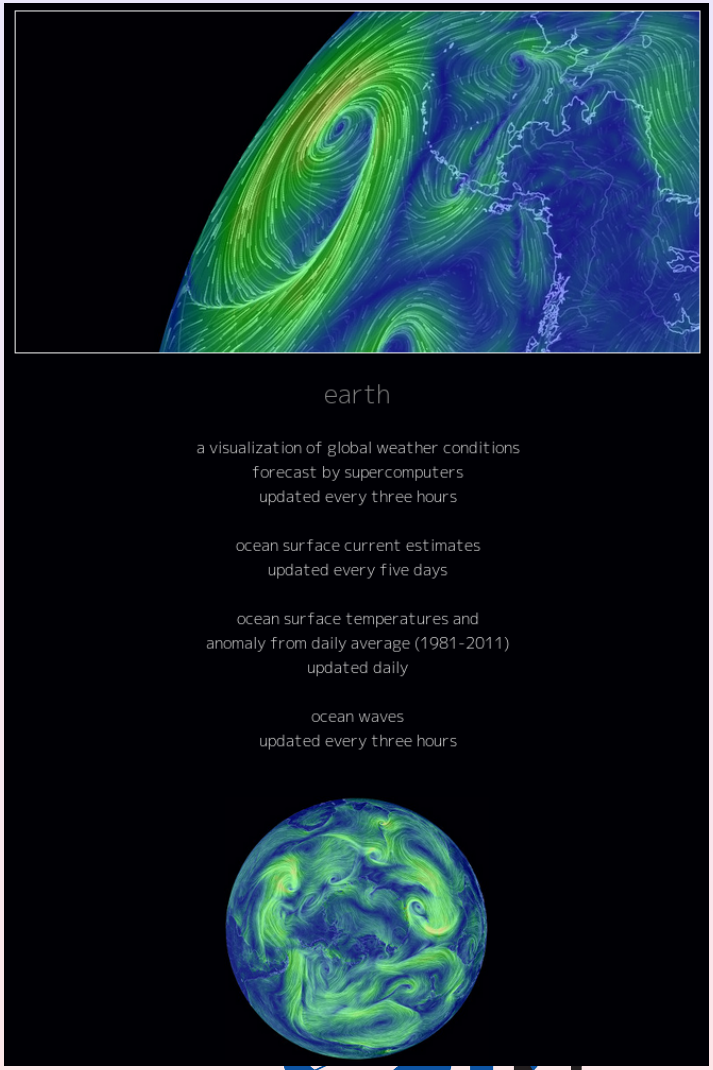
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

http://earth.nullschool.net



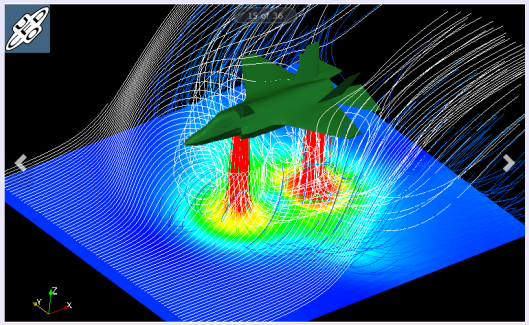
http://earth.nullschool.net

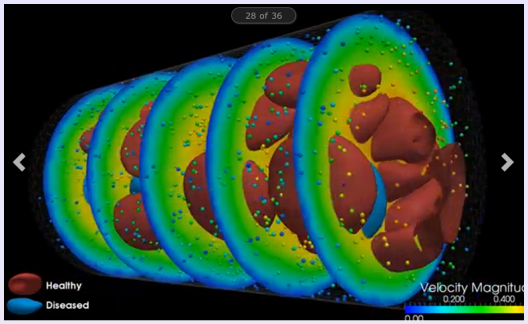




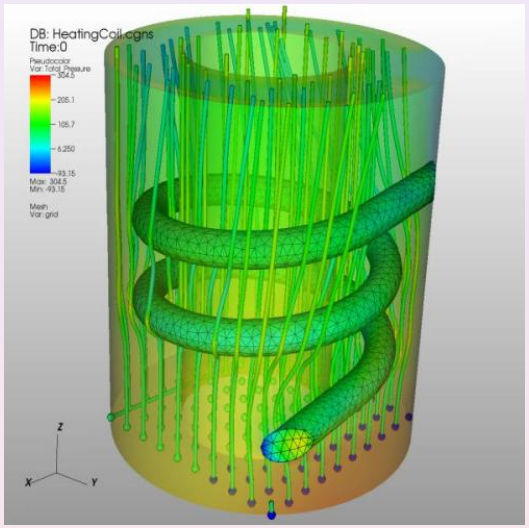
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

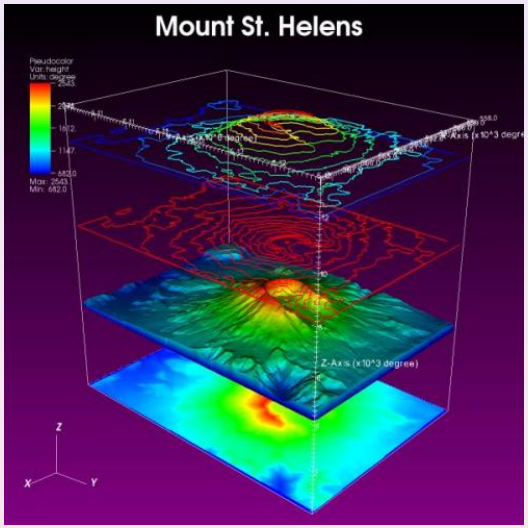
- Visualization is the process of mapping scientific data into "*visual form*"
- Much easier to understand images than a large set of numbers
- For interactive data exploration, debugging, communication with peers





ParaView





SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

2D/3D visualization packages

- **gnuplot**: command-driven interactive 2d and 3d plotting program
- **GraphViz**: represent strucrural information as diagrams of abstract graphs and networks
- **HDFview**: visual tool for browsing and editing HDF4 and HDF5 files
- **ImageMagick**: manipulation of image
- **Molden**: pre/post-processing for molecular and electronic structures
- **OpenCV**: library for real time computer vision
- **ParaView**: Parallel visualization application
- **SciLab**: open source platform for numerical computation
- **VisIt**: Visualization Tool for HPC
- **XCrysDen**: Crystaline and Molecular Structure Visualization
- **yt**: python-based package for visualization of AMR datasets
- **VMD**: Visualization for Molecular Dynamics
- **openDX**: very old, not mantained package, but really nice approach to visualization process (modules)

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

2D/3D visualization packages

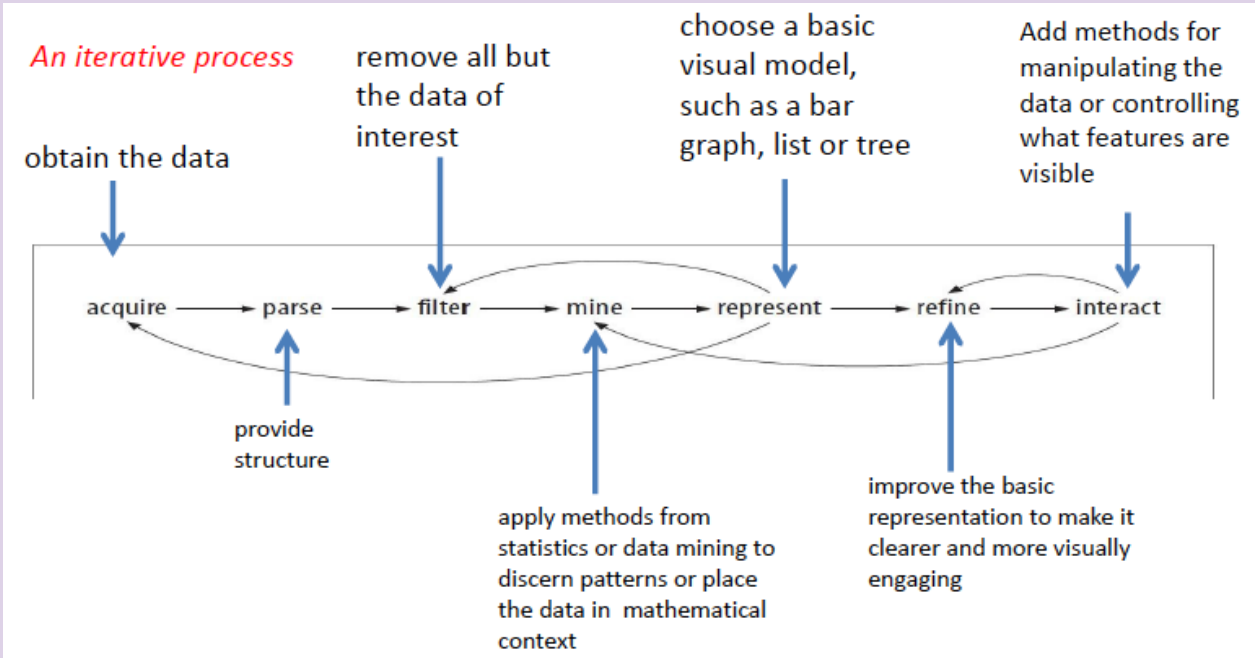
General Features

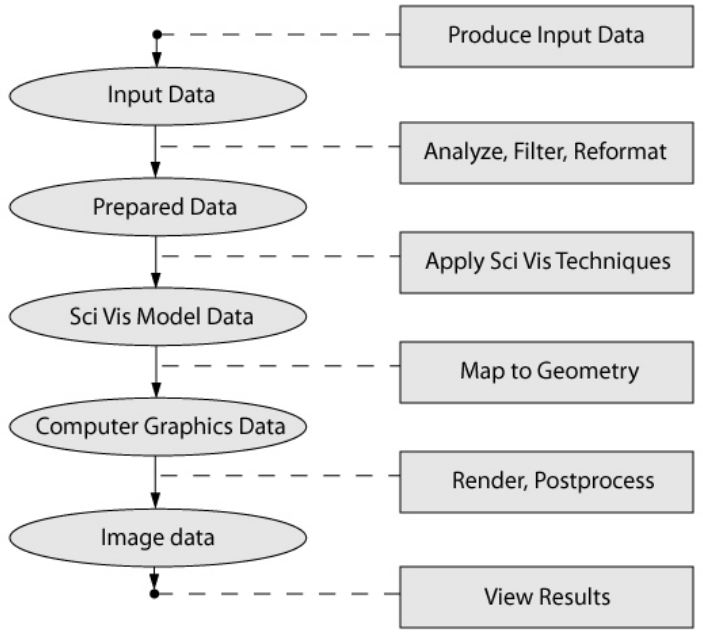
- visualize scalar and vector fields
- structured and unstructured meshes in 2D and 3D, particle data, polygonal data, irregular topologies
- ability to handle very large datasets (GBs to TBs)
- ability to scale to large ($10^3 - 10^5$ cores) computing facilities
- iteractive manipulation
- support for scripting, common data formats, parallel I/O
- open-source, multi-platform, and general-purpose

ParaView

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Data Visualization Process





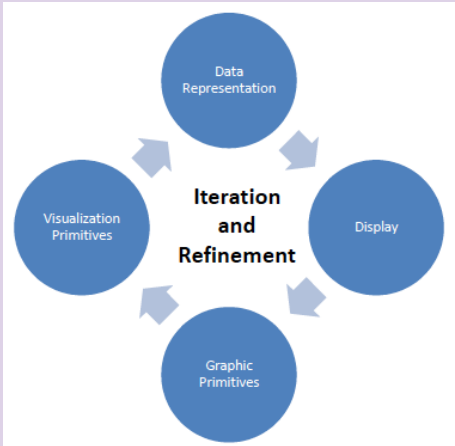
Scientific Visualization Techniques

- contours/isosurfaces, clips, thresholds, glyphs, streamlines, pseducolors, ...

Map to Geometry

- scalars, vectors, tensors
- 1D, 2D, 3D
- mesh/grid

Render/PostProcess



Visualization Toolkit (VTK)

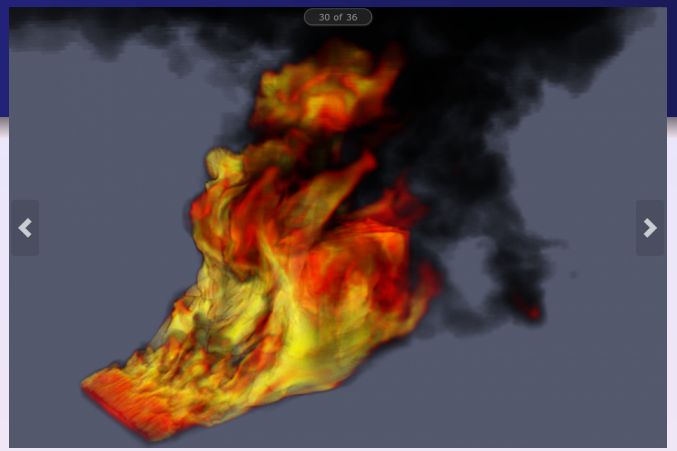
- Open source, multiplatform
- Supports distributed computation models
- Extensible modular architecture
- Available for 3D computer graphics, image processing and visualization
- Collection of C++ libraries
- Leveraged by many applications
- Divided into logical areas
  - Filtering
  - Information Visualization
  - Volume Rendering
- Cross platform, using OpenGL
- Wrapped in Python, Tool Command Language (Tcl) and Java

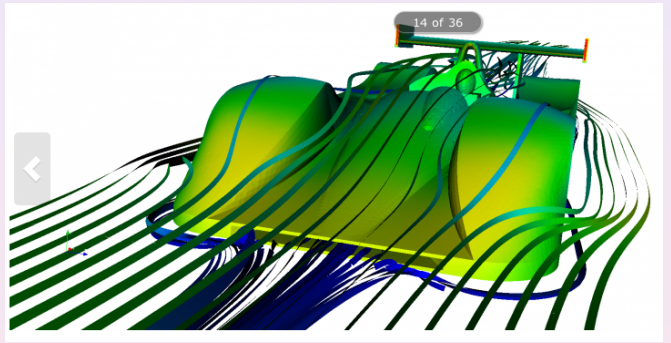
ParaView and **VisIt** are end-user applications with support:

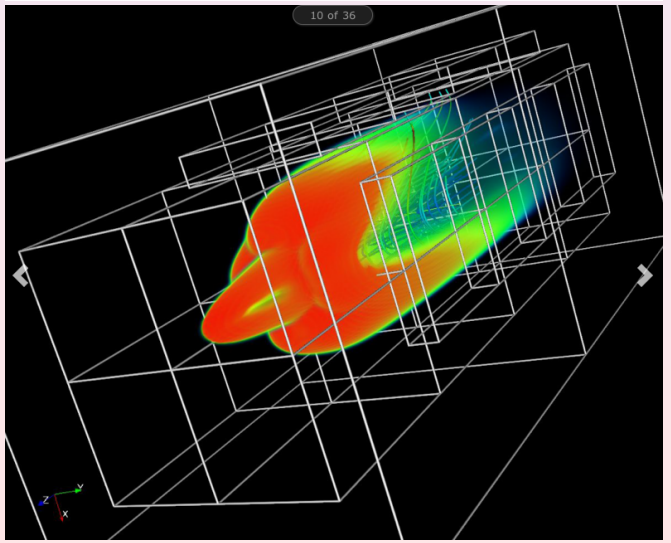
- *parallel* Data Archiving/Reading/Processing/Rendering
- single node, *Client-Server*, *MPI Cluster* Rendering

ParaView

- **http://www.paraview.org**
- Started in 2000 as a collaboration between *Los Alamos National Lab* and *Kitware Inc.*, later joined by *Sandia National Labs*, also the *Army Research Lab.*
- Latest stable release 5.1.2, available for Linux/Mac/Windows
- To visualize extremely large datasets on distributed memory machines
- Both interactive and Python scripting
- Uses MPI for distributed-memory parallelism on HPC clusters
- ParaView is based on VTK (Visualization Toolkit)







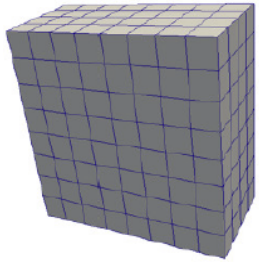
ParaView: Data Types, Grids, ...

- Supported Data Types

point data, polygonal data, images, multi-block, Adaptive Mesh Refinement (AMR), time series support

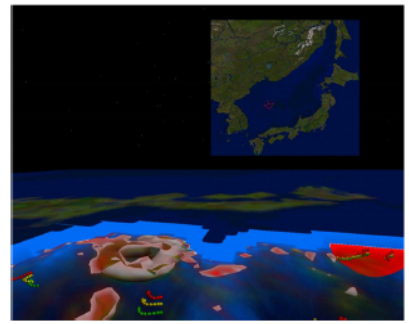
- Supported File Formats

Large number of file formats supported

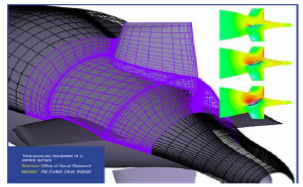


Uniform Rectilinear (Image Data)
A uniform rectilinear grid is a one- two- or three- dimensional array of data. The points are orthonormal to each other and are spaced regularly along each direction.

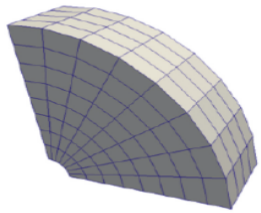
Grid – regular structure, all voxels (cells) are the same size and shape



Adopted from The ParaView Tutorial, The Basics of Visualization, version 3.98

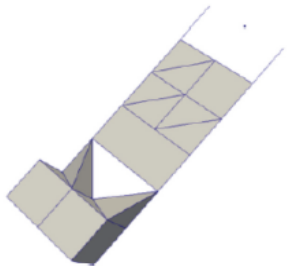


Curvilinear – regularly gridded mesh shaping function applied



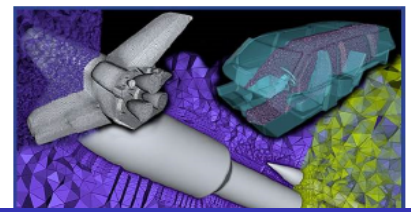
Curvilinear (Structured Grid)
Curvilinear grids have the same topology as rectilinear grids. However, each point in a curvilinear grid can be placed at an arbitrary coordinate (provided that it does not result in cells that overlap or self intersect). Curvilinear grids provide the more compact memory footprint and implicit topology of the rectilinear grids, but also allow for much more variation in the shape of the mesh.

