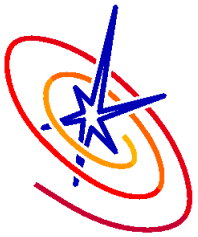


VisIt Visualization Tool

Anthony Scopatz

Original Slides by Randy Hudson

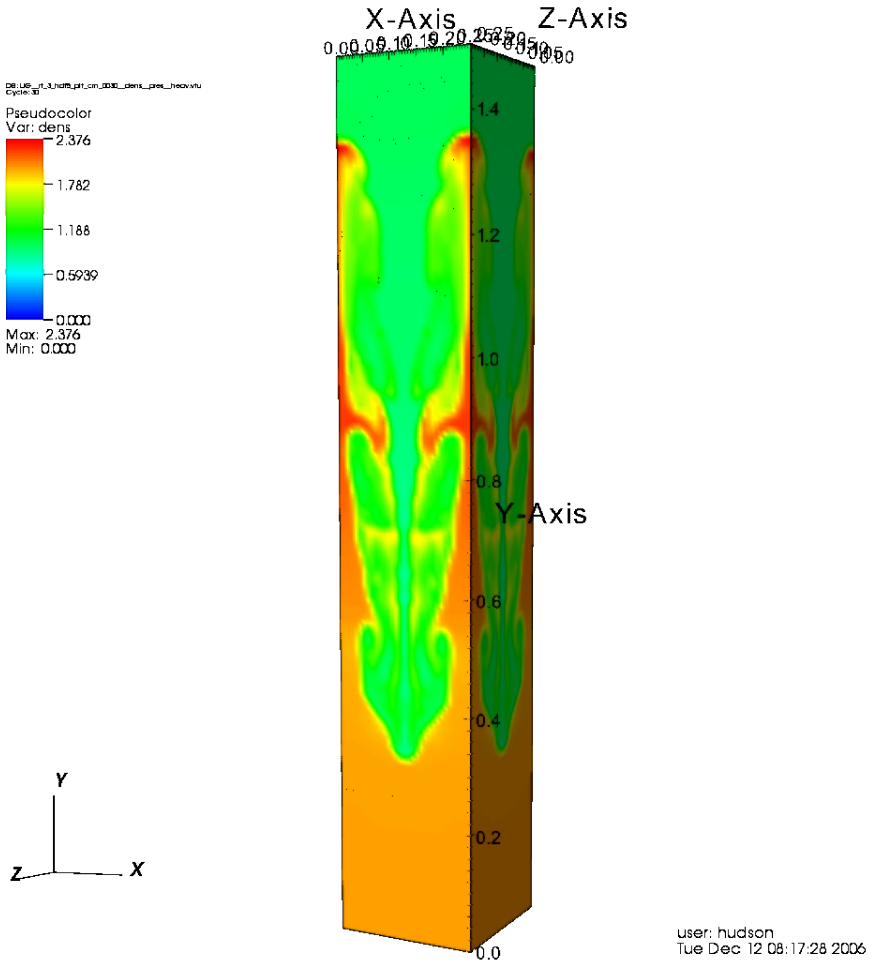
The FLASH Center, University of Chicago

Tutorial

- Lecture
- Practice

Lecture

- About VisIt
- Using VisIt



About VisIt

- Developed by DOE's Advanced Simulation & Computing Initiative
- Continued by DOE's Office of Nuclear Energy, ASC and Office of Science
- Free, open-source
- Built upon VTK (which is built upon OpenGL)

About VisIt

- Documentation
- Features
- Platforms
- Support

VisIt: documentation

- Online at https://wci.llnl.gov/codes/visit/manuals.html
  - Not new
  - Very useful
  - “VisIt Getting Started Manual”
    - Two tutorials
      - Basics
      - Running remotely
    - Appendix of command-line options
  - “Getting Data into VisIt”
    - Creating compatible files
    - Creating a reader plug-in
    - Instrumenting simulation code

VisIt: documentation

- Online, cont.
  - "VisIt User's Manual"
    - Using VisIt
  - "VisIt Python Interface Manual"
    - Writing Python scripts to control VisIt
  - More at https://wci.llnl.gov/codes/visit/doc.html
  - Many tips at https://wci.llnl.gov/codes/visit/FAQ.html
- "On board"
  - Newer
  - Menu: *Help* → *Help…*

VisIt: features

- 1d, 2d, 3d data
- Meshes
  - Structured & unstructured
  - Regular grids
- Parallel, distributed (client-server)
  - Data and visualization remote
  - Rendering remote or local
- Singular, local
  - Data, visualization, rendering local
- Rendering in software or hardware
- Handles files of several GB

VisIt: platforms

- Executables of current release for
  - Windows XP / Vista / 7
  - OSX 10.5 & 10.6
  - Ubuntu 11.04
  - Redhat 6
- Older releases, other platforms at https://wci.llnl.gov/codes/visit/executables.html
- Can build from source... But DON'T!

VisIt: support

- Mailing lists
  - General community support: visit-users@ornl.gov
  - Special others: http://visitusers.org/index.php?title=MailingLists
  - VisIt-development group is helpful / responsive
- Wiki
  - http://visitusers.org

Using VisIt

- Starting VisIt
- Settings
- Opening files
- Visualizing data (*plots* & *operators*)
- Colormapping
- Quantitative analysis
- Writing output
- Client-server
- Python
- Animation

Starting VisIt (GUI)

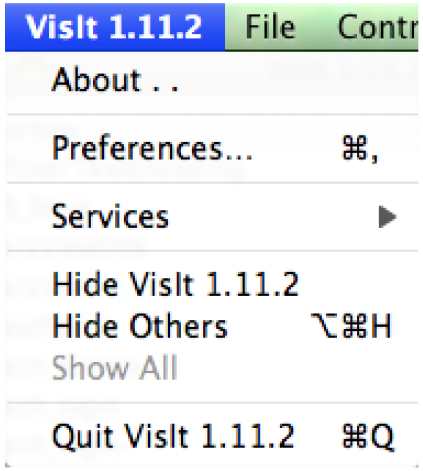
- Command line examples
  - *visit -debug <n>*
  - *visit -assume_format FLASH*
- OSX, Linux, Unix: recommend
  - Set *PATH* to *visit* directory
  - *cd* to data directory
  - Run *visit*
- Windows
  - Icon, command line, program menu
- Non-GUI alternatives
  - Python command-line interface (CLI)
  - Python script

Settings : adjusting & saving

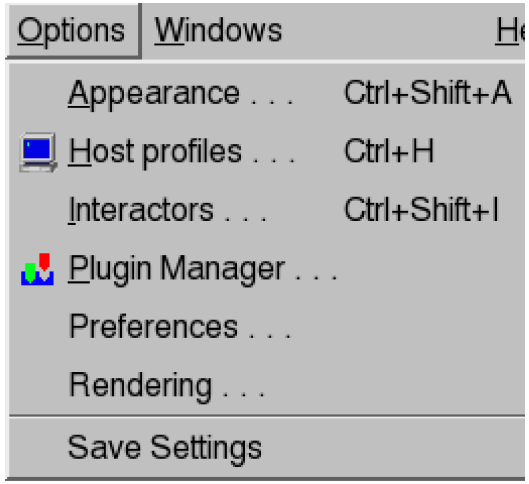
- Make changes in these
  - Preferences
  - *File* menu
  - *Controls* menu
  - *Options* menu
  - *Plot Attributes* menu
  - *Operator Attributes* menu
- Menu: *Options* → *Save Settings*
  - (Might require repetition)
- New settings should be permanent with restart

Preferences

- Mac
  - Menu: *VisIt → Preferences…*

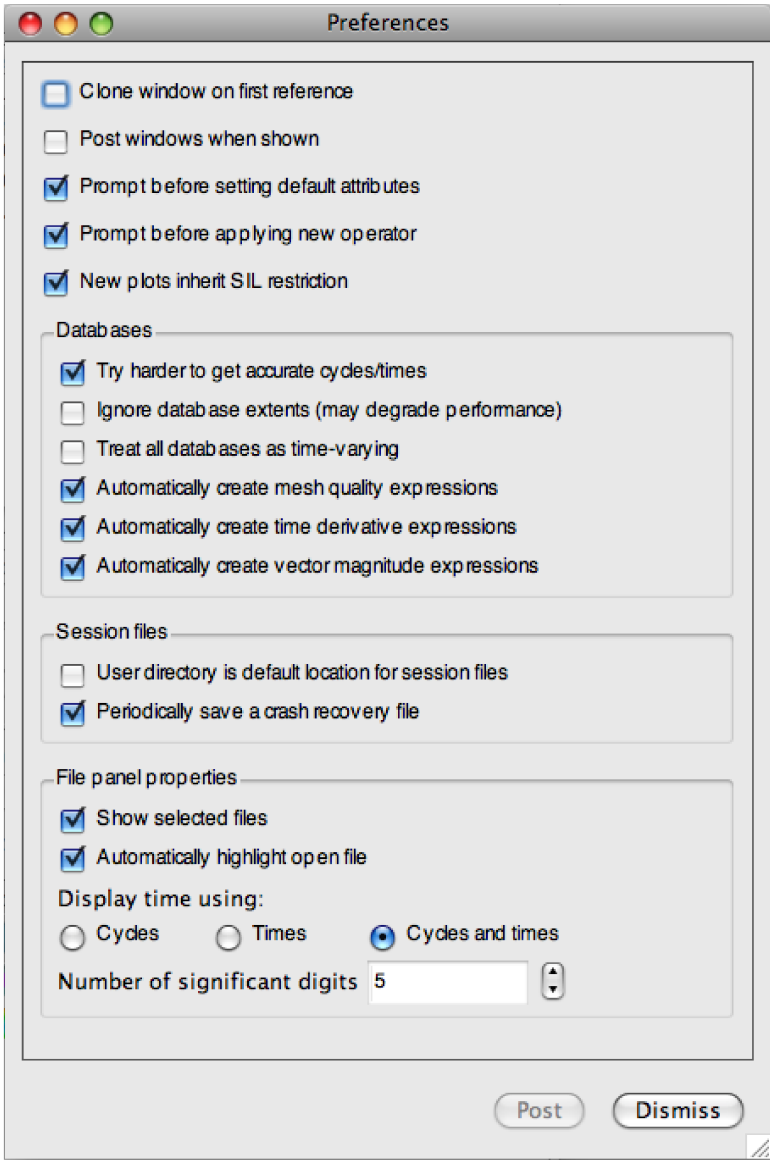


- XP & Linux
  - Menu: *Options → Preferences…*



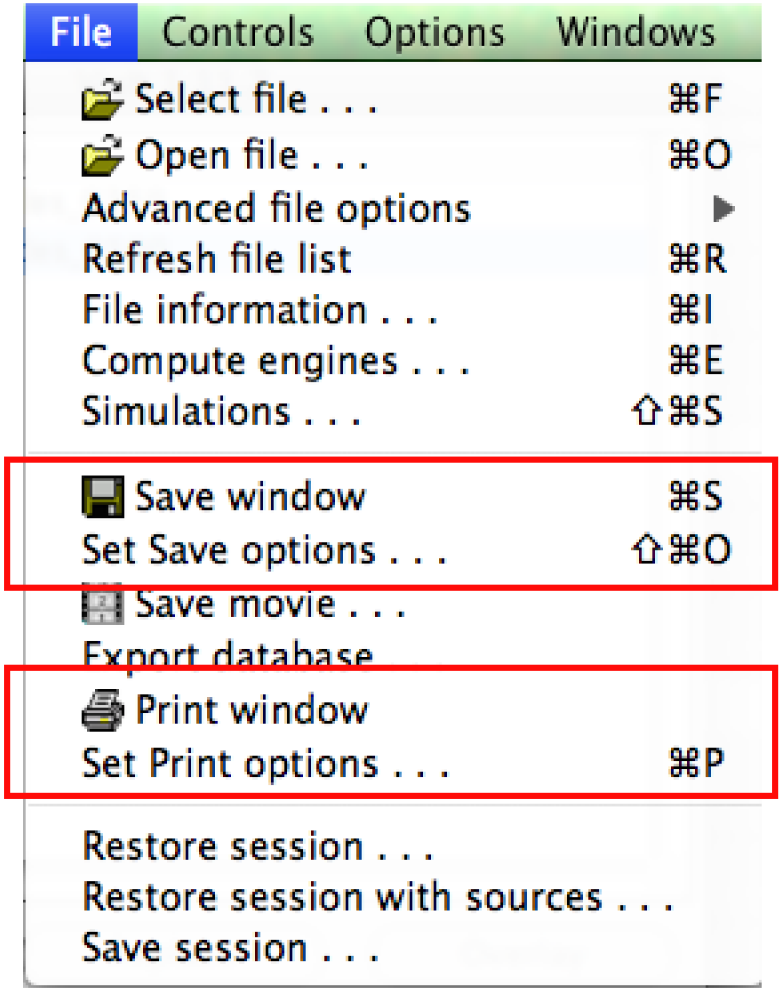
Preferences

- A few global settings:



