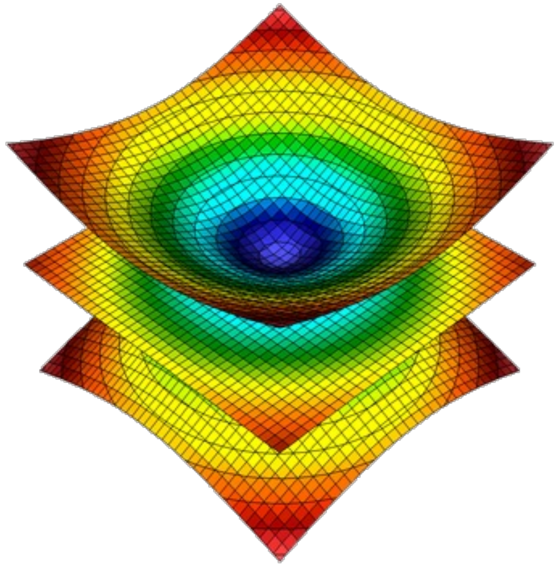
Advanced Scientific Visualization Workflows with VisIt



KAUST Visualization Core Lab

James Kress

Workshop Site: wiki.vis.kaust.edu.sa/training
Install VisIt 3.3.1: https://visit-dav.github.io/visit-website/releases-as-tables/#latest

KAUST
VISUALIZATION
CORE LAB

Resources

Presenter/KVL Contact Info:

- James Kress: james.kress@kaust.edu.sa
- KVL website: wiki.vis.kaust.edu.sa
- General Inquiries: help@vis.kaust.edu.sa
- KVL Vis Repo: https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes

User Resources:

- Main website: http://www.llnl.gov/visit
- Discussions: https://github.com/visit-dav/visit/discussions
- User Guide: https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/
- Wiki: http://www.visitusers.org

Developer Resources:

- Github: https://github.com/visit-dav/visit

Workshop Setup

- Never logged in to Ibex before?
  - Do so now so that your scratch directory will have time to get setup
    - `ssh -X <username>@ilogin.ibex.kaust.edu.sa`
- Clone example repo on local machine
  - `git clone https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes.git`
    - ex00 - This script shows how to load a data set and then query information about the mesh, variables, and more
    - ex01 - This script shows how to create a screenshot and save it to disk
    - ex02 - This script shows how to take a series of screenshots while moving the camera and creating a movie
    - ex03 - This script shows how to animate the visualization of multiple iso surface values, showing different segments of a static data set
    - ex04 - This script shows how to animate the progress of streamlines in a flow field
    - ex05 - This script shows how to load and step through a multi time step file and take a screenshot per step

Workshop Setup

mac machines

- Install VisIt
  - https://visit-dav.github.io/visit-website/releases-as-tables/#latest
- Install ffmpeg
  - https://www.ffmpeg.org/download.html
- Extra info
  - If you want to view ibex files locally without 'scp'
    - Download and install fuse and sshfs: https://osxfuse.github.io/
      - Install instructions: https://sbgrid.org/corewiki/faq-sshfs.md
  - If multiple versions of VisIt are installed, we need to add "-v 3.3.1" to local scripts in the examples later in the workshop

Workshop Setup

windows machines

- Install VisIt
  - https://visit-dav.github.io/visit-website/releases-as-tables/#latest
- Install ffmpeg
  - https://www.ffmpeg.org/download.html
    - Unzip this file by using any file archiver such as Winrar or 7z
    - Rename the extracted folder to ffmpeg and move it into the root of C: drive or location of your preference
    - Run the following in cmd: setx /m PATH "C:\ffmpeg\bin;%PATH%"
    - Reboot
- Extra info
  - Don't install VisIt in path with a space in it (`` '')
    - VisIt does not like this
  - I suggest running all the terminal examples in:
    - Ubuntu for Windows
      or
    - Visual Studio Code
  - If you want to view ibex files locally without 'scp'
    - Download and install SFTP Drive
      - https://www.nsoftware.com/sftp/drive/

Workshop Setup

linux machines

- Install VisIt
  - https://visit-dav.github.io/visit-website/releases-as-tables/#latest
- Install ffmpeg
  - https://www.ffmpeg.org/download.html
  - Use modules on KAUST machines
  - apt-get install ffmpeg

Visualization Core Lab

Overview

The Team

Dr. Sohaib Ghani
(LEAD STAFF SCIENTIST)

- VISUAL ANALYTICS
- INFORMATION VIS
- STATISTICAL ANALYSIS

Thomas Theussl
SCIVIS

- SCIENTIFIC VISUALIZATION
- LARGE DATA ANALYSIS
- DISTRIBUTED VISUALIZATION

Dr. James Kress
HPC SCIVIS

- VISUALIZATION SOFTWARE
- HPC INSITU VISUALIZATION
- DISTRIBUTED VISUALIZATION

Dr. Ronell Sicat
VR/AR

- SCIENTIFIC VISUALIZATION
- VR DEVELOPMENT
- 3D RECONSTRUCTION

Dr. Didier Barradas
Data Scientist

- DATA SCIENCE
- MACHINE LEARNING
- DEEP LEARNING

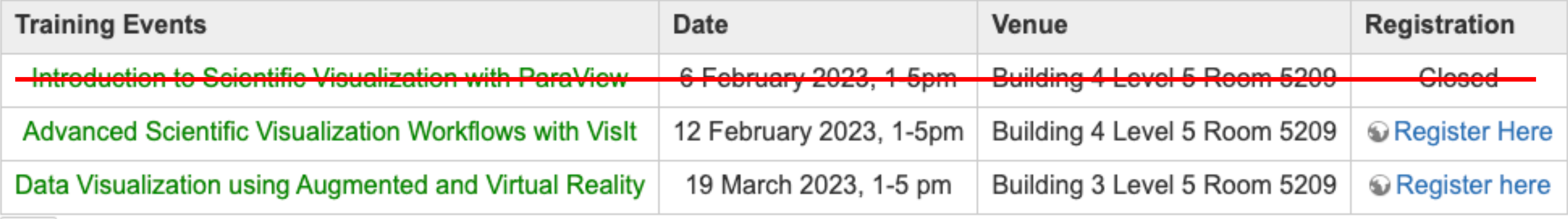
Dr. Abdelghafour Halimi
Data Scientist

- Data Science
- Machine Learning
- Deep Learning

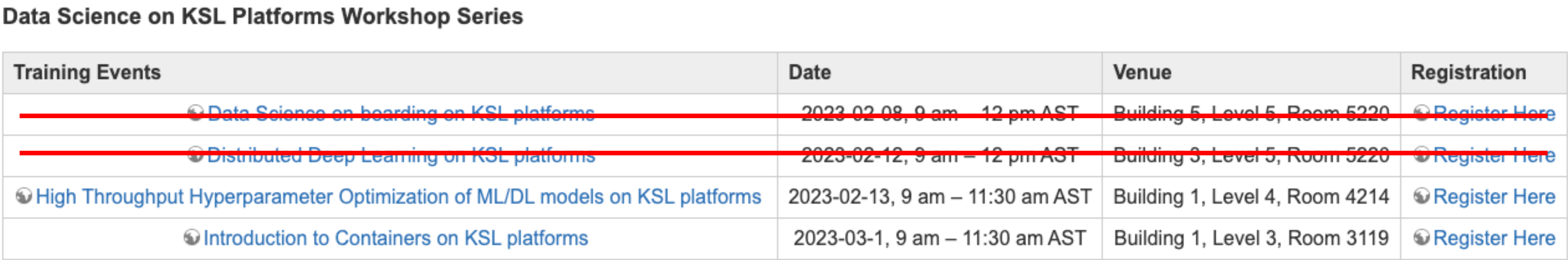
Collaborating with KVL

- Standard Request
  - Load data ‘X’ in program ‘P’ to produce a visualization ‘V’
- Advanced Support
  - Investigative visualization –
    - Asking “why?” & “what?” of your data
- Collaboration
  - Work with you through your research and discovery cycle
- Have an interest in HPC vis or in situ? Let me know!

Upcoming KVL Workshops



Training Events	Date	Venue	Registration
~~Introduction to Scientific Visualization with ParaView~~	~~6 February 2023, 1-5pm~~	~~Building 4 Level 5 Room 5209~~	~~Closed~~
Advanced Scientific Visualization Workflows with VisIt	12 February 2023, 1-5pm	Building 4 Level 5 Room 5209	Register Here
Data Visualization using Augmented and Virtual Reality	19 March 2023, 1-5 pm	Building 3 Level 5 Room 5209	Register here



Data Science on KSL Platforms Workshop Series

Training Events	Date	Venue	Registration
~~Data Science on-boarding on KSL platforms~~	~~2023-02-08, 9 am – 12 pm AST~~	~~Building 5, Level 5, Room 5220~~	~~Register Here~~
~~Distributed Deep Learning on KSL platforms~~	~~2023-02-12, 9 am – 12 pm AST~~	~~Building 3, Level 5, Room 5220~~	~~Register Here~~
High Throughput Hyperparameter Optimization of ML/DL models on KSL platforms	2023-02-13, 9 am – 11:30 am AST	Building 1, Level 4, Room 4214	Register Here
Introduction to Containers on KSL platforms	2023-03-1, 9 am – 11:30 am AST	Building 1, Level 3, Room 3119	Register Here

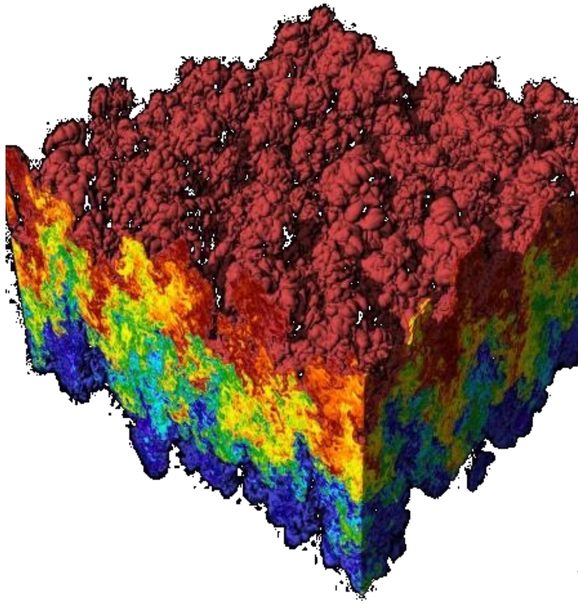
Workshop Goals

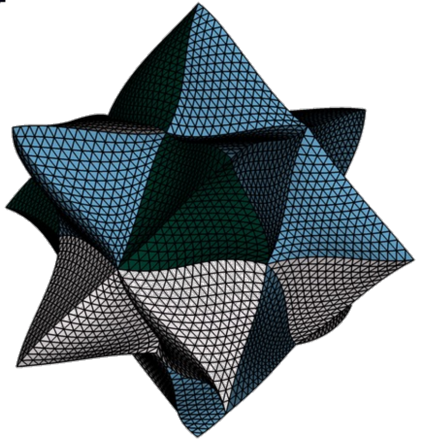
- Hands-on learning with VisIt
  - Intermediate / advanced course
    - Scripting and workflows from desktop to HPC
  - Interactive sessions!
- Why VisIt @ KAUST
  - Open source, scalable, multi-platform visualization application with users worldwide
  - Available on all major computation resources at KAUST
    - VisIt on Ibex and Shaheen
    - VisIt on IT Remote Workstations
    - VisIt on KVL Tiled-display walls

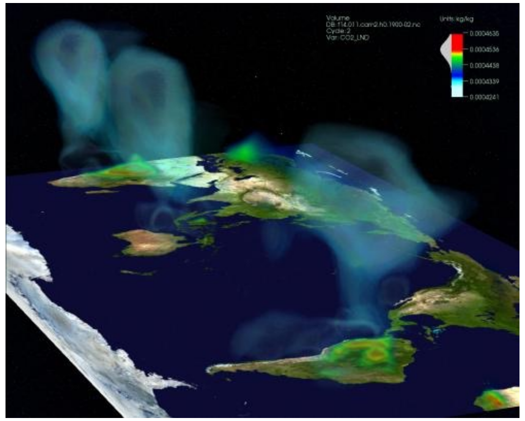
VisIt Basics

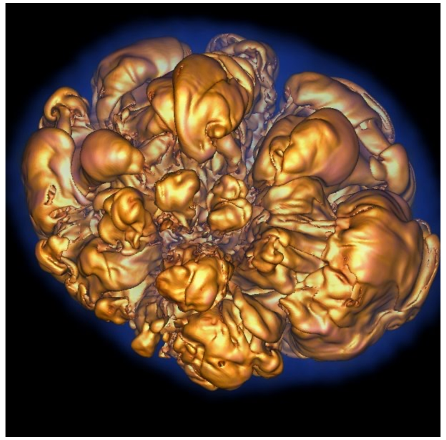
What is VisIt?

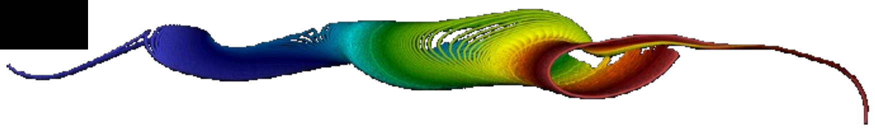
- Open source turnkey application for data analysis and visualization of mesh-based data
- Infrastructure for parallel post-processing that scales from laptops to HPC clusters
- Built-in in situ capabilities



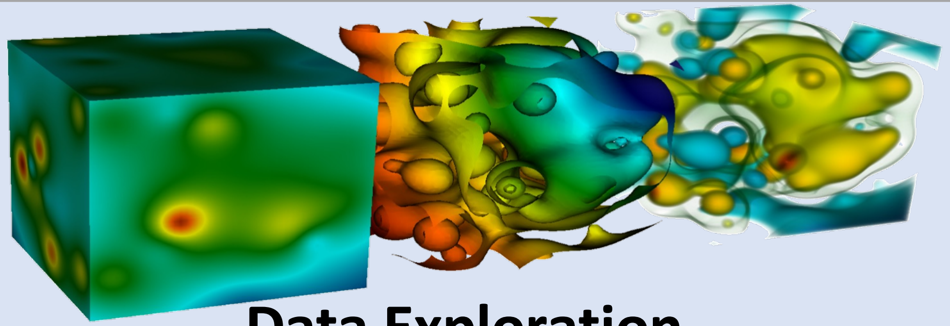




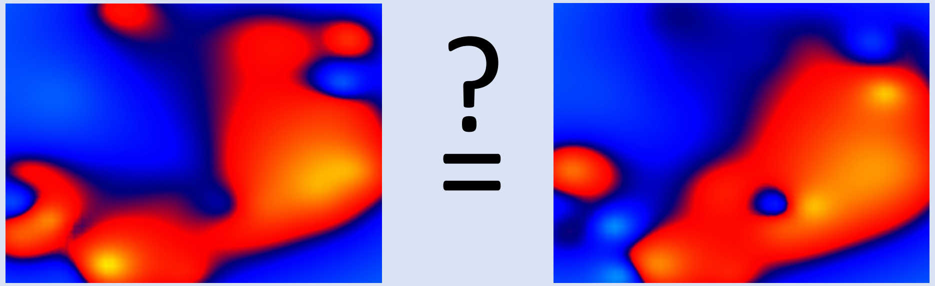




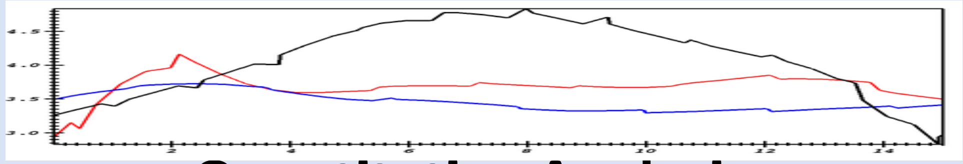
VisIt Supports a Wide Range of Use Cases



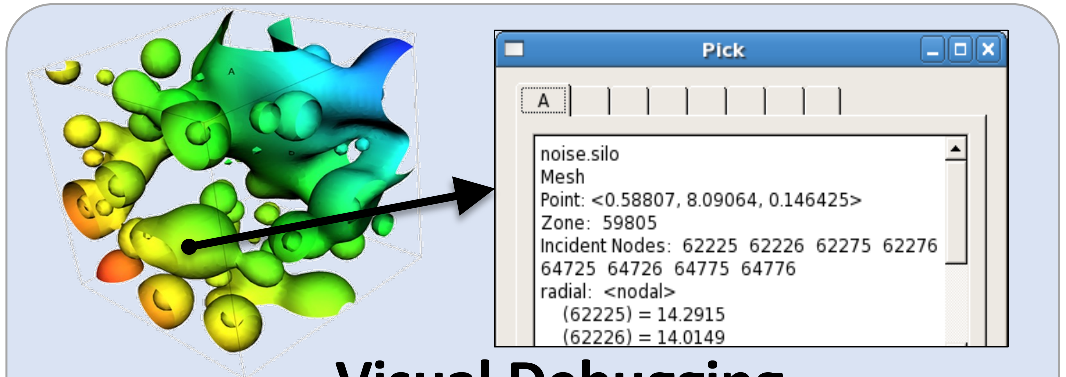
Data Exploration



Comparative Analysis

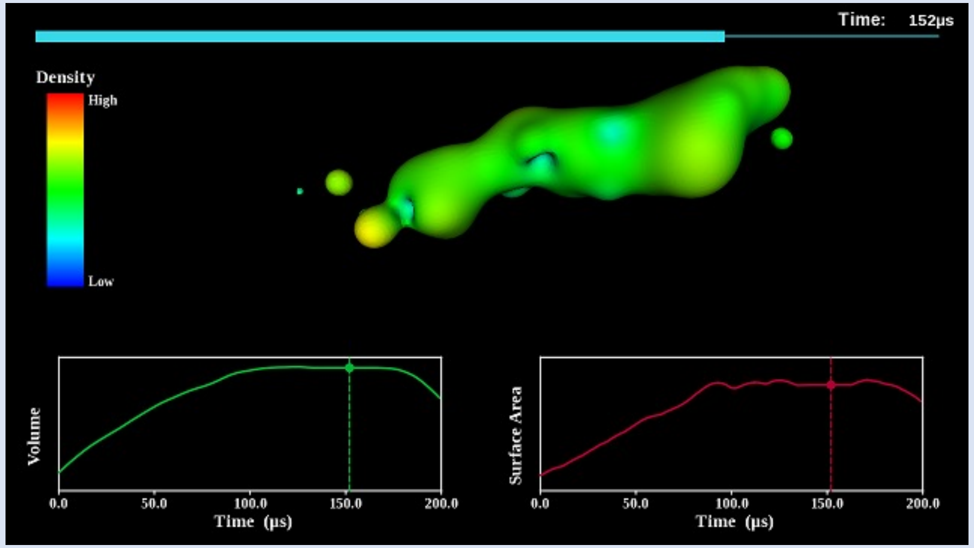


Quantitative Analysis



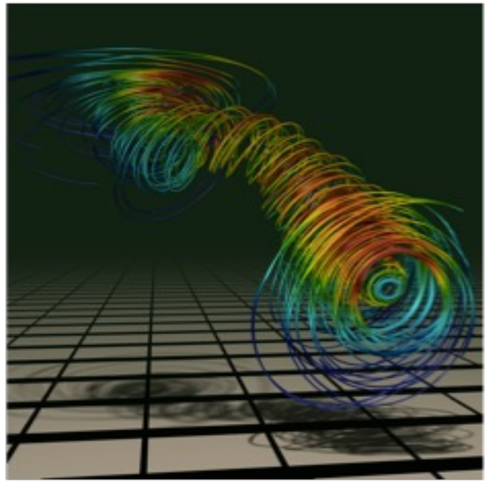
```
noise.silo
Mesh
Point: <0.58807, 8.09064, 0.146425>
Zone:  59805
Incident Nodes:  62225  62226  62275  62276
64725  64726  64775  64776
radial:  <nodal>
   (62225) = 14.2915
   (62226) = 14.0149
```