Adopted from The ParaView Tutorial, The Basics of Visualization, version 3.98



Unstructured Grid
Unstructured data sets are composed of points, lines, 2D polygons, 3D tetrahedra, and nonlinear cells. They are similar to polygonal data except that they can also represent 3D tetrahedra and nonlinear cells, which cannot be directly rendered.

Unstructured grid – irregular mesh typically composed of tetrahedra, prisms, pyramids, or hexahedra



Visualization Algorithms & Pipeline

- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Algorithms & Pipeline

- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Algorithms & Pipeline

- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Algorithms & Pipeline

- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Algorithms & Pipeline

- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Algorithms & Pipeline

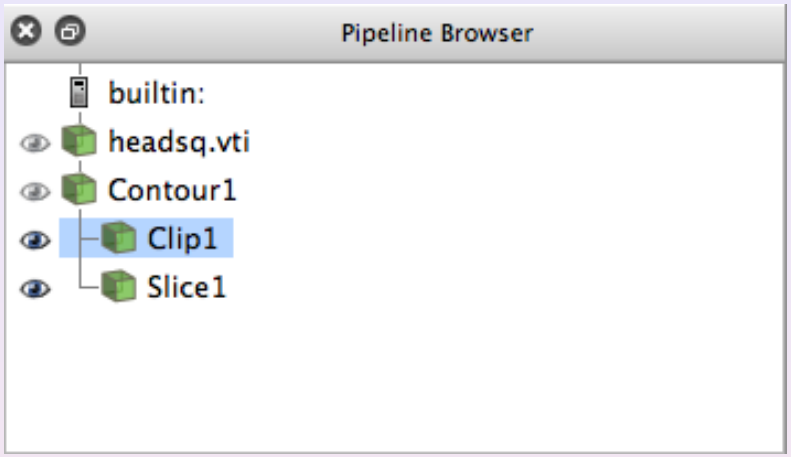
- Supported Visualization Algorithms
  - isosurfaces
  - cutting planes
  - streamlines
  - glyphs
  - volume rendering
  - clipping
  - height maps
  - ...

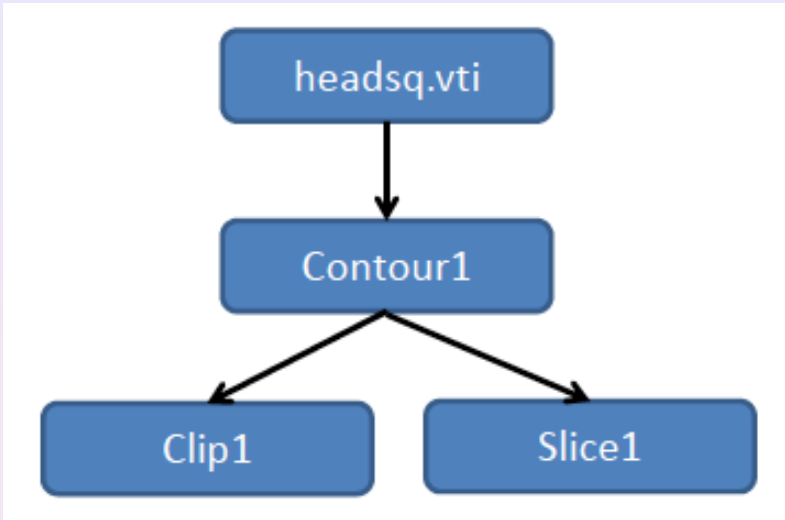
- generate animations
- can run in parallel/distributed mode for large data visualization

- Visualization Pipeline, in ParaView
  - read data into ParaView
  - apply (any number of) filters to process the original data ➟ generate, extract or derive features from the data
  - a viewable image is rendered from the data

- all processing operations (filters) ⇝ data sets
- by further process the result of every operation ⇝ build complex visualizations (eg. extract a cutting plane + apply glyphs ⇒ a plane of glyphs trought a 3D volume)

Visualization Pipeline

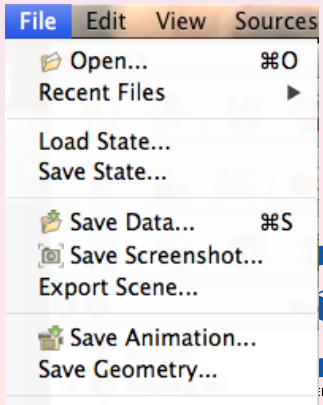




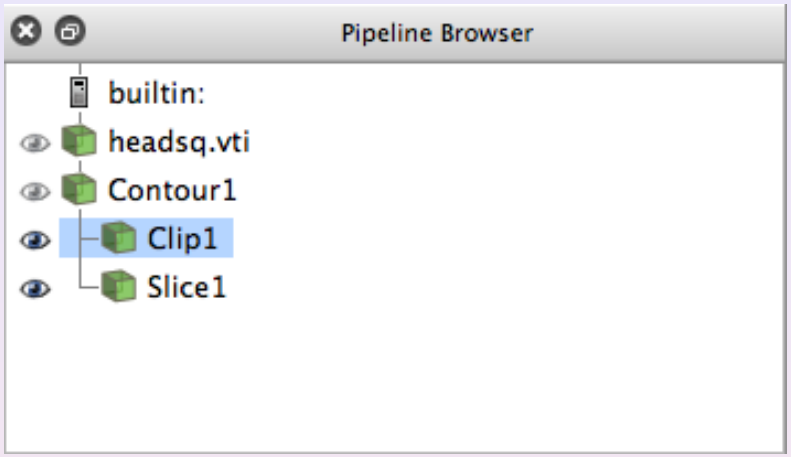
- read/import data ➠ apply operators (filters/...) ➠ render ➠ ... ➠ iterate

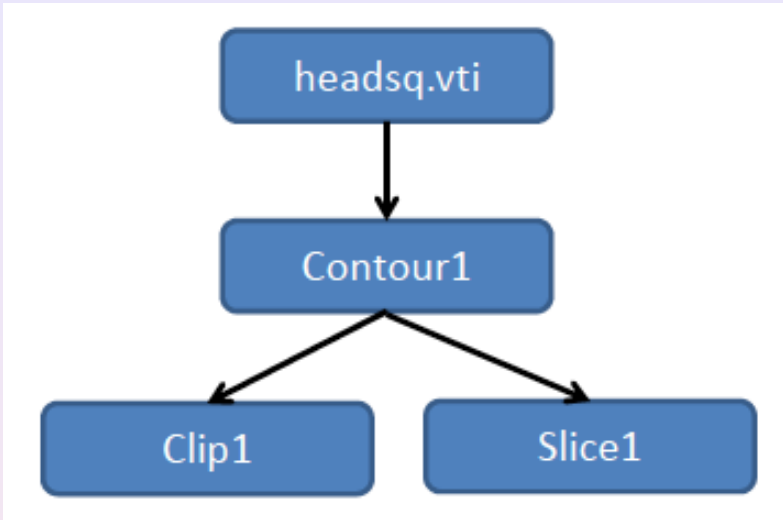
- Save your work!

File → Save ...



Visualization Pipeline

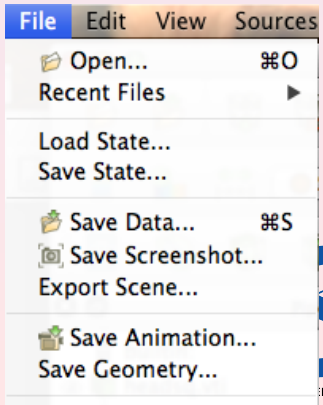




- read/import data ➠ apply operators (filters/...) ➠ render ➠ ... ➠ iterate

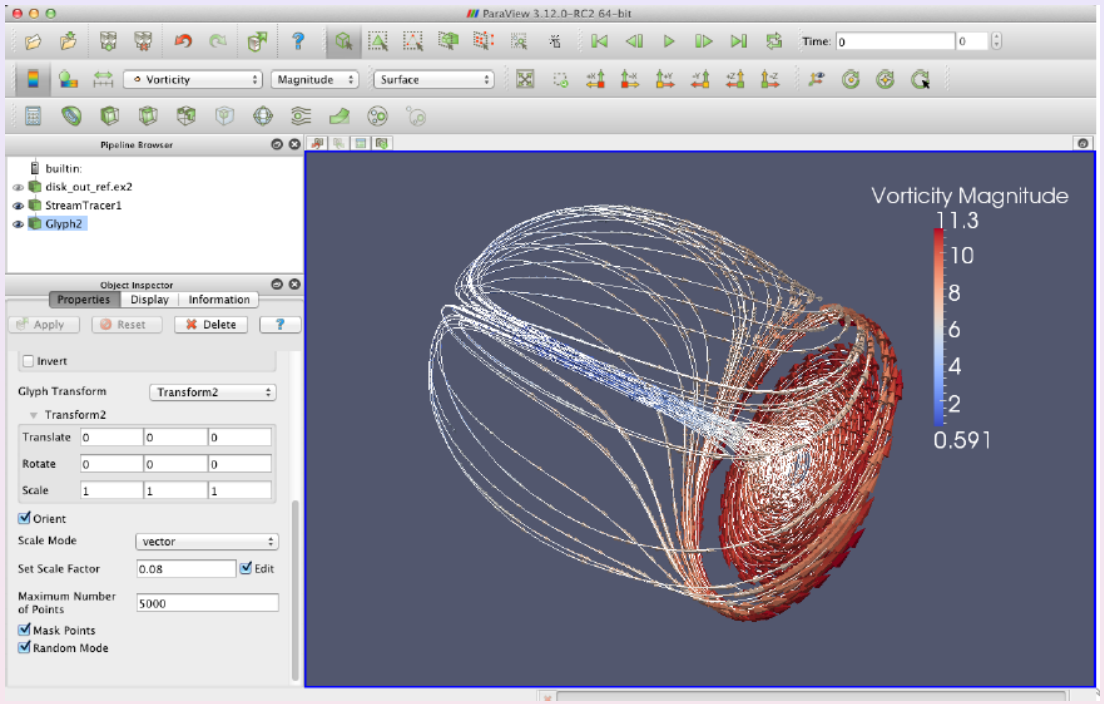
- Save your work!

File → **Save ...**



Streamlines & Glyphs

- Filters → Common ▸ → **StreamTracer**
  - seeds, Point/Line sources, other params...

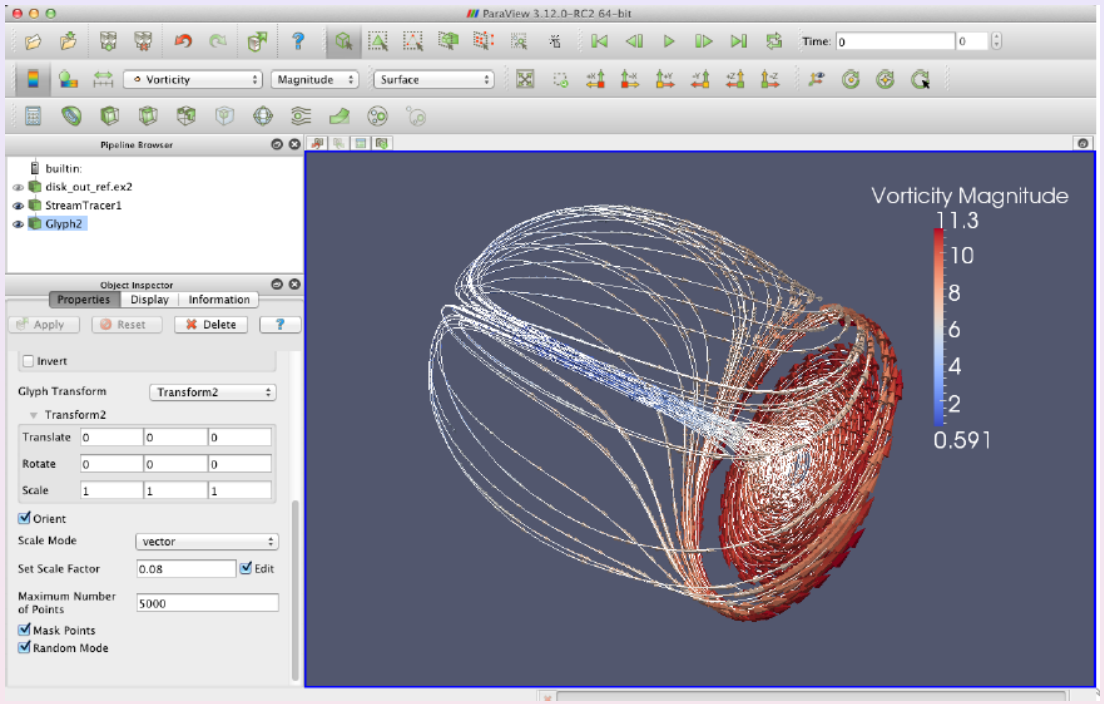


- Filters → Alphabetical ▸ → Tube
- Filters → Common ▸ → Glyphs

- in the object inspector:
  - change the Vectors option (second from the top) to *V*
  - change the Glyph Type option (third from the top) to "Cone", hit Apply
  - colour the glyphs with the *Temp* variable

Streamlines & Glyphs

- Filters → Common ▸ → **StreamTracer**
  - seeds, Point/Line sources, other params...
- Filters → Alphabetical ▸ → **Tube**
- Filters → Common ▸ → Glyphs

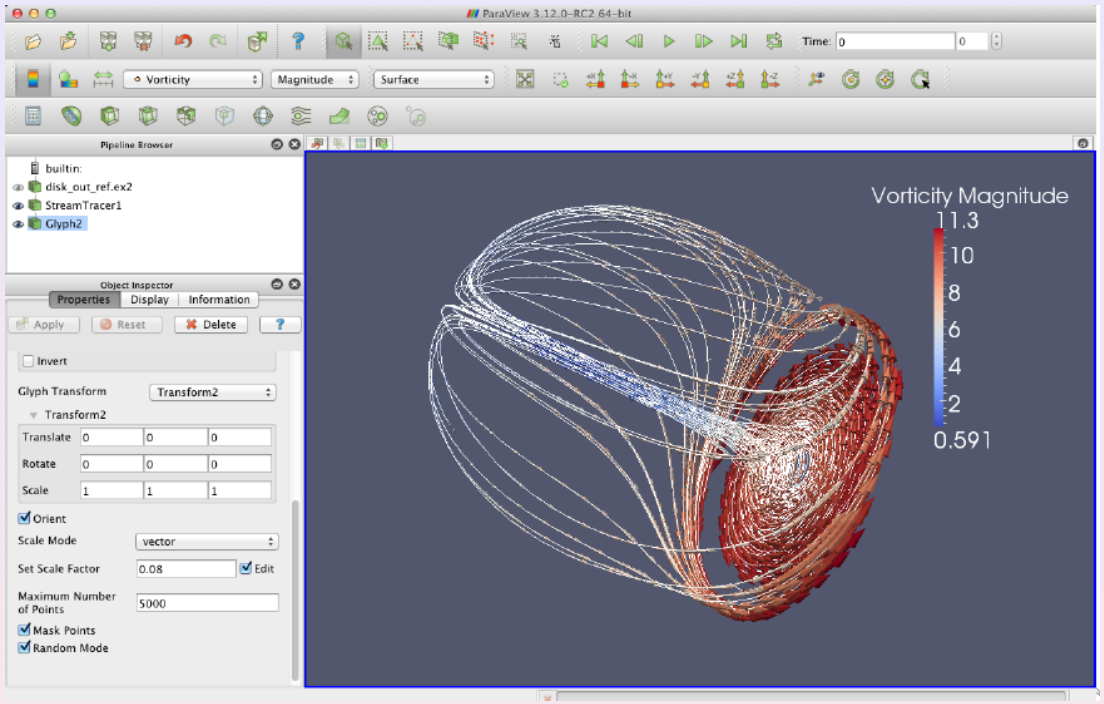


- in the object inspector:
  - change the Vectors option (second from the top) to *V*
  - change the Glyph Type option (third from the top) to "Cone", hit Apply
  - colour the glyphs with the *Temp* variable

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Streamlines & Glyphs

- Filters → Common ▸ → **StreamTracer**
  - seeds, Point/Line sources, other params...
- Filters → Alphabetical ▸ → **Tube**
- Filters → Common ▸ → **Glyphs**

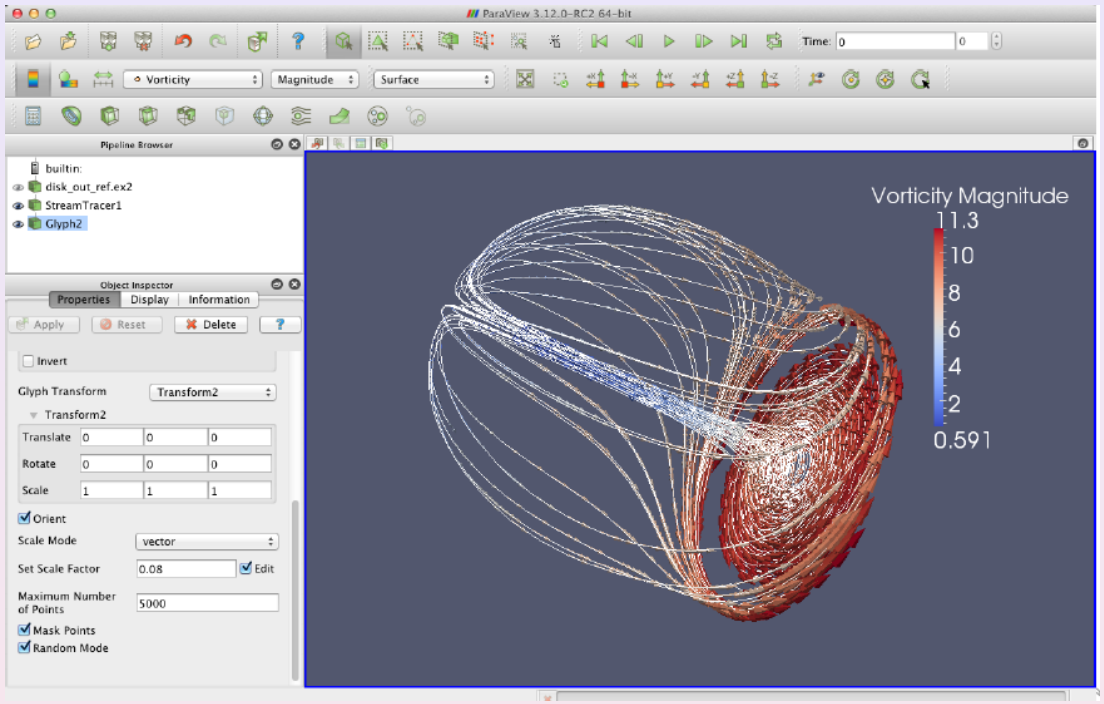


- in the object inspector:
  - change the Vectors option (second from the top) to *V*
  - change the Glyph Type option (third from the top) to "Cone", hit Apply
  - colour the glyphs with the *Temp* variable