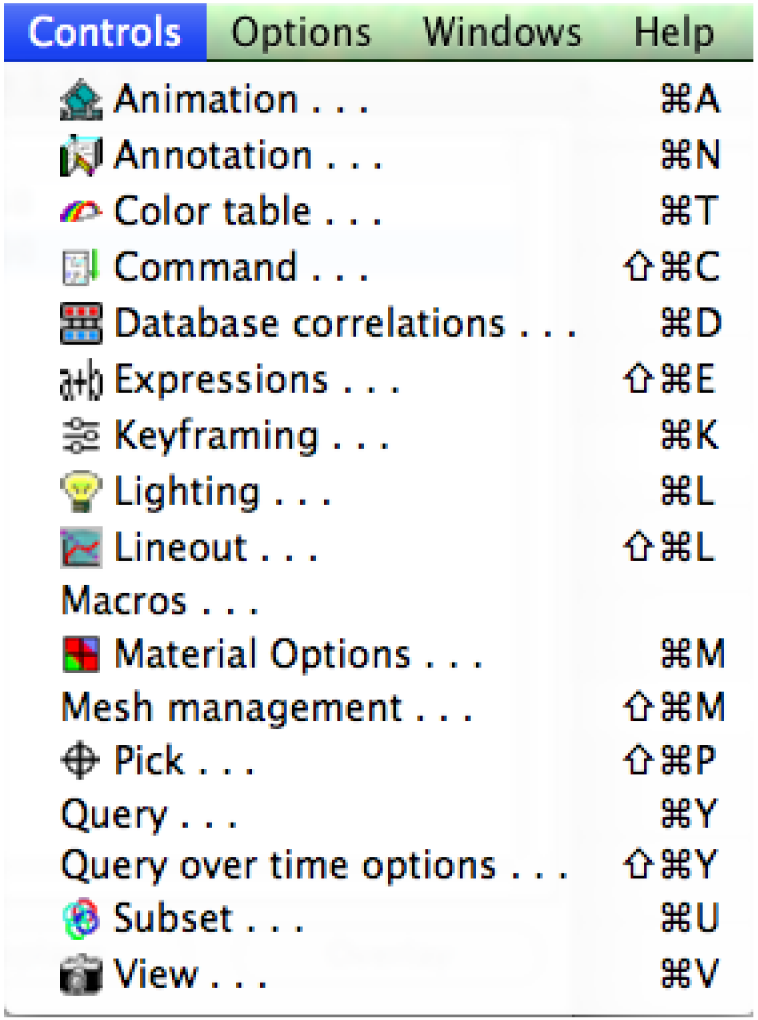
File menu

- Saving and printing
- Each function requires a pair of menu items



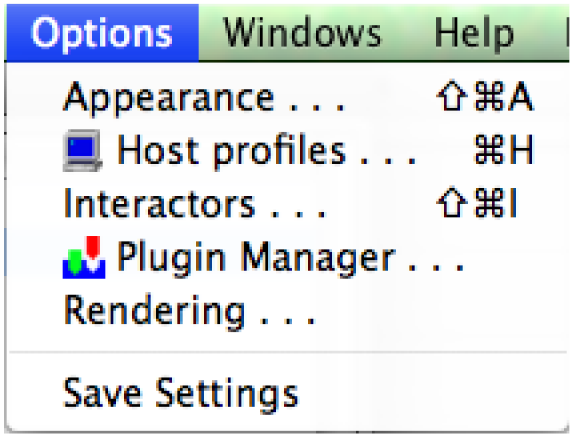
Controls menu

- Controls
- A few of these later

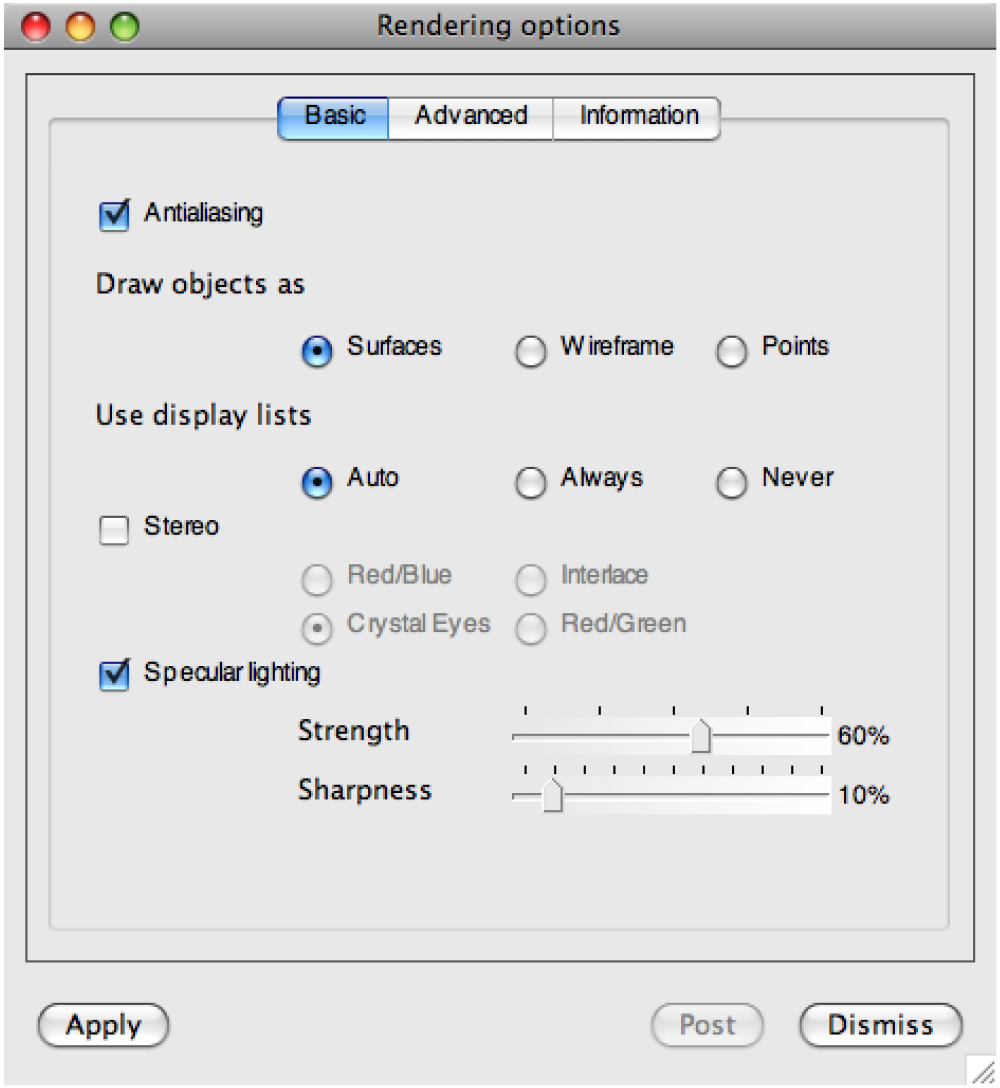


Options menu

- Appearance, rendering, plugins
- *Save Settings*

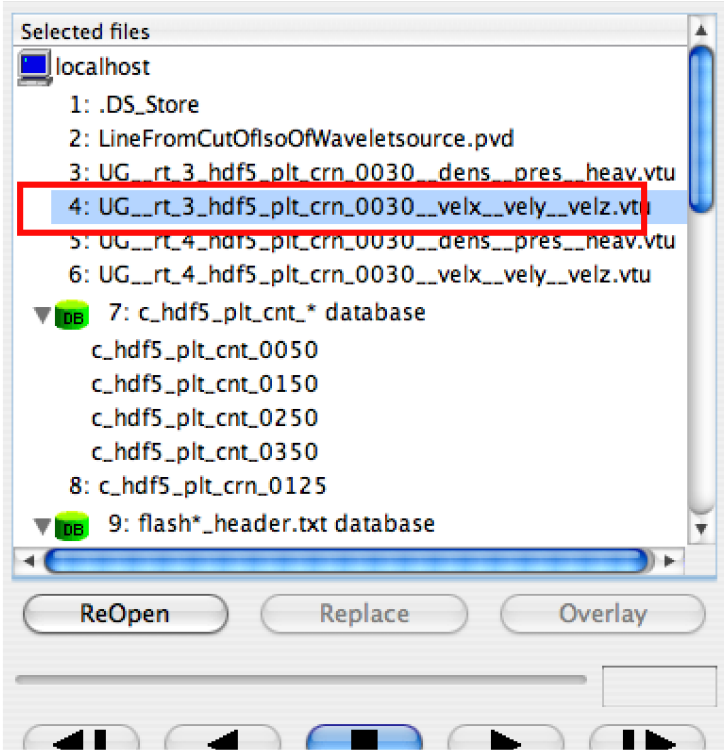


Options: rendering



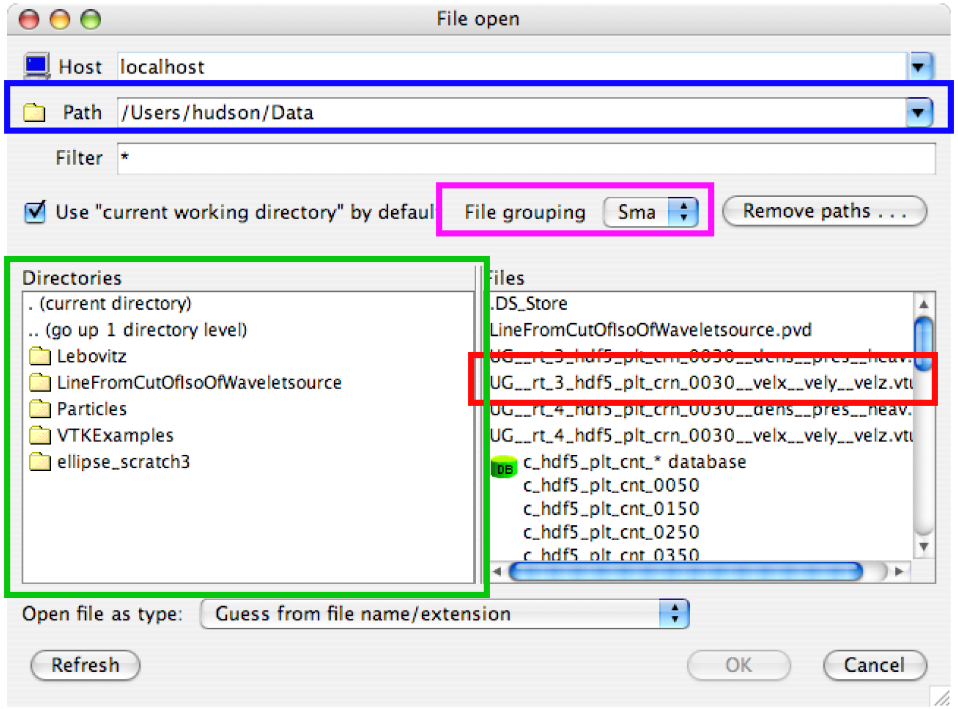
Opening files - this directory

- In *Selected files* panel of main window
  - Double-click file name
  - Or…



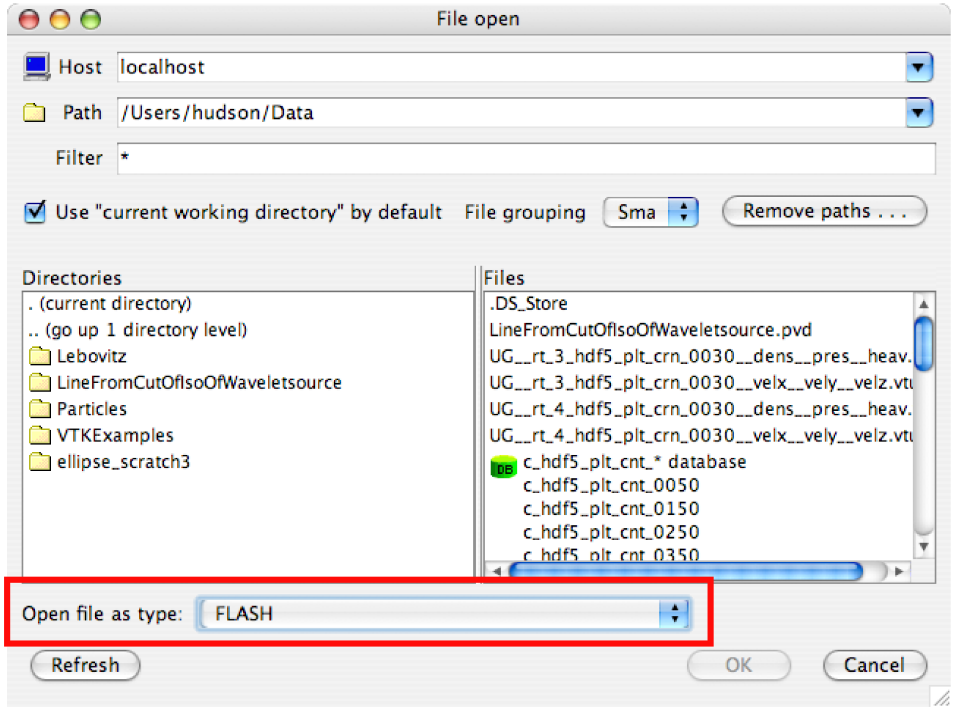
Opening files - other directory

- Open *File open* dialog with one of
  - Key: ⌘ *O*
  - Menu: *File* → *Open file …*
- In *File open* dialog, go to directory here or here
- Double-click file name
- Can group files