Visual Debugging

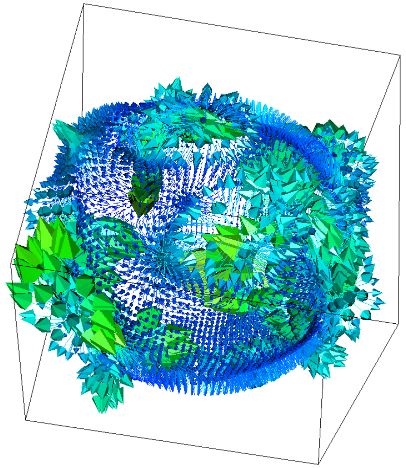


Presentation Graphics

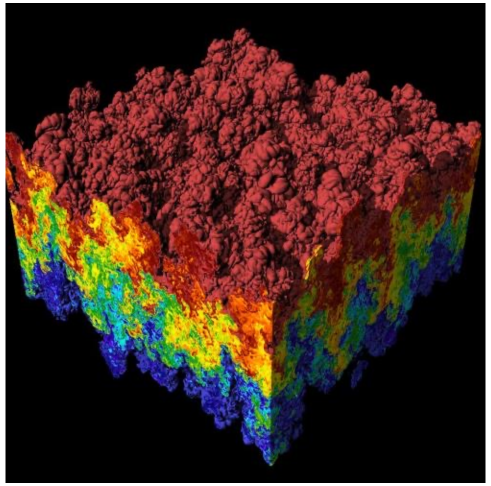
VisIt Supports a Wide Range of Plotting Types



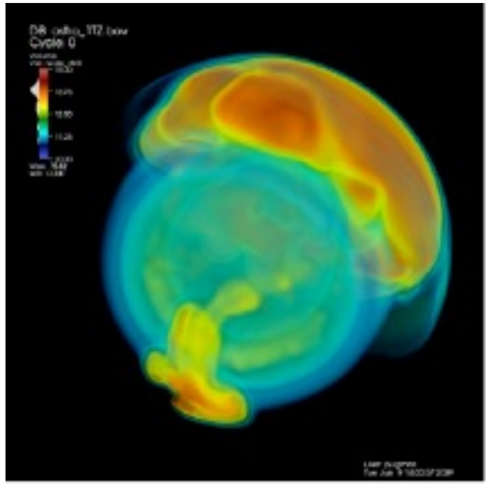
Streamlines / Pathlines



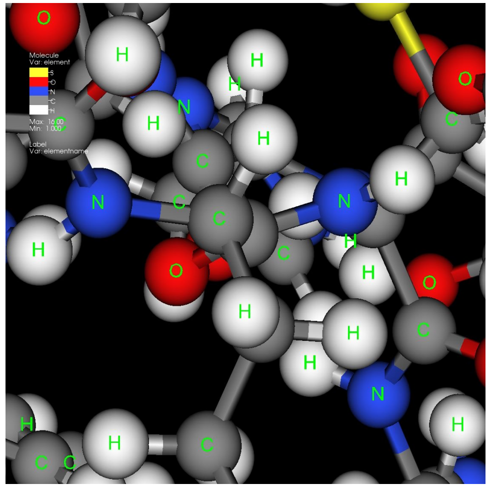
Vector / Tensor Glyphs



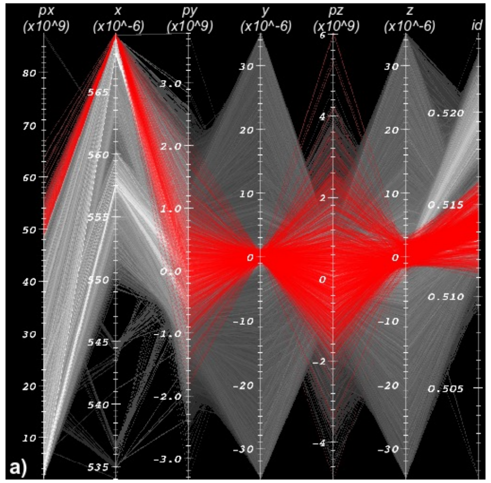
Pseudocolor Rendering



Volume Rendering



Molecular Visualization



Parallel Coordinates

How Do I Obtain VisIt?

- Use an existing build:
  - For your Laptop or Workstation:
    - Binaries for Windows, OSX, and Linux (RHEL + Ubuntu): (https://visit-dav.github.io/visit-website/releases-as-tables/#latest)
  - KVL team manages builds on Ibex and Shaheen
  - IT Remote Workstations
- Build VisIt yourself:
  - “build_visit” is a script that automates the process of building VisIt and its third-party dependencies. (docs: https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/building_visit/index.html)

How Do I Get My Data Into VisIt?

VisIt supports more than 110 file formats

- *VTK, Silo, Xdmf, PVTK*
- The *PlainText* database reader can read simple text files (CSV, etc)
    - https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/data_into_visit/PlainTextFormat.html
- *visit_writer* utility: code to write VTK files from your sim code
    - https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/data_into_visit/VTKFormat.html
- Support for Mesh-based data in Conduit Blueprint:
    - http://llnl-conduit.readthedocs.io/en/latest/blueprint_mesh.html

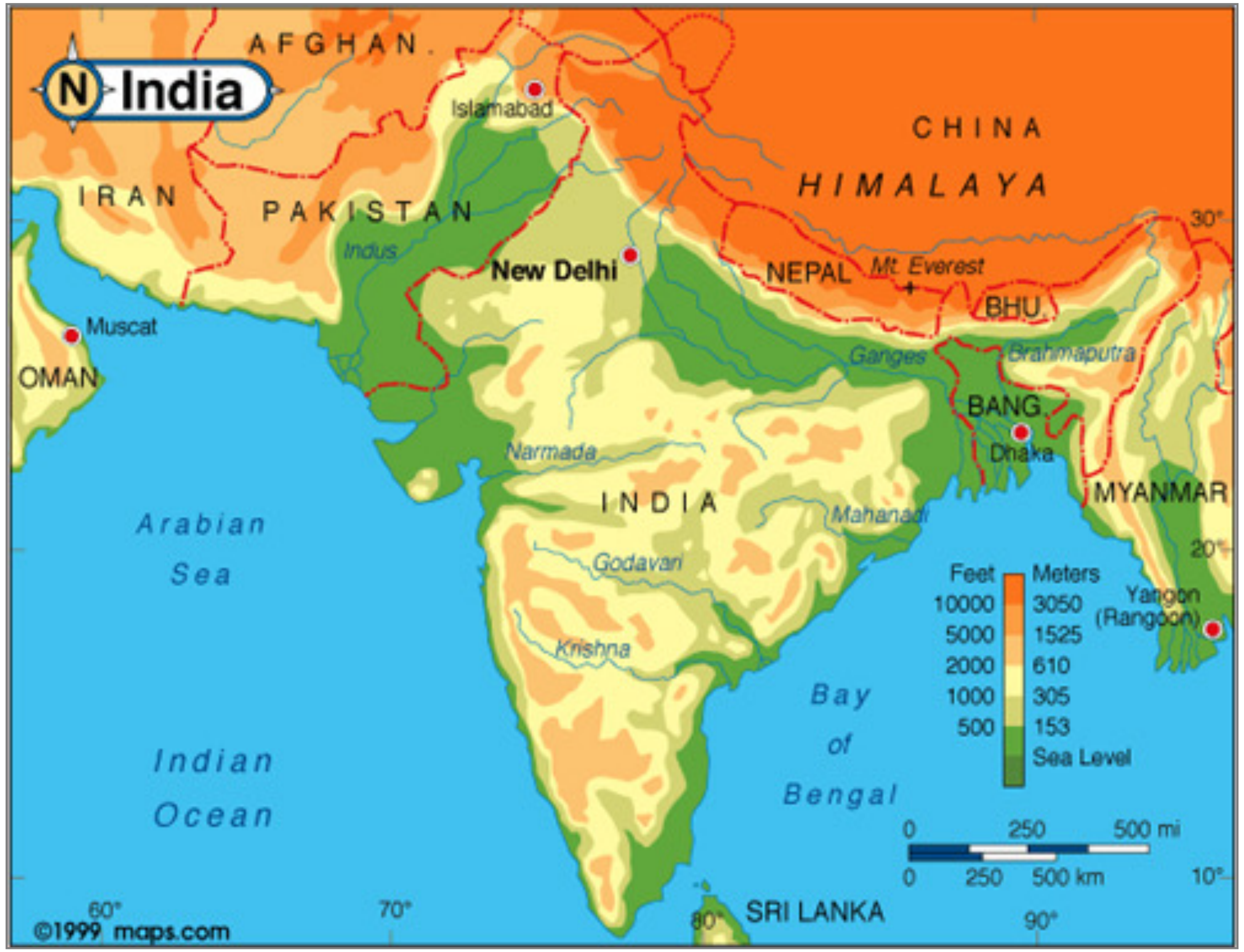
Read the docs: https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/data_into_visit/index.html

Visualization Techniques

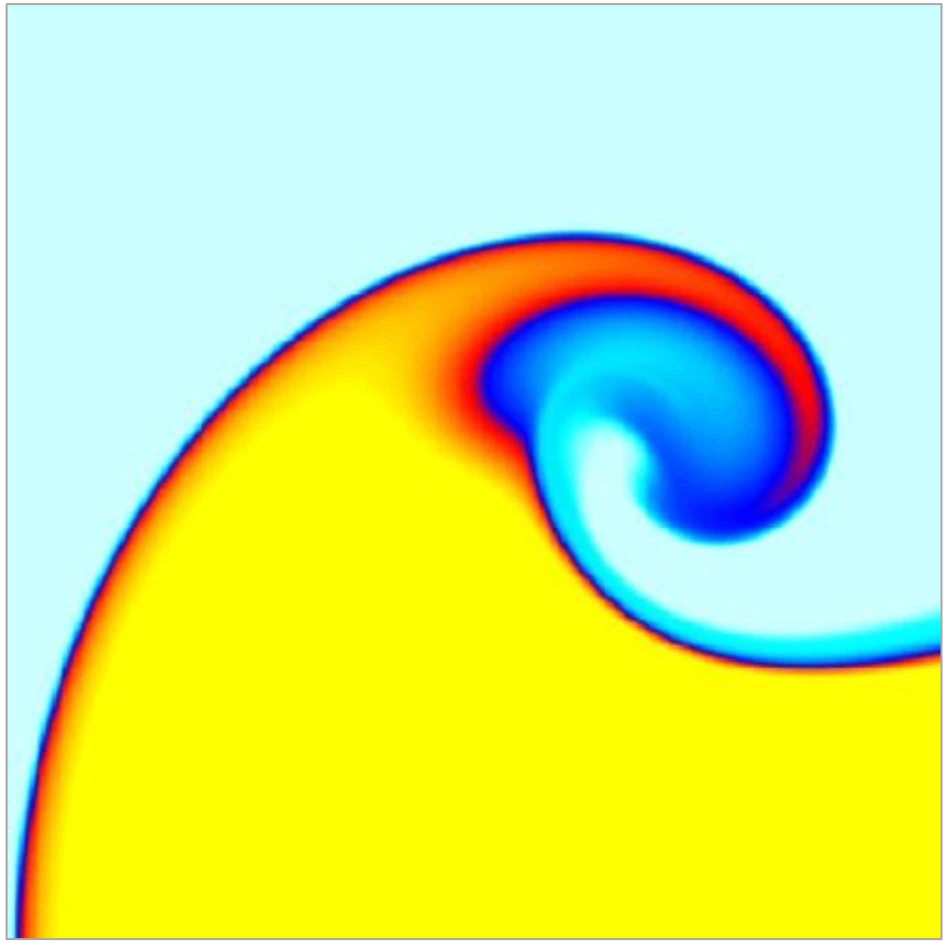
For Mesh Based Simulations

Pseudocolor Rendering

maps scalar fields to a range of colors



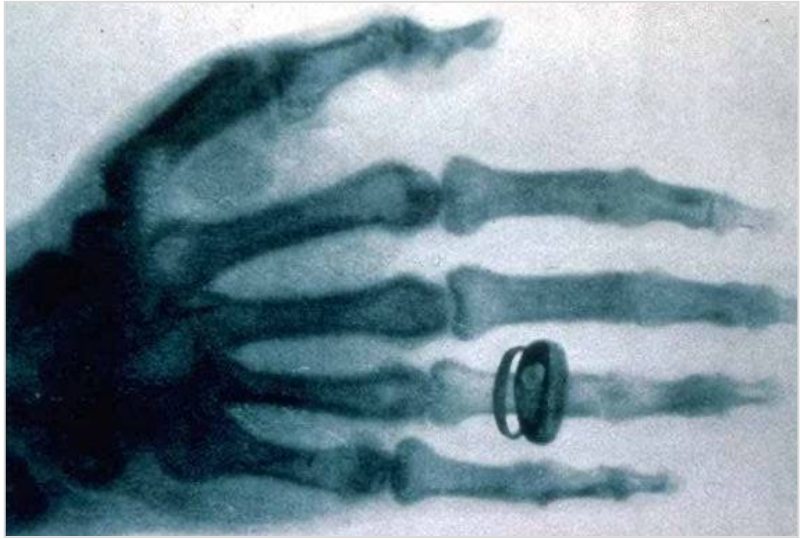
Pseudocolor rendering of Elevation

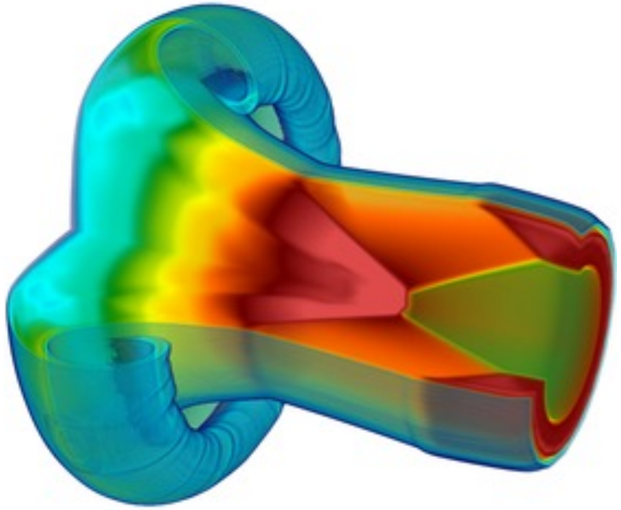


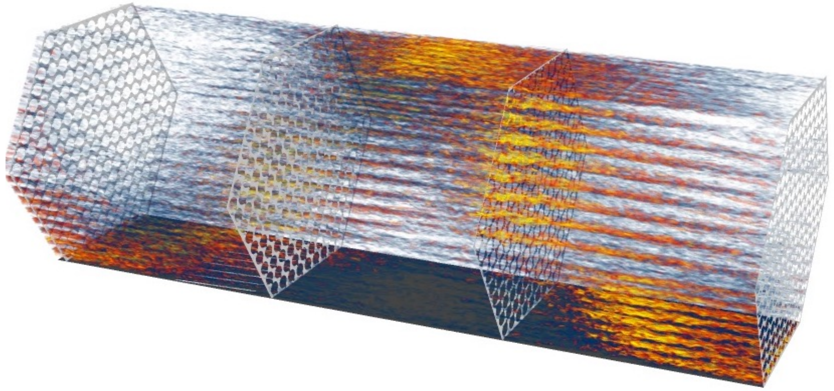
Pseudocolor rendering of Density

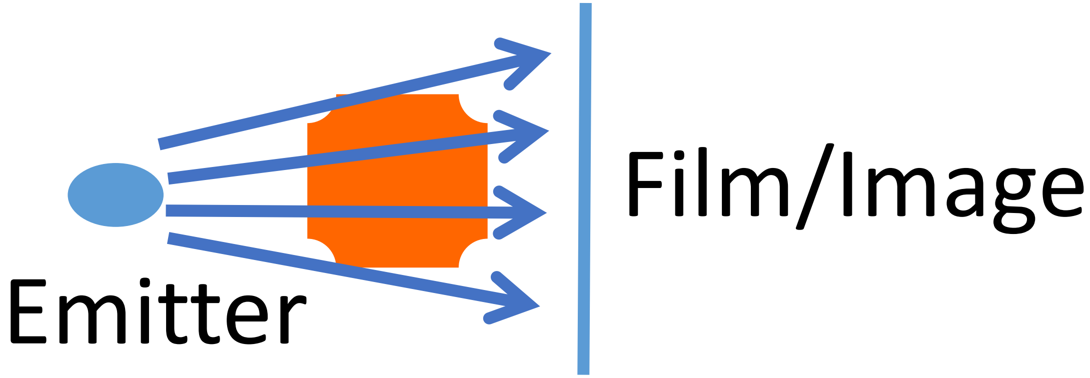
Volume Rendering

cast rays though data and applies transfer functions to produce an image



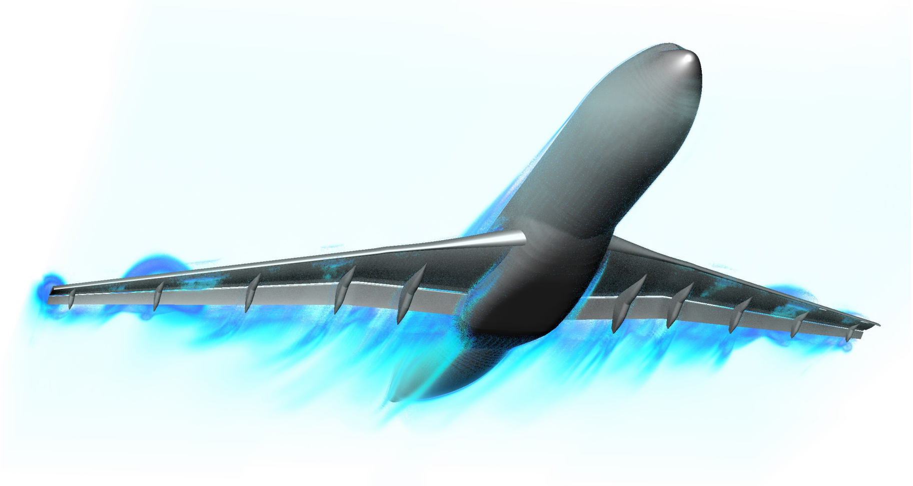






Film/Image

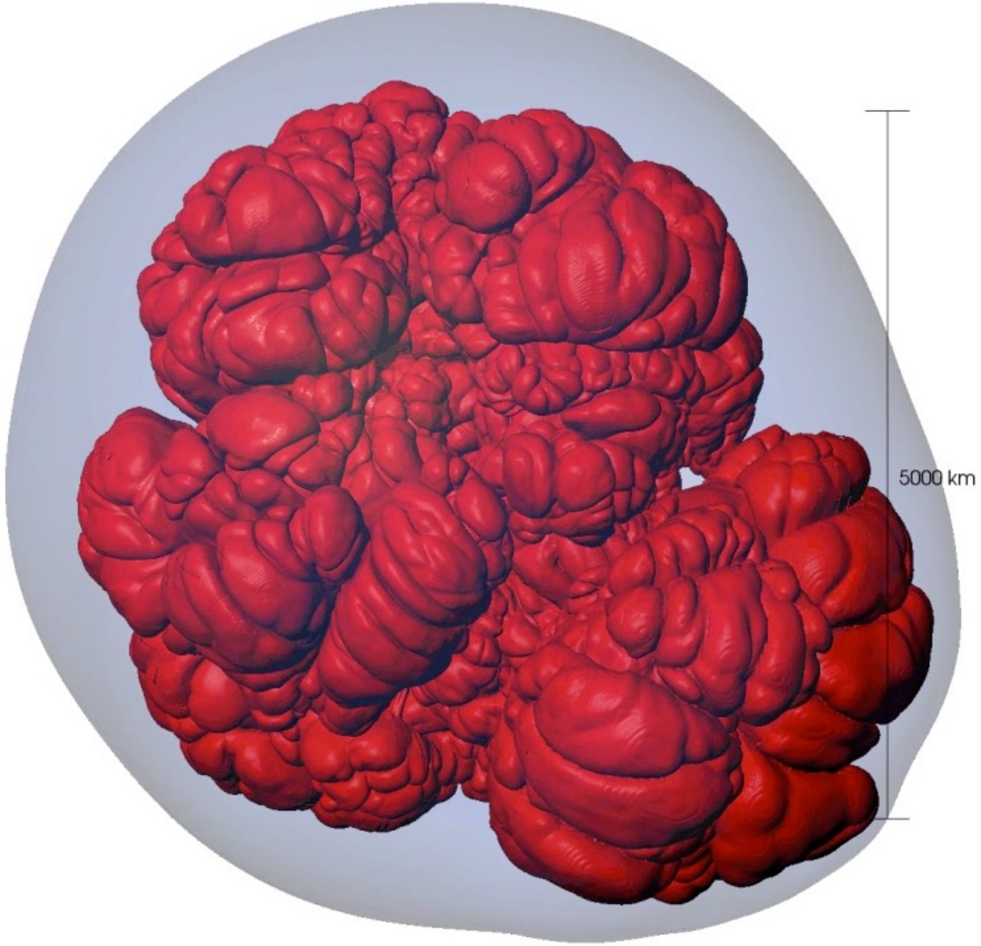
Emitter



Isosurfacing (Contouring)