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Streamlines & Glyphs

- Filters → Common ▸ → **StreamTracer**
  - seeds, Point/Line sources, other params...
- Filters → Alphabetical ▸ → **Tube**
- Filters → Common ▸ → **Glyphs**

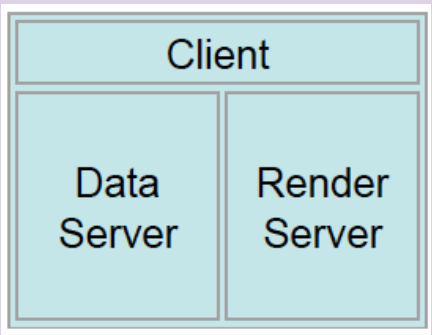


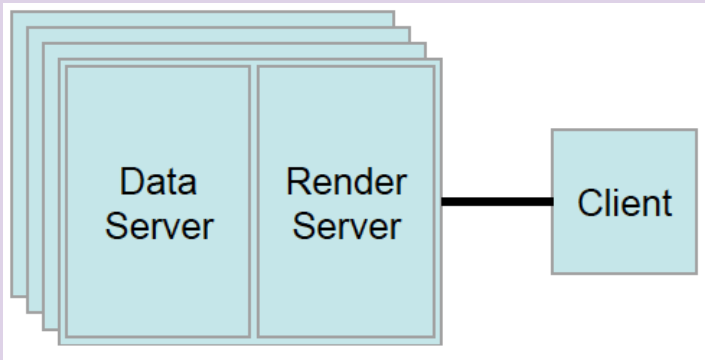
- in the object inspector:
  - change the Vectors option (second from the top) to *V*
  - change the Glyph Type option (third from the top) to "Cone", hit Apply
  - colour the glyphs with the *Temp* variable

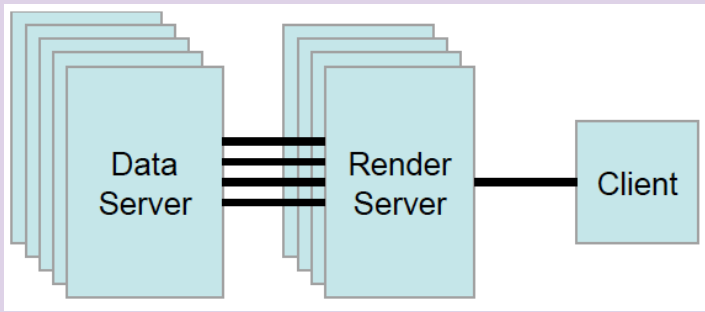
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

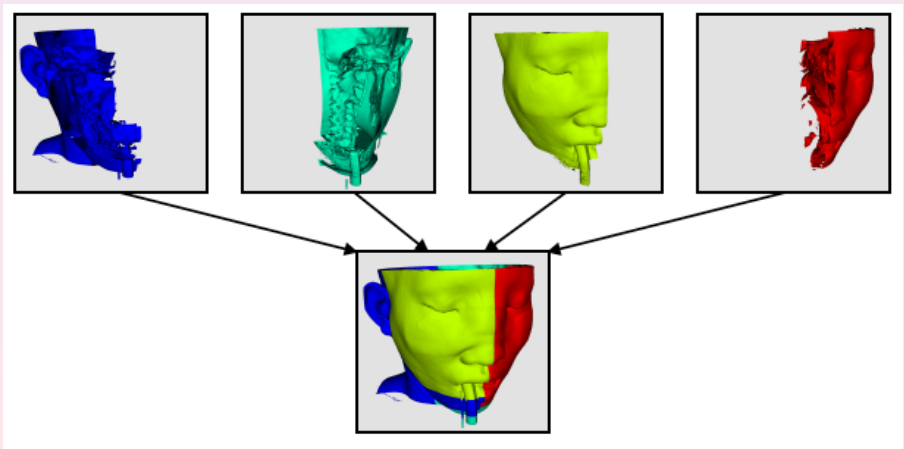
Parallelism

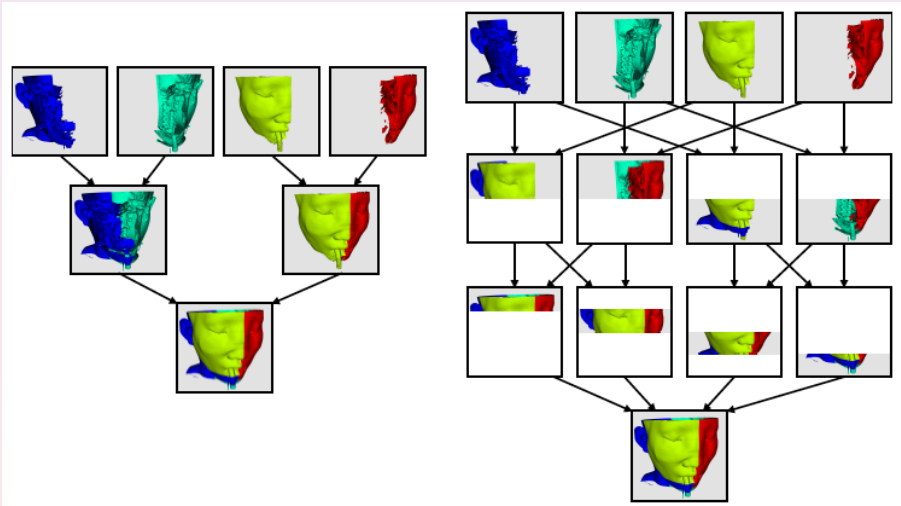
- Client-Server models: Parallel-Data & Rendering



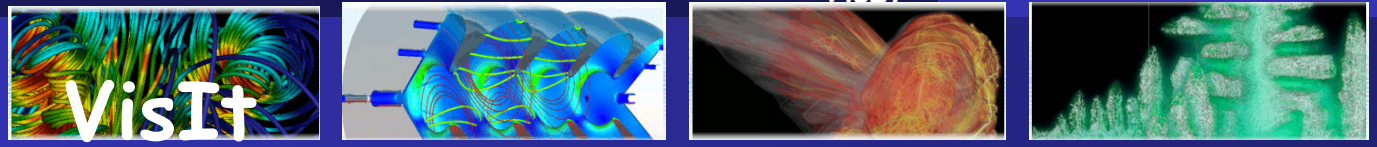






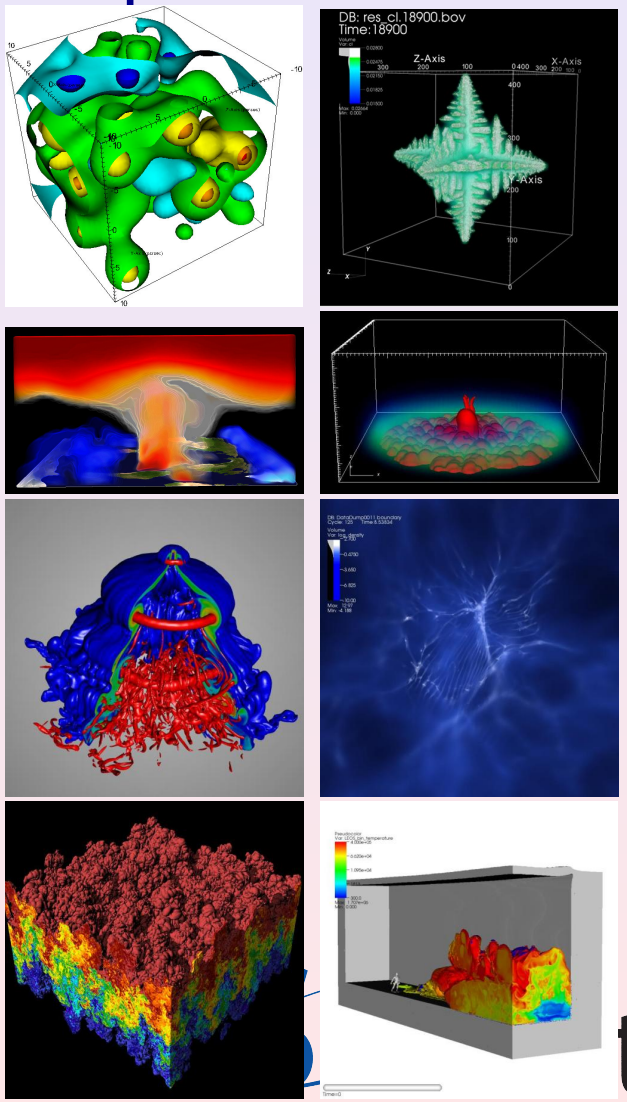


- File → **Connect...** / Disconnect

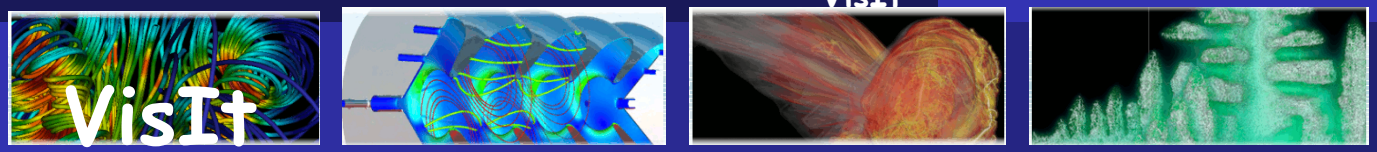


https://wci.llnl.gov/simulation/computer-codes/visit

- Developed by the *DOE Advanced Simulation and Computing Initiative*, to visualize results of terascale simulations, first release fall of 2002 - mantained by *LLNL*
- v2.10.3 available as source and binary for Linux/Mac/Windows
- Over 80 visualization features (contour, mesh, slice, volume, molecule, ...)
- Reads over 110 different file formats
- Interfaces with C++, Python, and Java
- Uses MPI for distributed-memory parallelism on HPC clusters

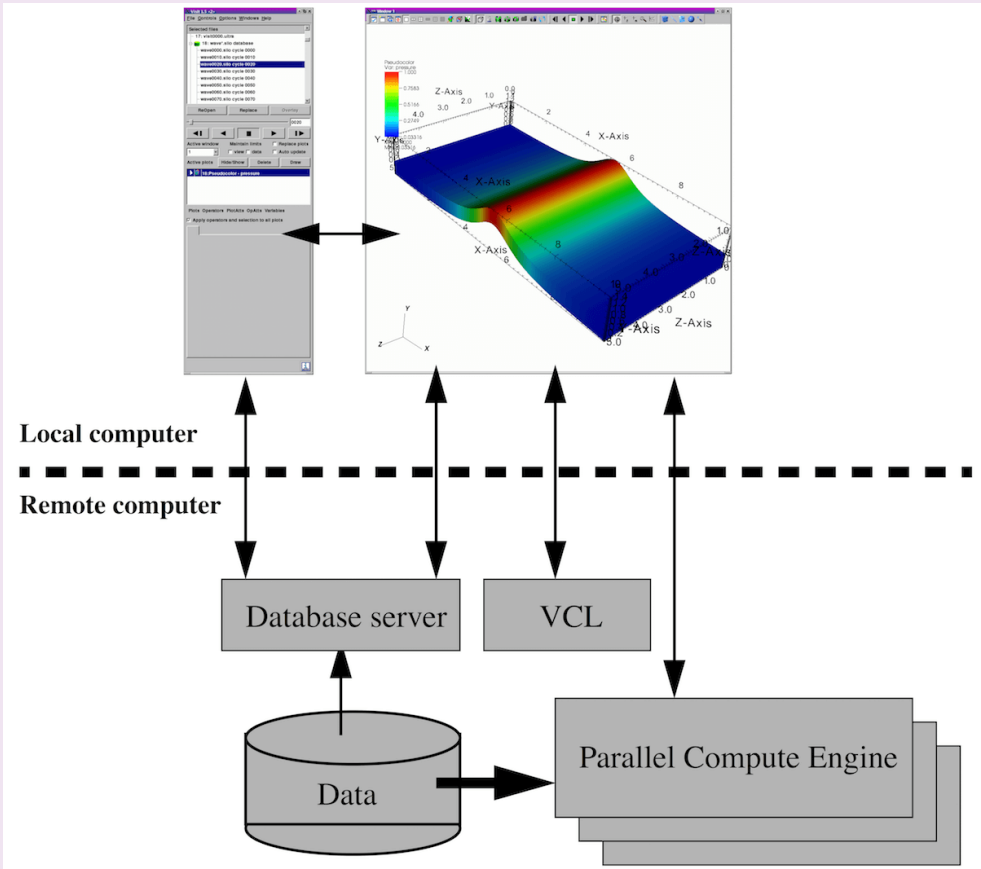


ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO



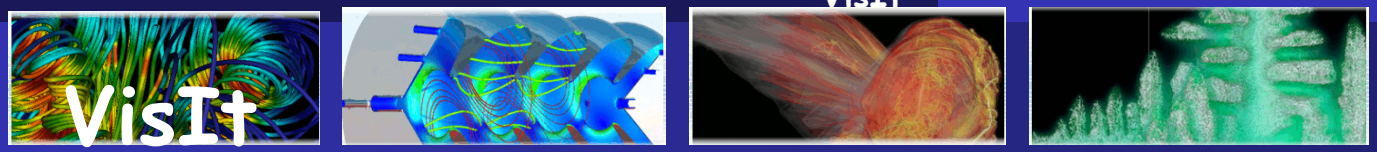
- can run: locally, remotelly, client/server mode
- interface pretty much looks the same on each platform
- can read over a 100 different data formats
- new database plugin readers can be developed

VisIt Architecture



Supported Mesh Types

- 1D Curves
- 2D/3D meshes: Rectilinear, Curvilinear, Unstructured, Points, AMR, Molecular, CSG

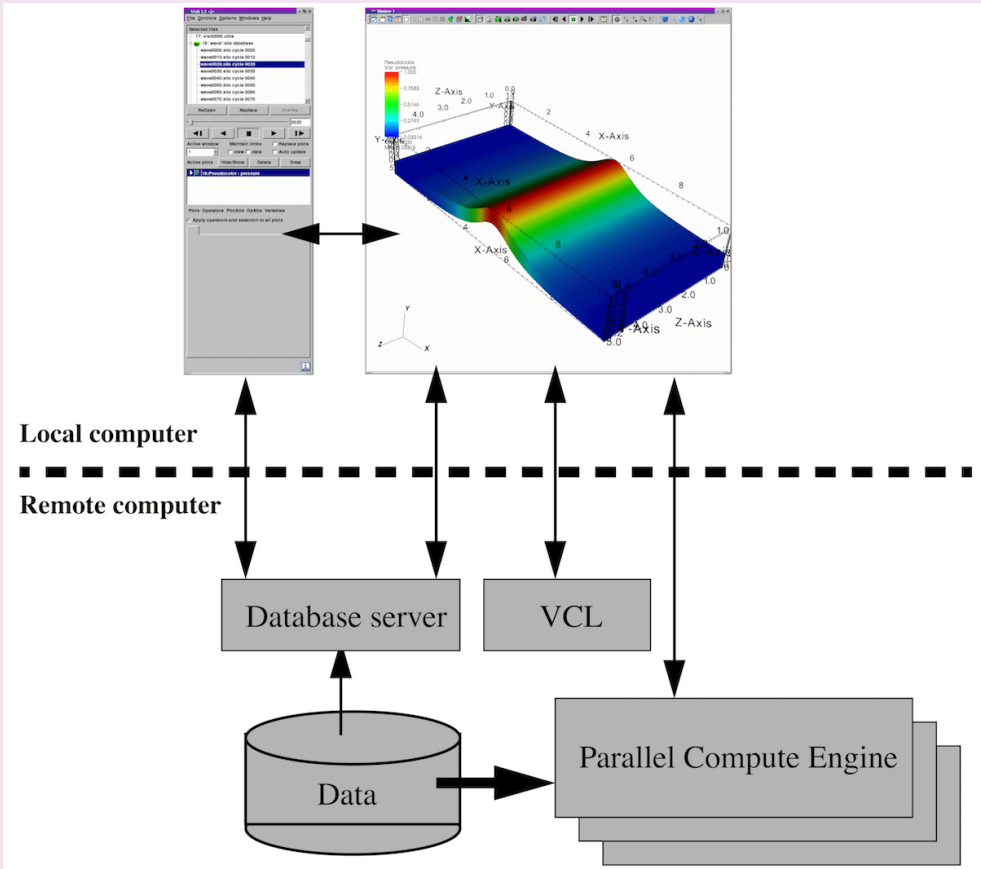


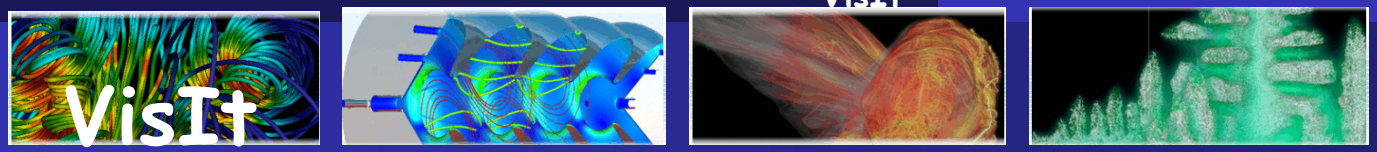
- can run: locally, remotelly, client/server mode
- interface pretty much looks the same on each platform
- can read over a 100 different data formats
- new database plugin readers can be developed

➡ Supported Mesh Types

- 1D Curves
- 2D/3D meshes: Rectilinear, Curvilinear, Unstructured, Points, AMR, Molecular, CSG