Opening files - by format

- In *File open* dialog
  - Select *FLASH* before opening

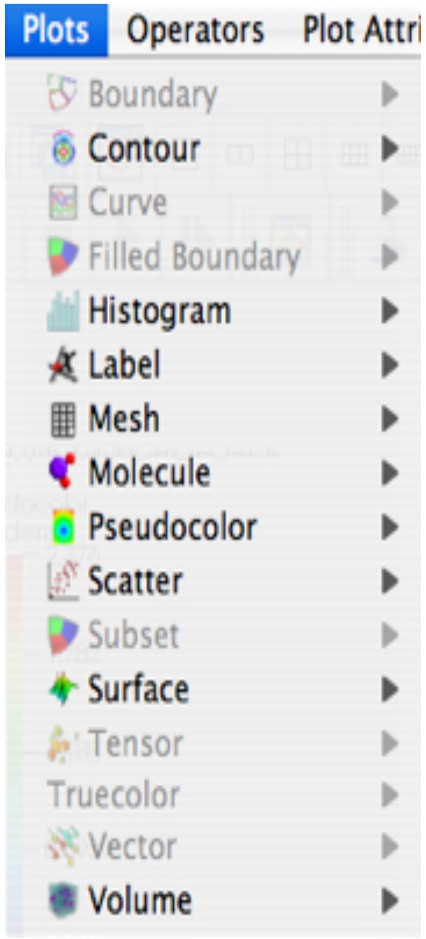


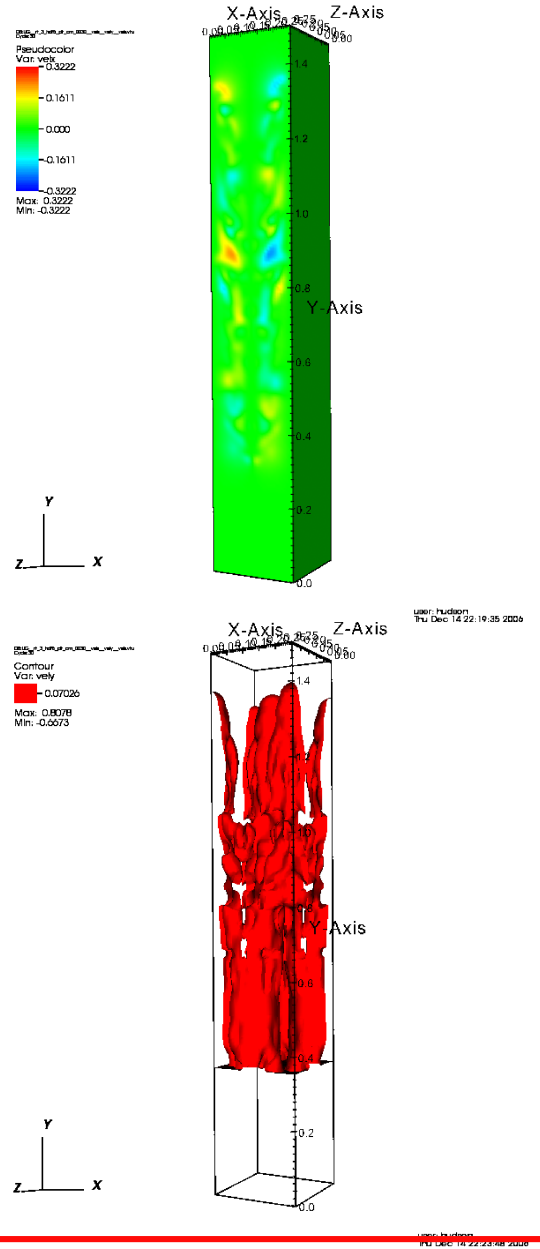
Visualizing data

- Plots and operators
  - Plots are chosen first
  - Operators operate first

Visualizing data

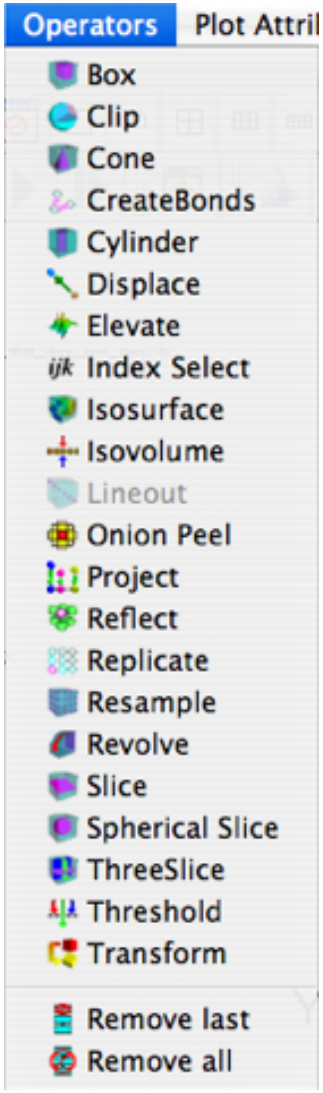
- Plots
  - Data or meshes, mapped to shapes or colors
  - Read data from file
  - Displayed in "visualization" windows (also "the viewer") on the local machine
  - Which plots appear on the menu is under user control

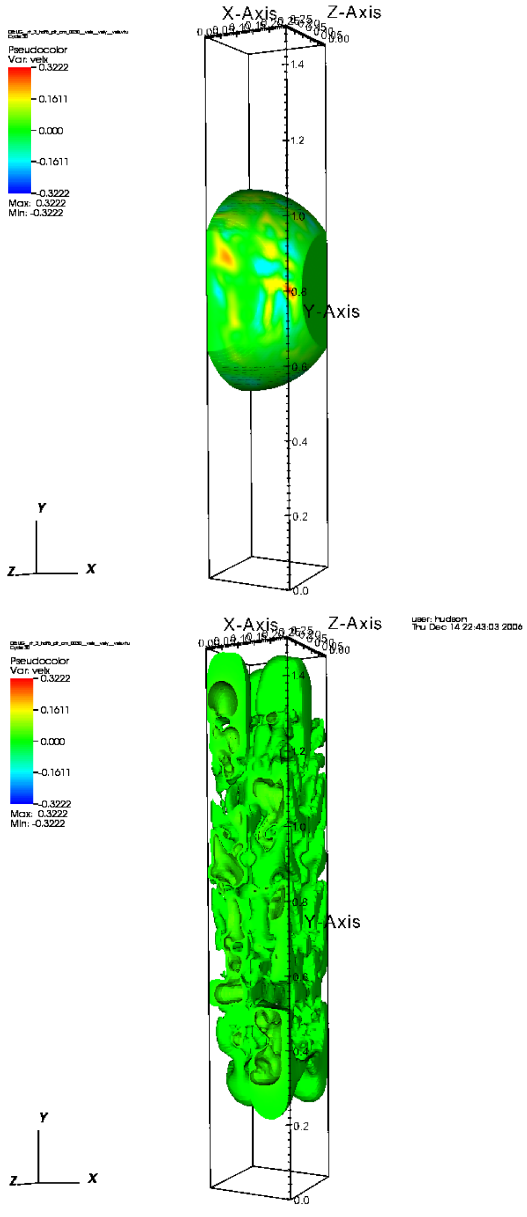




Visualizing data

- Operators
  - Subsets of data or meshes, selected for plotting
  - Which operators appear on the menu is under user control



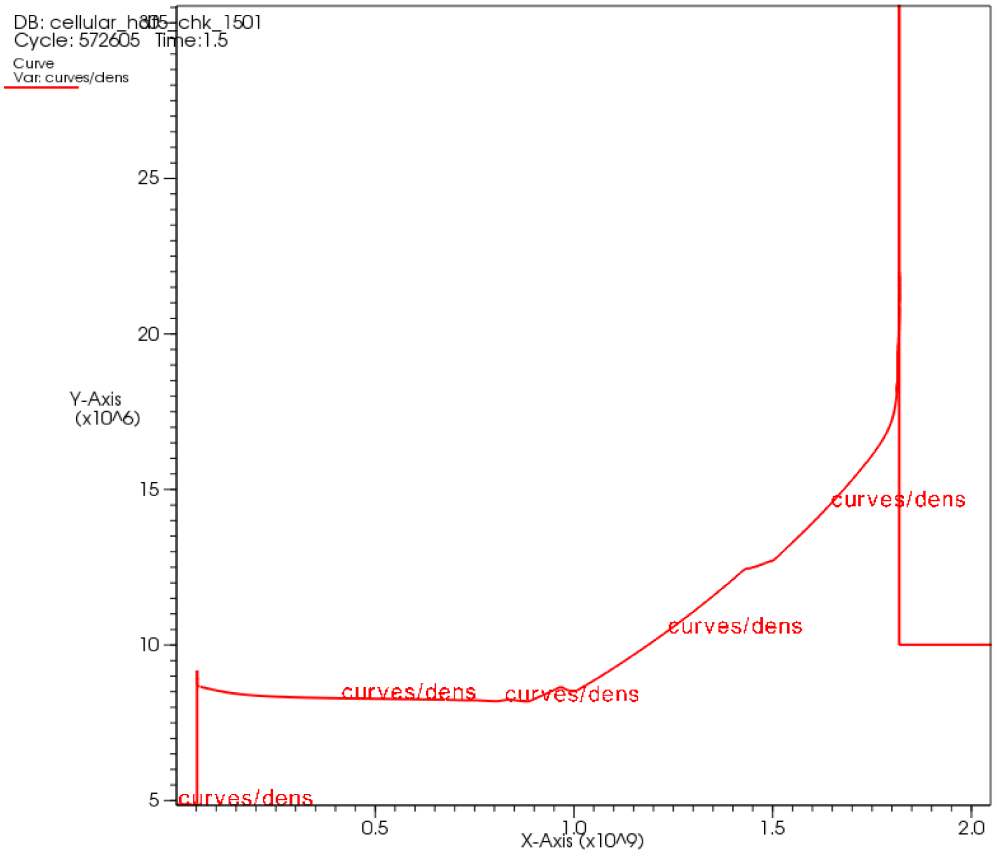


Visualizing data

- Scalar data: examples
  - 1d data
    - *Curve* plot
  - Cut Plane
  - Isosurface
    - Colored by same variable
    - Colored by other variable

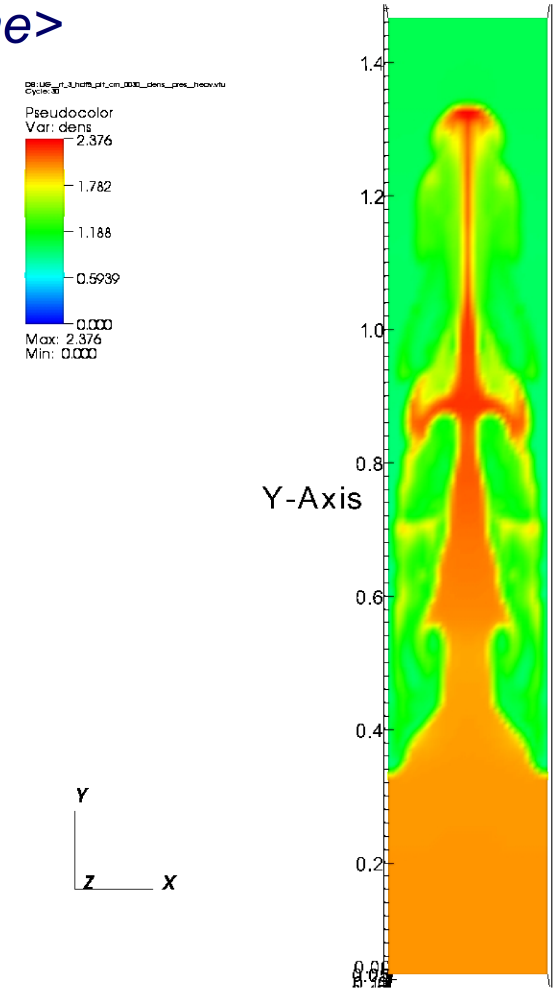
Scalar data - 1d data

- Menu: *Plots* → *Curve* → *<varname>*
- Optional settings change
  - Menu: *Plot Attributes* → *Curve…*



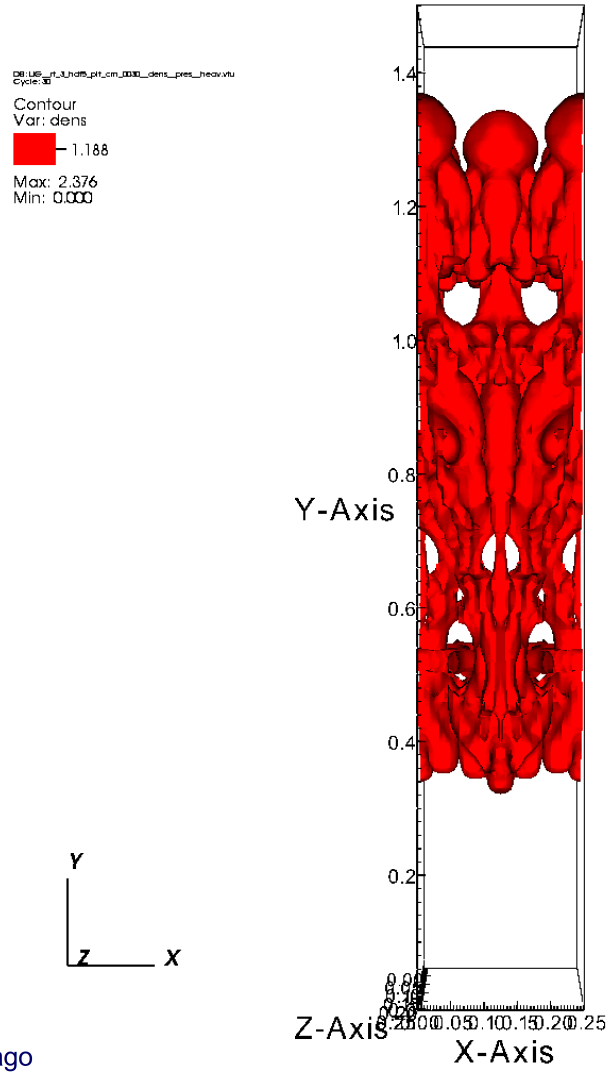
Scalar data - cut plane

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Menu: *Operators* → *Slice*
- Optional settings change
  - Menu: *Plot Attributes* → *Pseudocolor…*
  - Menu: *Operator Attributes* → *Slice…*