extracts surfaces of that represent level sets of field values

BM 670
700
Lower Villag
800
657
S
T
Little
700
800

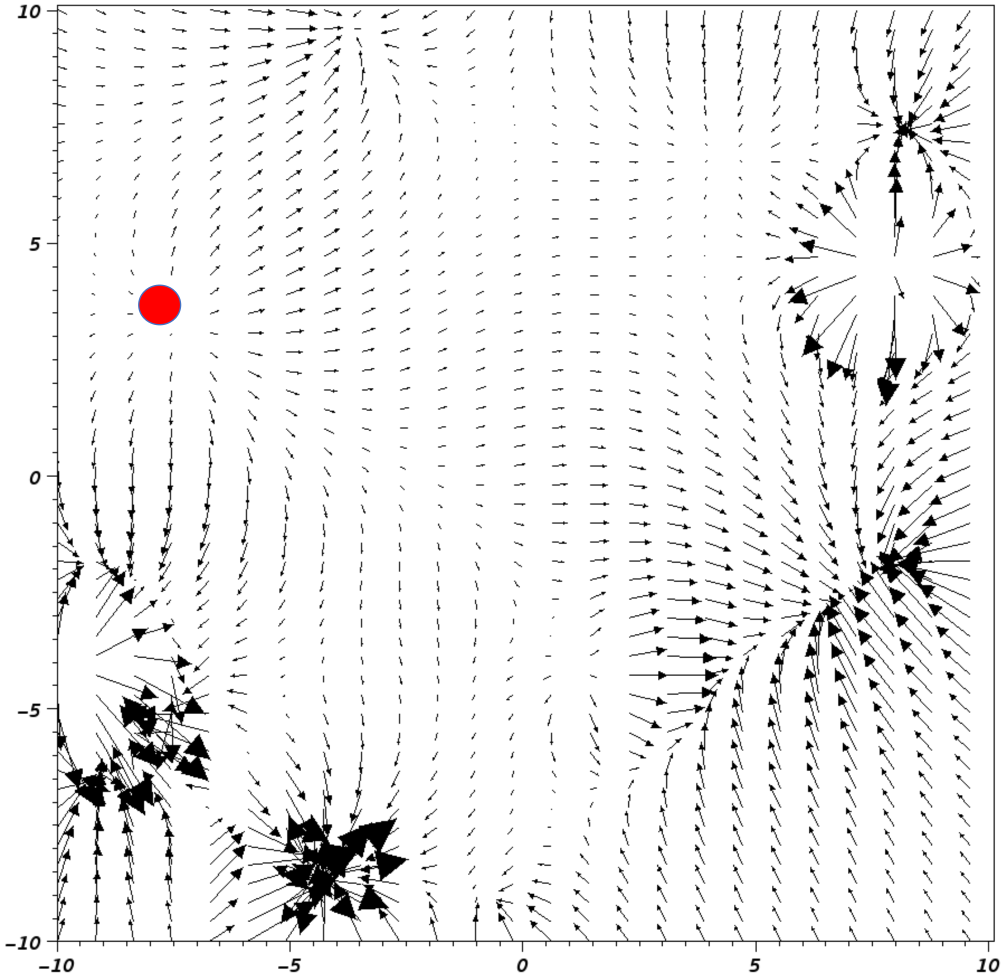


Particle advection

the foundation of several flow visualization techniques

- $S(t)$ = position of particle at time t
- $S(t_0) = p_0$
  - t_0: initial time
  - p_0: initial position
- $S'(t) = v(t, S(t))$
  - $v(t, p)$: velocity at time t and position p
  - $S'(t)$: derivative of the integral curve at time t

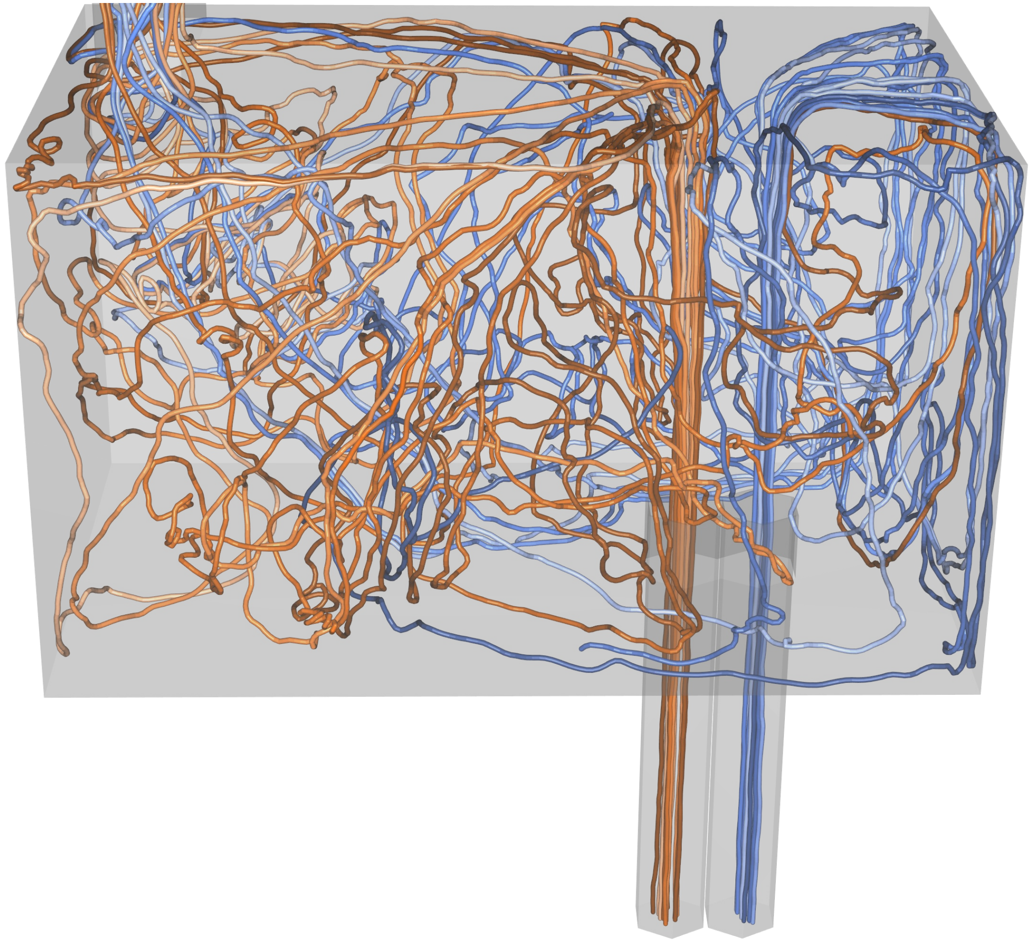
This is an ordinary differential equation.



Streamline and Pathline

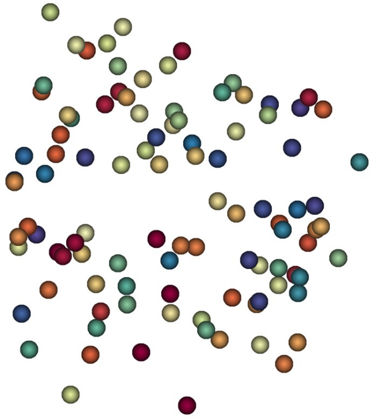
built on particle advection

- **Streamlines** – Instantaneous paths
- **Pathlines** – Time dependent paths

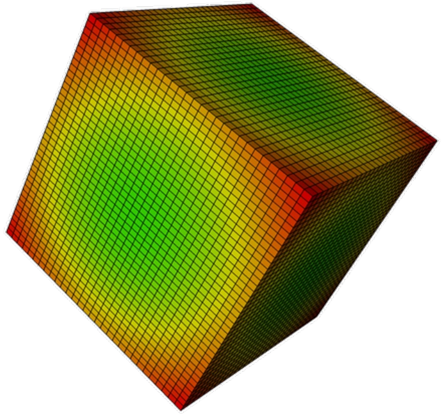


Meshes discretize continuous space

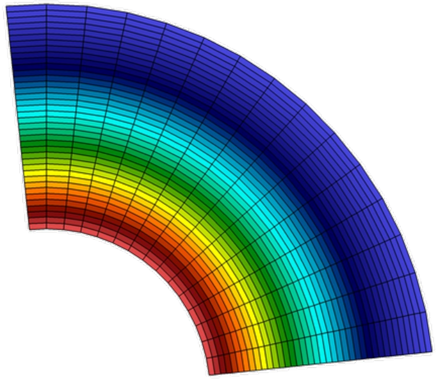
- **Simulations use a wide range of mesh types, defined in terms of:**
  - A set of coordinates (“nodes” / “points” / “vertices”)
  - A collection of “zones” / “cells” / “elements” on the coordinate set



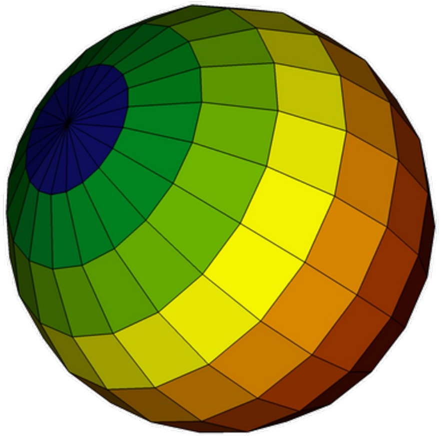
Points



Uniform



Curvilinear



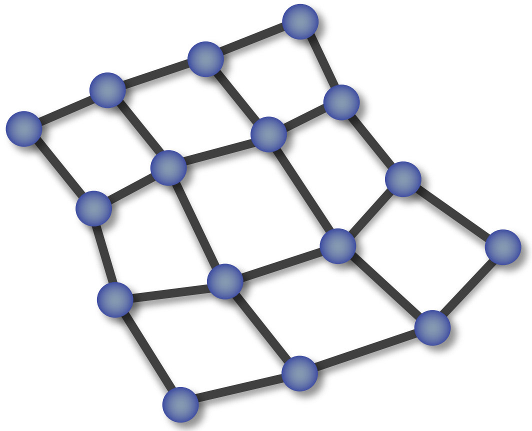
Unstructured

VisIt uses the “Zone” and “Node” nomenclature throughout its interface.

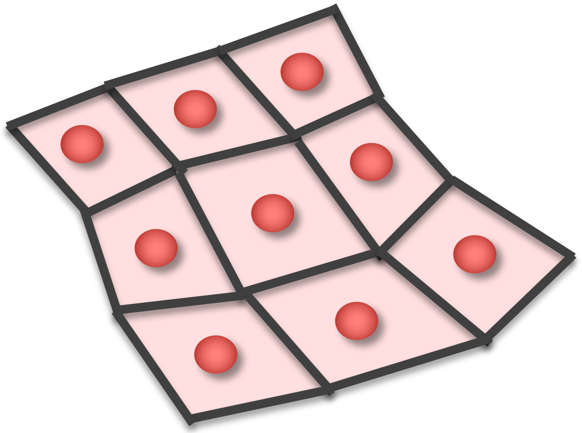
Mesh fields

variables associated with the mesh that hold simulation state

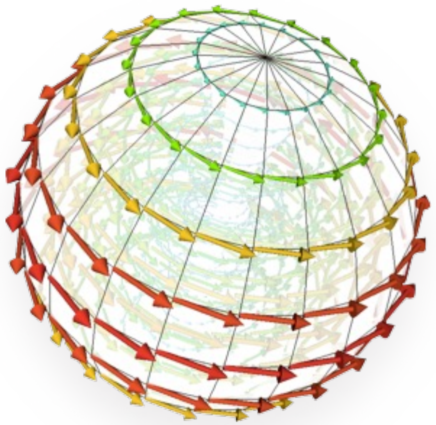
- Field values are associated with the zones or nodes of a mesh
  - Nodal: Linearly interpolated between the nodes of a zone
  - Zonal: Piecewise Constant across a zone
- Field values for each zone or node can be scalar, or multi-valued (vectors, tensors, etc.)



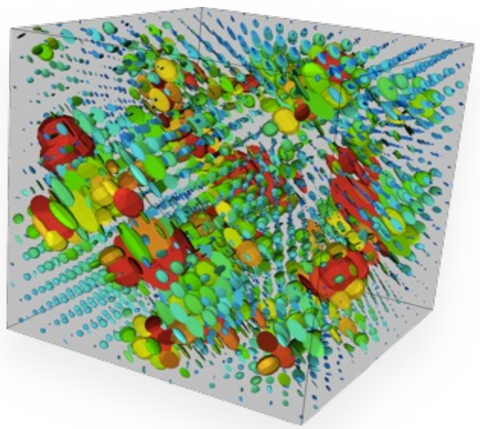
Nodal Association



Zonal Association



Vector Field

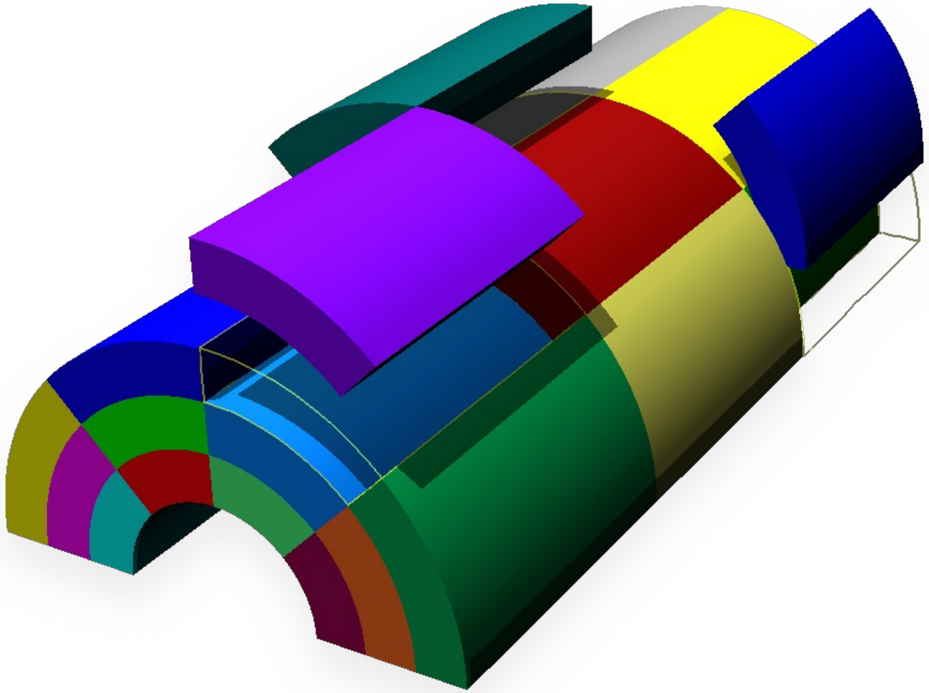


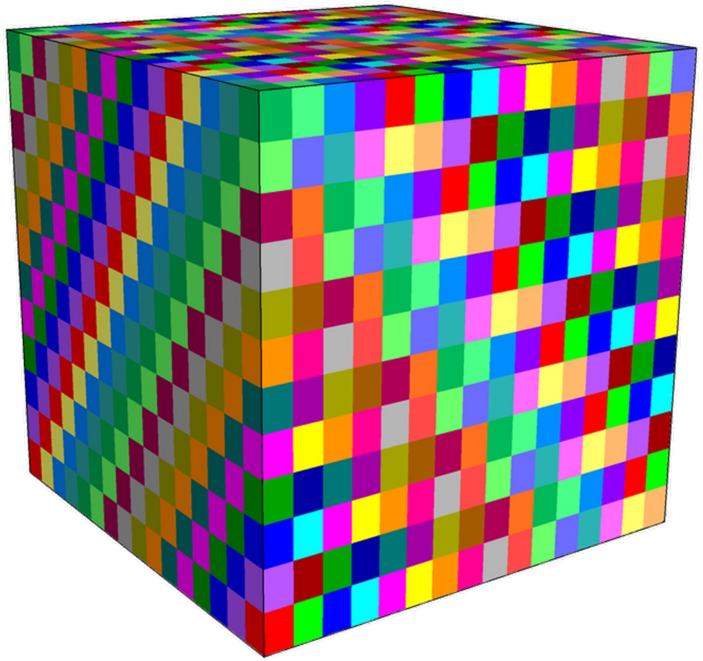
Tensor Field

Domain decomposed meshes

enable scalable parallel visualization and analysis algorithms

- Simulation meshes may be composed of smaller mesh “blocks” or “domains”
- Domains are partitioned across MPI tasks for processing

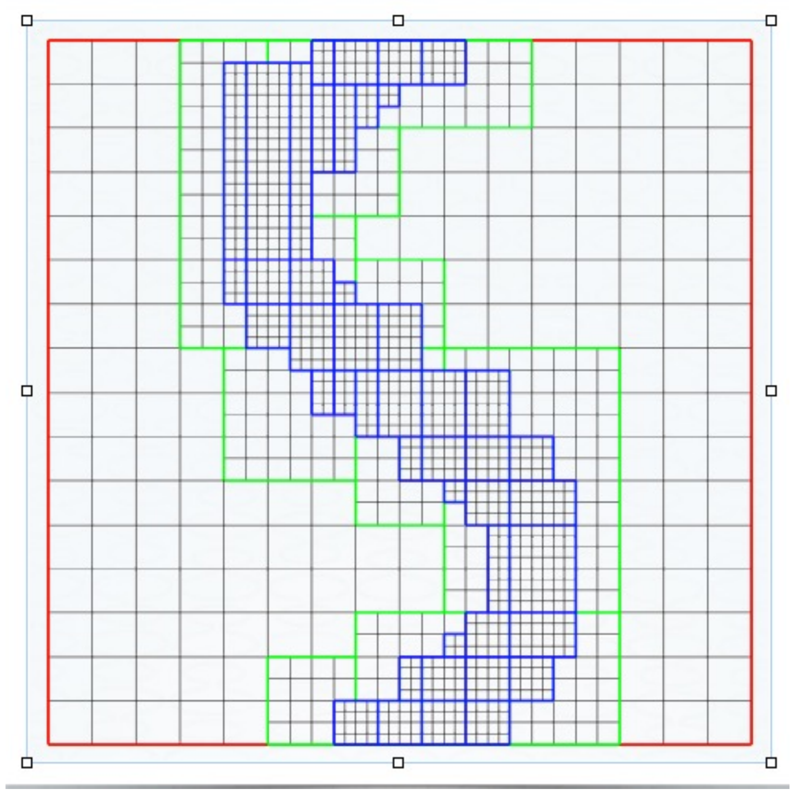


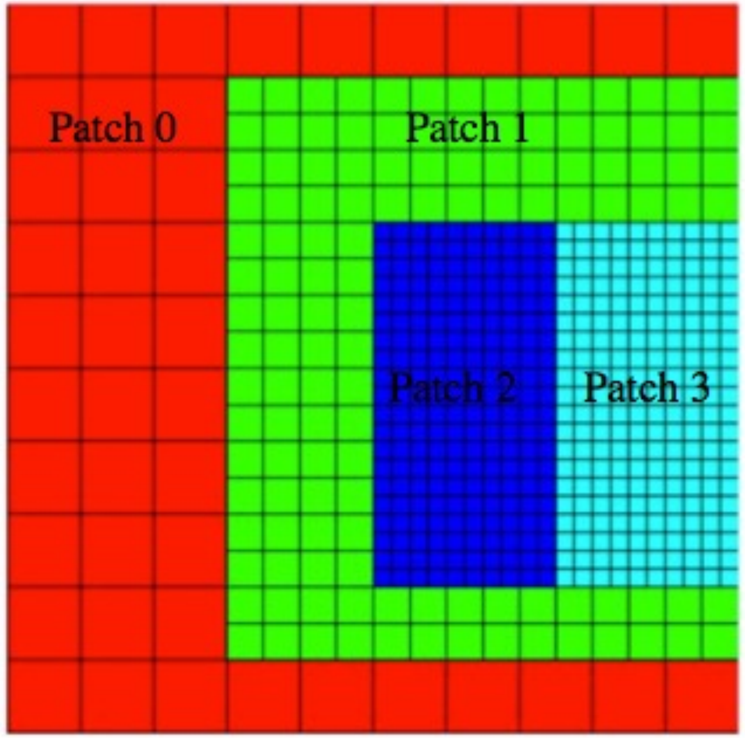


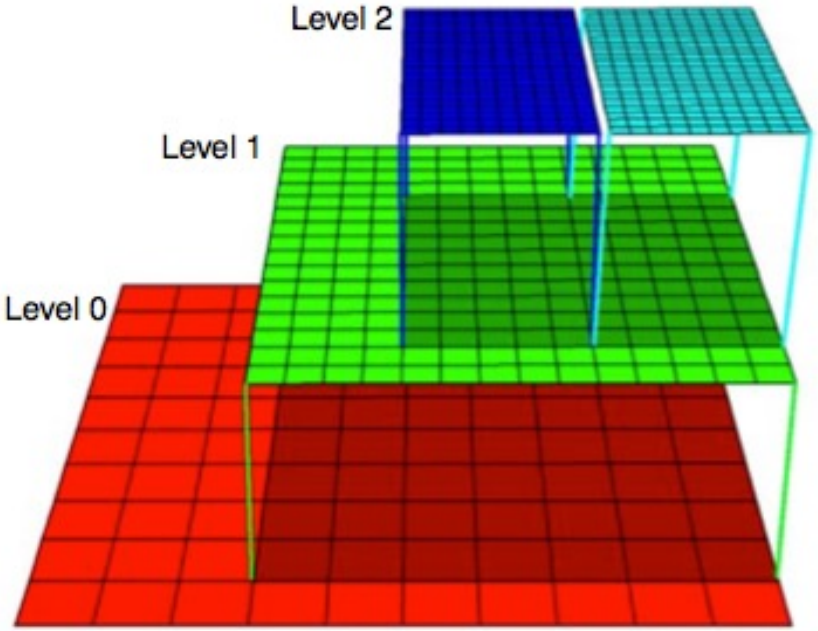
Adaptive Mesh Refinement (AMR)

refines meshes into patches that capture details across length scales

- Mesh domains are associated with patches and levels
- Patches are nested to form a AMR hierarchy