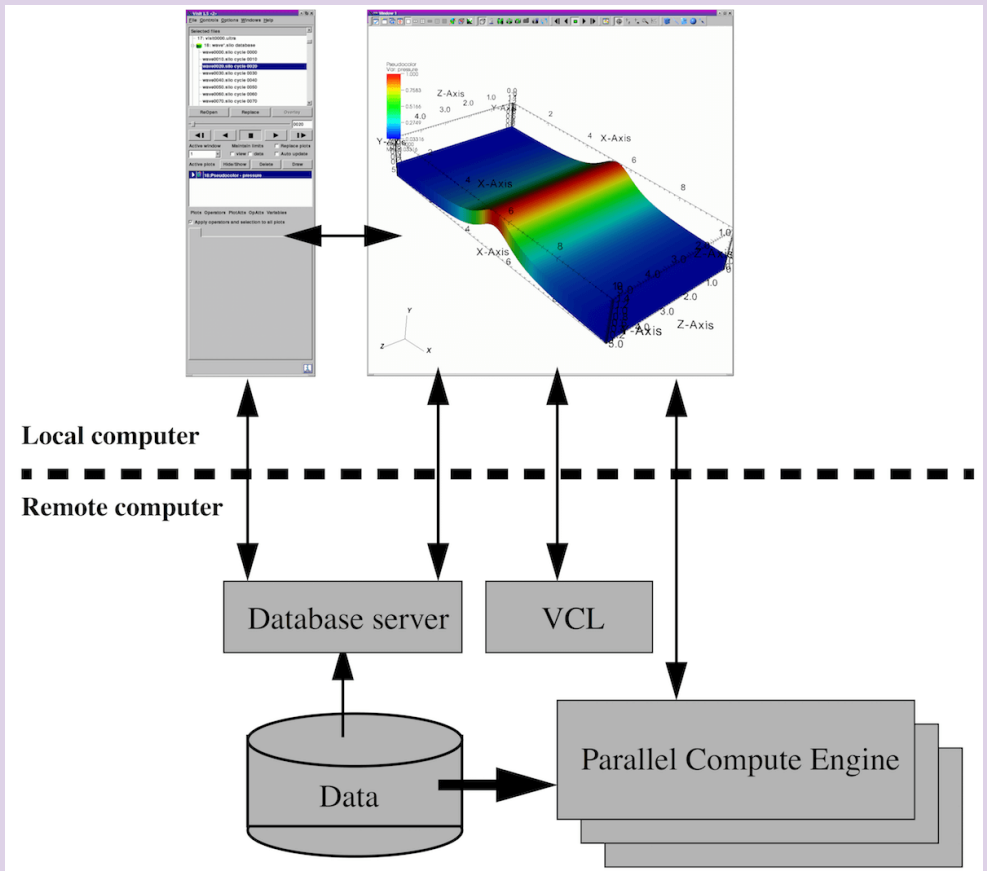
➡ VisIt Architecture





- can run: locally, remotelly, client/server mode
- interface pretty much looks the same on each platform
- can read over a 100 different data formats
- new database plugin readers can be developed

➡ VisIt Architecture



➡ Supported Mesh Types

- 1D Curves
- 2D/3D meshes: Rectilinear, Curvilinear, Unstructured, Points, AMR, Molecular, CSG

VisIt: Multiple Interfaces

- GUI (graphical user interface)
- Python programming interface
- Java programming interface
- C++ programming interface

- Use multiple interfaces simultaneously
- Use VisIt as an application or a library
- C++, Python, Java interfaces allow other applications to control VisIt

VisIt: Multiple Interfaces

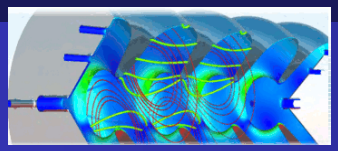
- GUI (graphical user interface)
- Python programming interface
- Java programming interface
- C++ programming interface

◗ Use multiple interfaces simultaneously

➡ Use VisIt as an application or a library

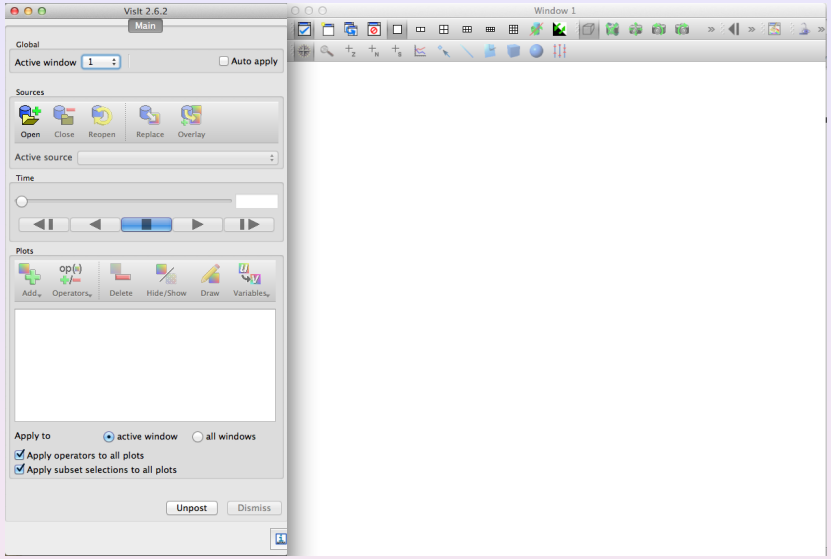
➡ C++, Python, Java interfaces allow other applications to control VisIt

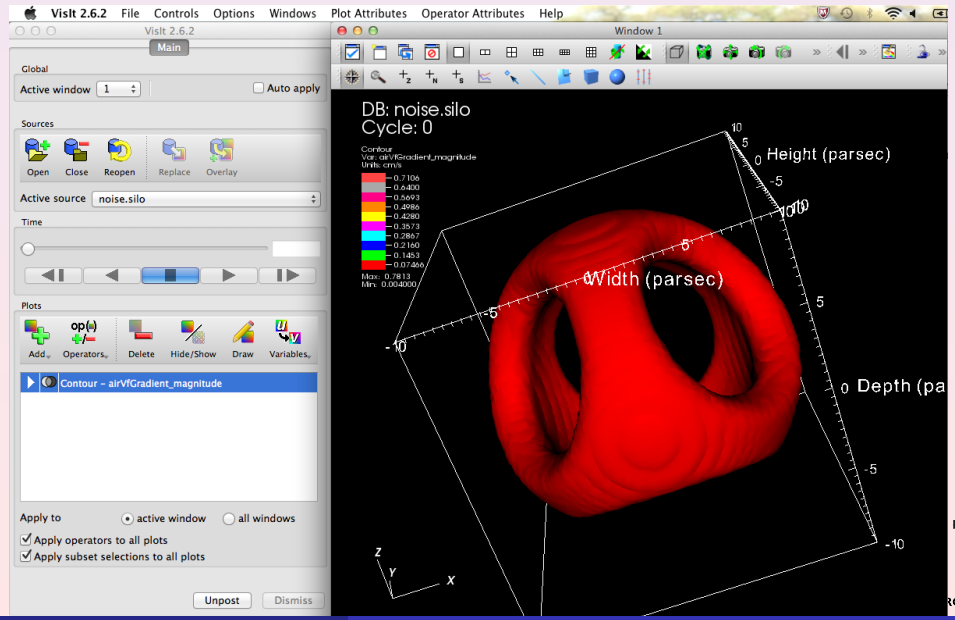
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

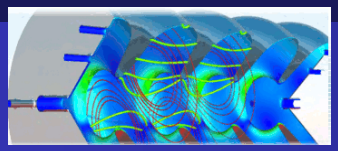


VisIt: GUI

- GUI
  - Select files to visualize
  - Create and manage plots
  - Set plot attributes
  - Add operators
  - Set look and props. for visualization
- Viewer
  - display all of the data being visualized
  - Mouse navigation
  - up to 16 vis windows
  - Popup menu
  - Toolbars

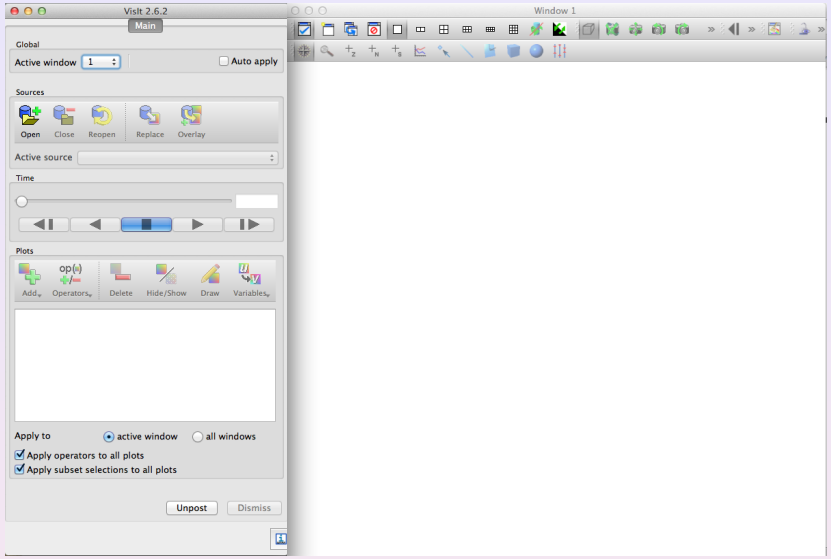


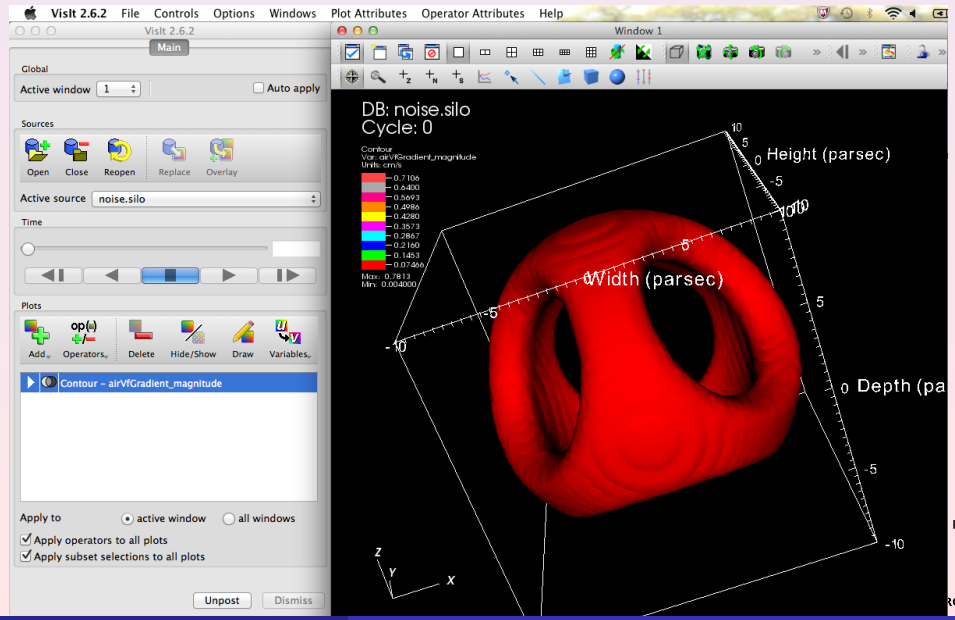


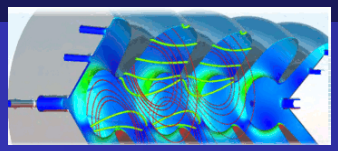


VisIt: GUI

- GUI
  - Select files to visualize
  - Create and manage plots
  - Set plot attributes
  - Add operators
  - Set look and props. for visualization
- Viewer
  - display all of the data being visualized
  - Mouse navigation
  - up to 16 vis windows
  - Popup menu
  - Toolbars

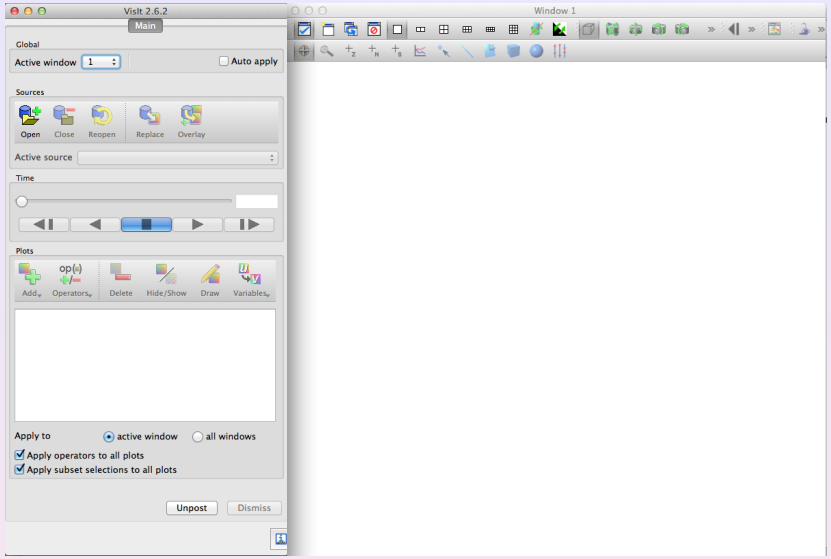


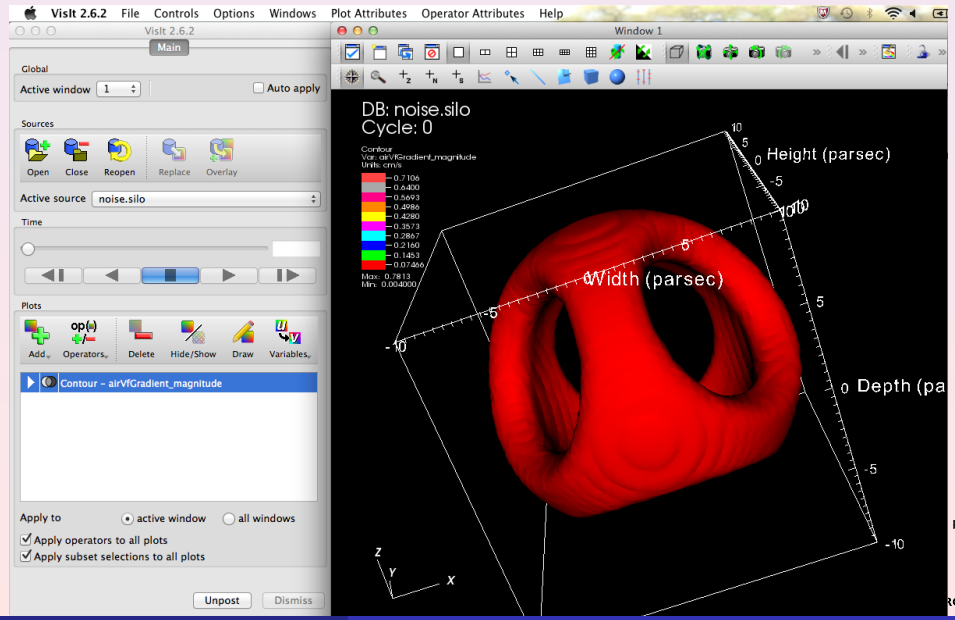


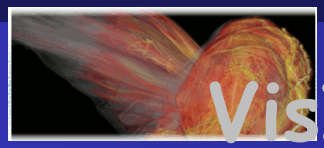


VisIt: GUI

- GUI
  - Select files to visualize
  - Create and manage plots
  - Set plot attributes
  - Add operators
  - Set look and props. for visualization
- Viewer
  - display all of the data being visualized
  - Mouse navigation
  - up to 16 vis windows
  - Popup menu
  - Toolbars

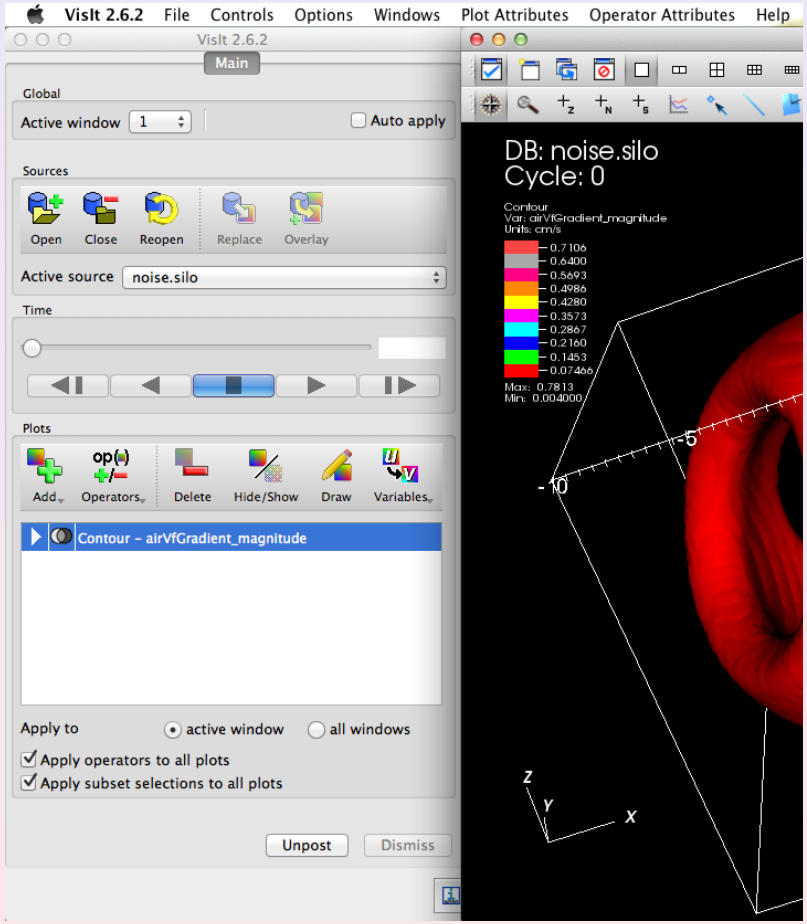


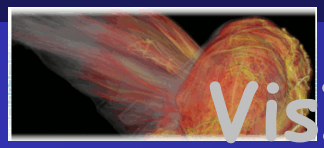




VisIt: GUIs

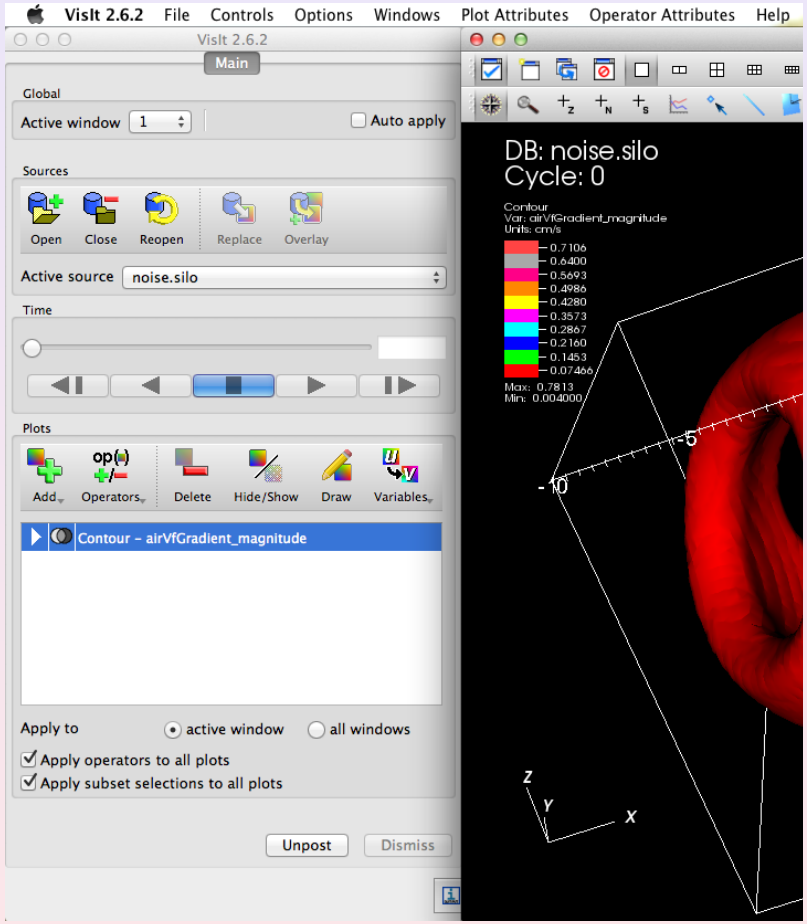
- Main window in GUI
  - Access other important windows
  - Open files
  - Set animation time state
  - Set active window
  - Create and manage filters (pipeline)
  - Displays progress from compute engine
- Really useful ones
  - Appy to... check-boxes