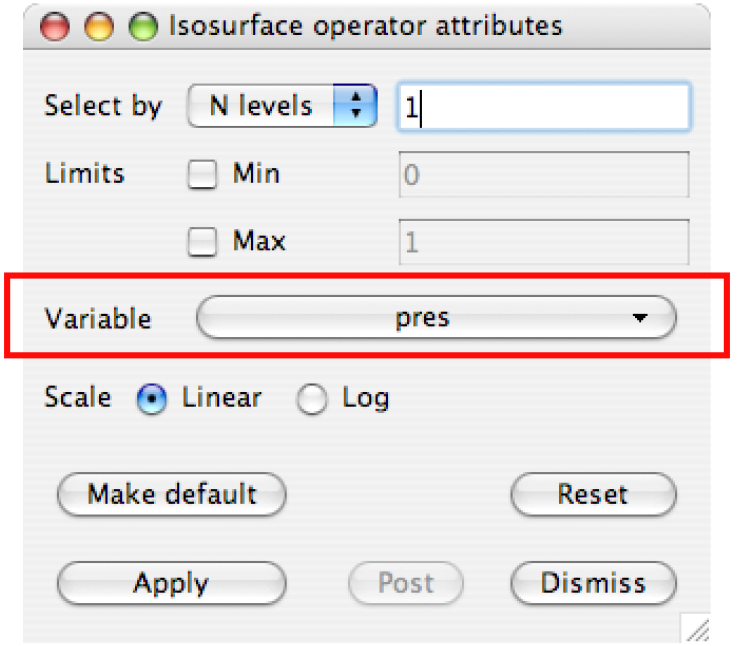
Isosurface - same variable

- Menu: *Plots* → *Contour* → *<varname>*
- Optional settings change
  - Menu: *Plot Attributes* → *Contour…*



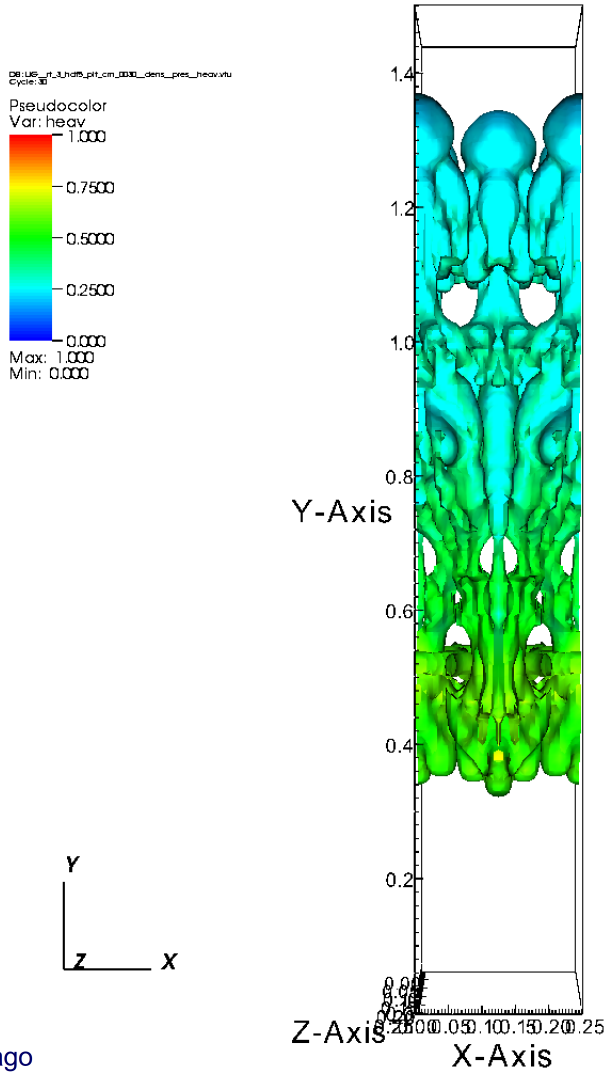
Isosurface - other variable

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Menu: *Operators* → *Isosurface*
- Required settings change
  - Menu: *Operator Attributes* → *Isosurface…*
  - Change *Variable*



Isosurface - other variable

- Optional settings change
  - Menu: *Operator Attributes* → *Isosurface…*

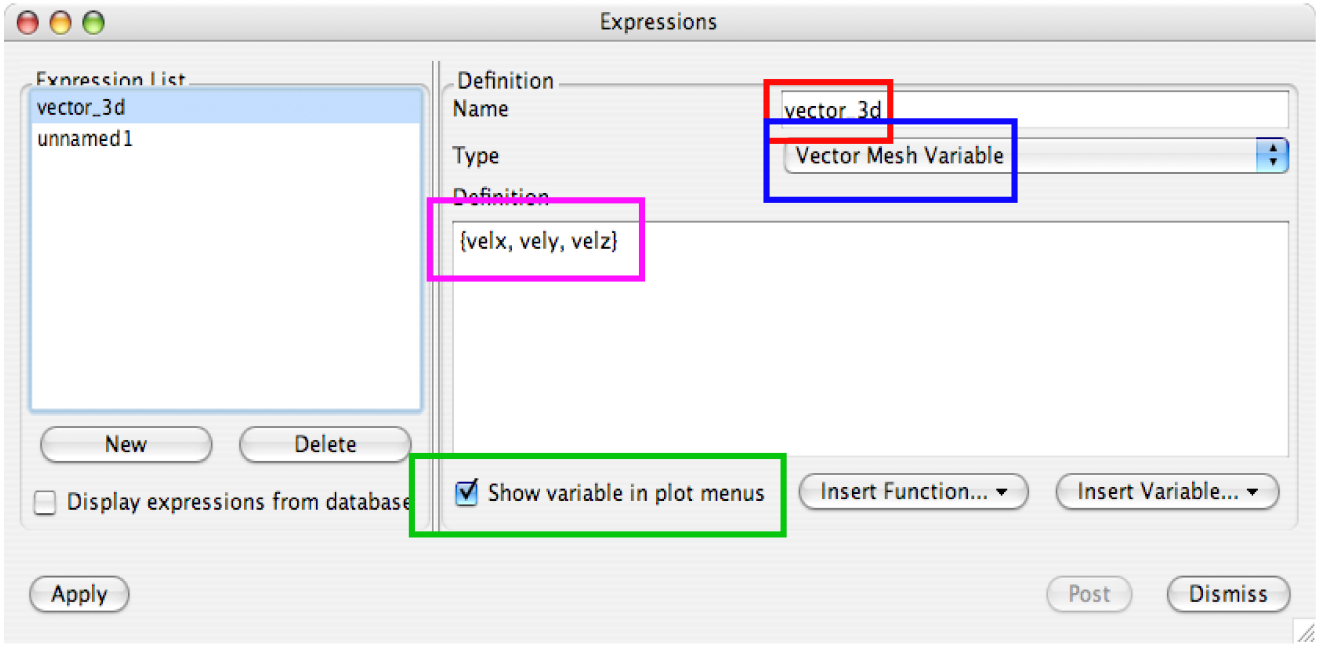


Visualizing data

- Vector data: examples
  - (Read vector data set)
  - (Define vector from input scalars)
  - Glyphs
  - Streamlines
    - As tubes
    - As lines

Vector data - define vector

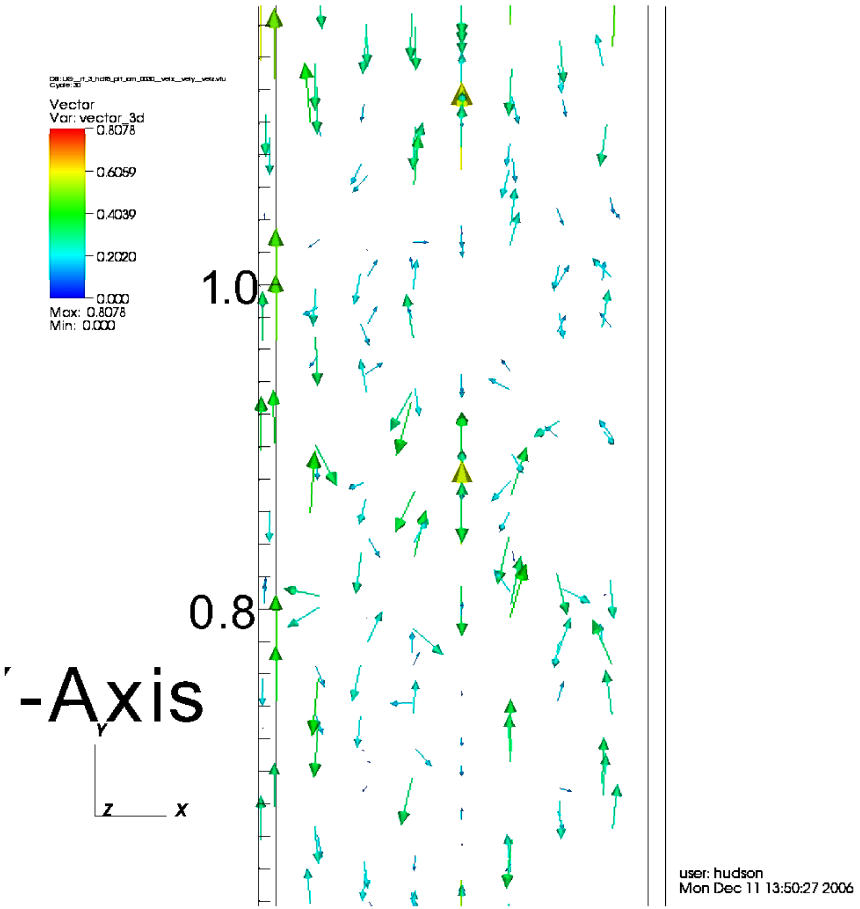
- Menu: *Controls → Expressions…*



- Define *<vectorvarname>* as *Vector Mesh Variable* with *{<xname>,<yname>,<zname>}*
- Check *Show variable in plot menus*

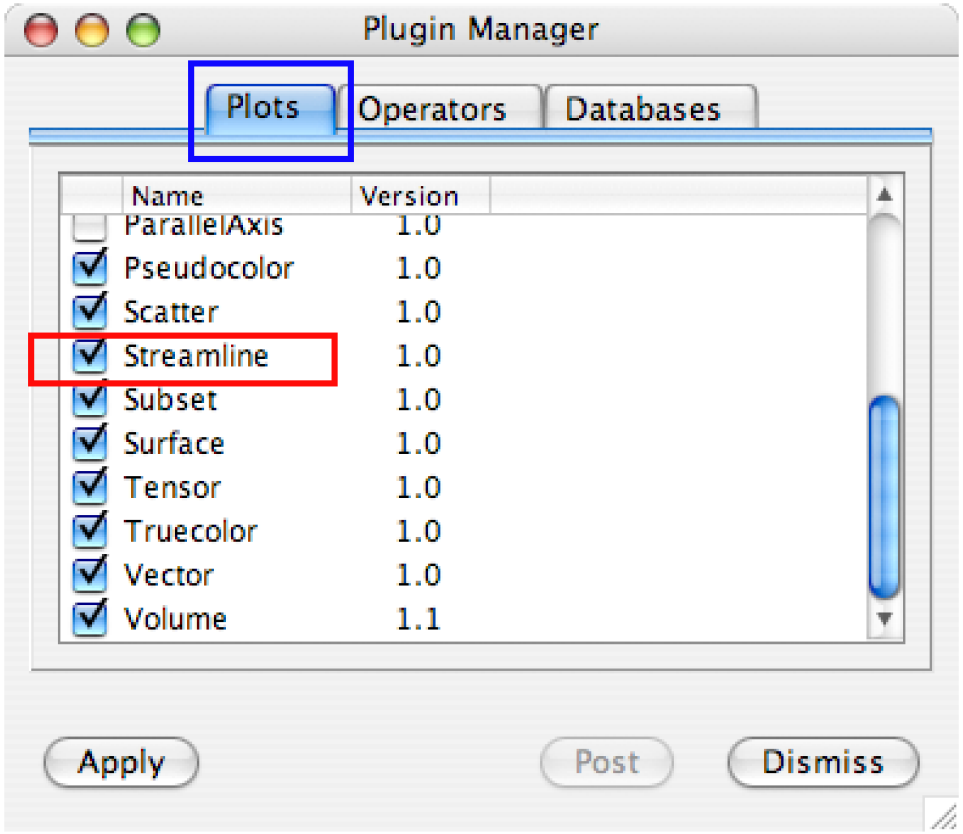
Vector data - glyphs

- Menu: *Plots* → *Vector* → *<vectorvarname>*
- Optional settings change
  - Menu: *Plot Attributes* → *Vector…*



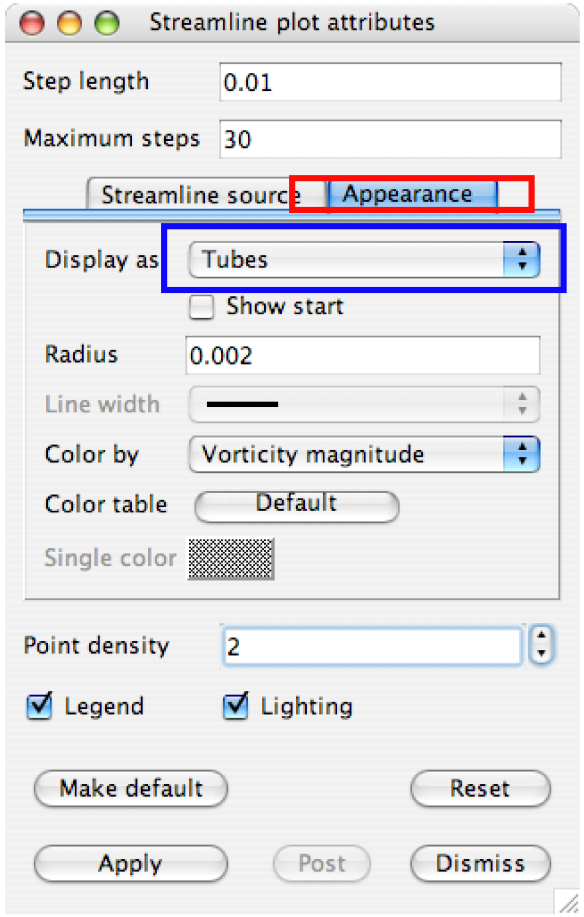
Vector data - streamlines

- Add *Streamline* plugin to *Plot* menu
  - Menu: *Options* → *Plugin Manager …*
  - *Plots* tab: click *Streamline*
  - Menu: *Options* → *Save Settings*
  - Restart VisIt