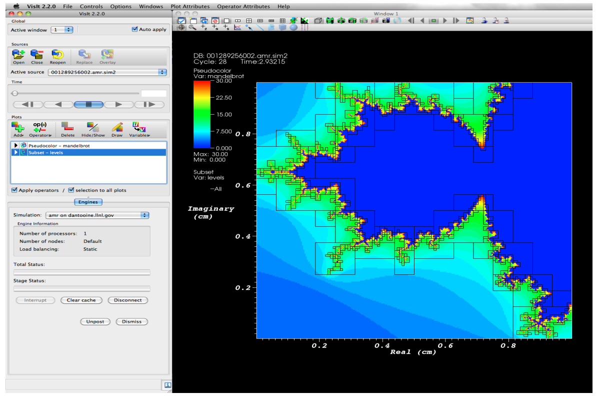


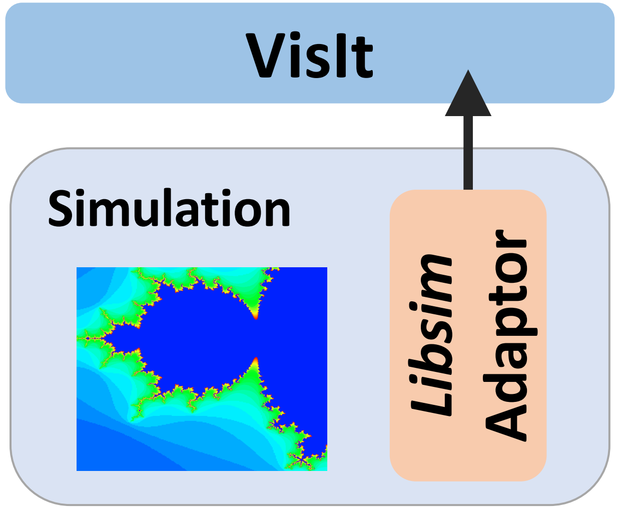


VisIt Core Concepts

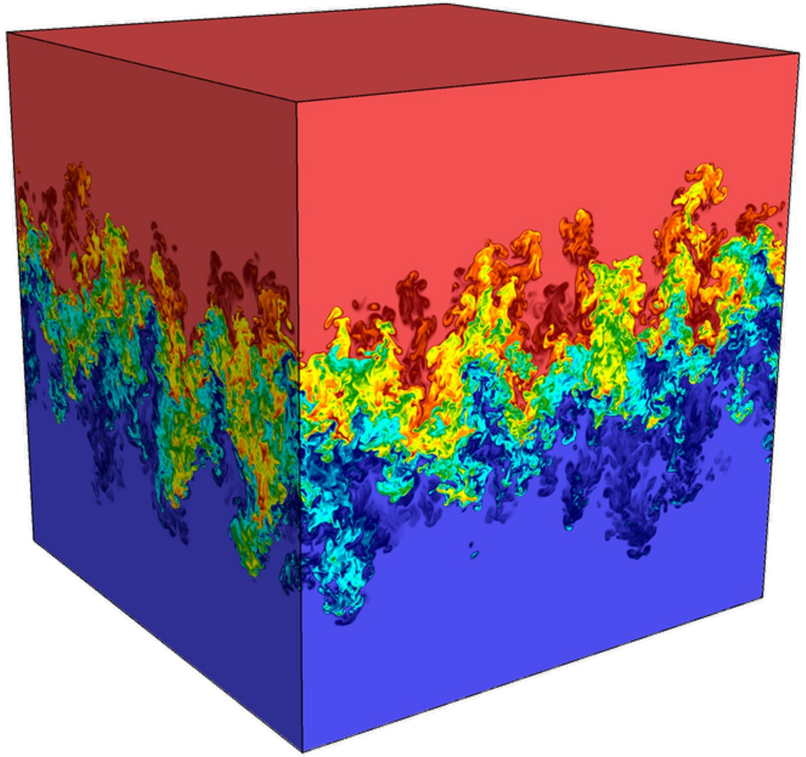
VisIt’s Infrastructure Provides a Flexible Platform for Custom Workflows

- **C++ Plugin Architecture**
  - Custom File formats, Plots, Operators
  - Interface for custom GUIs in Python, C++ and Java
- **Python Interfaces**
  - Python scripting and batch processing
  - Data analysis via Python Expressions and Queries
- **In-Situ Coupling**
  - VisIt’s *Libsim* library allows simulation codes to link in VisIt’s engine for in situ visualization

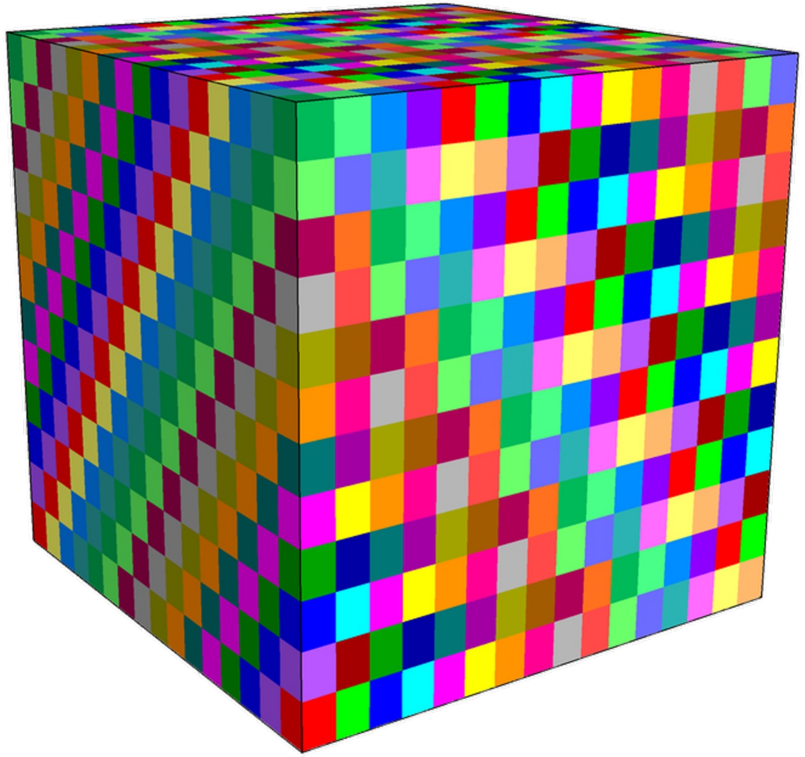




VisIt Uses MPI for Distributed-Memory Parallelism



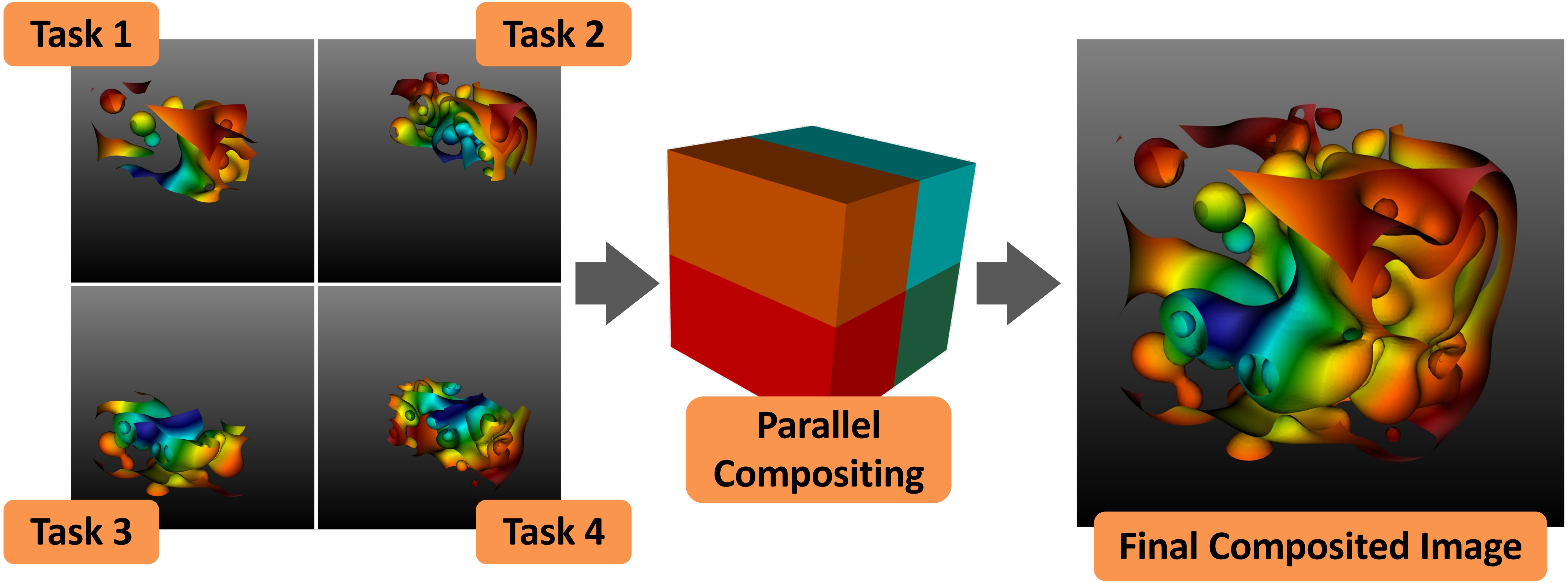
Full Dataset
(27 billion total elements)



3072 sub-grids
(each 192x129x256 cells)

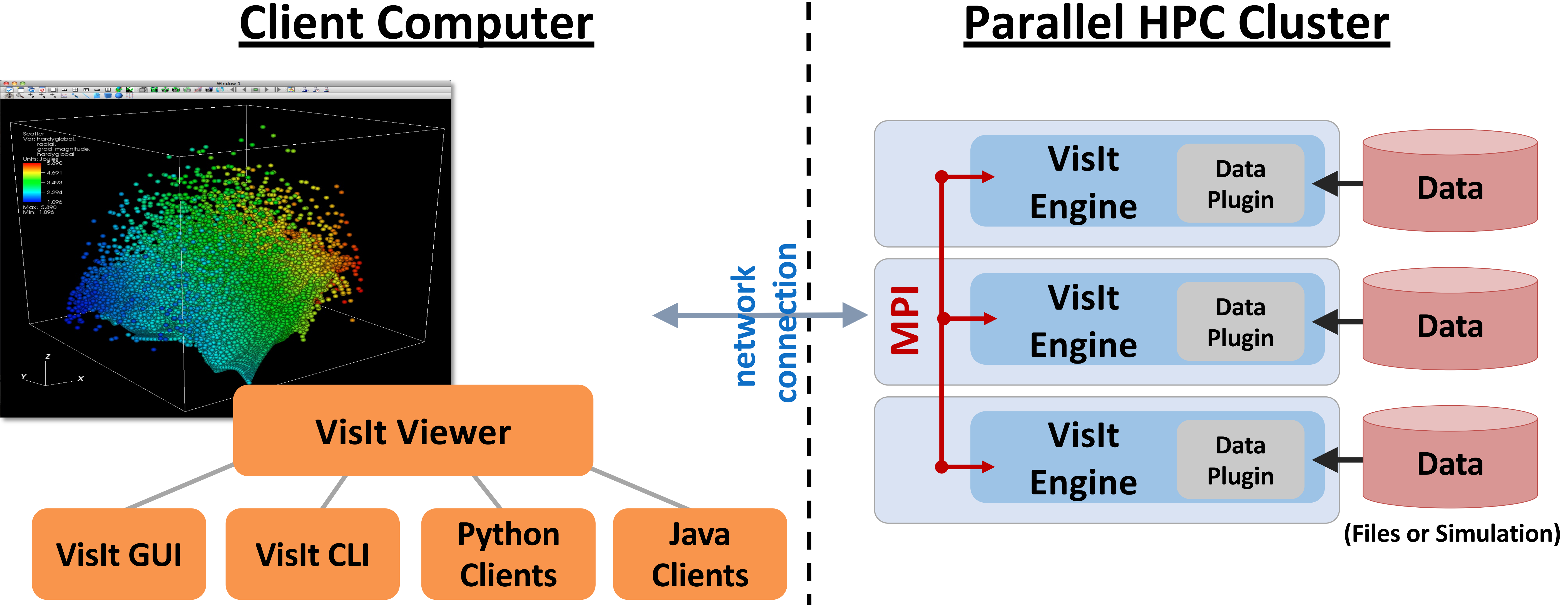
We are enhancing VisIt’s pipeline infrastructure to support threaded processing and many-core architectures

VisIt Uses Scalable Rendering for Large Data



In addition to scalable surface rendering, VisIt also provides scalable volume rendering

VisIt Employs a Client-Server Architecture



VisIt's Interface is Based on Five Abstractions

- **Databases**: Read data
- **Plots:** Render data
- **Operators:** Manipulate data
- **Expressions:** Generate derived quantities
- **Queries:** Summarize data

Examples of VisIt Pipelines

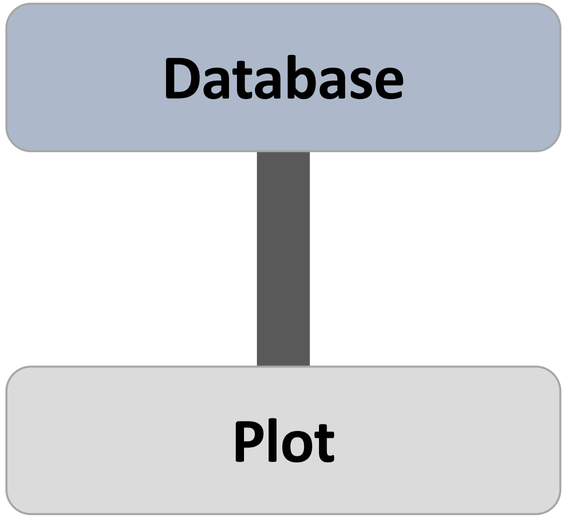
Databases: Read data

Plots: Render data

Operators: Manipulate data

Expressions: Generate derived quantities

Queries: Summarize data



Open a database, which reads from a file
(Example: Open file1.hdf5)

Make a plot of a field in the database
(Example: Pseudocolor plot of *density*)

Examples of VisIt Pipelines

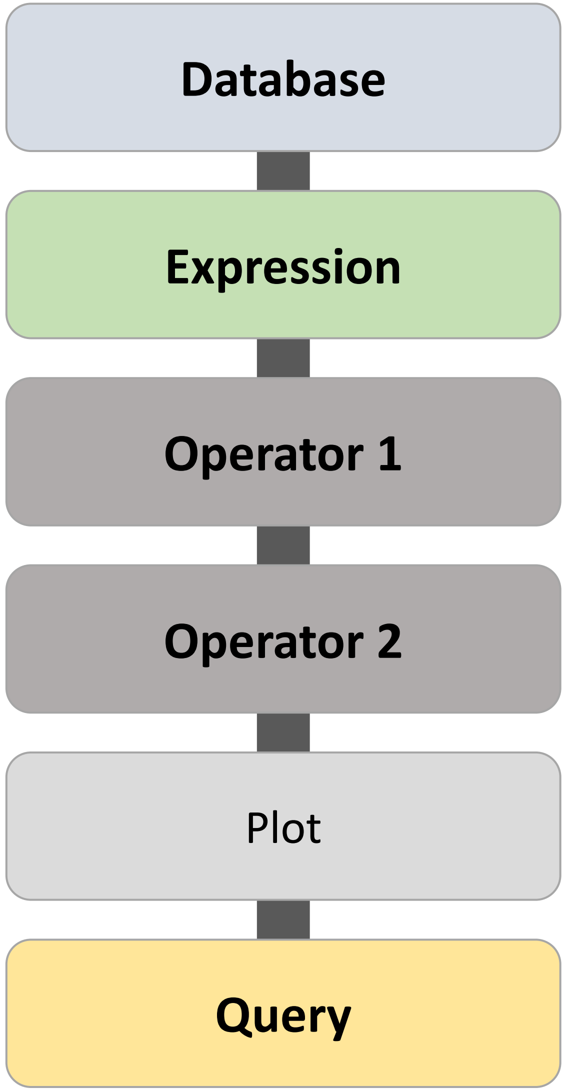
Databases: Read data

Plots: Render data

Operators: Manipulate data

Expressions: Generate derived quantities

Queries: Summarize data



Database

Open a database, which reads from a file
(Example: Open file1.hdf5)

Expression

Create derived quantity from existing fields
(Example: *speed = magnitude(velocity)*)

Operator 1

Apply an operator to transform the data
(Example: Slice operator)

Operator 2

Apply a second operator to transform the data
(Example: Elevate operator)

Plot

Plot a field
(Example: Pseudocolor plot of speed)

Query

Extract quantitative information
(Example: Maximum speed over cross-section)