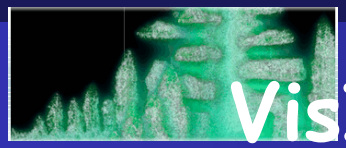


VisIt: GUIs

- Main window in GUI
  - Access other important windows
  - Open files
  - Set animation time state
  - Set active window
  - Create and manage filters (pipeline)
  - Displays progress from compute engine
- Really useful ones
  - Appy to... check-boxes



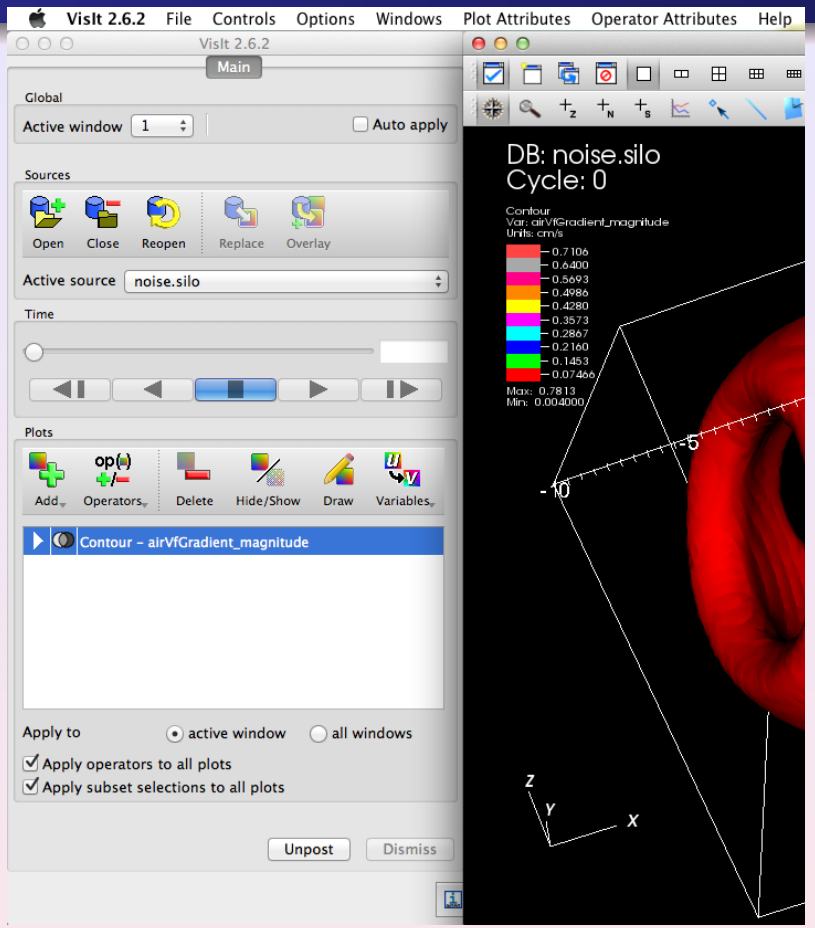
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO



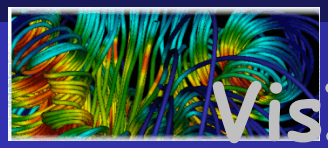
VisIt: GUIs

Main Menu

- File
- Controls
- Options
- Windows
- Plot Attributes
- Operator Attributes
- Help

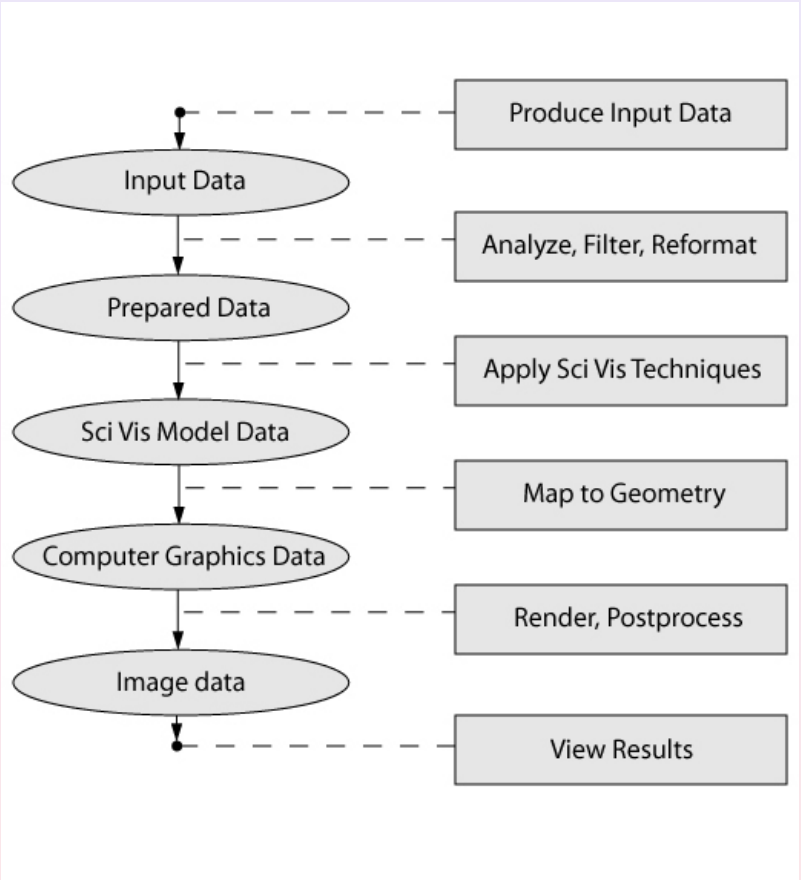


Visualization Toolbar

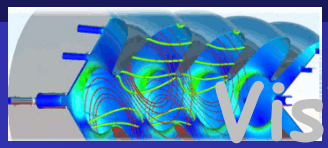


VisIt: Visualization Pipeline

1. Open database (or file)
2. Create a plot
3. Set plot attributes
4. Apply operators to plot to modify data
5. Set operator attributes
6. Compute engine generates plot
7. Plot displayed in vis window
8. iterate//repeat...

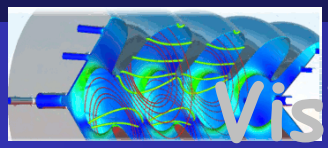


SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO



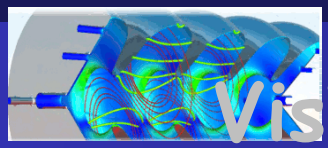
VisIt: Import Data

- File → **Open file...**
  - ↦ choose data file ("noise.silo")
- Become available
  - Active source: "noise.silo"
  - Add button
- File → File information...



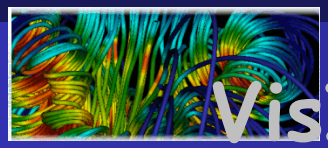
VisIt: Import Data

- File → **Open file...**
  - ↦ choose data file ("noise.silo")
- Become available
  - ➡ Active source: "noise.silo"
  - ➡ Add button
- File → File information...

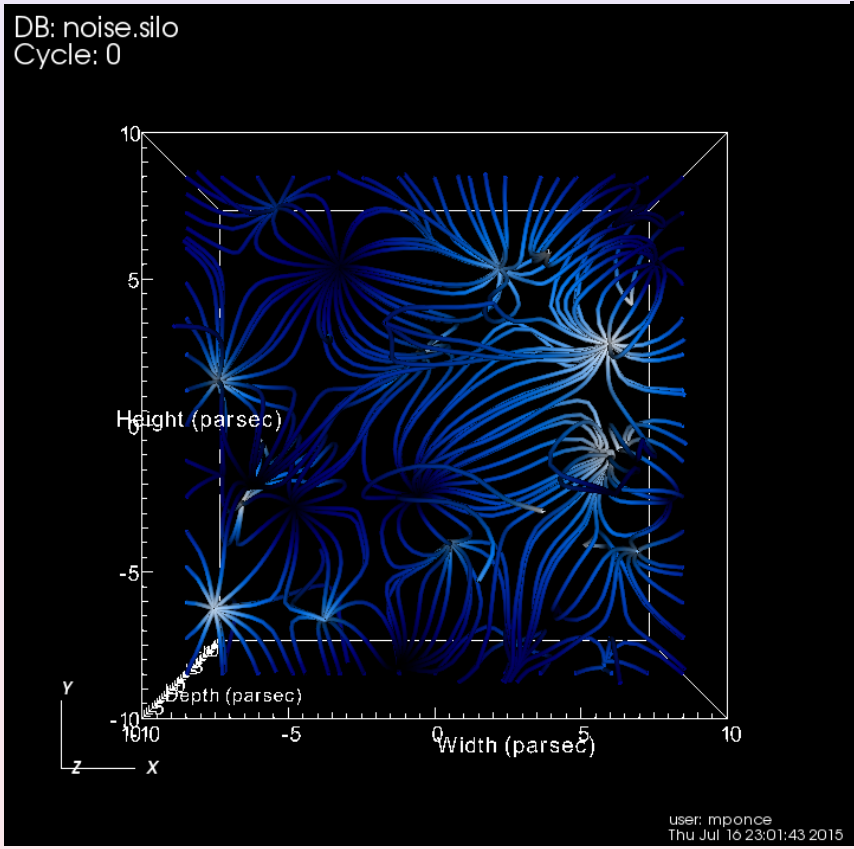


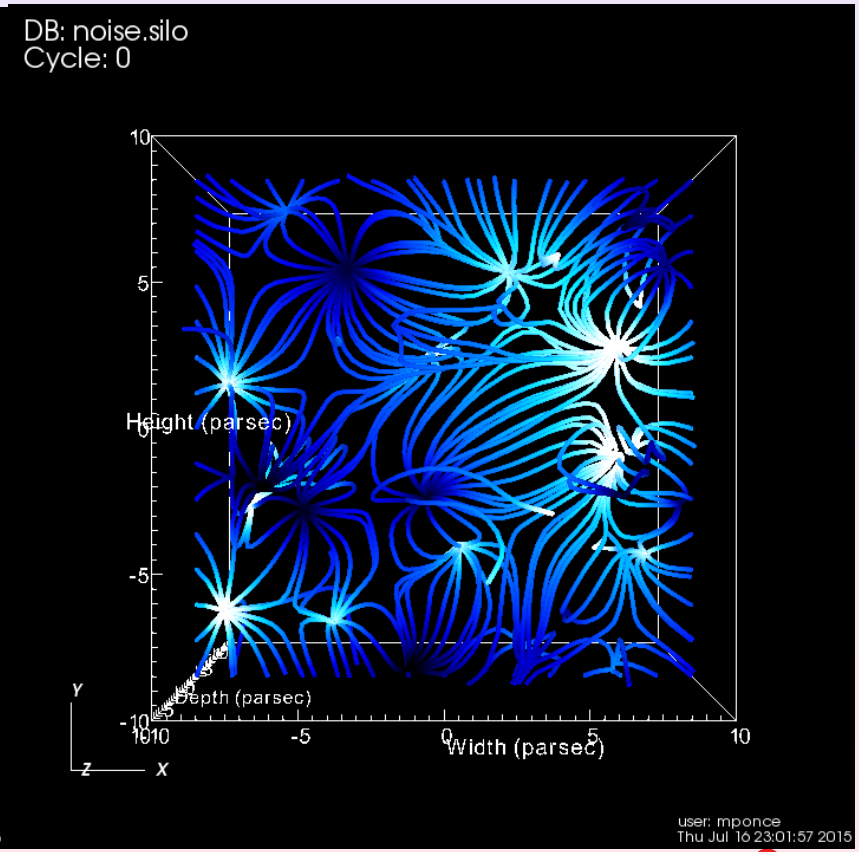
VisIt: Import Data

- File → **Open file...**
  - ↦ choose data file ("noise.silo")
- Become available
  - Active source: "noise.silo"
  - Add button
- File → **File information...**

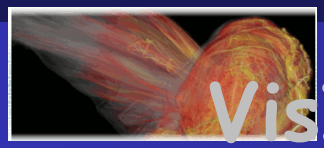


VisIt: Vector Field representations - Streamlines



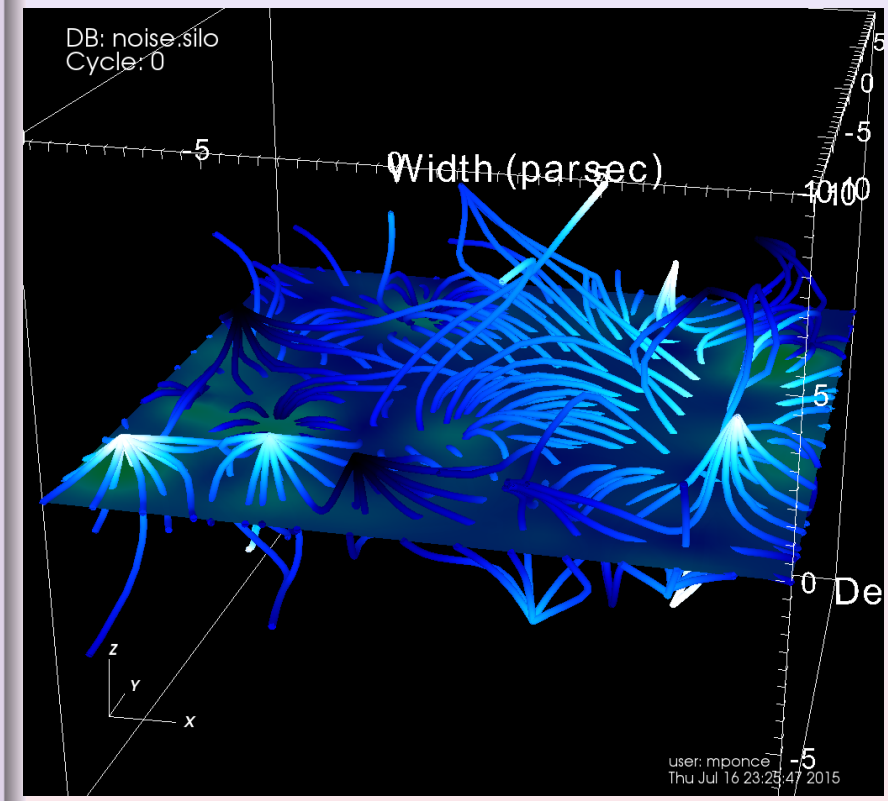


SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

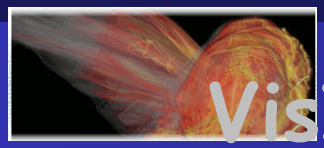


VisIt: Slices

- Add ▾ → Pseudocolor
  - ⇝ grad_magnitude
- Operator ▾ → Slice ⇝ Slicing
- double-click on [Slice]
  - select ⇝ Z-axis
  - unselect Project to 2D
  - Apply
  - Draw
  - Hide/Show

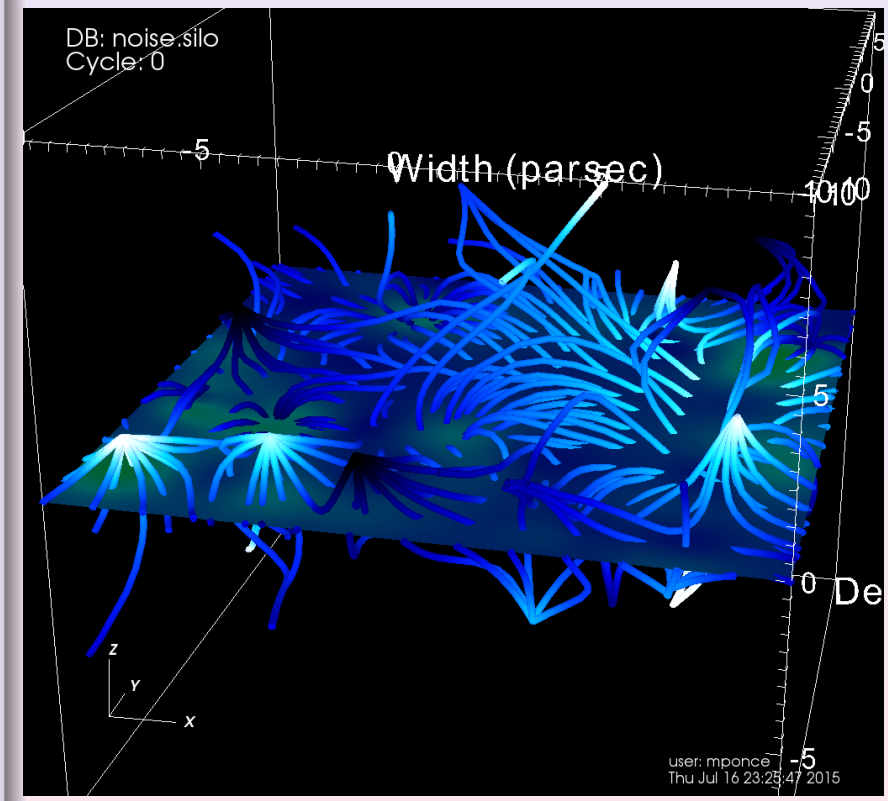


SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

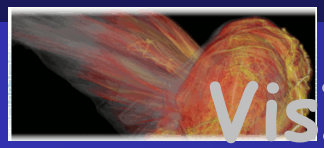


VisIt: Slices

- Add ▾ → Pseudocolor
  - ⇝ grad_magnitude
- Operator ▾ → Slice ⇝ Slicing
- double-click on [Slice]
  - select ⇝ Z-axis
  - unselect Project to 2D
  - Apply
  - Draw
  - Hide/Show