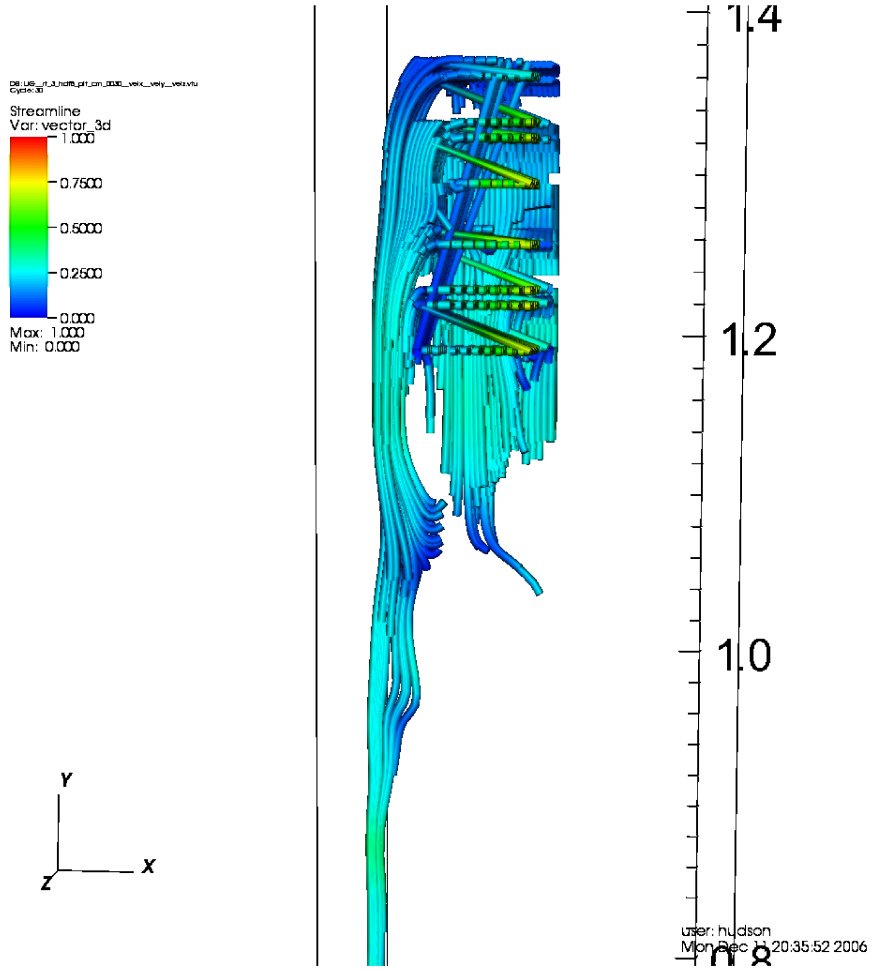
Streamlines - as tubes

- Menu: *Plots* → *Streamline* → *<vectorvarname>*
- Required settings change
  - Menu: *Plot Attributes* → *Streamline…*
  - On *Appearance* tab, select *Tubes*



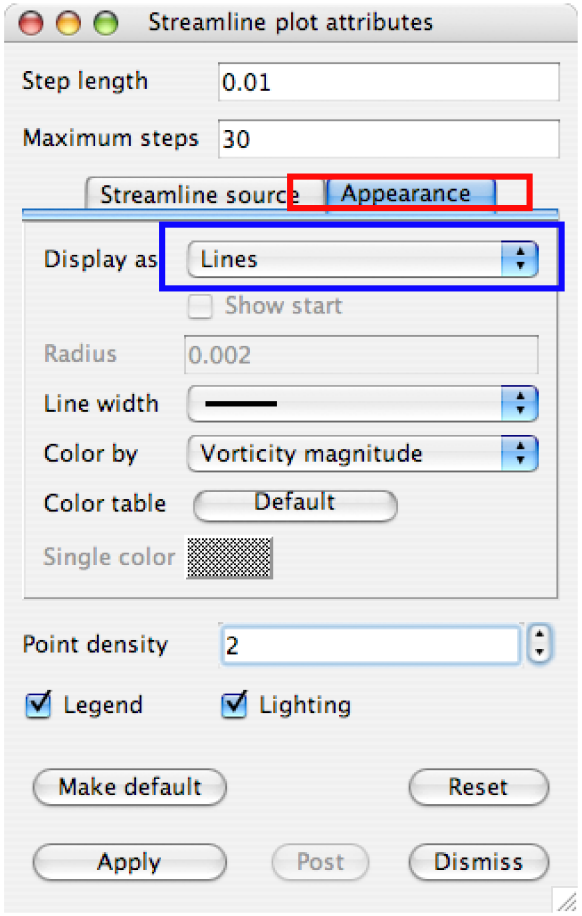
Streamlines - as tubes

- Optional settings change
  - Menu: *Plot Attributes* → *Streamline…*



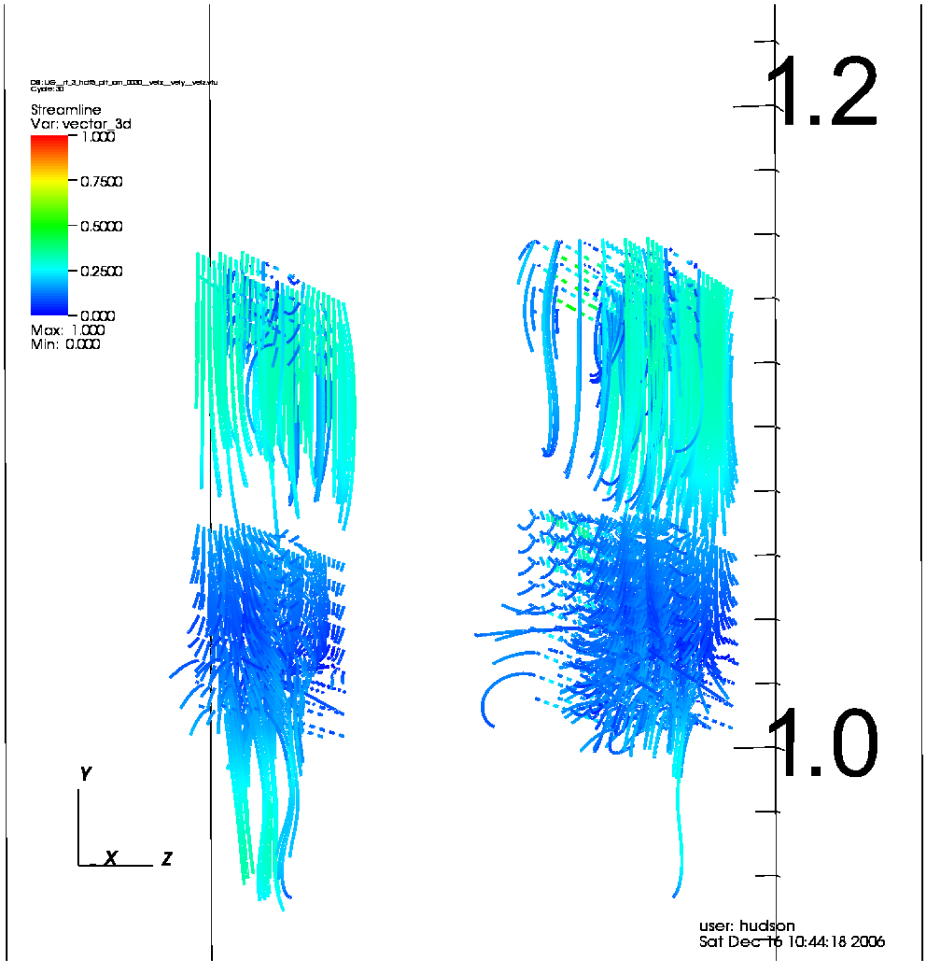
Streamlines - as lines

- Menu: *Plots* → *Streamline* → *<vectorvarname>*
- Required settings change
  - Menu: *Plot Attributes* → *Streamline…*
  - On *Appearance* tab, select *Lines*



Streamlines - as lines

- Optional settings change
  - Menu: *Plot Attributes* → *Streamline…*

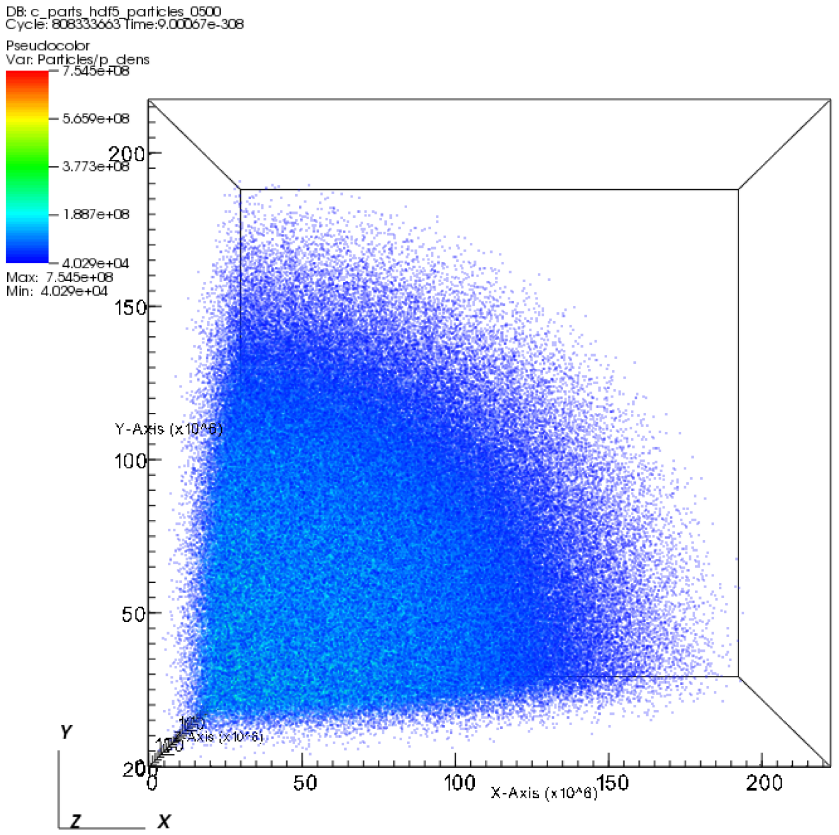


Visualizing data

- Particle data: examples
  - Pseudocolor
  - Cut Plane
  - Isosurface
  - Volume rendering
  - Particle tracks

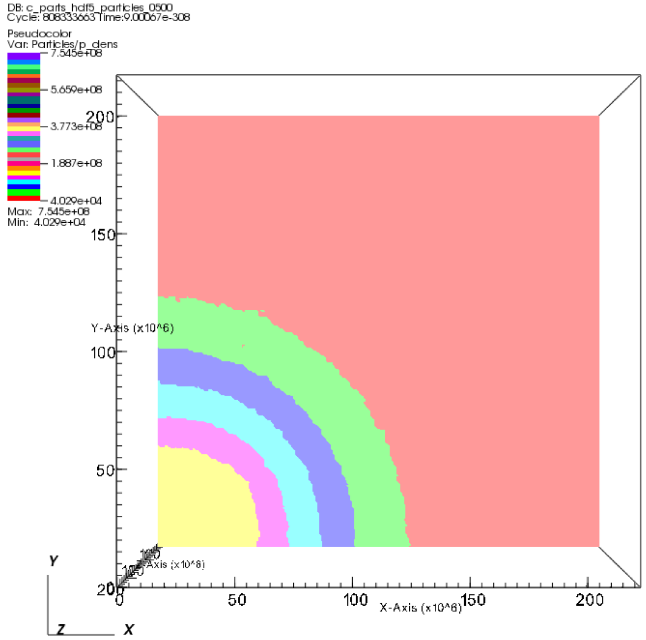
Particle data - pseudocolor

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Optional settings change
  - Menu: *Plot Attributes* → *Pseudocolor…*



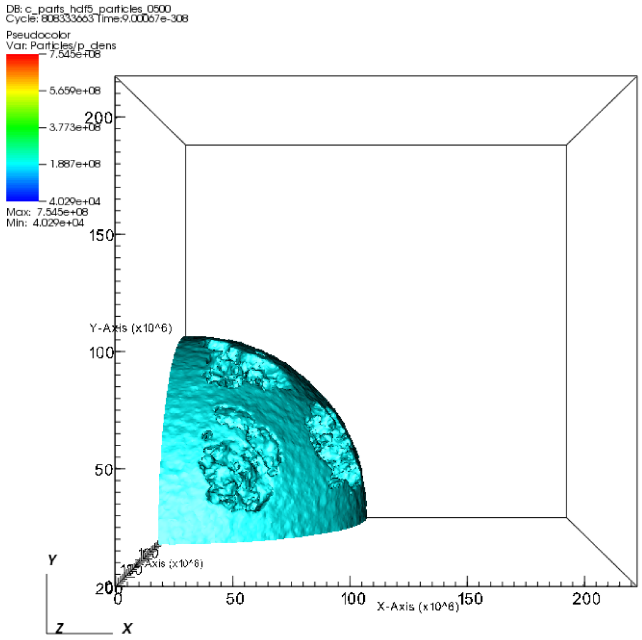
Particle data - cut plane

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Menu: *Operators* → ***Resample***
- Menu: *Operators* → *Slice*
- Optional settings change
  - Menu: *Plot Attributes* → *Pseudocolor…*
  - Menu: *Operator Attributes* → *Resample…*
  - Menu: *Operator Attributes* → *Slice…*