VisIt Interface Tour

VisIt GUI

Pulldown Menus

Plot Window

Active Window Selector

Data Sources

Time Slider

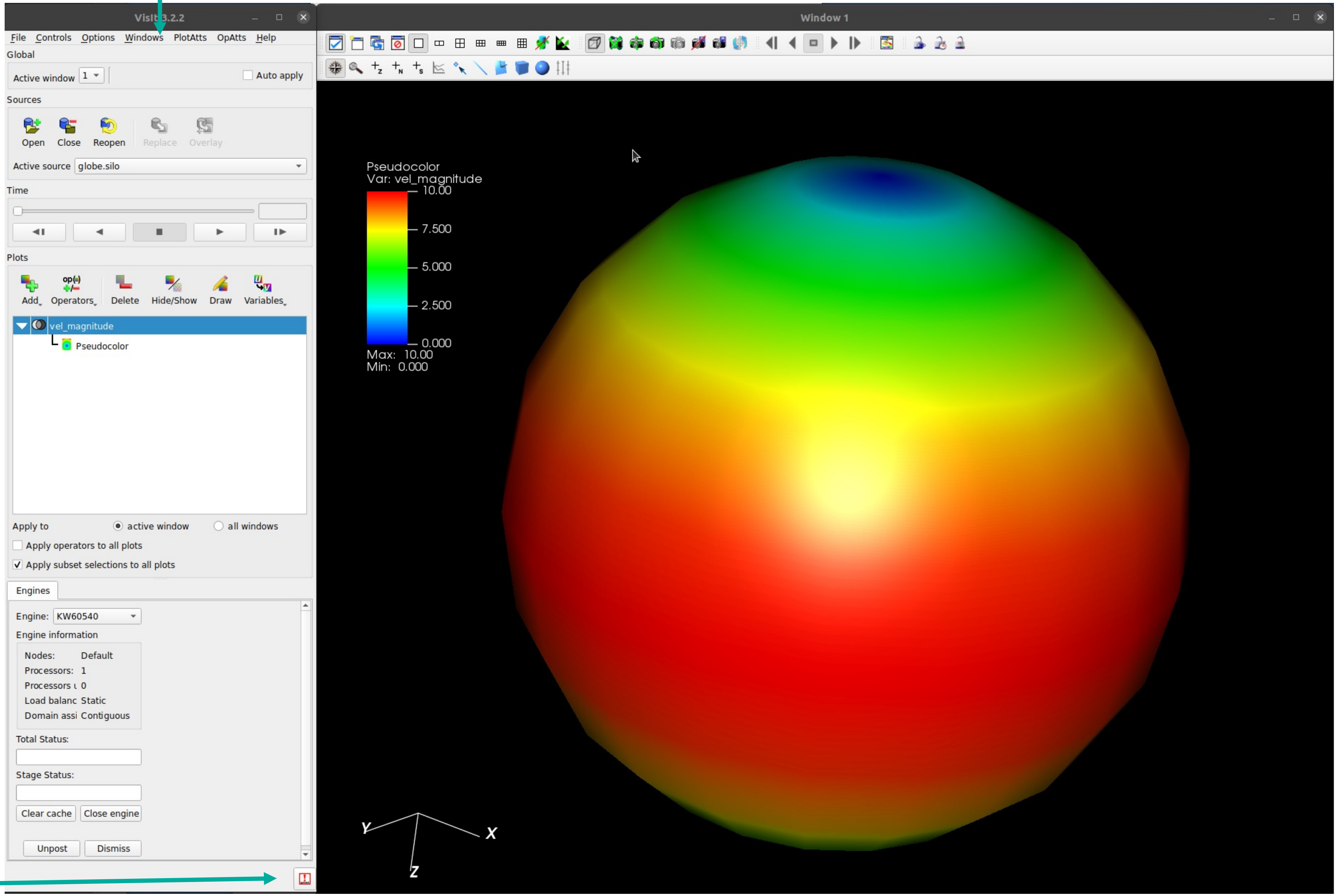
Plots & Operators

Pipeline Browser

Apply To

Notepad

Output Indicator



Active window 1

□ Auto apply

Active source globe.silo

vel_magnitude
Pseudocolor

Apply to ● active window ○ all windows

□ Apply operators to all plots

☑ Apply subset selections to all plots

Engine: KW60540

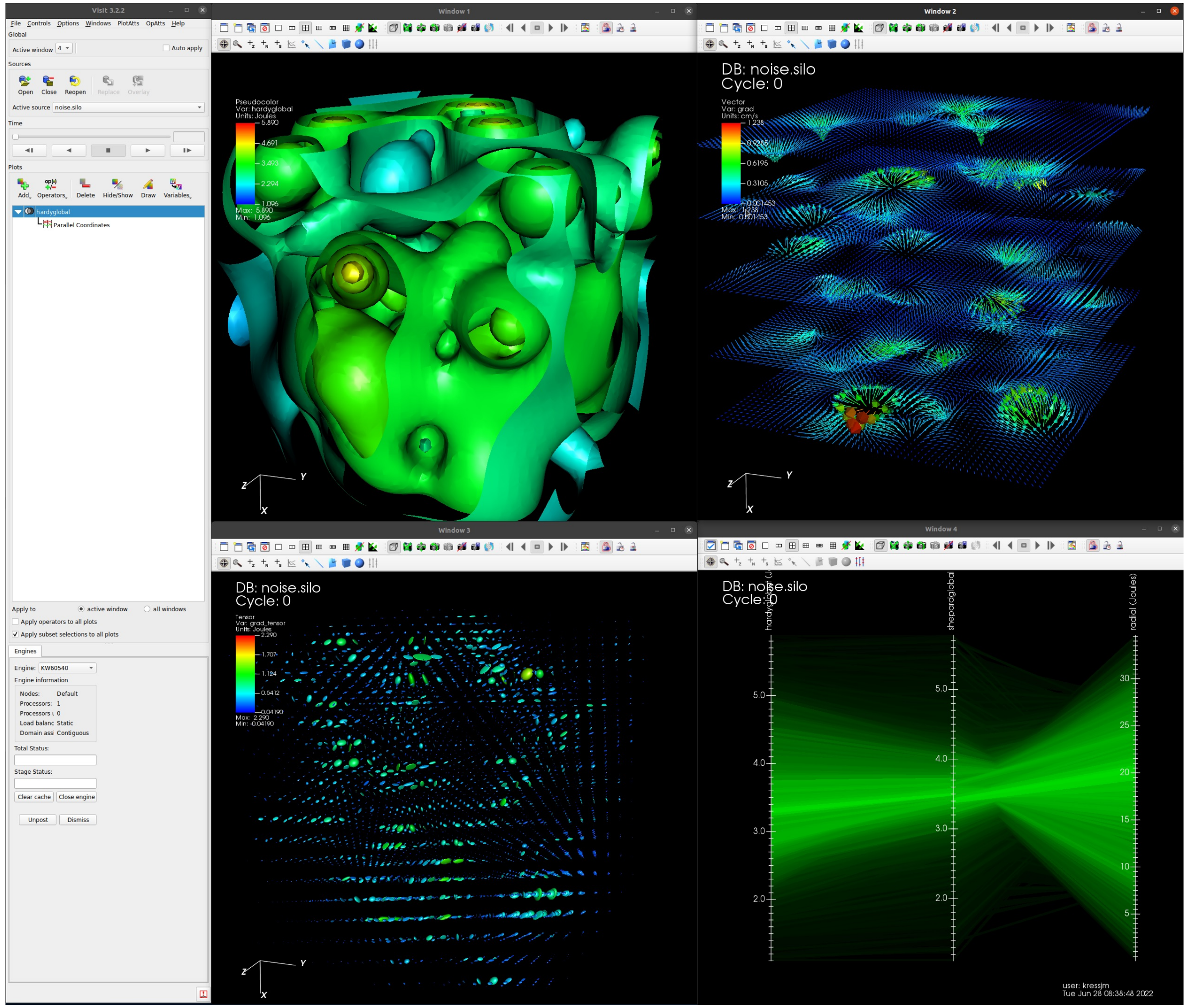
Engine information

Nodes: Default
Processors: 1
Processors ι 0
Load balanc Static
Domain assi Contiguous

Total Status:

Stage Status:

VisIt GUI Tour



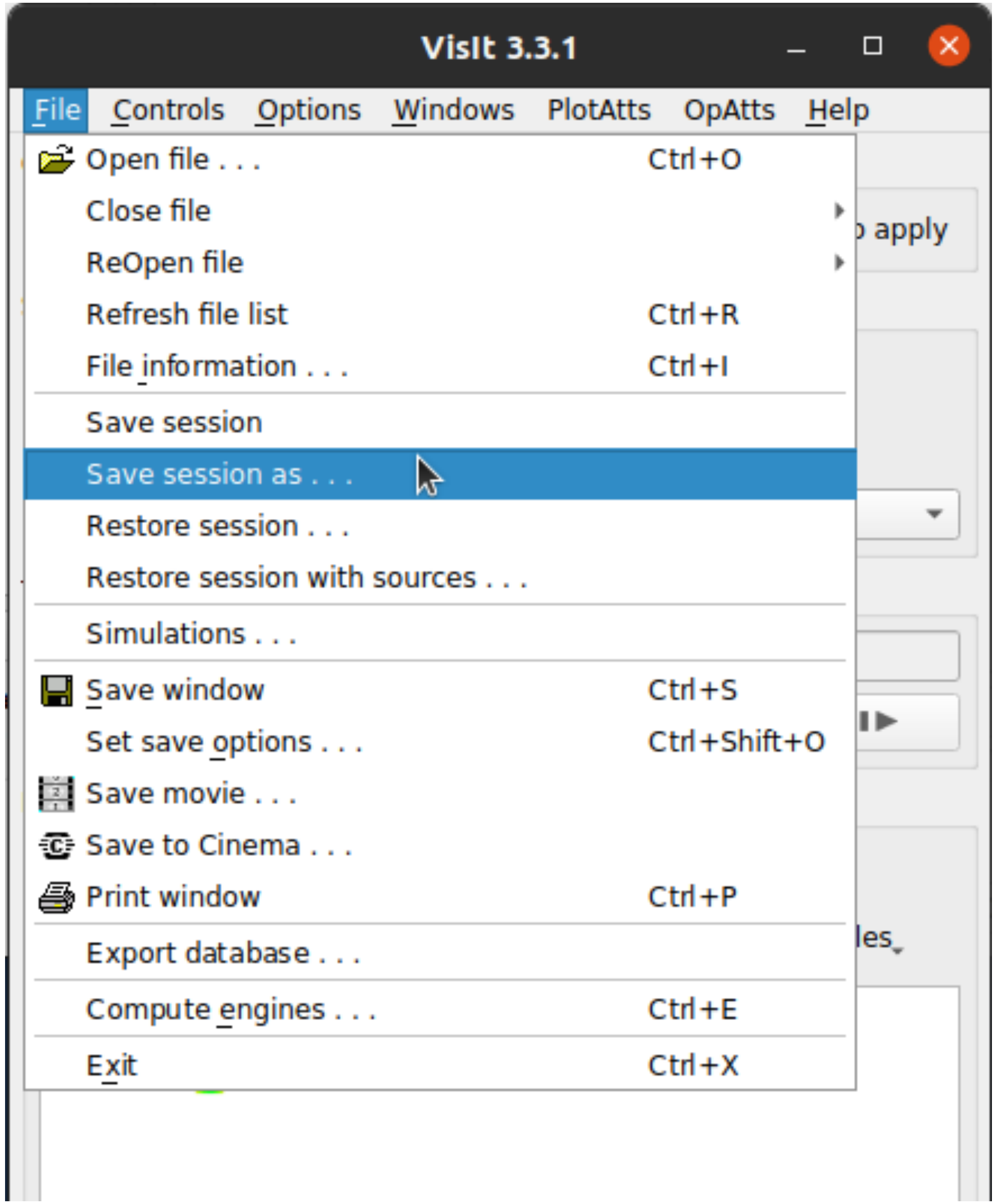
- Opening files / file types
- View file info
- Navigating views
- Multiple views
- Window tools
- Add plot / add operator
- Change plot / operator attributes
- Selectively applying operators
- Link views

Warm-up

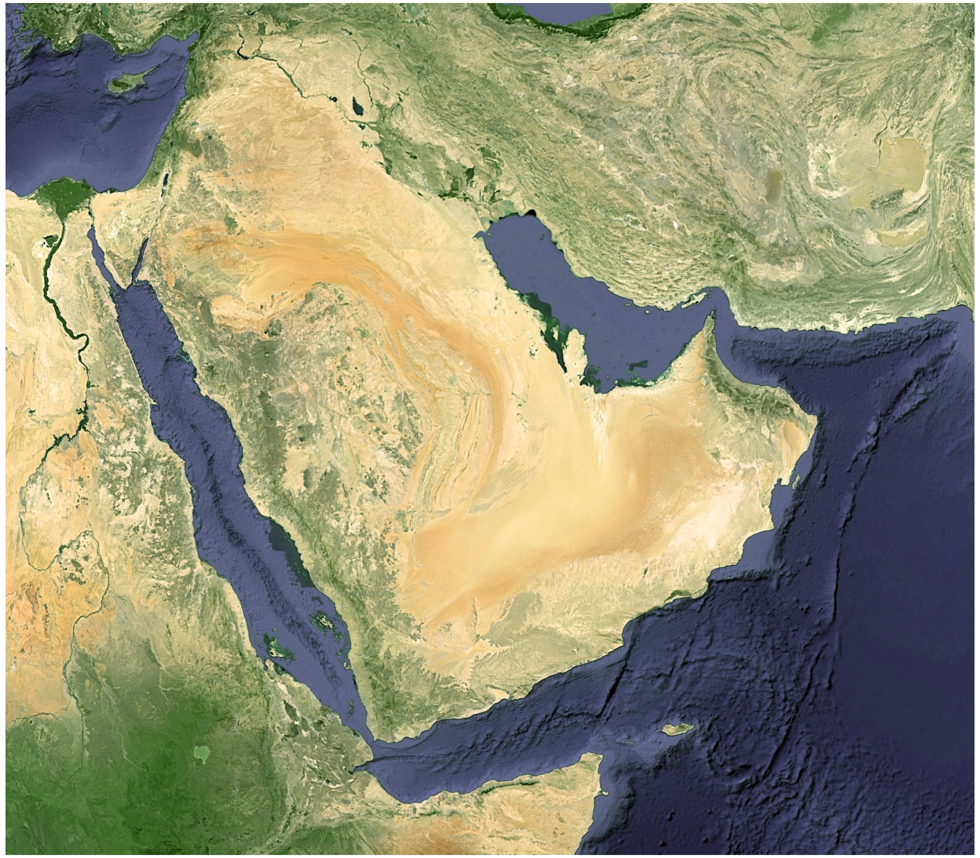
Basic plots, session files, small set of interesting operators

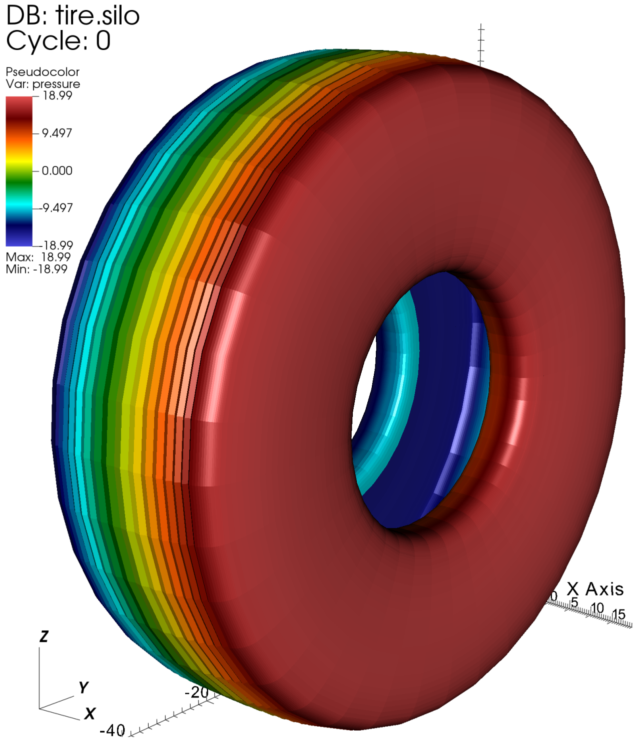
Session Files

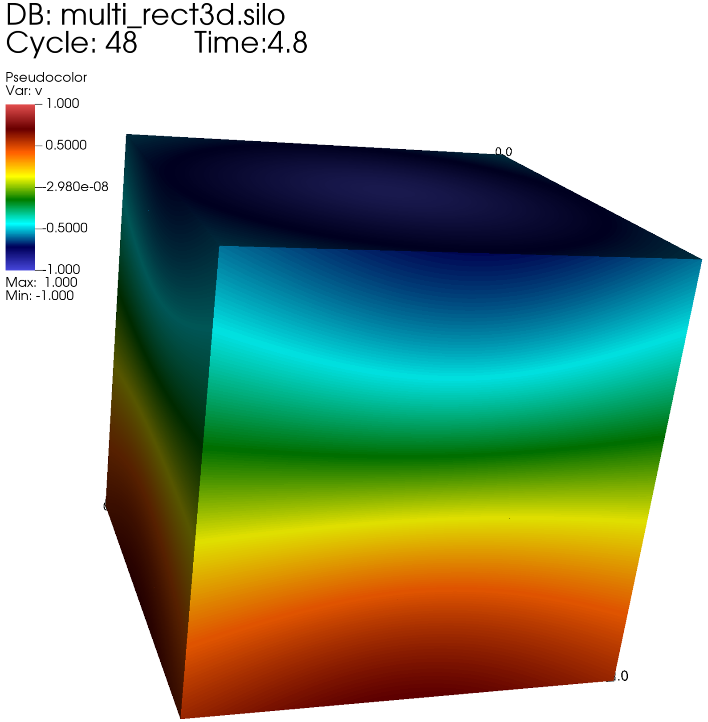
- A **session file** is an XML file that contains all of the necessary information to recreate the plots and visualization windows used in a VisIt session

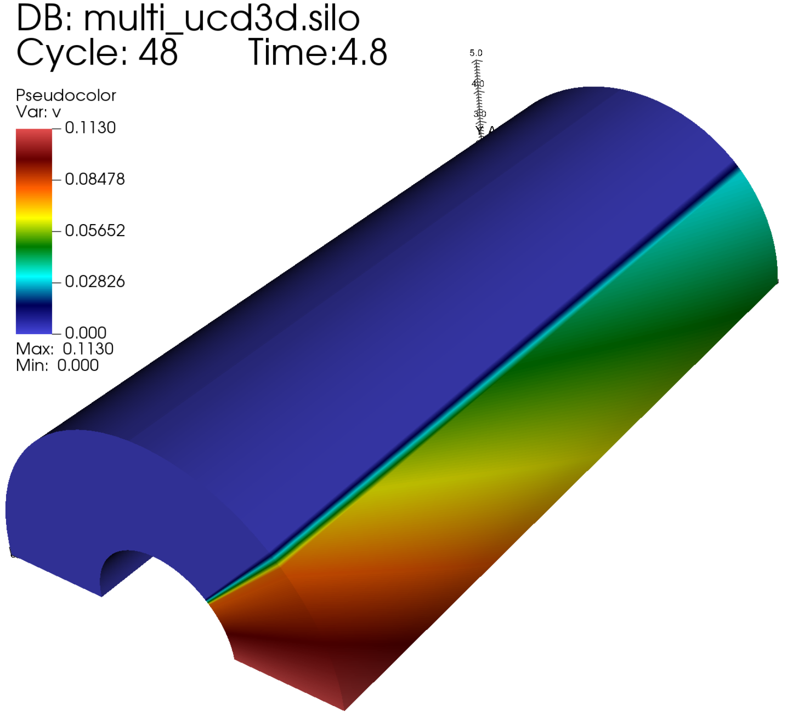


Data

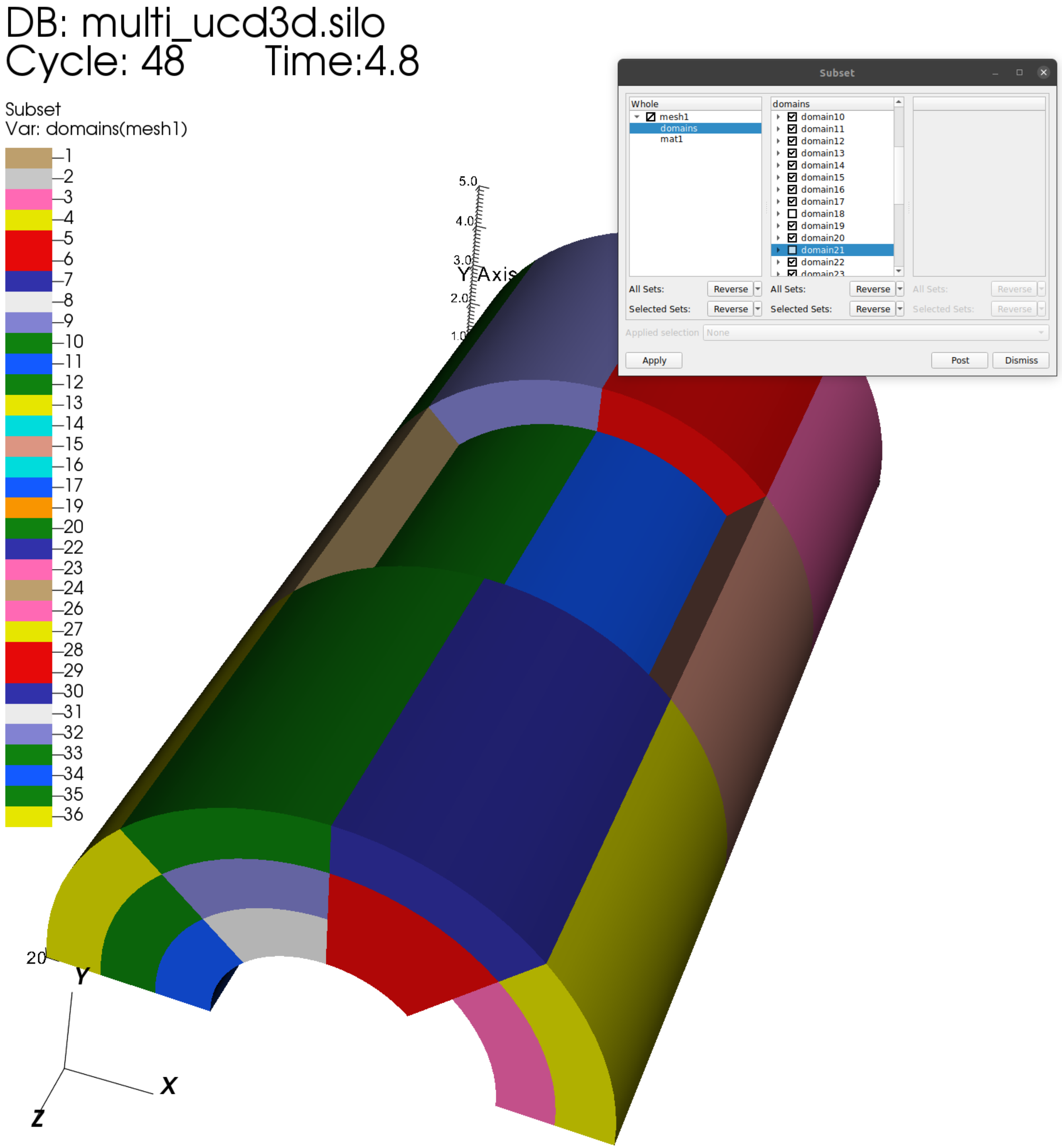



DB: tire.silo
Cycle: 0
Pseudocolor
Var: pressure
18.99
9.497
0.000
-9.497
-18.99
Max: 18.99
Min: -18.99
X Axis
0
5
10
15
Z
Y
X
-20
-40