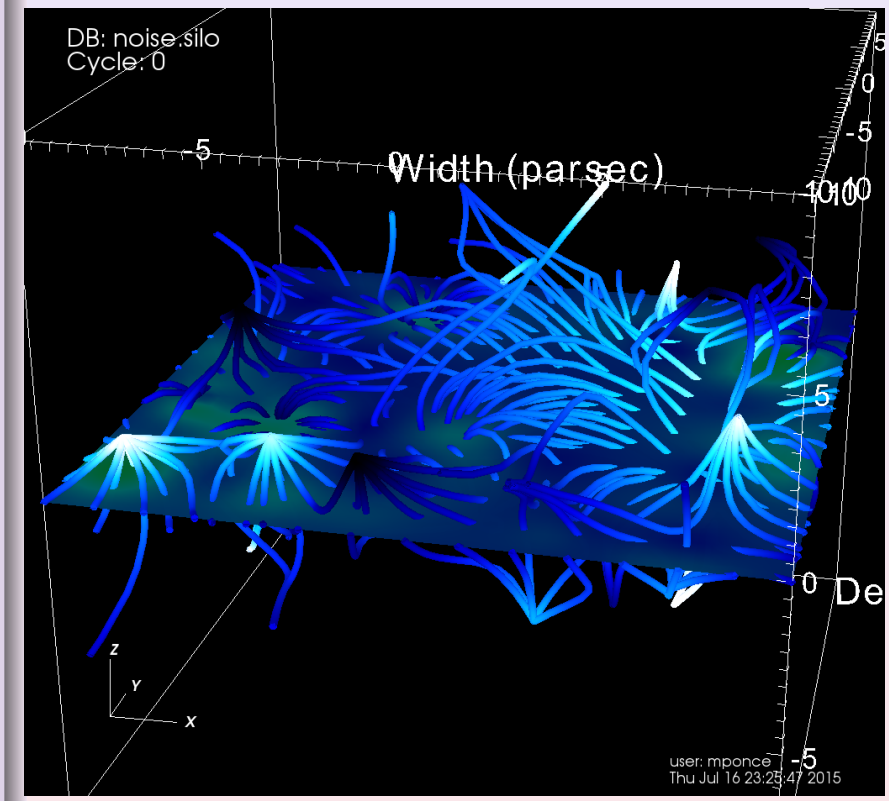


SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

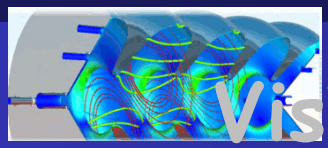


VisIt: Slices

- Add ▾ → Pseudocolor
  - ⇝ grad_magnitude
- Operator ▾ → Slice ⇝ Slicing
- double-click on [Slice]
  - select ⇝ Z-axis
  - unselect Project to 2D
  - Apply
  - Draw
  - Hide/Show

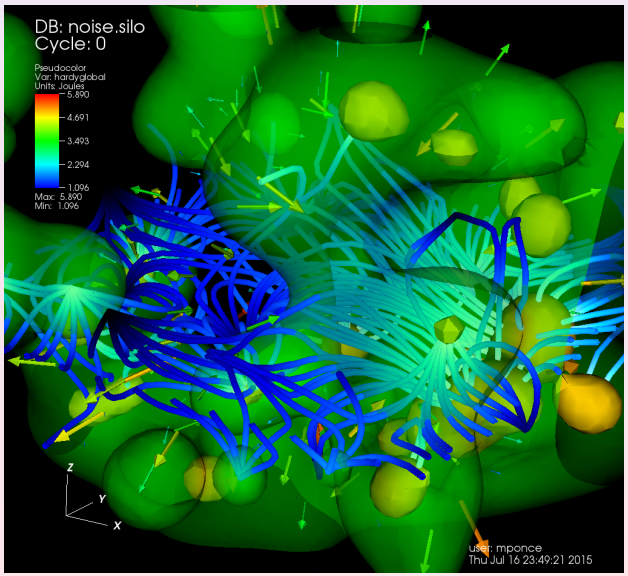


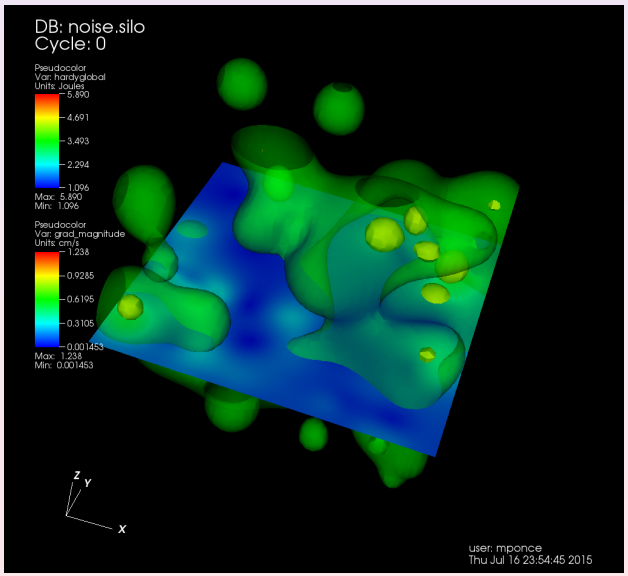
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

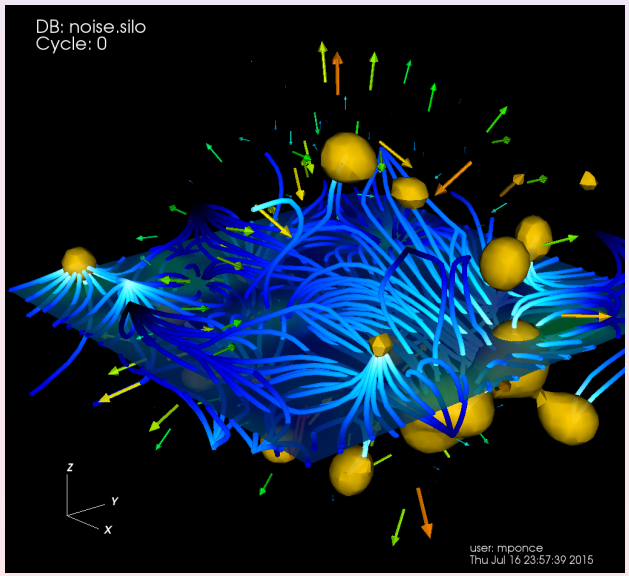


VisIt: Aesthetics & Final Products

- Legends, axes, ...
  - Controls → Annotantion...
- "hard-copies": images, ...
  - File → Save Window

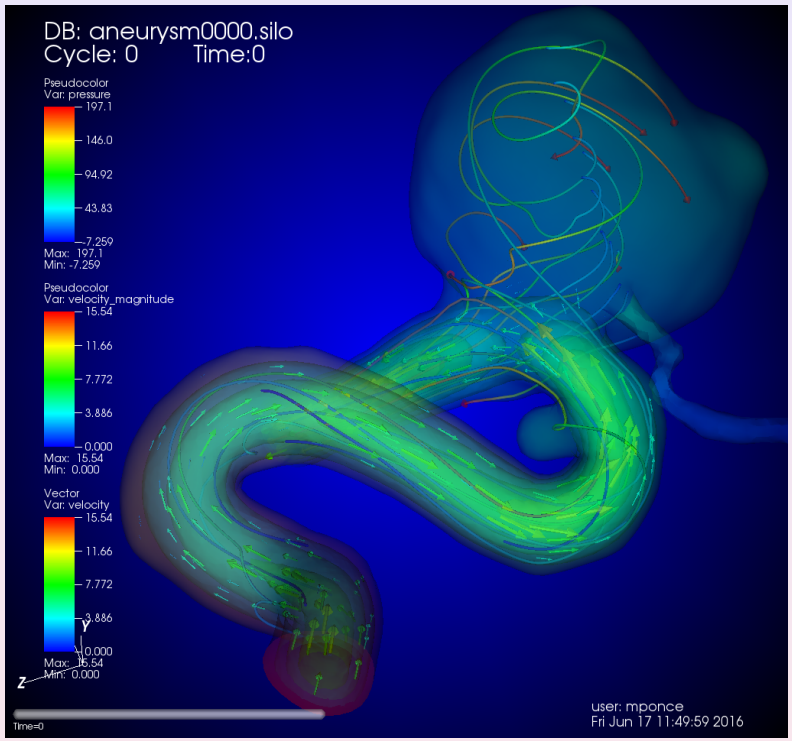






SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

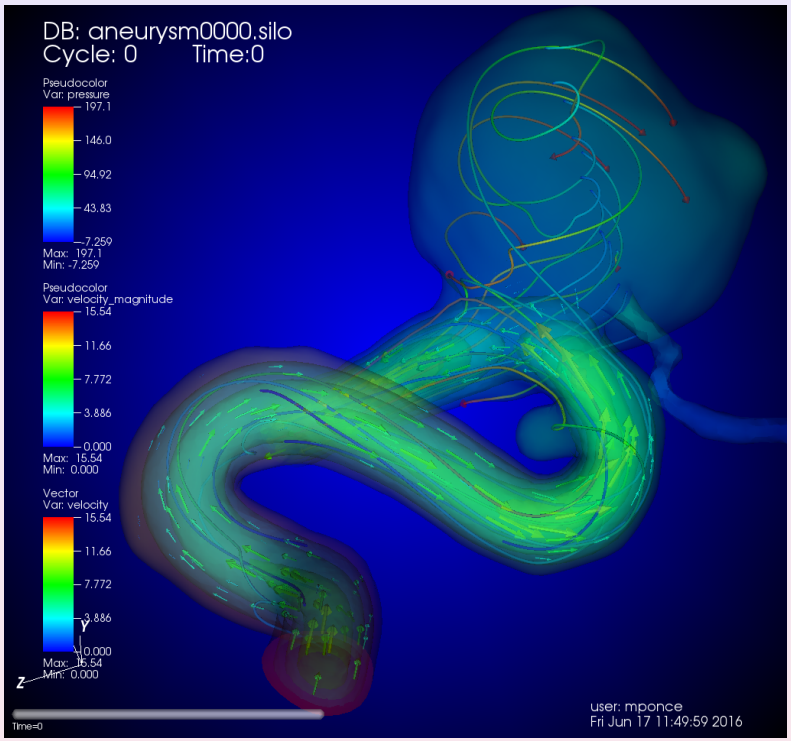
Aneurism



More info about this dataset at:

http://www.visitusers.org/index.php?title=Blood_Flow_Aneurysm_Tutorial_Dataset_Exploration

Aneurism



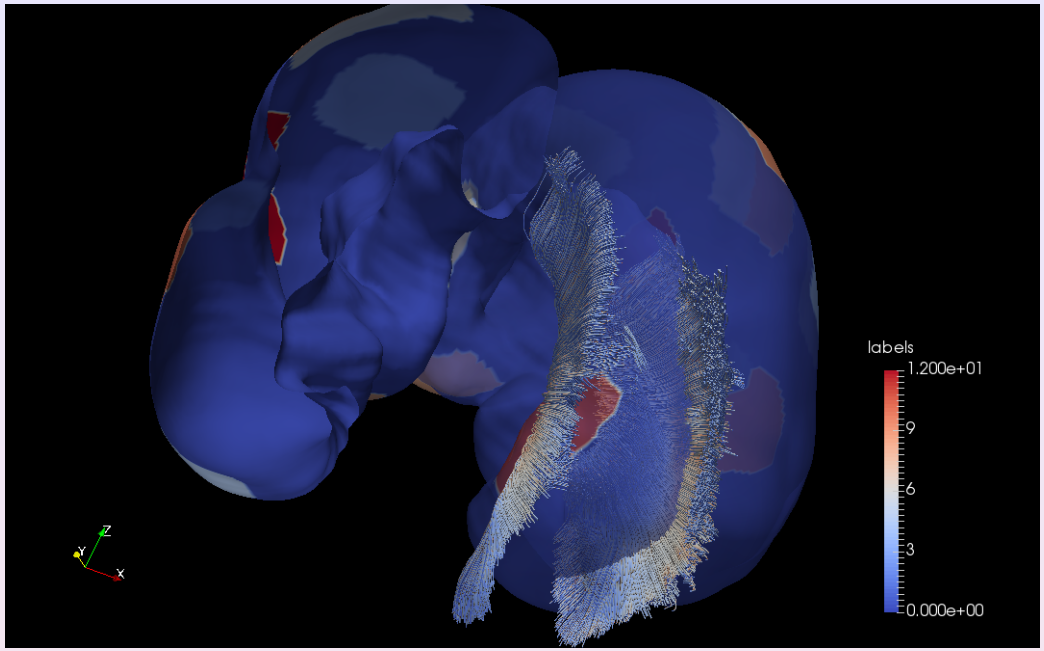
More info about this dataset at:

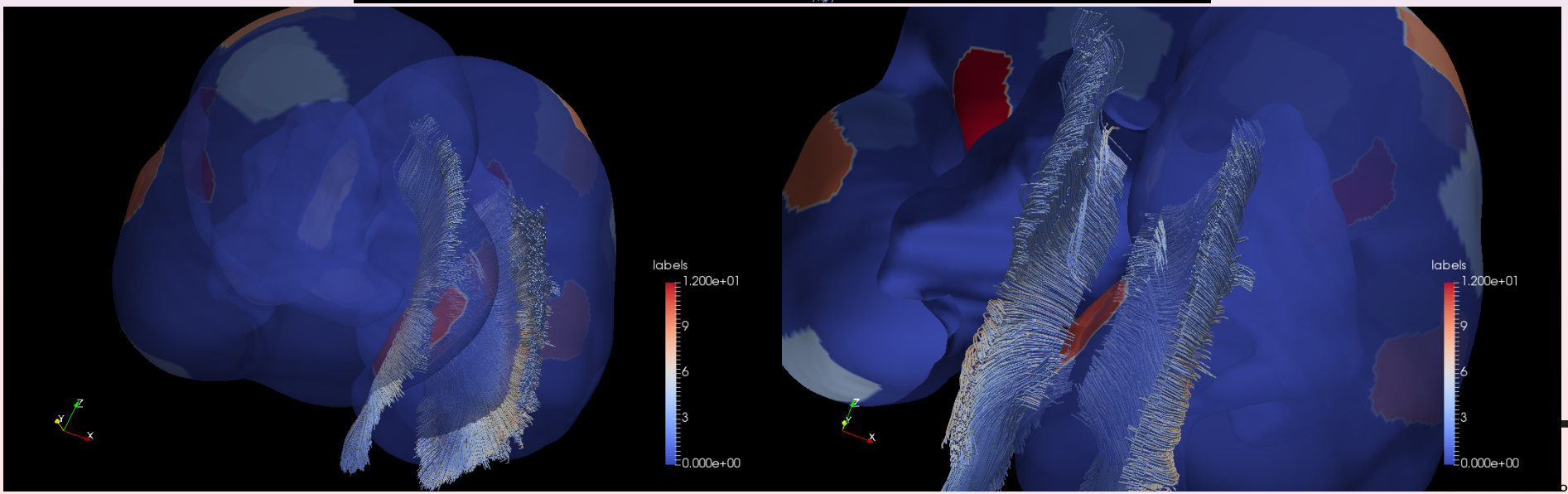
http:

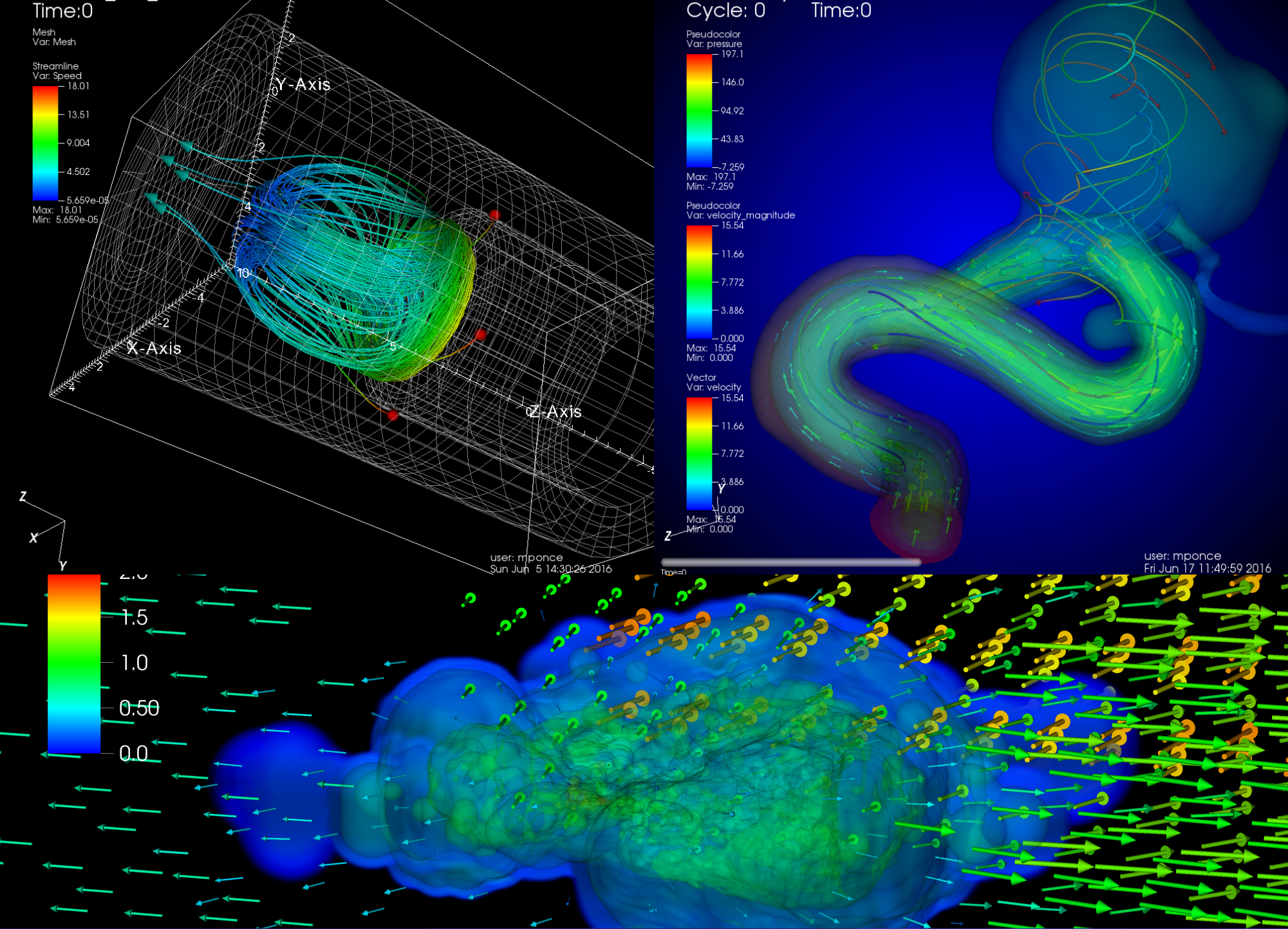
//www.visitusers.org/index.php?title=Blood_Flow_Aneurysm_Tutorial_Dataset_Exploration

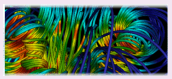
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Brain topology and neuro-paths...