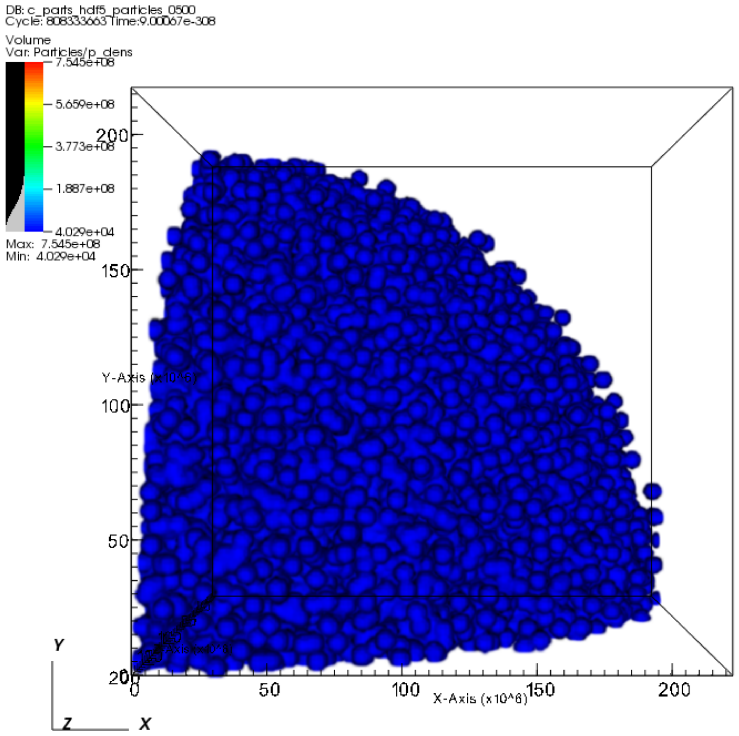
Particle data - isosurface

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Menu: *Operators* → ***Resample***
- Menu: *Operators* → *Isosurface*
- Optional settings change
  - Menu: *Plot Attributes* → *Pseudocolor…*
  - Menu: *Operator Attributes* → *Resample …*
  - Menu: *Operator Attributes* → *Isosurface…*



Particle data - volume rendering

- Menu: *Plots* → *Pseudocolor* → *<varname>*
- Menu: *Operators* → *Volume*
- (***Implicit resampling***)
- Optional settings change
  - Menu: *Plot Attributes* → *Pseudocolor…*
  - Menu: *Operator Attributes* → *Volume…*

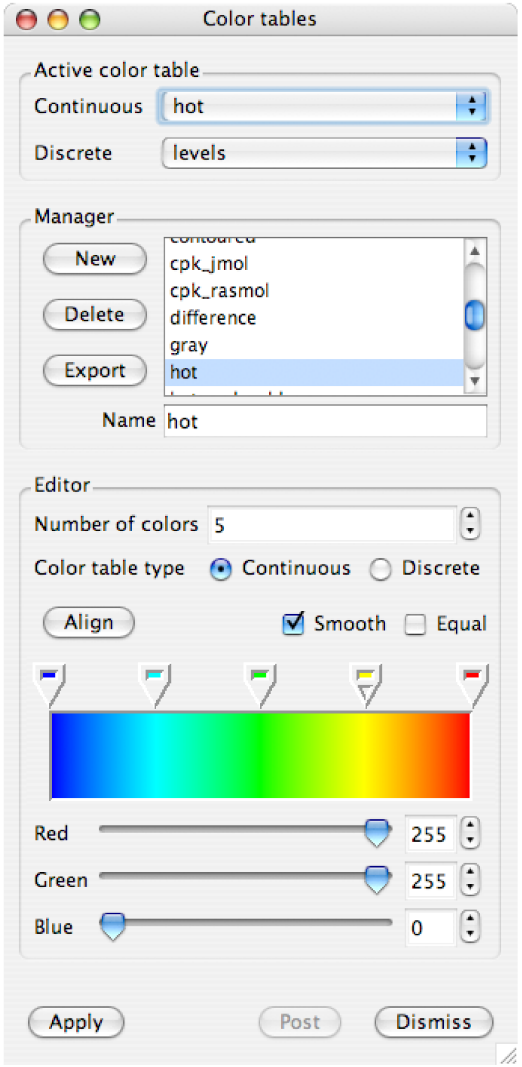


Colormapping

- Pseudocolor (seen earlier)
- Custom color table
- (Truecolor takes color from color vector in dataset)

Custom color table

- Menu: *Controls* → *Color table…*



Quantitative Analysis

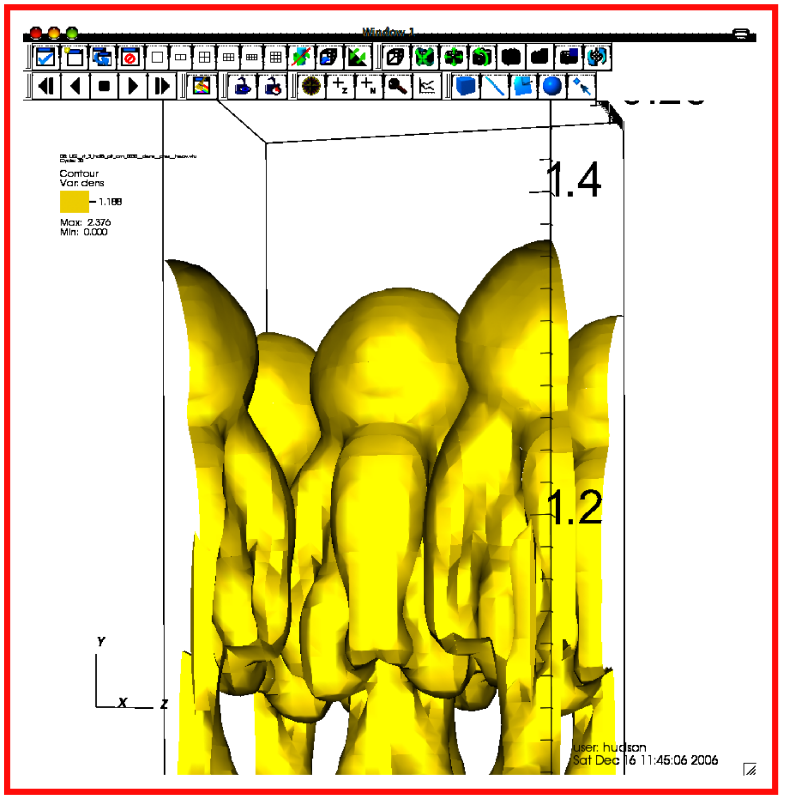
- Of data
  - Samples at points
  - Samples in cells
  - Samples along lines
- Of meshes
  - Size (area, volume)
  - Number elements
- Via expressions: derived data

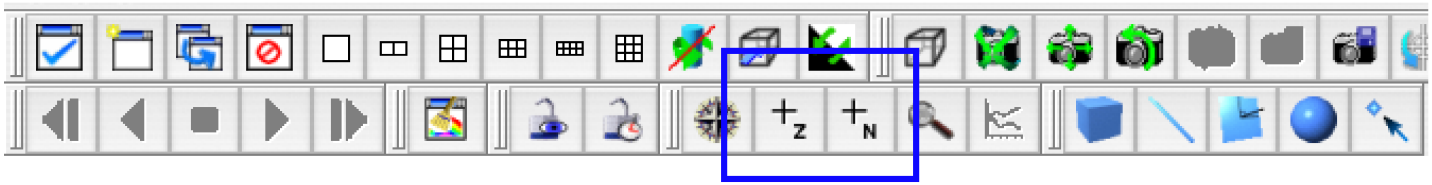
Quantitative Analysis

- Zone Pick: cell the pick point falls in
- Node Pick: nearest node of data set

Data analysis - points & cells

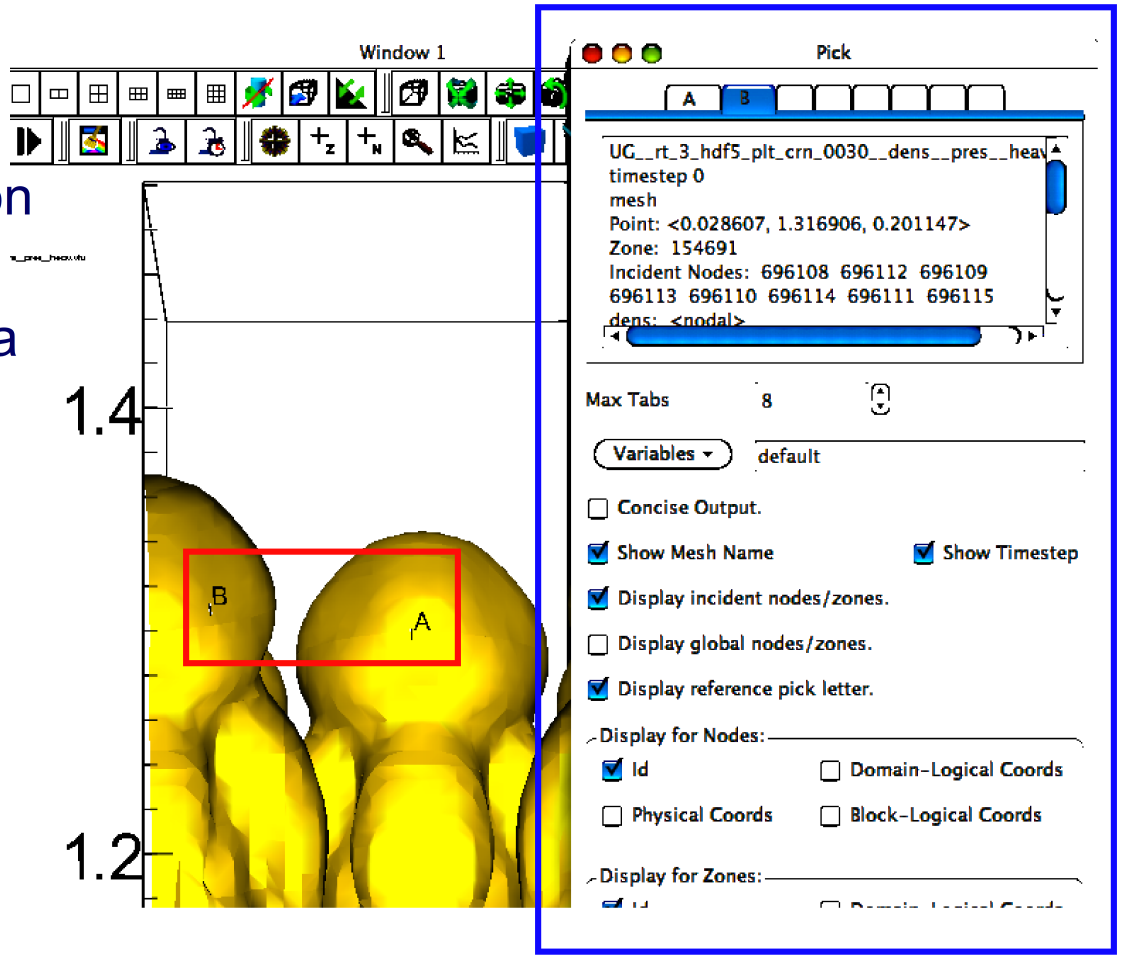
- Create isosurface
- Select visualization window
- Click on “zone pick” or “node pick” button:
- Click somewhere on plot





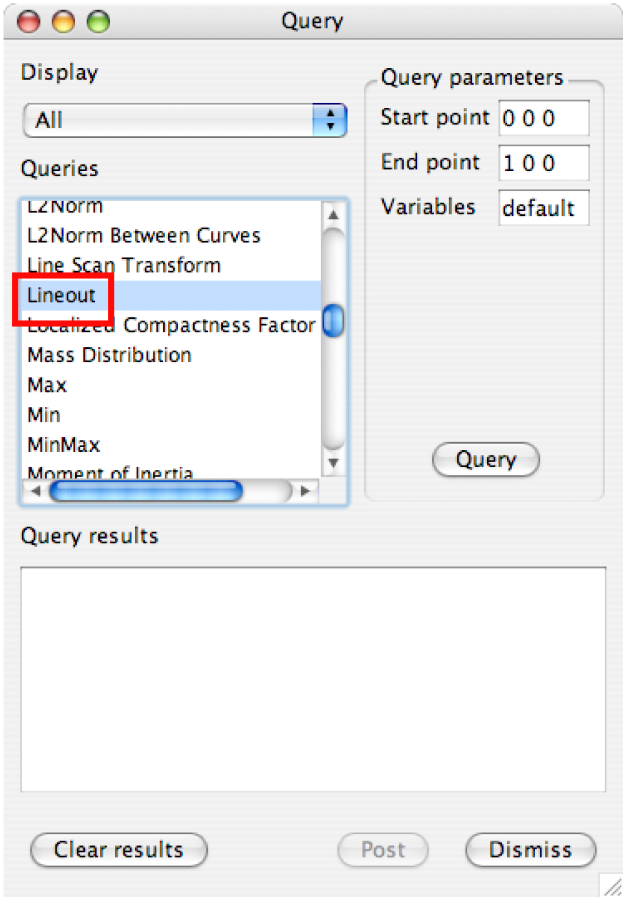
Data analysis - points & cells

- Picked points are displayed in visualization window
- *Pick* panel displays data at picked points