DB: multi_rect3d.silo
Cycle: 48 Time:4.8
Pseudocolor
Var: v
1.000
0.5000
2.980e-08
-0.5000
-1.000
Max: 1.000
Min: -1.000
0.0


DB: multi_ucd3d.silo
Cycle: 48 Time:4.8
Pseudocolor
Var: v
0.1130
0.08478
0.05652
0.02826
0.000
Max: 0.1130
Min: 0.000
5.0

Subsets



DB: multi_ucd3d.silo
Cycle: 48 Time:4.8

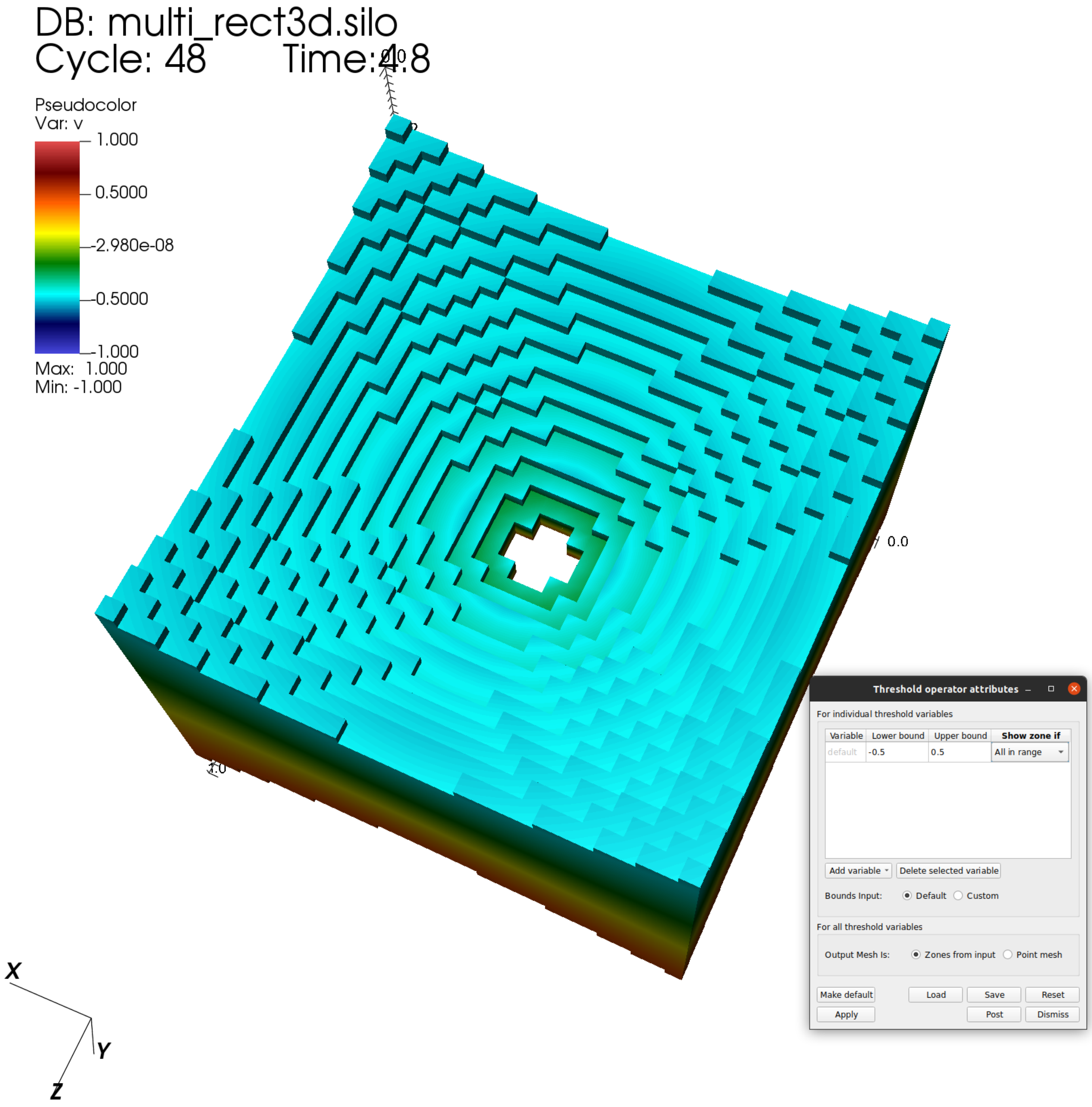
Subset
Var: domains(mesh1)

Whole
- ☒ mesh1
  - domains
  - mat1

domains
- ☑ domain10
- ☑ domain11
- ☑ domain12
- ☑ domain13
- ☑ domain14
- ☑ domain15
- ☑ domain16
- ☑ domain17
- ☐ domain18
- ☑ domain19
- ☑ domain20
- ☐ domain21
- ☑ domain22
- ☑ domain23

Applied selection: None

Threshold Operator



Threshold operator attributes

For individual threshold variables

Variable	Lower bound	Upper bound	Show zone if
default	-0.5	0.5	All in range

Add variable | Delete selected variable

Bounds Input: ● Default ○ Custom

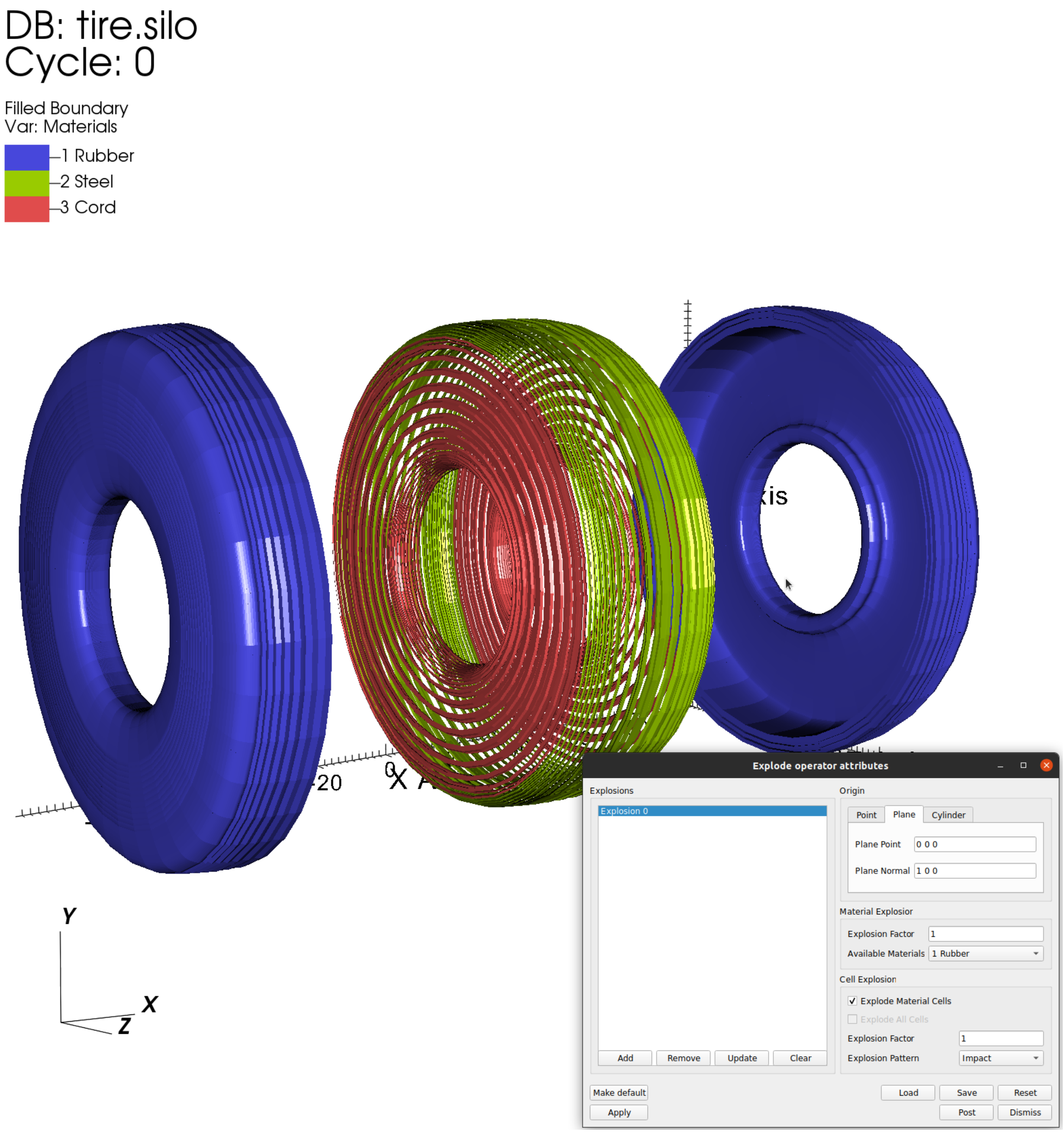
For all threshold variables

Output Mesh Is: ● Zones from input ○ Point mesh

Make default | Load | Save | Reset

Apply | Post | Dismiss

Explode Operator



Explode operator attributes

Explosions

- Explosion 0

Origin

Point | Plane | Cylinder

Plane Point: 0 0 0

Plane Normal: 1 0 0

Material Explosior

Explosion Factor: 1

Available Materials: 1 Rubber

Cell Explosion

☑ Explode Material Cells

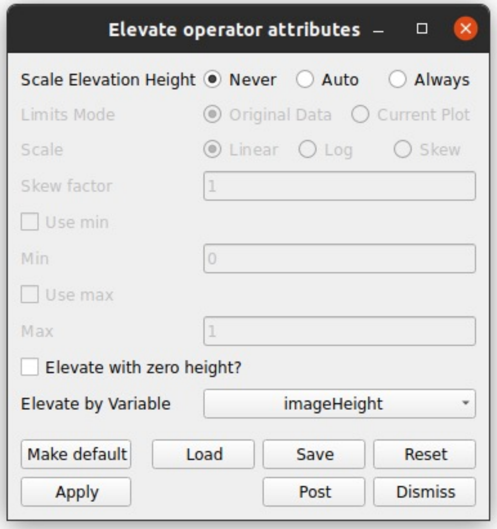
☐ Explode All Cells

Explosion Factor: 1

Explosion Pattern: Impact

Elevation Operator

- Elevate image by pixel intensity
  - Cross-mesh field evaluation



Elevate operator attributes

Scale Elevation Height ● Never ○ Auto ○ Always

Limits Mode ● Original Data ○ Current Plot

Scale ● Linear ○ Log ○ Skew

Skew factor 1

□ Use min

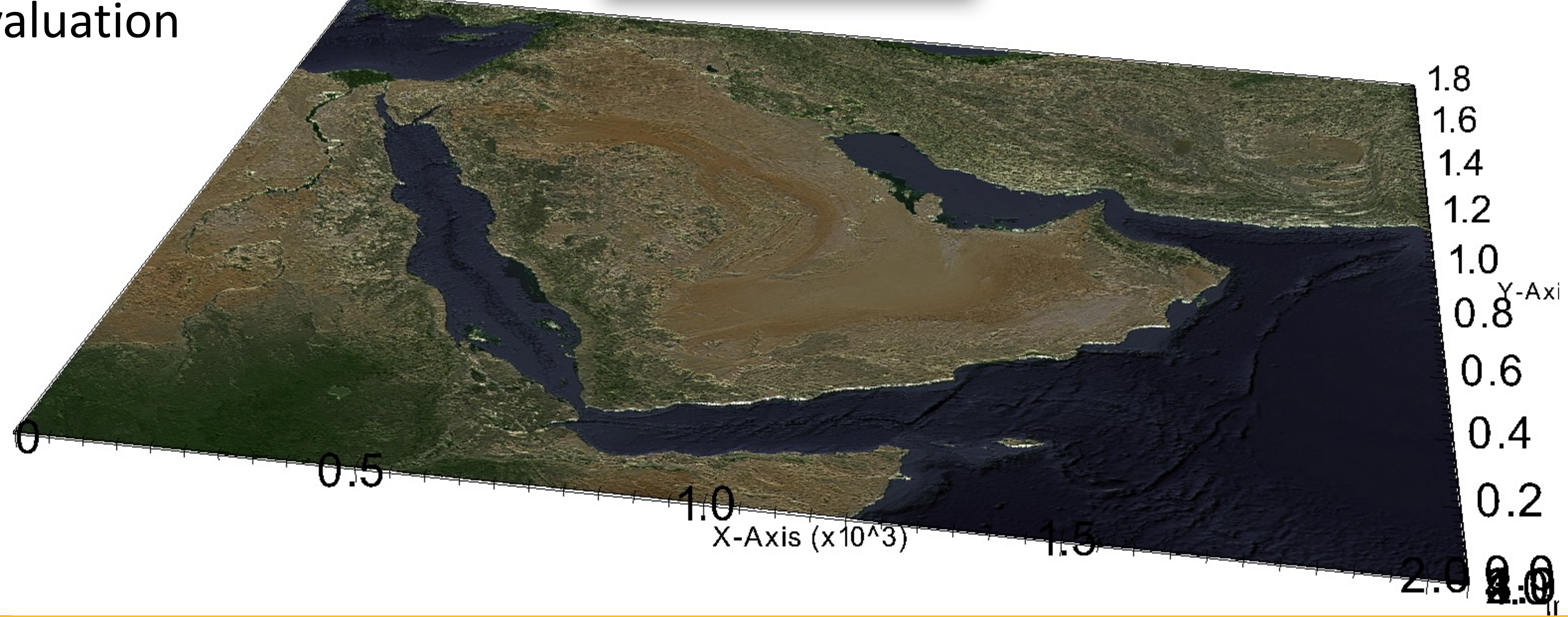
Min 0

□ Use max

Max 1

□ Elevate with zero height?

Elevate by Variable imageHeight



Hands–On Session 1

Ibex Interactive Visualization

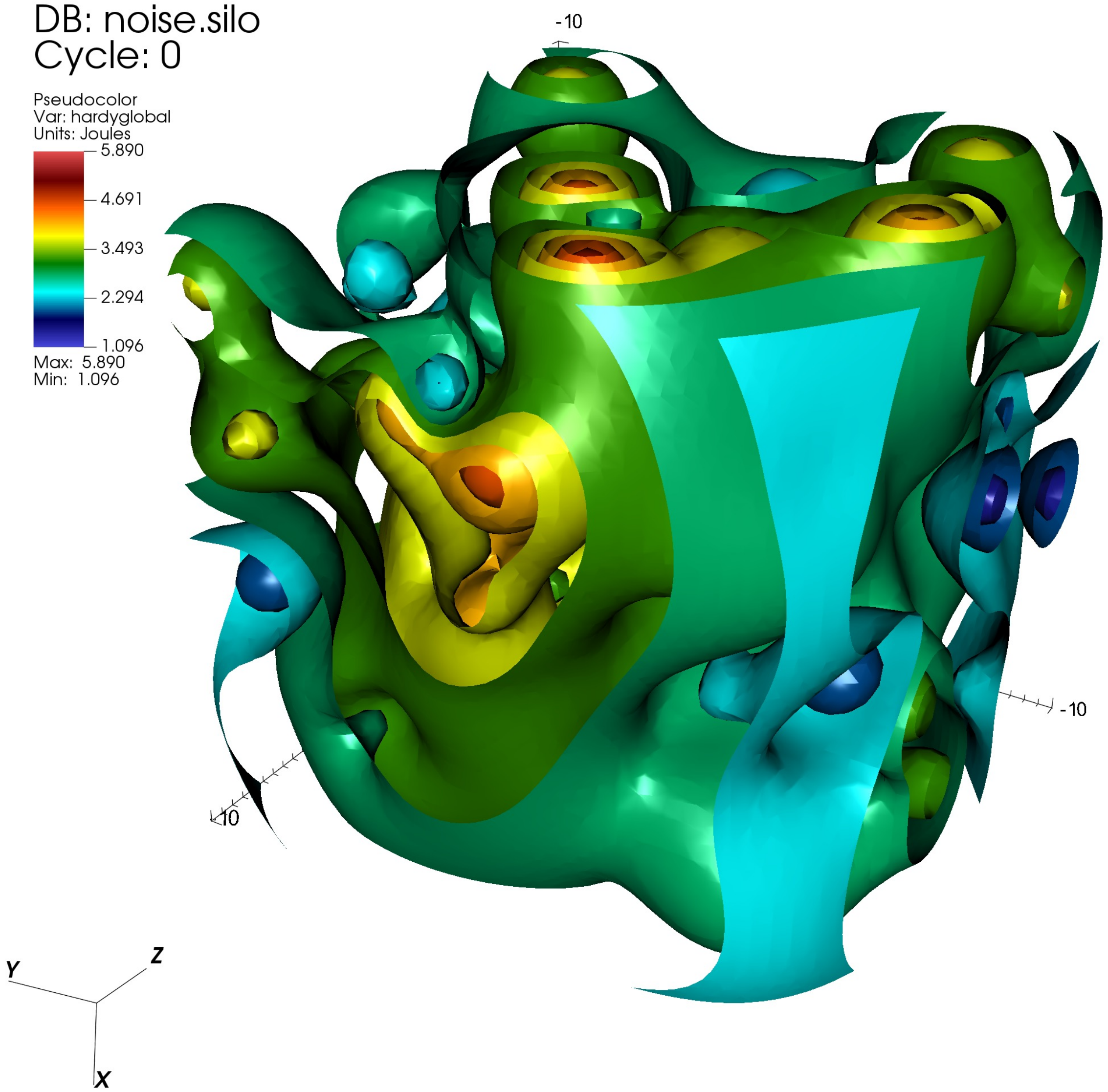
Why use VisIt on Ibex?