Time:0
Mesh
Var: Mesh
Streamline
Var: Speed
18.01
13.51
9.004
4.502
5.659e-05
Max: 18.01
Min: 5.659e-05
Y-Axis
X-Axis
Z-Axis
user: mponce
Sun Jun 5 14:30:26 2016
Cycle: 0
Time:0
Pseudocolor
Var: pressure
197.1
146.0
94.92
43.83
-7.259
Max: 197.1
Min: -7.259
Pseudocolor
Var: velocity_magnitude
15.54
11.66
7.772
3.886
0.000
Max: 15.54
Min: 0.000
Vector
Var: velocity
15.54
11.66
7.772
3.886
0.000
Max: 15.54
Min: 0.000
user: mponce
Fri Jun 17 11:49:59 2016
1.5
1.0
0.50
0.0

 VisIt

- Website:

  https://wci.llnl.gov/simulation/computer-codes/visit/

  https://wci.llnl.gov/codes/visit/
- Documentation:
  https://wci.llnl.gov/simulation/computer-codes/visit/manuals
- Gallery:

  https://wci.llnl.gov/simulation/computer-codes/visit/gallery
- Visit Users wiki:
  http://www.visitusers.org
- Tutorials:

  http://www.visitusers.org/index.php?title=VisIt_Tutorial
- Examples Datasets:
  http://www.visitusers.org/index.php?title=Tutorial_

ParaView

- Tutorial,
  http://www.paraview.org/Wiki/The_ParaView_Tutorial
- ParaView official documentation,
  http://www.paraview.org/documentation
- ParaView gallery,
  http://www.paraview.org/gallery
- ParaView wiki,
  http://www.paraview.org/Wiki/ParaView
- ParaView Python scripting,
  http://www.paraview.org/Wiki/ParaView/Python_Scripting
- using ParaView on SciNet

Others...

SciNet

- Visualization Sessions at the 2016 HPC-SummerSchool
- Tutorial on Scientific Visualization using VisIt
- Visualization Software @ SciNet
- Visualization Material
- VNC
- https://wiki.scinet.utoronto.ca/wiki/index.php/Visualization_Nodes
- **https://support.scinet.utoronto.ca/education**
- **https://wiki.scinet.utoronto.ca/**

- Visualization Support
  - https://docs.computecanada.ca/wiki/Visualization
  - https://www.computecanada.ca/research-portal/national-services/visualization/
  - **vis-support@computecanada.ca**
  - **support@scinet.utoronto.ca**

X Forwarding

- Remote graphics

ssh X forwarding - if an X server has been installed locally (for Linux and MacOS this is often already there by default)

```
$ ssh -X login.scinet.utoronto.ca
$ ssh -X viz01          #gpc0x
```

This can be slow, depending on various factors, eg. low-bandwidth/high-latency connections (such as, home internet connections).

➡ Tunneling: allows you to keep *forwarding* through several consecutive connections by carrying on the "-X"

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

X Forwarding

- Remote graphics

ssh X forwarding - if an X server has been installed locally (for Linux and MacOS this is often already there by default)

```
$ ssh -X login.scinet.utoronto.ca
$ ssh -X viz01          #gpc0x
```

This can be slow, depending on various factors, eg. low-bandwidth/high-latency connections (such as, home internet connections).

- Tunneling: allows you to keep *forwarding* through several consecutive connections by carrying on the "-X"

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

X Forwarding

- Remote graphics

ssh X forwarding - if an X server has been installed locally (for Linux and MacOS this is often already there by default)

```
$ ssh -X login.scinet.utoronto.ca
$ ssh -X viz01          #gpc0x
```

This can be slow, depending on various factors, eg. low-bandwidth/high-latency connections (such as, home internet connections).

- Tunneling: allows you to keep *forwarding* through several consecutive connections by carrying on the "-X"

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

X Forwarding

- Remote graphics

ssh X forwarding - if an X server has been installed locally (for Linux and MacOS this is often already there by default)

```
$ ssh -X login.scinet.utoronto.ca
$ ssh -X viz01          #gpc0x
```

This can be slow, depending on various factors, eg. low-bandwidth/high-latency connections (such as, home internet connections).

➡ Tunneling: allows you to keep *forwarding* through several consecutive connections by carrying on the "**-X**"

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Virtual Network Computing

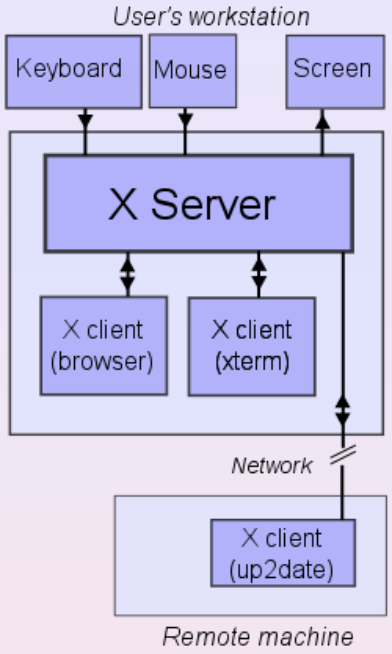
- VNC: Virtual Network Computing
- VNC behaves as if taking continuous desktop snapshots
- it uses compression techniques to reduce the required bandwidth, and trasnfers only the parts of the desktop that have changed
- using VNC with an SSH tunnel and a password is quick and secure
- the connection procedure is a bit more involve than tunneling but it may be worth it

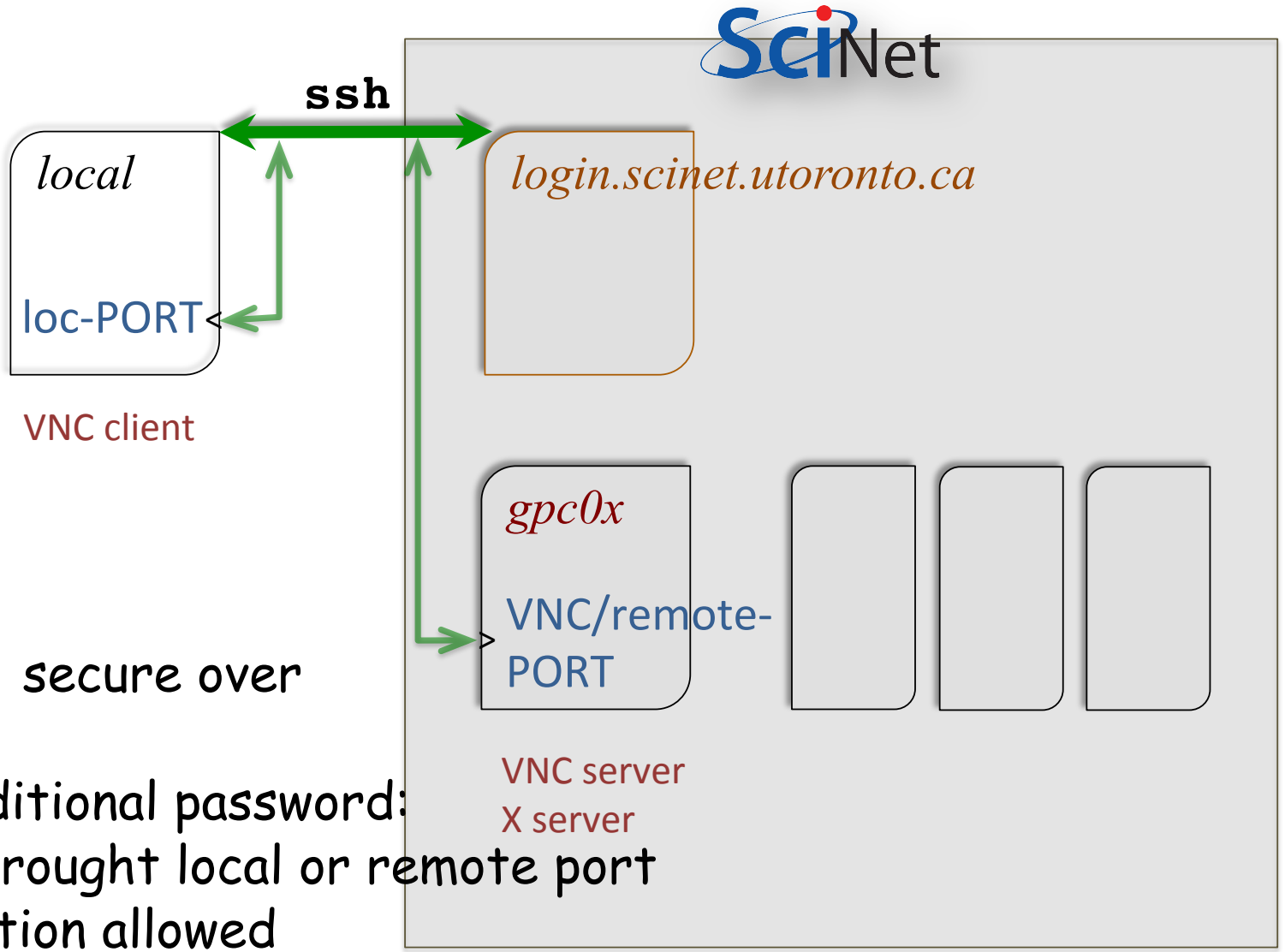
VNC vs X tunneling

- X tunneling will work and be just fine in many cases
- VNC offers a potentially more suitable protocol for such remote connections
- Remote X graphics applications require a local X server, and transmit many little events and data messages. On a network with high latency, the number of roundtrips needed makes X slow and less responsive.
- X's speed depends more on the type of application than VNC (eg. java applications tend to be very slow over X, but are OK over VNC).
- VNC typically requires fewer roundtrip, hence is often more responsive.

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

How does it work?





- ssh encrypts: secure over network
- VNC uses additional password: no exposure throught local or remote port
- only 1 connection allowed
  - ➡ a high level of security

VIZ-nodes @ SciNet

- two dedicated visualization nodes
  - **viz01**, aimed for interactive use (eg. via X-forwarding or VNC)
  - **viz02**, aimed for batch proccessing, accessible through the **viz**-queue

https://wiki.scinet.utoronto.ca/wiki/index.php/Visualization_Nodes

- **two quad** core Intel Xeon X5550 2.67GHz CPUs: 8 cores/per-node
- 64GB of RAM per node
- **two** NVIDIA Tesla M2070 **GPUs** with CUDA Capability 2.0 (Fermi) each with 448 CUDA Cores@1.15GHz and 6GB of RAM
- DDR Infiniband interconnection
- GPFS mounted
- 60GB local scratch disk (**/localscratch**)

Software Prerequisites

➡ on your *local machines*

- Install an **ssh client**
  - Linux and MacOS: come with them!
  - Windows: MobaXterm, Cygwin, PuTTY
- Install a **VNC client**
  - Linux and MacOS: most likely are there already
  - If not: TightVNC or TigerVNC are a good option

➡ on the *remote machine* (on GPC@SciNet)

- Require a **VNC server**
  - we will be using *modules*
  - module `vnc` ⇝ VNC server & scripts
  - This module requires, the `Xlibraries` module
  - `module load Xlibraries vnc`

Software Prerequisites

➡ on your *local machines*

- Install an **ssh client**
  - Linux and MacOS: come with them!
  - Windows: MobaXterm, Cygwin, PuTTY
- Install a **VNC client**
  - Linux and MacOS: most likely are there already
  - If not: TightVNC or TigerVNC are a good option

➡ on the *remote machine* (on GPC@SciNet)

- Require a **VNC server**
  - we will be using *modules*
  - module **vnc** $\rightsquigarrow$ VNC server & scripts
  - This module requires, the **Xlibraries** module
  - **module load Xlibraries vnc**

X forwarding, via the viz-nodes...

- Establish the Remote Connection with X-Forwarding

```
$ ssh -X login.scinet.utoronto.ca
$ ssh -X gpc0y                    y⇝ 1,...,8
```

one extra-step...

```
$ qsub -I -X -q viz -l nodes=1:ppn=8,walltime=
01:00:00
```

- Alternatively you can "*bypass*" the gpc-nodes, and from the login-nodes connect directly into the `viz01`-node

```
$ ssh -X viz01
```

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

Set up a VNC session

1. Start a VNC server on a compute node using **qsub** (on an usual day, this would be a GPC devel node: *gpc01-08*)
2. **ssh tunnel** from your local machine to the GPC
3. start the VNC client on local machine

Terminal 1

```
mponce~ $ ssh mponce@login.scinet.utoronto.ca
mponce@login.scinet.utoronto.ca's password:
Last login: Wed Jul  8 10:07:30 2015 from ophiuchus.scinet.utoronto.ca

==================================================

This SciNet login node is to be used only as a
gateway to the GPC and TCS.

DO NOT USE THIS NODE TO COMPILE OR RUN CODE.

Read the continually updated wiki for important
information about using these systems as well as
user FAQs - https://support.scinet.utoronto.ca/wiki

Remember that /scratch is never backed-up.

Please report any problems to <support@scinet.utoronto.ca>

**************************************************

Current messages:


**************************************************

scinet01-ib0:/home/s/scinet/mponce $ ssh gpc03Last login: Wed Ju
015 from scinet04
gpc-f103n084-ib0:/home/s/scinet/mponce $ qsub -I -q debug -l nodes=1:ppn=8,wall
time=00:30:00
qsub: waiting for job 29831175.gpc-sched-ib0 to start
qsub: job 29831175.gpc-sched-ib0 ready

----------------------------------------
Begin PBS Prologue Wed Jul  8 11:24:56 EDT 2015 1436369096
Job ID:         29831175.gpc-sched-ib0
Username:       mponce
Group:          scinet
Nodes:          gpc-f101n027-ib0
End PBS Prologue Wed Jul  8 11:24:57 EDT 2015 1436369097
----------------------------------------
gpc-f101n027-ib0:/home/s/scinet/mponce $ module load Xlibraries vnc
gpc-f101n027-ib0:/home/s/scinet/mponce $ vnc start
vncstart: Starting Xvfb
vncstart: Resolution set to 800x544x16
vncstart: Connecting x11 to vnc serverNo protocol specified
No protocol specified

vncstart: Connecting x11 to vnc server
PORT=11950
ALT.PORT=6050
```

Terminal 2

```
mponce~ $ ssh login.scinet.utoronto.ca -L15901:gpc-f101n027-ib0:11950 -N
mponce@login.scinet.utoronto.ca's password:
```

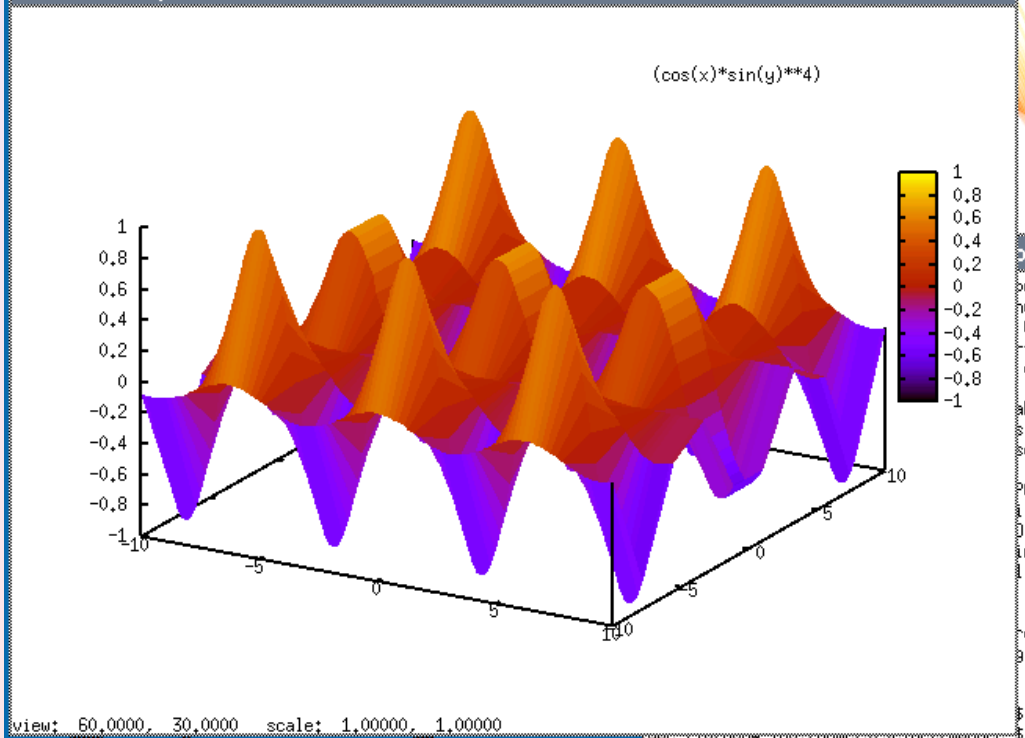
Terminal 3

```
mponce~ $ open vnc://localhost:15901
mponce~ $
```

mponce@gpc-f102n084-ib0:~

```
gpc-f102n084-ib0:/home/s/scinet/mponce $ gnuplot
bash: gnuplot: command not found
gpc-f102n084-ib0:/home/s/scinet/mponce $ module load gnuplot
gpc-f102n084-ib0:/home/s/scinet/mponce $ gnuplot

        G N U P L O T
        Version 4.6 patchlevel 1    last modified 2012-09-26
        Build System: Linux x86_64

        Copyright (C) 1986-1993, 1998, 2004, 2007-2012
        Thomas Williams, Colin Kelley and many others

        gnuplot home:     http://www.gnuplot.info
        faq, bugs, etc:   type "help FAQ"
        immediate help:   type "help"  (plot window: hit 'h')

Terminal type set to 'x11'
gnuplot> set ticslevel 0
gnuplot> splot (cos(x)*sin(y)**4) w pm3d
gnuplot>
gnuplot>
```

Gnuplot



view: 60.0000, 30.0000 scale: 1.00000, 1.00000

Allinea DDT 4.2.1-36484

File View Control Search Tools Window Help

Run
Run and debug a program.