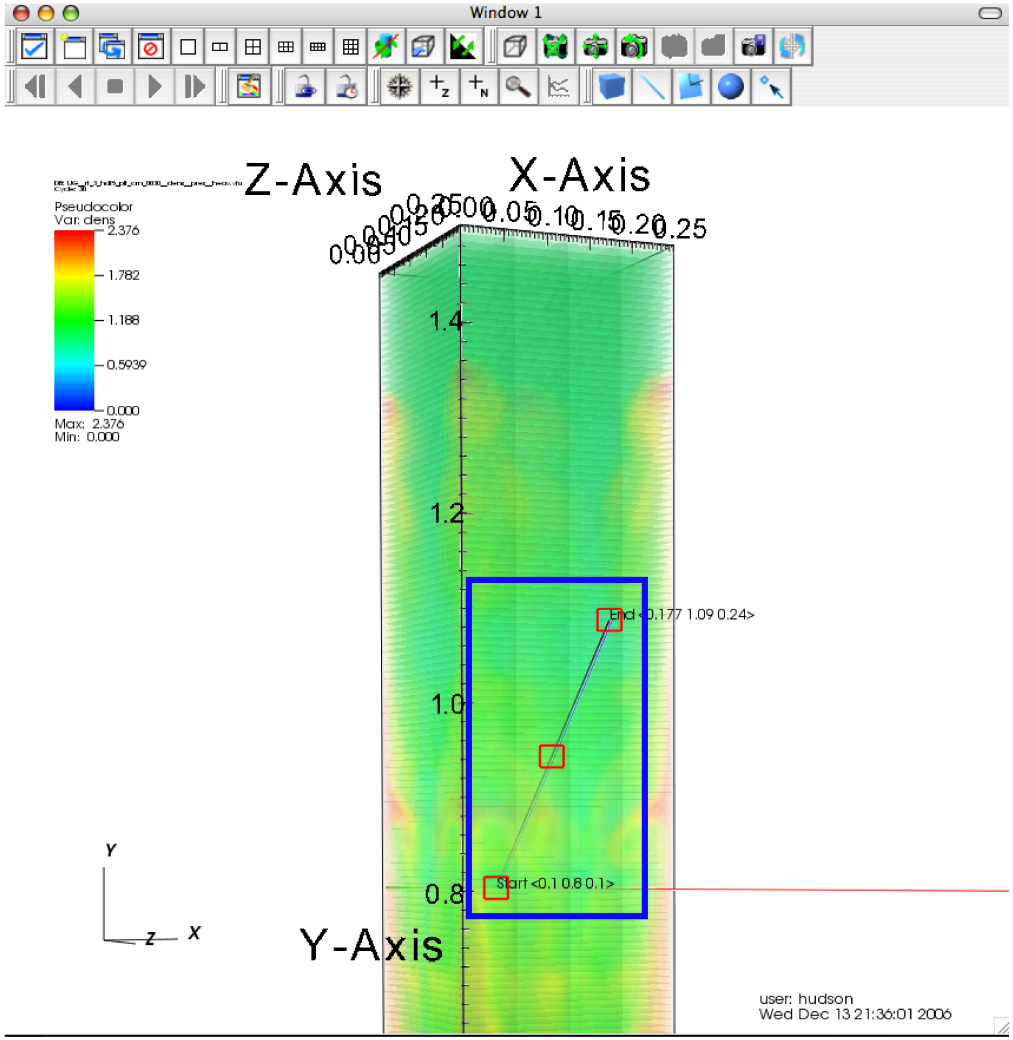
Data analysis - along line

- Create, e.g., *Pseudocolor* plot
- Menu: *Controls* → *Query…*
- In *Queries* list of *Query* panel
  - Select *Lineout*
- Select visualization window
- Click on "line tool" button:



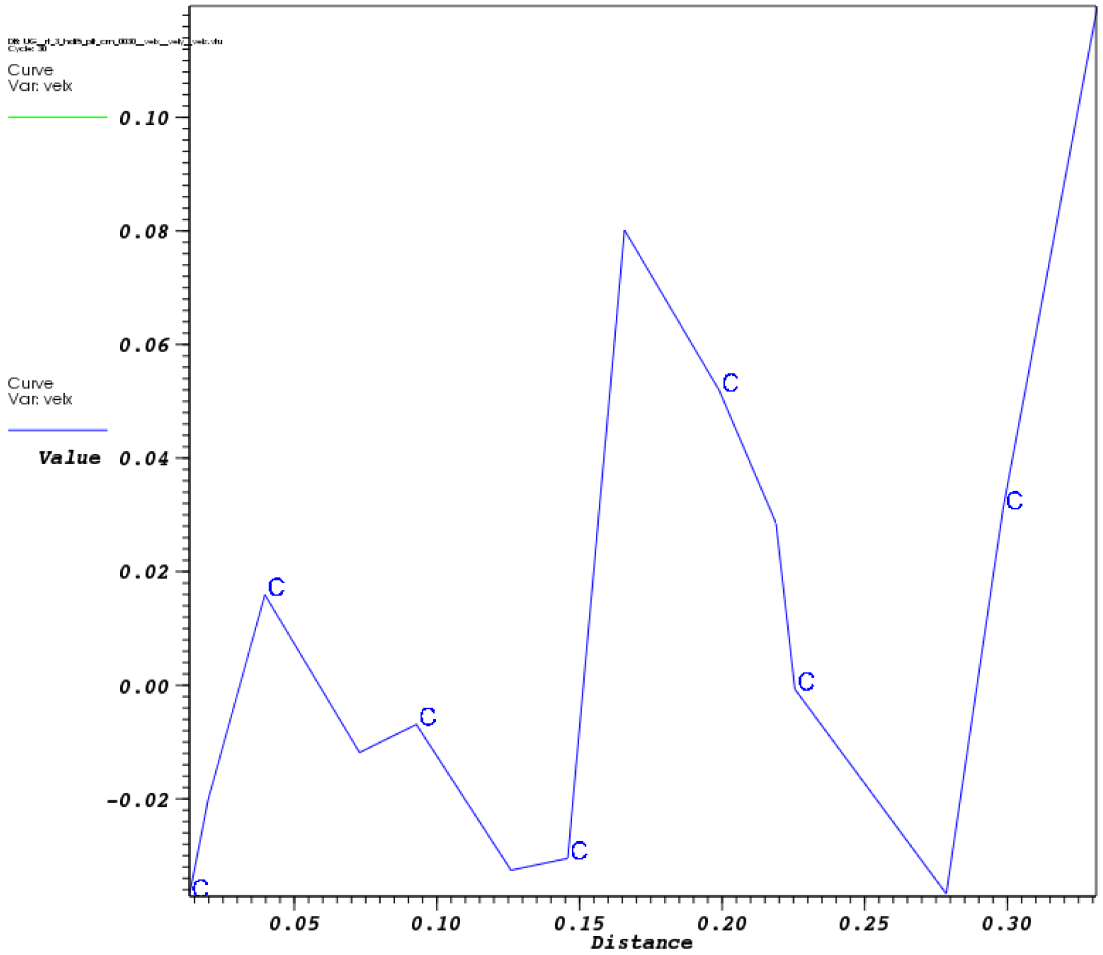
Data analysis - along line

- A “widget”
- Line tool controls sampling line
- Change the line tool via its red control points to change the sampling line