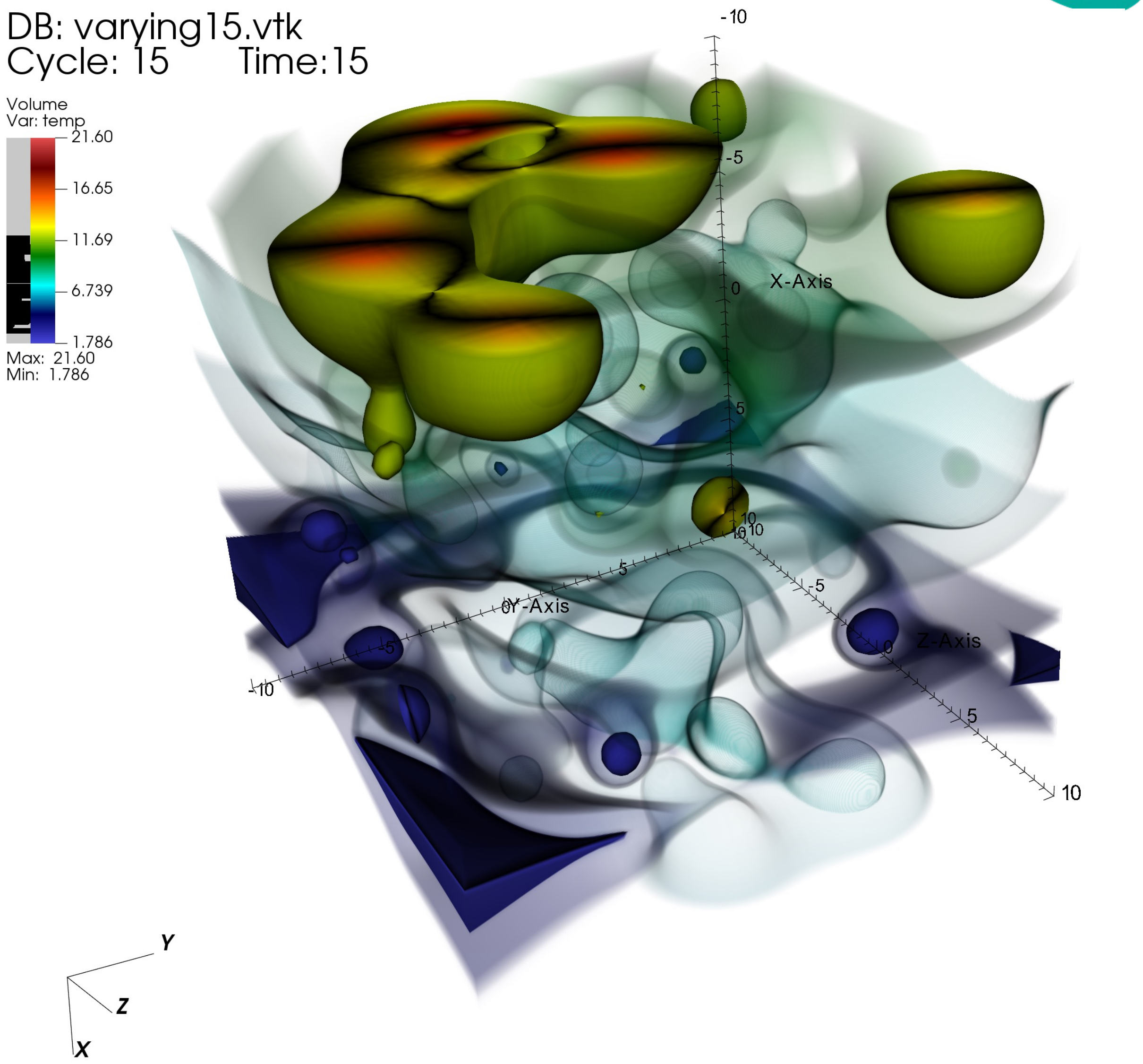
- Access to data generated on Ibex or Shaheen w/out copying
- Can use distributed computation and rendering for very large data
- Ability to run scripted batch visualization
- Ability to run client-server mode
  - VisIt GUI runs locally, all computation is done on Ibex
  - Allows for fast GUI interactions and distributed computation

Download Example Repo on Ibex

- Login to Ibex
  - *ssh <username>@ilogin.ibex.kaust.edu.sa*
- Navigate to scratch dir
  - *cd /ibex/scratch/<username>*
- Clone repo
  - *git clone https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes.git*

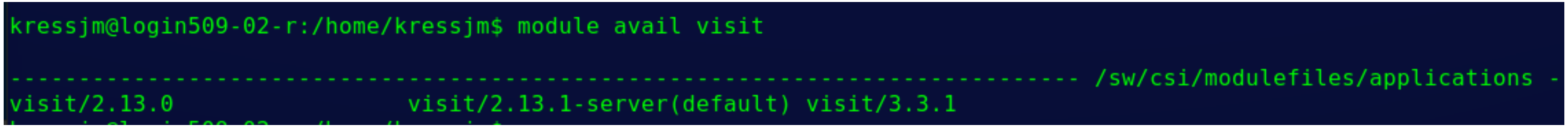
Repo Data


DB: noise.silo
Cycle: 0
Pseudocolor
Var: hardyglobal
Units: Joules
5.890
4.691
3.493
2.294
1.096
Max: 5.890
Min: 1.096
-10
-10
10
Y
Z
X


DB: varying15.vtk
Cycle: 15 Time:15
Volume
Var: temp
21.60
16.65
11.69
6.739
1.786
Max: 21.60
Min: 1.786
-10
-5
0
5
10
X-Axis
Y-Axis
Z-Axis
-10
-5
0
5
10
Y
Z
X

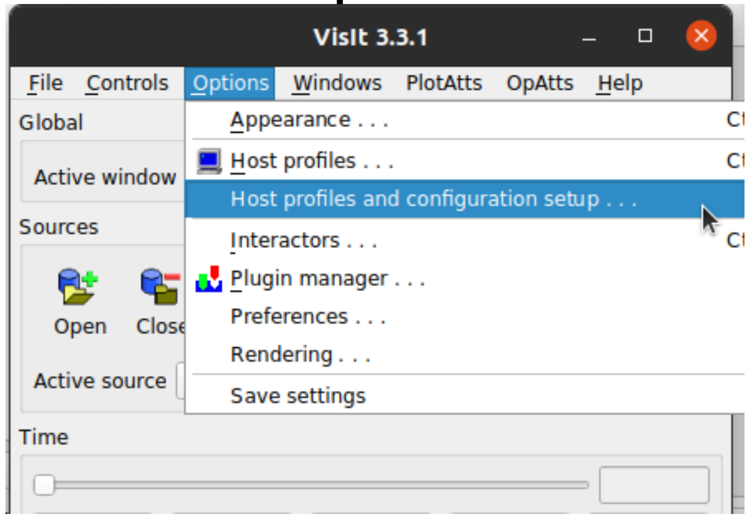
Initially Accessing VisIt on Ibex

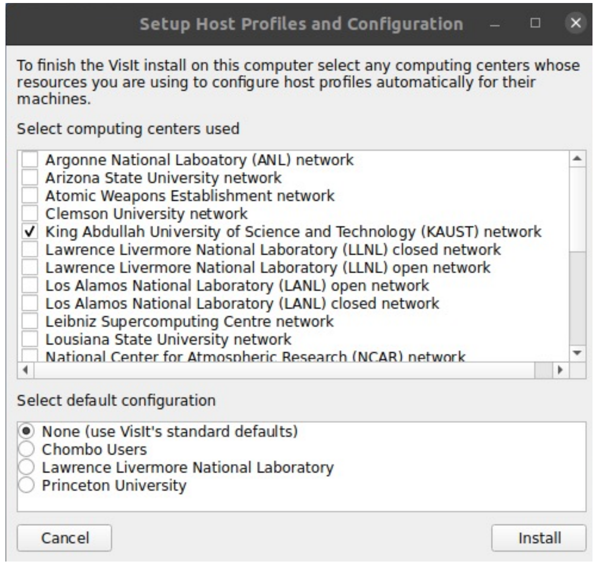
1. Check available VisIt versions on Ibex



```
kressjm@login509-02-r:/home/kressjm$ module avail visit

---------------------------------------------------------------- /sw/csi/modulefiles/applications -
visit/2.13.0             visit/2.13.1-server(default) visit/3.3.1
```

2. Download/use the latest VisIt version that ***matches*** Ibex
3. If first time using VisIt on Ibex, load the KAUST profiles
    1. Click *<Options/Host profiles and …>*
    2. Select KAUST network
    3. Click *<Install>*
    4. Save settings *<Options/Save Settings>*
    5. Relaunch VisIt





To finish the VisIt install on this computer select any computing centers whose resources you are using to configure host profiles automatically for their machines.

Select computing centers used

- □ Argonne National Laboatory (ANL) network
- □ Arizona State University network
- □ Atomic Weapons Establishment network
- □ Clemson University network
- ☑ King Abdullah University of Science and Technology (KAUST) network
- □ Lawrence Livermore National Laboratory (LLNL) closed network
- □ Lawrence Livermore National Laboratory (LLNL) open network
- □ Los Alamos National Laboratory (LANL) open network
- □ Los Alamos National Laboratory (LANL) closed network
- □ Leibniz Supercomputing Centre network
- □ Lousiana State University network
- □ National Center for Atmospheric Research (NCAR) network

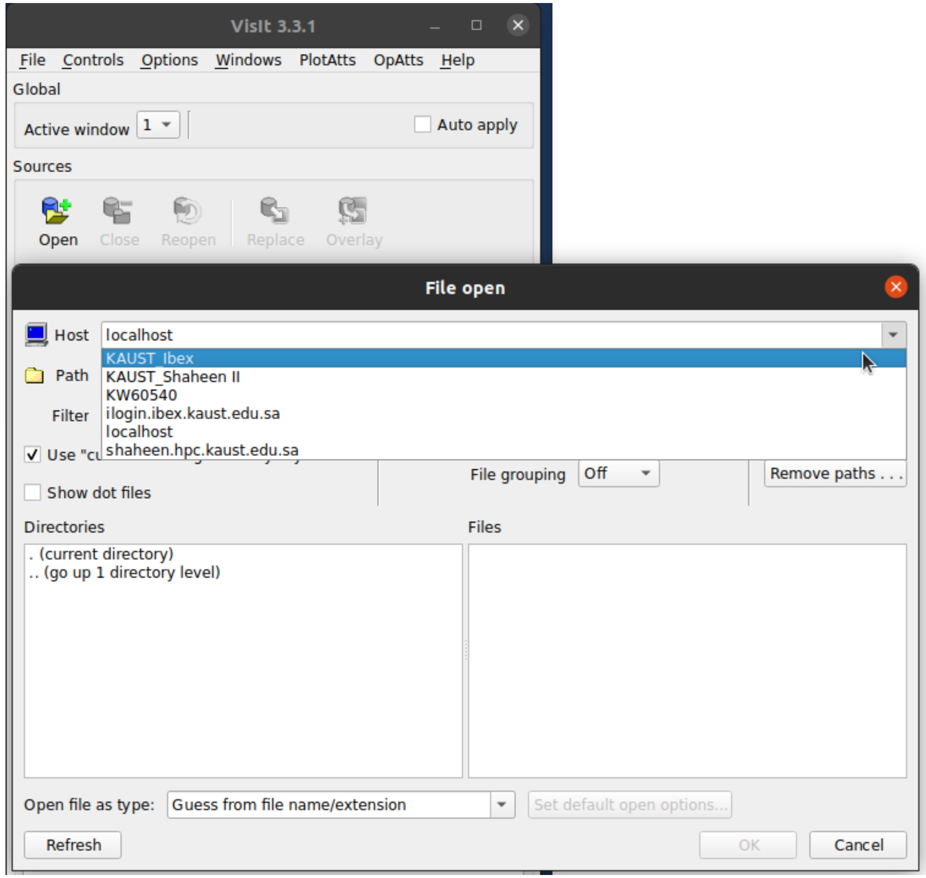
Select default configuration

- ● None (use VisIt's standard defaults)
- ○ Chombo Users
- ○ Lawrence Livermore National Laboratory
- ○ Princeton University

Using VisIt Interactively on Ibex

- Open VisIt on your local computer
- Go to: *<File/Open file>* or click the *<Open>* button on the GUI
- Click the *<Host>* dropdown menu on the "File open" window that popped up and choose "Ibex"
- This will prompt you for your Ibex password, unless you have passwordless ssh setup
- Navigate to the file you want to process
- Once you choose a file, you will be prompted for the number of nodes and processors you would like to use ***(for now, use 2 processes and 1 node)***
- Once specified, the server side of VisIt will be launched, and you can interact with your data (after the job launches and reaches to top of the Ibex queue)

Using VisIt Interactively on Ibex Cont.



Active window 1 ▾ □ Auto apply

Host localhost

KAUST_Ibex
KAUST_Shaheen II
KW60540
ilogin.ibex.kaust.edu.sa
localhost
shaheen.hpc.kaust.edu.sa

Path

Filter

☑ Use "cu

File grouping Off

□ Show dot files

Directories

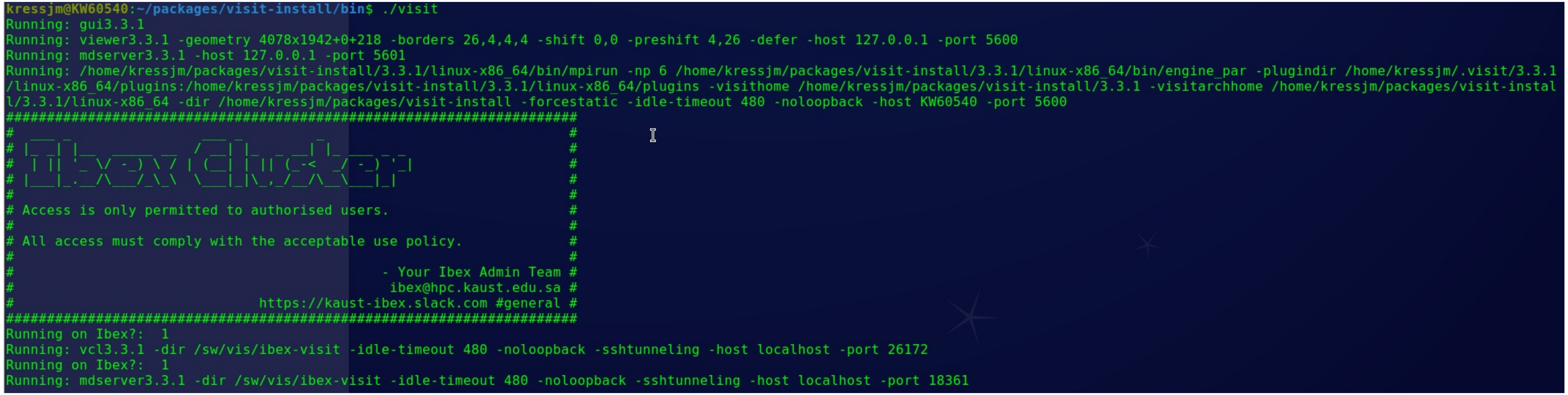
. (current directory)
.. (go up 1 directory level)

Files

Open file as type: Guess from file name/extension

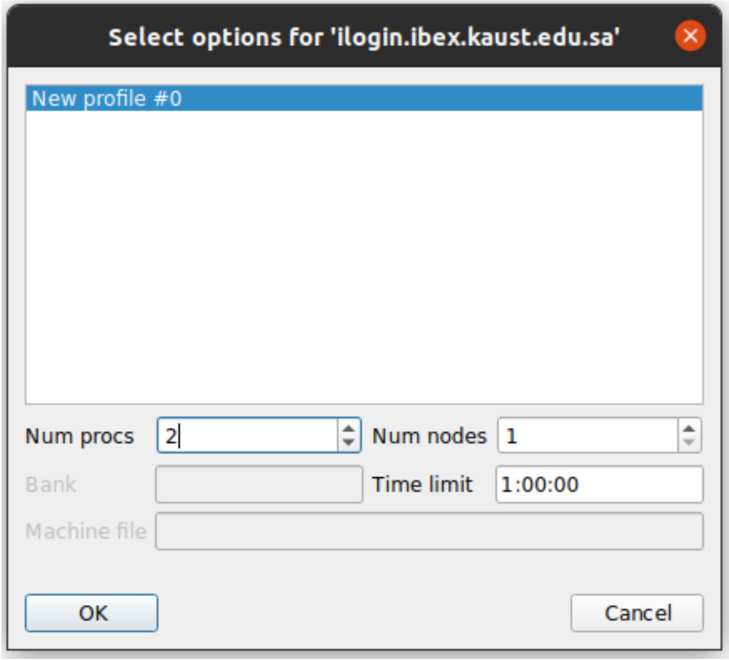
vcl – *VisIt Component Launcher (manages VisIt session)*

mdserver – *VisIt metadata server (interacts with GUI and databases)*

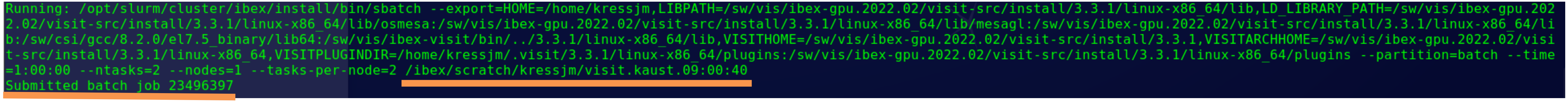


```
kressjm@KW60540:~/packages/visit-install/bin$ ./visit
Running: gui3.3.1
Running: viewer3.3.1 -geometry 4078x1942+0+218 -borders 26,4,4,4 -shift 0,0 -preshift 4,26 -defer -host 127.0.0.1 -port 5600
Running: mdserver3.3.1 -host 127.0.0.1 -port 5601
Running: /home/kressjm/packages/visit-install/3.3.1/linux-x86_64/bin/mpirun -np 6 /home/kressjm/packages/visit-install/3.3.1/linux-x86_64/bin/engine_par -plugindir /home/kressjm/.visit/3.3.1
/linux-x86_64/plugins:/home/kressjm/packages/visit-install/3.3.1/linux-x86_64/plugins -visithome /home/kressjm/packages/visit-install/3.3.1 -visitarchhome /home/kressjm/packages/visit-instal
l/3.3.1/linux-x86_64 -dir /home/kressjm/packages/visit-install -forcestatic -idle-timeout 480 -noloopback -host KW60540 -port 5600
####################################################################
#   ___ _              ___ _         _                             #
#  |_ _| |__  _____ __/ __| |_  _ __| |_ ___ _ _                   #
#   | || '_ \/ -_) \ / (__| | || (_-<  _/ -_) '_|                  #
#  |___|_.__/\___/_\_\\___|_|\_,_/__/\__\___|_|                    #
#                                                                  #
# Access is only permitted to authorised users.                    #
#                                                                  #
# All access must comply with the acceptable use policy.           #
#                                                                  #
#                                             - Your Ibex Admin Team #
#                                               ibex@hpc.kaust.edu.sa #
#                              https://kaust-ibex.slack.com #general #
####################################################################
Running on Ibex?:  1
Running: vcl3.3.1 -dir /sw/vis/ibex-visit -idle-timeout 480 -noloopback -sshtunneling -host localhost -port 26172
Running on Ibex?:  1
Running: mdserver3.3.1 -dir /sw/vis/ibex-visit -idle-timeout 480 -noloopback -sshtunneling -host localhost -port 18361
```

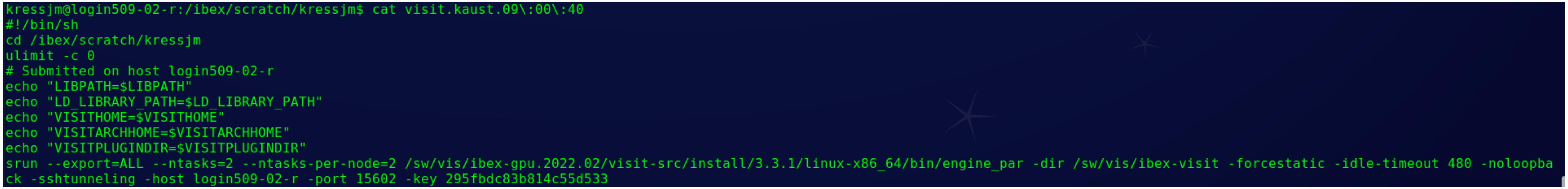
Using VisIt Interactively on Ibex Cont.