Attach
Attach to an already running program.

Open Core
Open a core file from a previous run.

Manual Launch (Advanced)
Manually launch the backend yourself.

Options

Remote Launch:
Off

Quit

```
0:~
ublished work using VMD:
ulten, K., `VMD - Visual
Graphics 1996, 14.1, 33-38.
--------------------------
detected.

able.
ion: if incorrect display occurs
erver feature.

OT PP PS GLSL(OVF)
ilable.
) (512x512x512), Multitexture (8)
irectory:
ib/plugins/LINUXAMD64/molfile


ency,
core)with the same signal


$ module load ddt
$ ddt
```

Available Tools:

Allinea DDT Support Expires 2016-03-26

Allinea MAP Support Expires 2016-03-26

Useful Packages

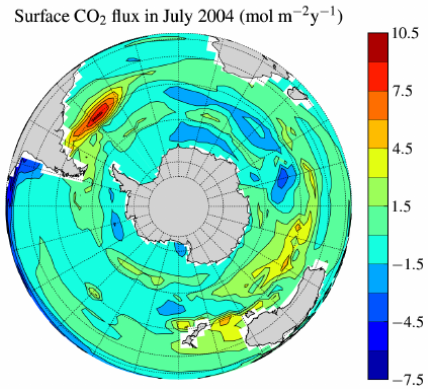
matplotlib.org

www.scipy.org

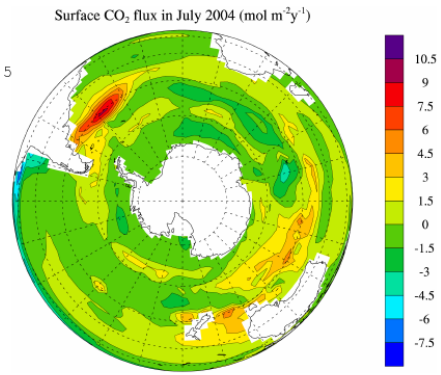
www.numpy.org

Python's Matplotlib BaseMaps

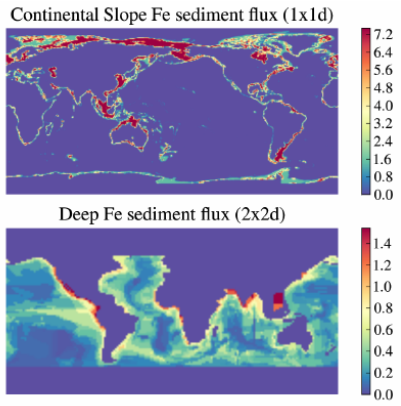
```
>>> import numpy as N
>>> import pylab as pl
>>> import netCDF4
>>> from numpy import ma as MA
>>> from matplotlib.toolkits.basemap import Basemap
>>> fpin = netCDF4.Dataset('ccsm_840-12.nc','r')
>>> tlat      = fpin.variables['TLAT'][:]
>>> tlon      = fpin.variables['TLONG'][:]
>>> fgco2     = fpin.variables['FG_CO2'][:]
>>> fillvalue = fpin.variables['FG_CO2']._FillValue
>>> fpin.close()
>>> fgco2 = MA.masked_values(fgco2,fillvalue)
>>> fgco2 = fgco2*1.e-2*1.e-3*60.*60.*24.*365.
>>> map = Basemap(projection='ortho',lat_0=-90,/
...    lon_0=0,resolution='c')
>>> x,y = map(tlon,tlat)
>>> ax  = pl.axes([0.1,0.1,0.7,0.7],axisbg='white')
>>> CS  = map.contourf(x,y,fgco2,15,cmap=pl.cm.jet)
>>> map.drawcoastlines(linewidth=0.5)
>>> map.drawmeridians(N.arange(-180,180,15))
>>> map.drawparallels(N.arange(-90,90,15))
>>> map.fillcontinents(color='lightgrey')
>>> pl.title('Surface CO$_2$ flux in July 2004 /
...    (mol m$^-2$y$^-1$)')
>>> l,b,w,h = ax.get_position()
>>> cax      = pl.axes([l+w+0.025, b, 0.025, h])
>>> pl.colorbar(CS,cax=cax,drawedges=True)
```

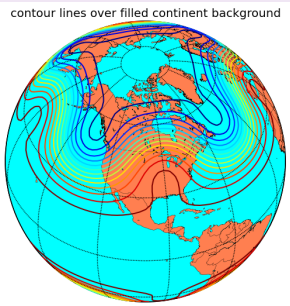


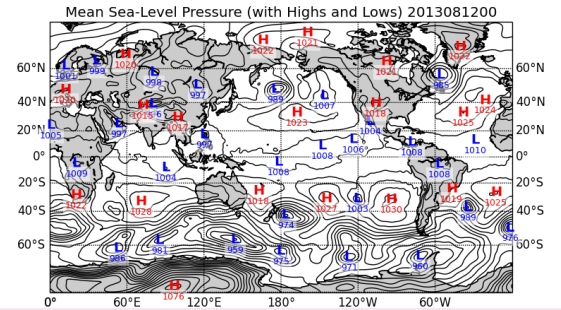

```
(import modules, read data)
>>> mres = Ngl.Resources()
>>> mres.sfXArray              = tlon[:]
>>> mres.sfYArray              = tlat[:]
>>> mres.sfMissingValueV       = fillvalue
>>> mres.cnLevelSelectionMode  = 'ExplicitLevels'
>>> mres.cnLevels              = N.arange(-7.5,11.5
>>> mres.cnLineThicknessF      = 0.5
>>> mres.pmLabelBarDisplayMode = 'Always'
>>> mres.pmTickMarkDisplayMode = 'Never'
>>> mres.mpProjection          = 'Orthographic'
>>> mres.mpMinLonF             = -180
>>> mres.mpMaxLonF             = 180
>>> mres.mpMinLatF             = -90
>>> mres.mpMaxLatF             = 90
>>> mres.mpCenterLonF          = 0
>>> mres.mpCenterLatF          = -90
>>> mres.mpGridSpacingF        = 15
>>> mres.mpPerimOn             = False
>>> mres.tiMainString          = 'Surface /
...    CO~B~2~N~ flux in July 2004 /
...    (mol m~S~- 2~N~y~S~-1~N~)'
>>> wres = Ngl.Resources()
>>> wres.wkColorMap = 'BlAqGrYeOrReVi200'
>>> wks = Ngl.open_wks('ps','example_07',wres)
>>> map = Ngl.contour_map(wks,MA.filled(fgco2,fillvalue),mres)
```

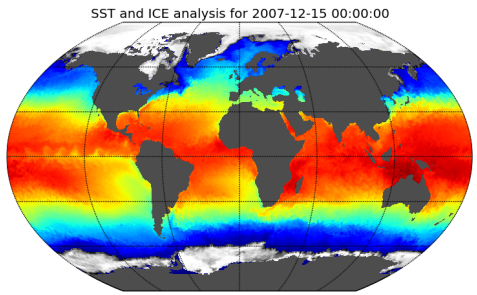


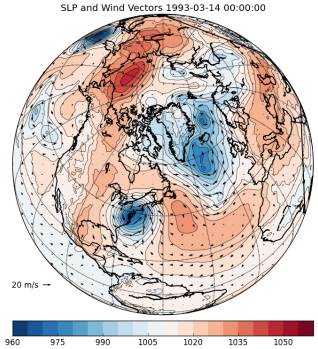

```
>>> import numpy as N
>>> import pylab as pl
>>> from scipy.io import fread, bswap, read_array
>>> fbin = open('slopeFeflux_1x1d.bin','r')
>>> feflux_slope = fread(fbin,180*360,'f')
>>> fbin.close()
>>> feflux_slope = bswap(feflux_slope)
>>> feflux_slope = N.reshape(feflux_slope,/
...     (180,360))
>>> pl.subplot(2,1,1)
>>> pl.imshow(feflux_slope,cmap=pl.cm.Spectral_r)
>>> pl.colorbar()
>>> pl.title('Continental Slope Fe sediment flux /
...     (1x1d)' ,fontsize=18)
>>> pl.axis('off')
>>> feflux_deep = read_array('deepFeflux_2x2d.txt')
>>> pl.subplot(2,1,2)
>>> pl.imshow(feflux_deep,cmap=pl.cm.Spectral_r)
>>> pl.colorbar()
>>> pl.title('Deep Fe sediment flux (2x2d)',/
...     fontsize=18)
>>> pl.axis('off')
```









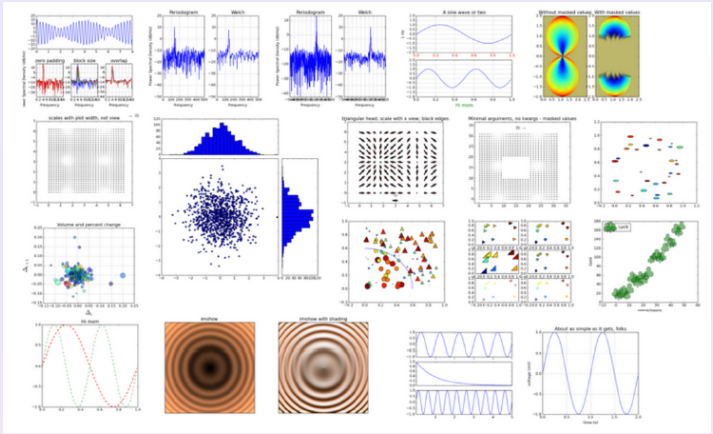


➠ matplotlib.org/basemap/examples.html

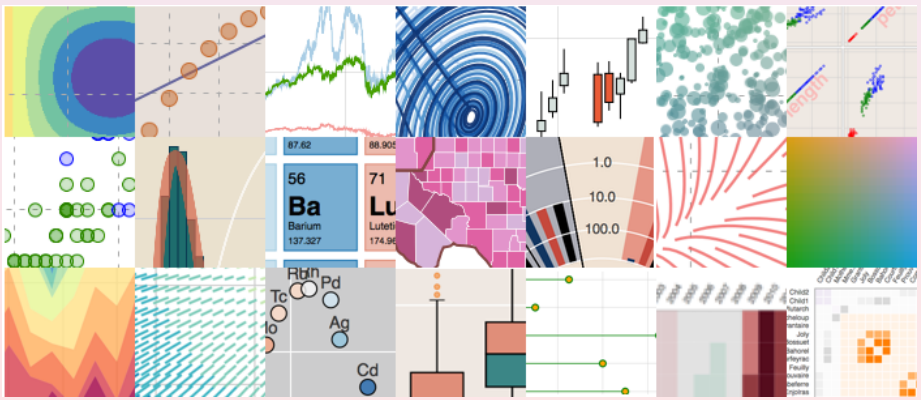
SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

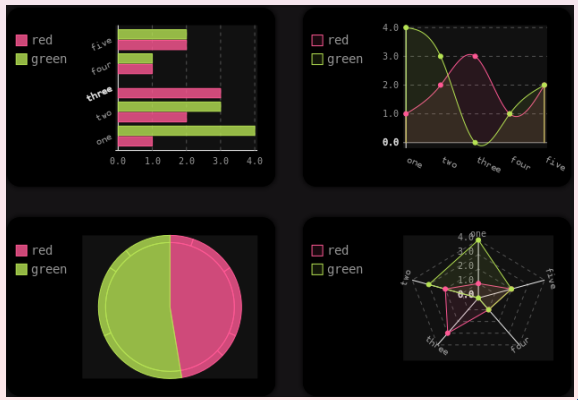
- **matplotlib**

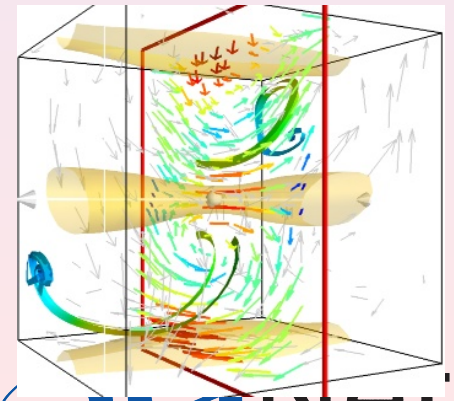
  Gallery: click on any plot to get its source code



- Tons of many others...
  - bokeh, plotly, seaborn, ggplot, pygal, vispy, mayavi, yt-project, ...
  - tkinter







Othet python graphics and visualization libraries I

bokeh

Open-source project from Continuum Analytics: **http://bokeh.pydata.org/docs/gallery.html**
Produces dynamic data visualizations in the web browser via html5

Mayavi2

3D scientific data visualizer (Python + VTK)

yt project

analysis/visualization of volumetric, multi-resolution data from astrophysical simulations (Enzo, Orion, FLASH, etc.), recently used in other fields

Neuronvision

GUI for NEURON simulator enviroment

VPython

3D graphics library

Othet python graphics and visualization libraries II

- **PyVisfile**

  storing data in a variety of scientific visualization file formats

- **PyVTK**

  tools for manipulating VTK files in Python

- **ScientificPython**

  various Python modules for scientific computing and visualization

- **Chaco**

  interactive 2D plotting

- **NodeBox**

  for OpenGL 2D animations (originally for game development)

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO

- ImageMagic
  - command line utility ➠ **convert**
- Movies' command lines utilities (linux/macOS)
  - ➠ **ffmpeg**, **mencoder**, ...
- File Formats conversion - command lines utilities (linux/macOS)
  - ➠ **epspdf**, **ps2pdf**, ...
- Libraries - for C, Fortran
  - ➠ **pgplot**, ...
- Visualization Tools available at SciNet
  - **Grace** (xmgr), **gnuplot**
  - **ParaView**, **VisIt**
  - **ImageMagick**
  - **VMD**
  - **NCL/NCARG**, **ROOT**
  - **pgplot**

compute canada | calcul canada

Portal Home

Account Management

Accessing Resources

Technical Support

National Services

Compute

Storage

Compute Canada Cloud

Data Movement (Globus)

Visualization

Research News

Grant Support

Sustainable Planning for Advanced Research in Canada (SPARC)

Feedback

RESEARCH PORTAL

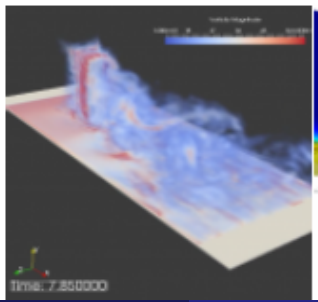
Visualization

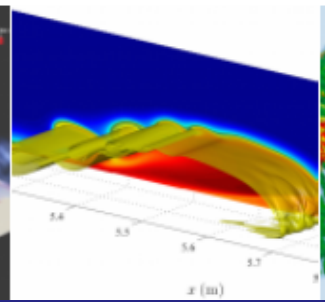
French ➤

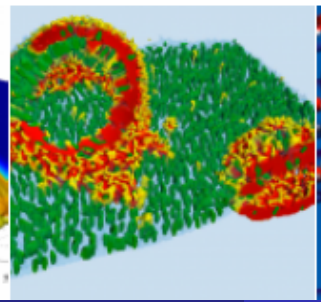
Using Compute Canada's resources and technical expert help, you can easily convert the results of your numerical simulations or your experimental dat into engaging images or movies to share with colleagues, to put online, or into a publication. Our technical staff have extensive experience in scientific visualization and visual data analysis, primarily using open-source tools such as ParaView, VisIt, VTK, Blender, VMD, and various Python libraries to wo with a wide variety of data types. Large multi-dimensional datasets can be visualized directly on Compute Canada clusters without having to move them to your desktop. We can help you with all stages of visualization, from preparing data in the right format to interactive analysis. For more information, please contact us at vis-support@computecanada.ca

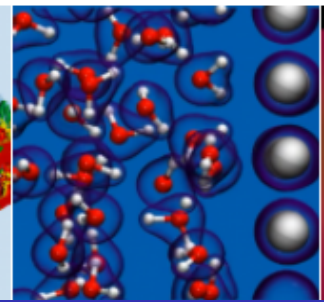
Compute Canada Visualization Working Group

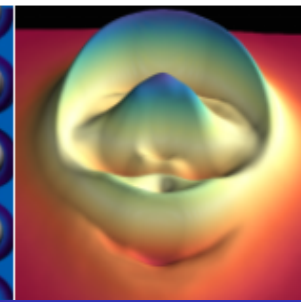
Alex Razoumov, Compute Canada (Lead)
Belaid Moa, UVic
Chris Want, Compute Canada
Dmitri Rozmanov, U of Calgary
Doug Phillips, U of Calgary
Frederick Lefebvre, Laval
Joey Bernard, UNB
Marcelo Ponce, U of Toronto
Maxime Boissonneault, Laval
Michael Hanlan, Queen's
Oliver Stueker, Memorial U
Phil Romkey, SMU
Tyson Whitehead, Western
Weiguang Guan, McMaster

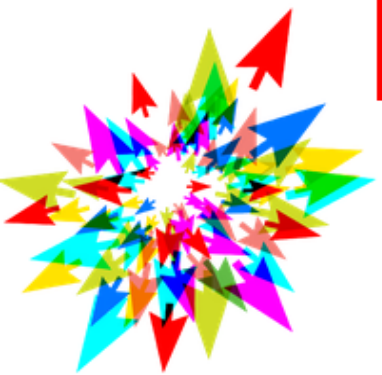












RESEARCH PORTAL

Visualization

French ➤

Portal Home

Account Management

Accessing Resources

Technical Support

National Services

Using Compute Canada's resources and technical expert help, you can easily convert the results of your numerical simulations or your experimental dat into engaging images or movies to share with colleagues, to put online, or into a publication. Our technical staff have extensive experience in scientific visualization and visual data analysis, primarily using open-source tools such as ParaView, VisIt, VTK, Blender, VMD, and various Python libraries to wo with a wide variety of data types. Large multi-dimensional datasets can be visualized directly on Compute Canada clusters without having to move them to your desktop. We can help you with all stages of visualization, from preparing data in the right format to interactive analysis. For more information, please contact us at vis-support@computecanada.ca

VISUALIZATION CHALLENGE

- To be announced on **Sept.20th**
- **Live Q&A session** on **Sept.26th**

https://www.computecanada.ca/events/visualization-challenge

Visualization Challenge

Alex Razoumov

SciNet
ADVANCED RESEARCH COMPUTING at the UNIVERSITY OF TORONTO