Num procs 2
Num nodes 1
Time limit 1:00:00



```
Running: /opt/slurm/cluster/ibex/install/bin/sbatch --export=HOME=/home/kressjm,LIBPATH=/sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1/linux-x86_64/lib,LD_LIBRARY_PATH=/sw/vis/ibex-gpu.202
2.02/visit-src/install/3.3.1/linux-x86_64/lib/osmesa:/sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1/linux-x86_64/lib/mesagl:/sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1/linux-x86_64/li
b:/sw/csi/gcc/8.2.0/el7.5_binary/lib64:/sw/vis/ibex-visit/bin/../3.3.1/linux-x86_64/lib,VISITHOME=/sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1,VISITARCHHOME=/sw/vis/ibex-gpu.2022.02/visi
t-src/install/3.3.1/linux-x86_64,VISITPLUGINDIR=/home/kressjm/.visit/3.3.1/linux-x86_64/plugins:/sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1/linux-x86_64/plugins --partition=batch --time
=1:00:00 --ntasks=2 --nodes=1 --tasks-per-node=2 /ibex/scratch/kressjm/visit.kaust.09:00:40
Submitted batch job 23496397
```

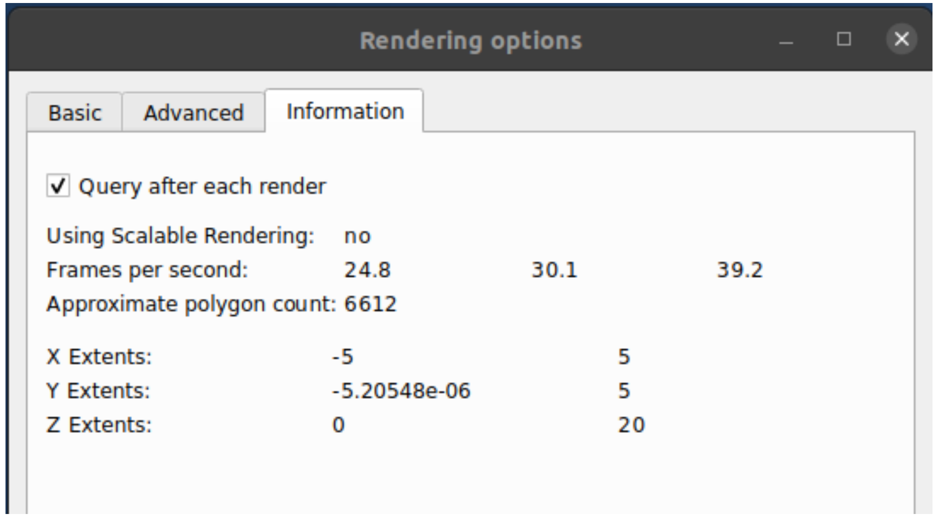


```
kressjm@login509-02-r:/ibex/scratch/kressjm$ cat visit.kaust.09\:00\:40
#!/bin/sh
cd /ibex/scratch/kressjm
ulimit -c 0
# Submitted on host login509-02-r
echo "LIBPATH=$LIBPATH"
echo "LD_LIBRARY_PATH=$LD_LIBRARY_PATH"
echo "VISITHOME=$VISITHOME"
echo "VISITARCHHOME=$VISITARCHHOME"
echo "VISITPLUGINDIR=$VISITPLUGINDIR"
srun --export=ALL --ntasks=2 --ntasks-per-node=2 /sw/vis/ibex-gpu.2022.02/visit-src/install/3.3.1/linux-x86_64/bin/engine_par -dir /sw/vis/ibex-visit -forcestatic -idle-timeout 480 -noloopba
ck -sshtunneling -host login509-02-r -port 15602 -key 295fbdc83b814c55d533
```

engine_par– *VisIt parallel computation engine*

Explore Example Repo Data Sets

- Load each of the example data sets and try different visualizations
- Note on rendering
  - VisIt has two rendering modes
    - Transfer data to client for rendering
      - Done when data is small
    - Transfer images to client, rendering on the server
      - This is how VisIt can render extremely large data on clusters
      - This is called scalable rendering
      - You can turn on/off scalable rendering, see stats, and other options @ <*Options/Rendering*>



☑ Query after each render

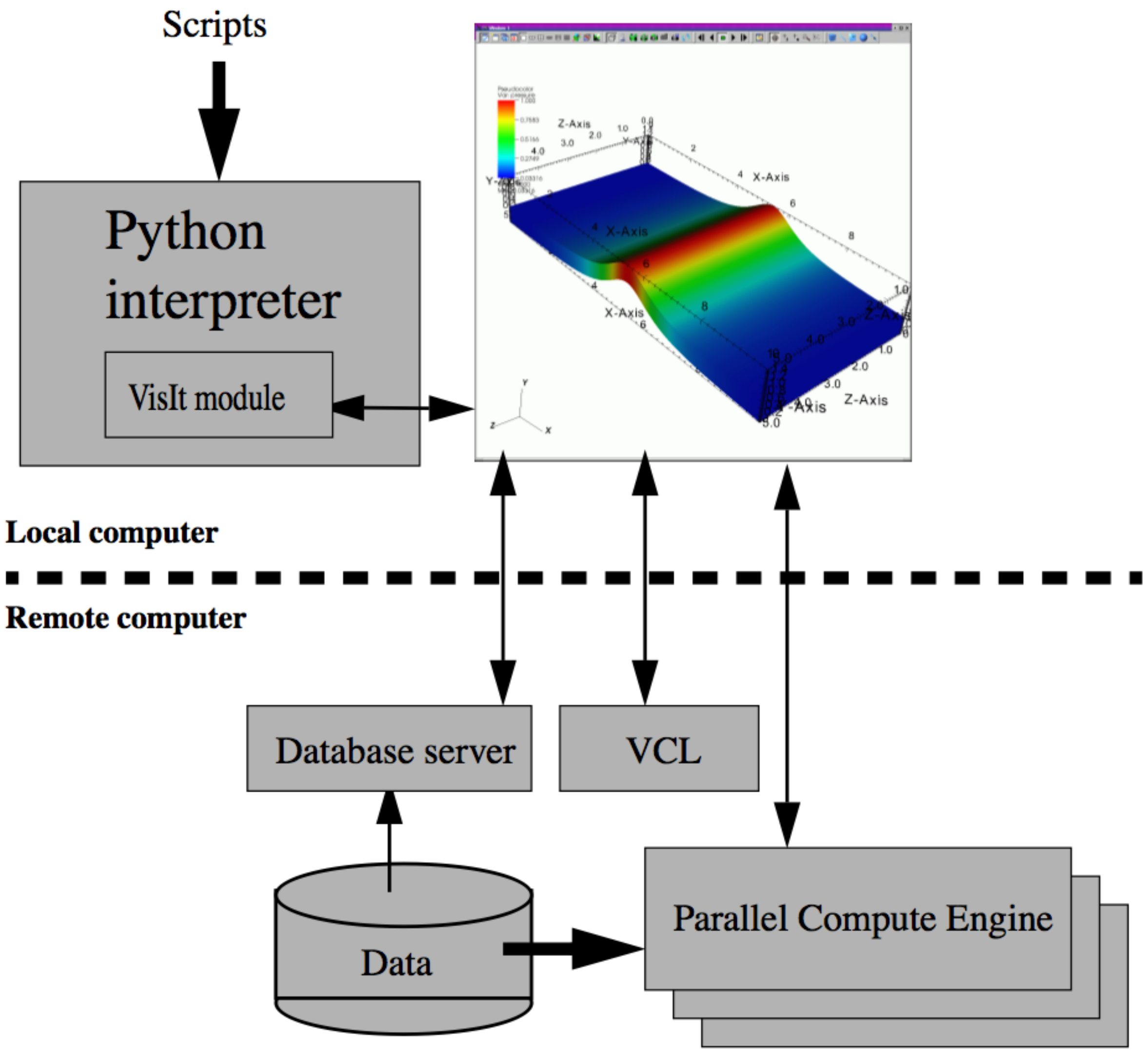
```
Using Scalable Rendering:   no
Frames per second:          24.8          30.1          39.2
Approximate polygon count:  6612

X Extents:                 -5                    5
Y Extents:                 -5.20548e-06          5
Z Extents:                  0                    20
```

Hands–On Session 2

Scripting Visualization within VisIt

VisIt and Python

- VisIt can be used from python
  - *import sys*
  - *sys.path.append("/path/to/visit/<version>/<architecture>/lib/site-packages")*
    import visit
  - *visit.Launch()*
- Python can be used within VisIt



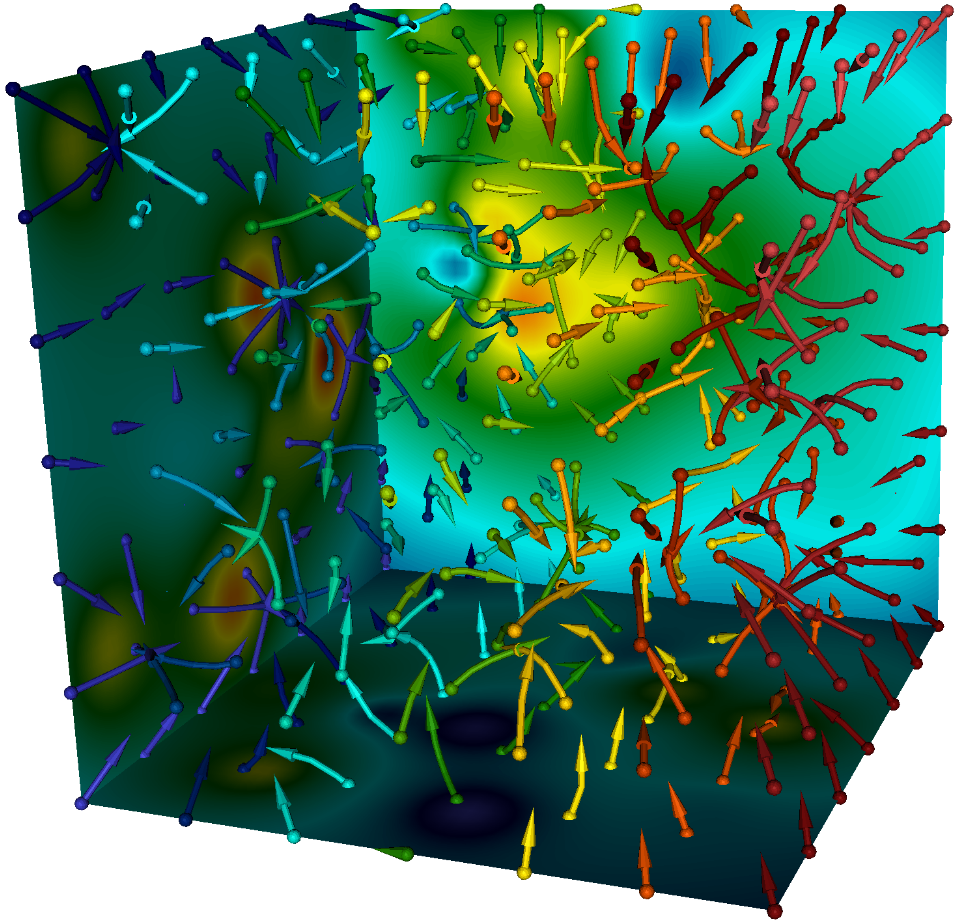
Using VisIt GUI and cli Simultaneously

- Open VisIt
  - Open command window: *<Controls/Command>*

- *Go to VisIt Docs*
  - https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/tutorials/Scripting.html
  - We'll walk through some of the initial copy-paste examples

Running a Script Interactively in VisIt

- Works just like the previous examples, but code is in a file
  - Use your favorite editor
  - Have more complicated multi-file scripts



- Enter the following in the cli and click <*Execute*>
  - Source("/path/to/KAUST_Visualization_Vignettes/VisIt_Vignettes/ex04_visitStreamlineAnimation/ex04_visitStreamlineAnimation.py")

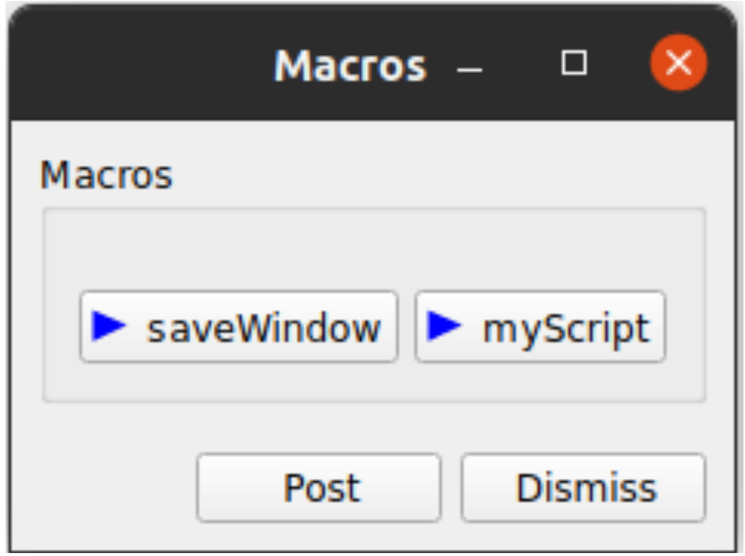
Tracing Your Actions

- Open VisIt
  - Open command window: *<Controls/Command>*
  - Open an empty tab
  - Click *<Record>*
  - Interact with the GUI to do the vis you want
  - Click *<Stop>*
  - A lengthy trace will reproduce your actions
    - VisIt prints all possible options for your actions, so you can prune lots of the code away if you are not changing default options

Creating a Macro

save window example

- Open VisIt
  - Create a basic plot
  - Open command window: *<Controls/Command>*
    - Open an empty tab
    - Click *<Record>*
    - *<File/Save Window>*
    - Click *<Stop>*
    - Click *<Make macro>*
  - Click *<Controls/Macros>*
    - You can now run this script anytime with 1-click

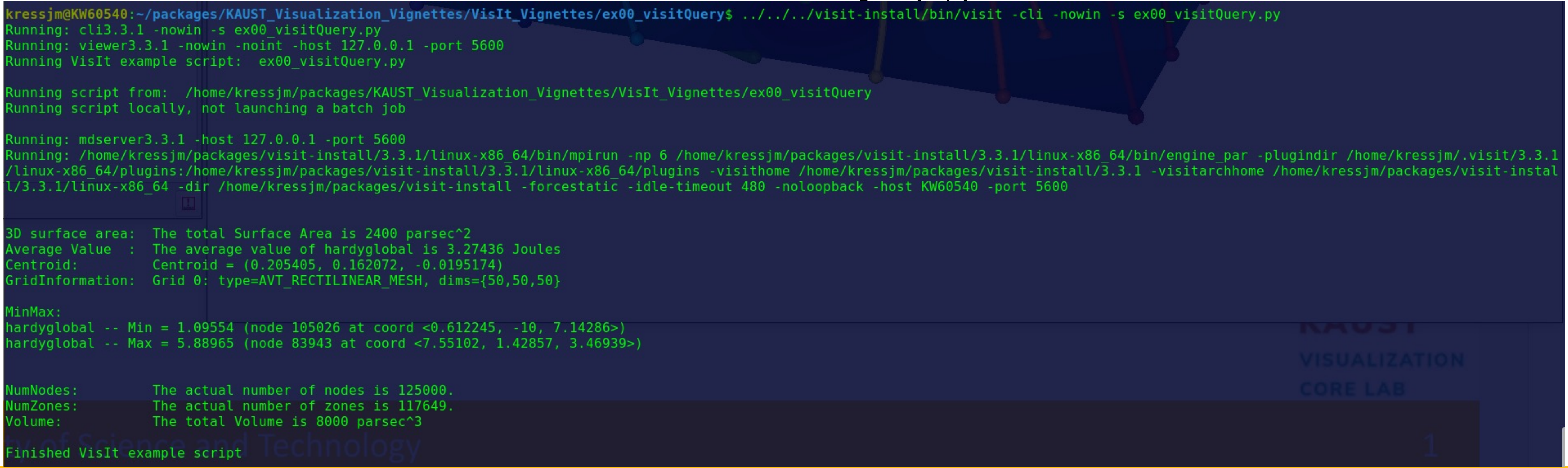


Hands–On Session 3

Scripting Visualization from Command Line

Running Scripts without VisIt GUI

- Navigate to VisIt_Vignettes repo folder on your local computer
  - Run each of the six examples
    - cd to individual example directory
    - *visit –nowin –cli –v 3.3.1 -s ex00_visitQuery.py*



```
kressjm@KW60540:~/packages/KAUST_Visualization_Vignettes/VisIt_Vignettes/ex00_visitQuery$ ../../../visit-install/bin/visit -cli -nowin -s ex00_visitQuery.py
Running: cli3.3.1 -nowin -s ex00_visitQuery.py
Running: viewer3.3.1 -nowin -noint -host 127.0.0.1 -port 5600
Running VisIt example script:  ex00_visitQuery.py

Running script from:  /home/kressjm/packages/KAUST_Visualization_Vignettes/VisIt_Vignettes/ex00_visitQuery
Running script locally, not launching a batch job

Running: mdserver3.3.1 -host 127.0.0.1 -port 5600
Running: /home/kressjm/packages/visit-install/3.3.1/linux-x86_64/bin/mpirun -np 6 /home/kressjm/packages/visit-install/3.3.1/linux-x86_64/bin/engine_par -plugindir /home/kressjm/.visit/3.3.1
/linux-x86_64/plugins:/home/kressjm/packages/visit-install/3.3.1/linux-x86_64/plugins -visithome /home/kressjm/packages/visit-install/3.3.1 -visitarchhome /home/kressjm/packages/visit-instal
l/3.3.1/linux-x86_64 -dir /home/kressjm/packages/visit-install -forcestatic -idle-timeout 480 -noloopback -host KW60540 -port 5600


3D surface area:  The total Surface Area is 2400 parsec^2
Average Value  :  The average value of hardyglobal is 3.27436 Joules
Centroid:         Centroid = (0.205405, 0.162072, -0.0195174)
GridInformation:  Grid 0: type=AVT_RECTILINEAR_MESH, dims={50,50,50}

MinMax:
hardyglobal -- Min = 1.09554 (node 105026 at coord <0.612245, -10, 7.14286>)
hardyglobal -- Max = 5.88965 (node 83943 at coord <7.55102, 1.42857, 3.46939>)


NumNodes:         The actual number of nodes is 125000.
NumZones:         The actual number of zones is 117649.
Volume:           The total Volume is 8000 parsec^3

Finished VisIt example script
```

Hands–On Session 4

Scripting Visualization on Ibex

Running Examples on Ibex

- Login to Ibex
  - *ssh -X <username>@ilogin.ibex.kaust.edu.sa*
- Navigate to example dir
  - *cd /ibex/scratch/<username>/KAUST_Visualization_Vignettes/Visit_Vignettes*
- Run individual examples
  - *cd ex00_visitQuery*
    - *sbatch ex00_ibex_runScript.sbat*
    - View queue info: *squeue –u username*
    - When job finishes view results: *cat ex00.ibex.*.out*
  - *cd ex01_visitScreenshot*
    - *sbatch ex01_ibex_runScript.sbat*
    - When job finishes view image: *display output/ex01_visit000.png*
  - *ex02… etc.*

Access Data on Ibex

- Linux
  - scp –r username@ilogin.ibex.kaust.edu.sa:/path/to/files <local/destination>
  - Mount scratch locally as a folder:
    - sshfs username@mover.ibex.kaust.edu.sa:/ibex/scratch/username <local/destination>
- Mac
  - scp –r username@ilogin.ibex.kaust.edu.sa:/path/to/files <local/destination>
  - Mount scratch locally as a folder:
    - sshfs username@mover.ibex.kaust.edu.sa:/ibex/scratch/username <local/destination>
- Windows
  - scp –r username@ilogin.ibex.kaust.edu.sa:/path/to/files <local/destination>
  - Mount scratch locally as lettered drive:
    - Run SFTP Drive and connect: mover.ibex.kaust.edu.sa; drive path: /ibex/scratch/<username>

Demo

Scripting Visualization on Shaheen

Running Examples on Shaheen

- Login to Shaheen
  - *ssh -X <username>@shaheen.hpc.kaust.edu.sa*
- Navigate to example dir (download repo as before)
  - *cd /scratch/<username>/KAUST_Visualization_Vignettes/Visit_Vignettes*
- Run individual examples
  - *cd ex00_visitQuery*
    - Modify run script with your project #: *--account=<##>*
    - *sbatch ex00_shaheen_runScript.sbat*
    - View queue info: *squeue –u username*
    - When job finishes view results: *cat ex00.ibex.*.out*
  - *cd ex01_visitScreenshot*
    - Modify run script with your project #: *--account=<##>*
    - *sbatch ex01_shaheen_runScript.sbat*
    - When job finishes view image: eog *output/ex01_visit000.png*
  - *ex02… etc.*

VisIt Wrap-up

Best Practices

How do I use ParaView or VisIt?

- If your data is small/manageable
  - Do your visualizations on your laptop, desktop, or IT Remote Workstation
- If your data is medium/large
  - Do interactive visualization on Ibex
    - Run it on your local machine and connect directly to Ibex to load/process/visualize
    - https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes/-/tree/master/ParaView_Vignettes#using-paraview-interactively-on-ibex
    - https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes/-/tree/master/VisIt_Vignettes#using-visit-interactively-on-ibex
- If your data is large/huge and you have a defined workflow
  - Do batch visualization on Shaheen
    - https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes/-/tree/master/VisIt_Vignettes#expy
    - https://gitlab.kaust.edu.sa/kvl/KAUST_Visualization_Vignettes/-/tree/master/ParaView_Vignettes#expy
- If you have repeatable repetitive tasks
  - Do scripted or batch visualization

Thanks!

Contacts:

james.kress@kaust.edu.sa

help@vis.kaust.edu