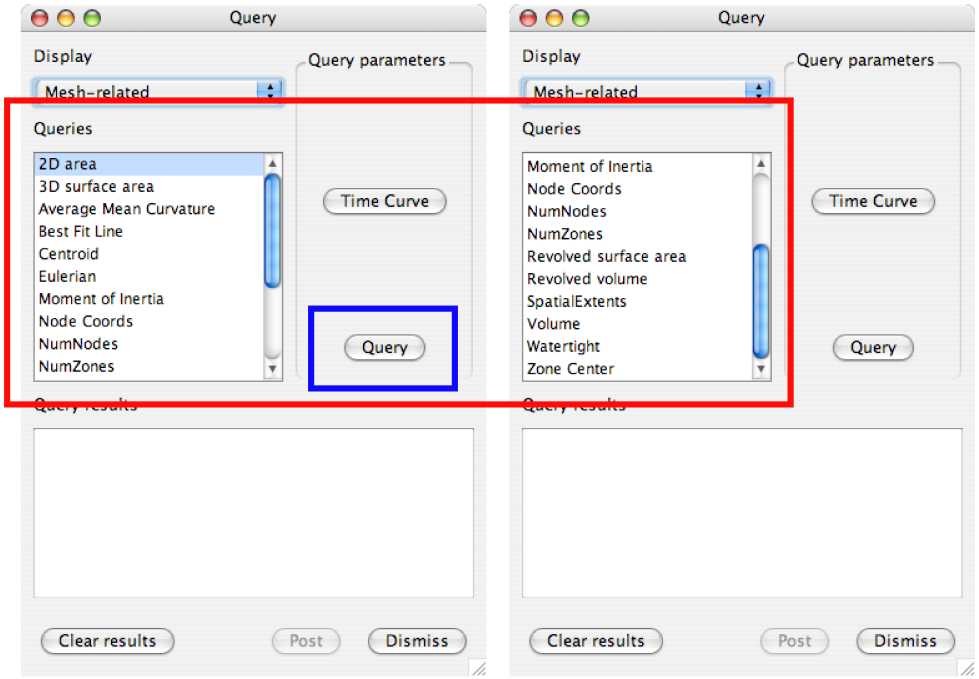
Data analysis - along line

- Curve is displayed in a visualization window



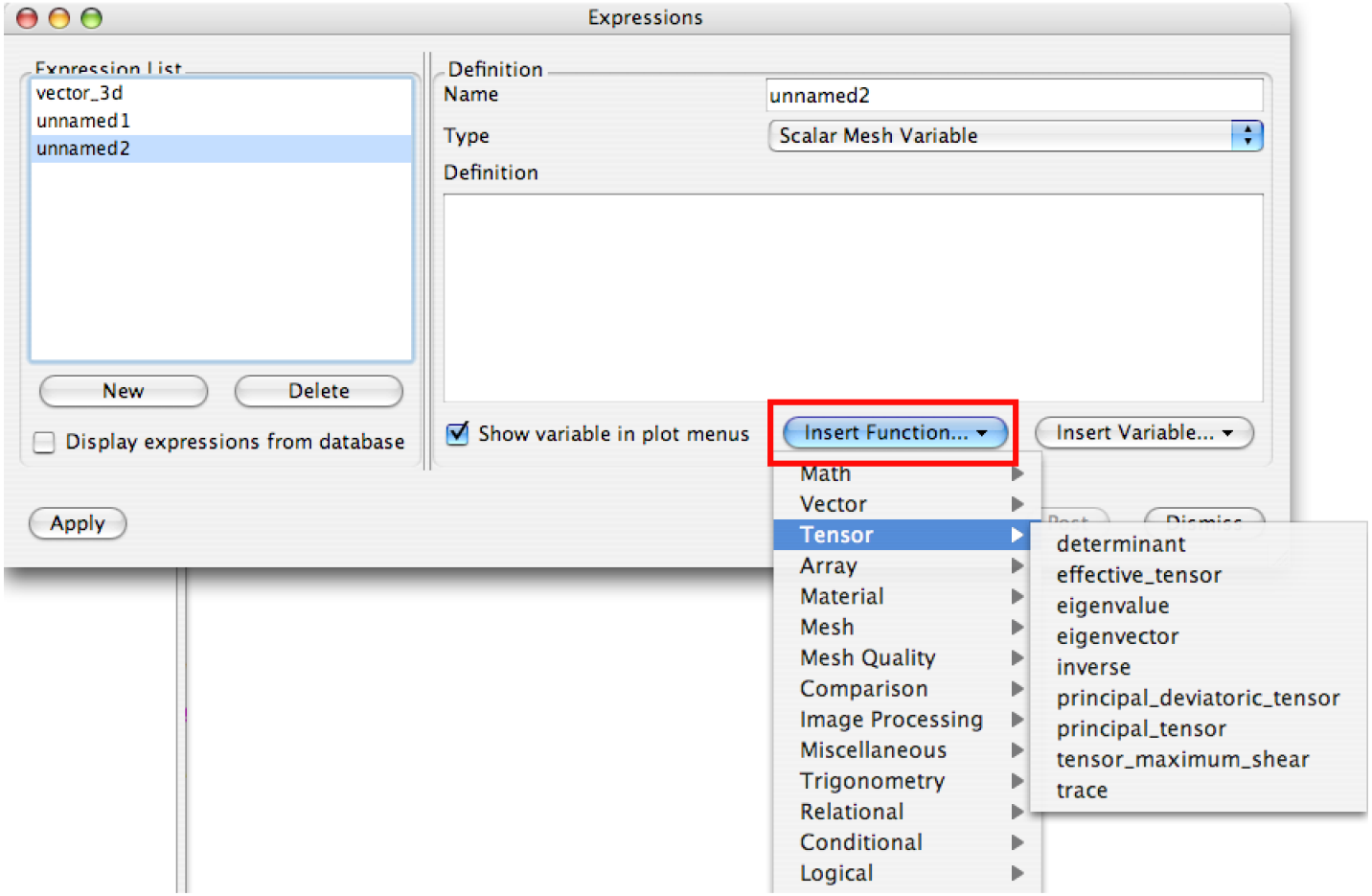
Mesh analysis

- Menu: *Controls* → *Query…*
- Select a query
- Click query button



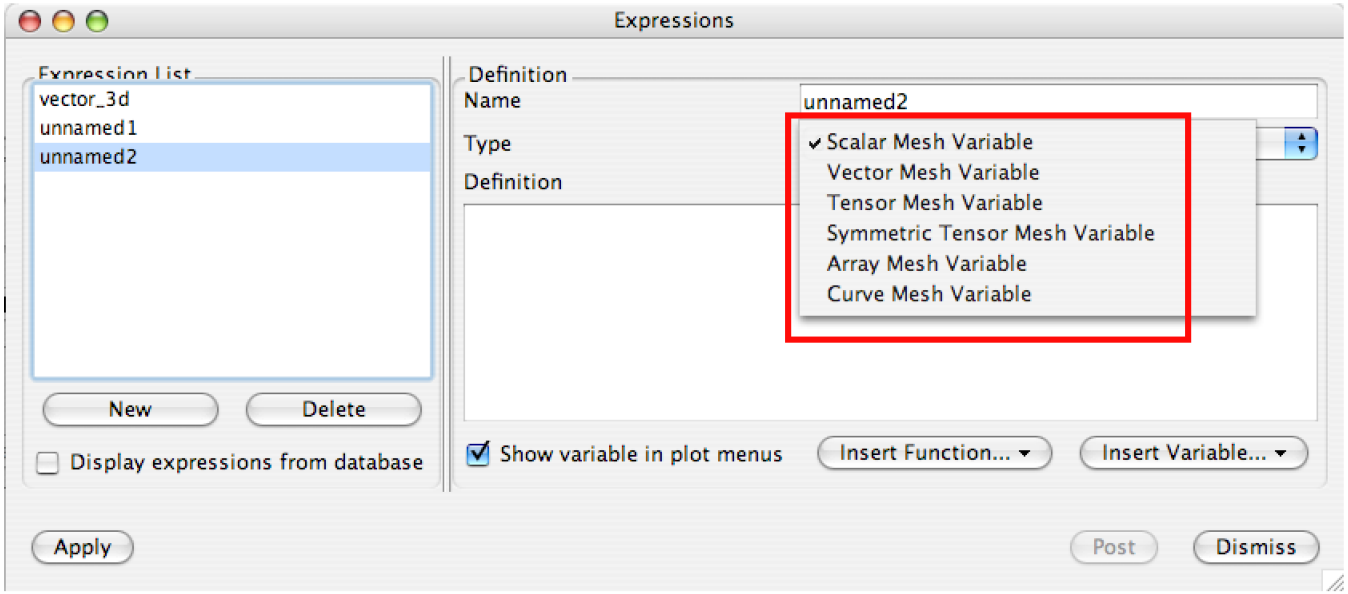
Data analysis and synthesis

- Expressions: function



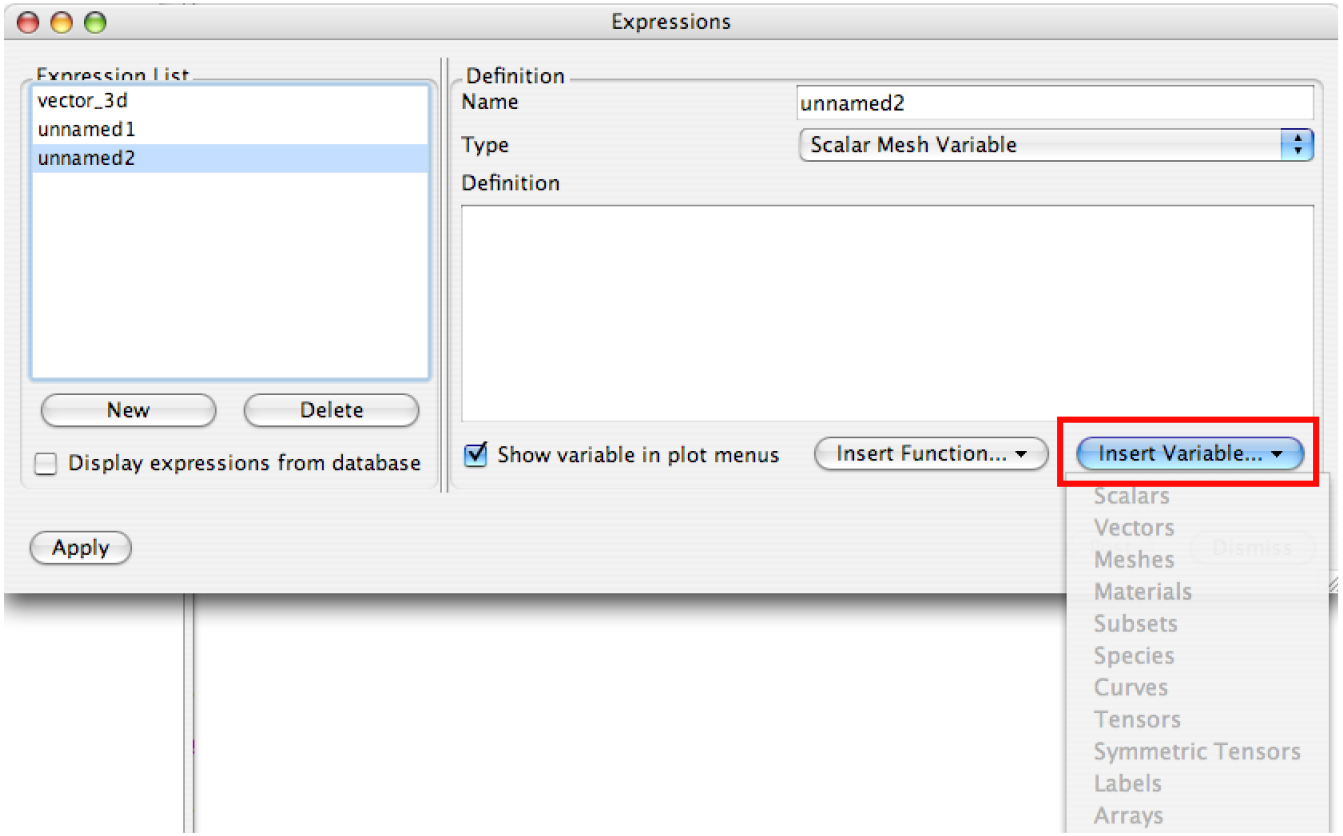
Data analysis and synthesis

- Expressions: output variable



Data analysis and synthesis

- Expressions: input variable

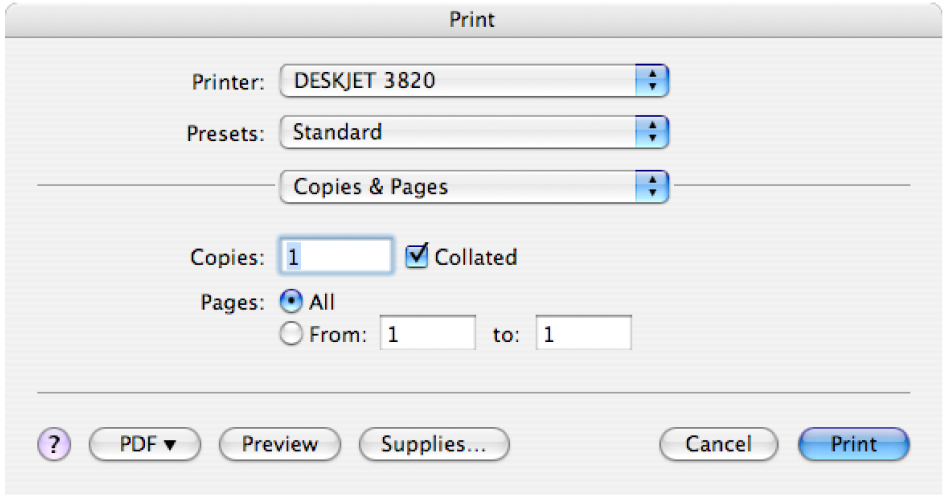


Writing output

- Hardcopy
- Image
- Geometry
- Export database
- Animation
  - (Animations can be saved, but I don’t cover that in these slides)

Output - hardcopy

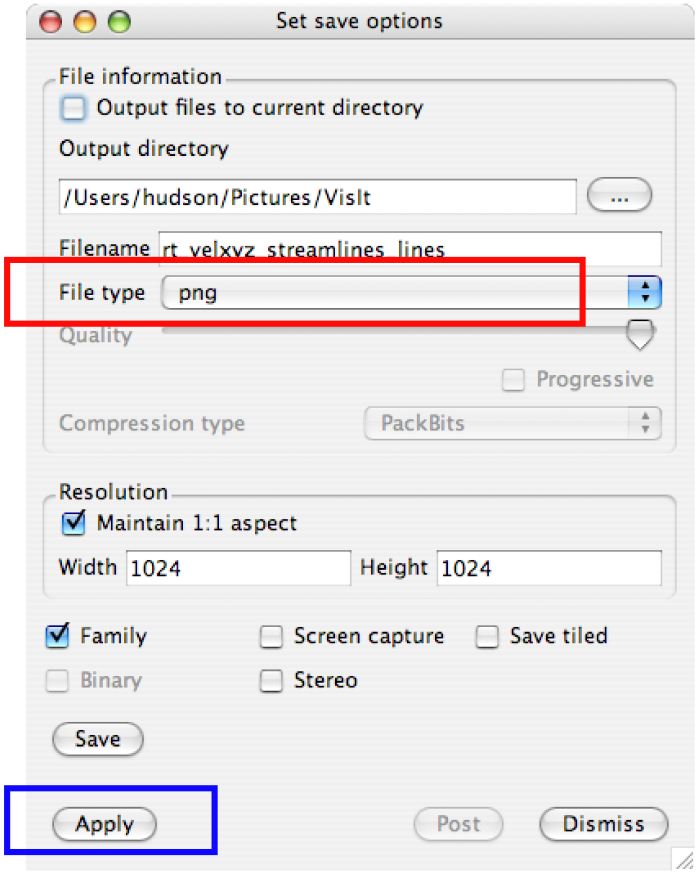
- Hardcopy of visualization window
  - Menu: *File → Set Print options…*



  - Menu: *File → Print window*

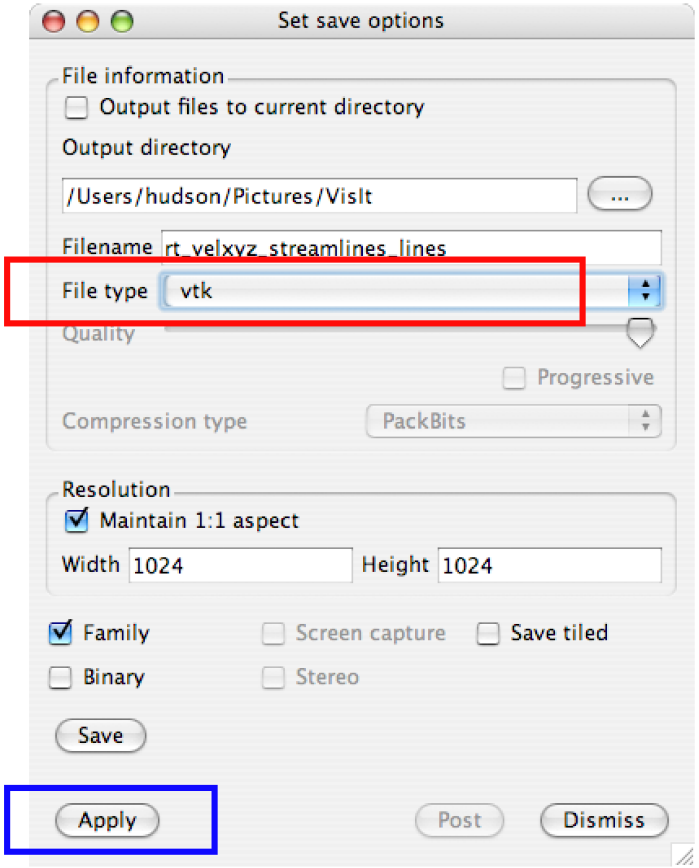
Output - image file

- Image of visualization window
  - Menu: *File → Set Save options…*
  - Select an image format
  - Click apply
  - Menu: *File → Save window*



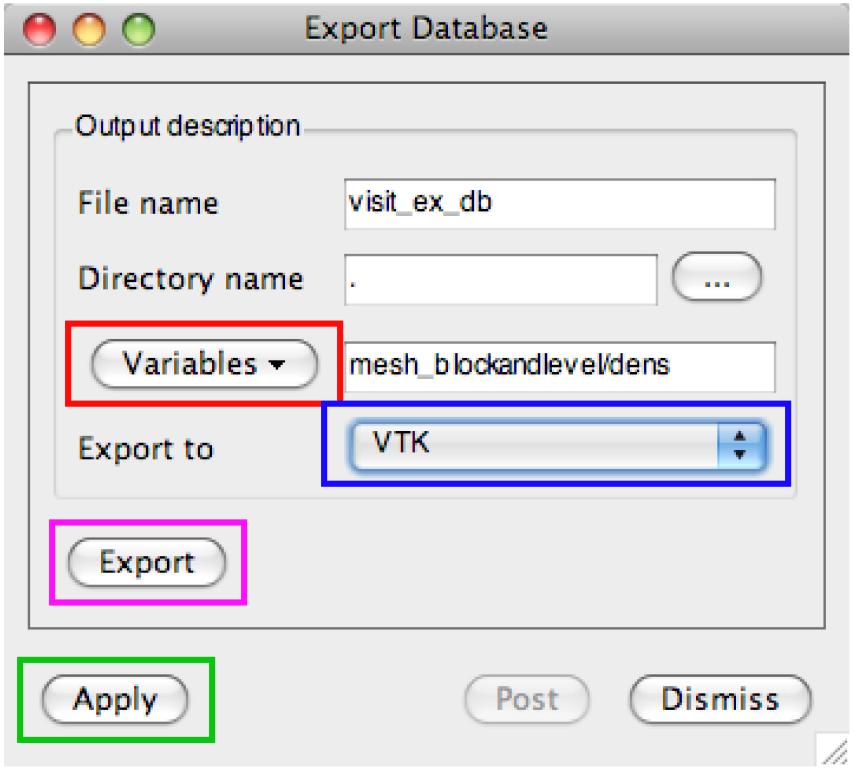
Output - geometry file

- (Outer, viewable) geometry of visualization window
  - Menu: *File → Set Save options…*
  - Select a geometry format
  - Click apply
  - Menu: *File → Save window*



Output - data base

- Entire data base (all (changed) geometry and data)
  - Menu: *File* → *Export database…*
  - Select variables
  - Select format
  - Click *Apply*
  - Click *Export*



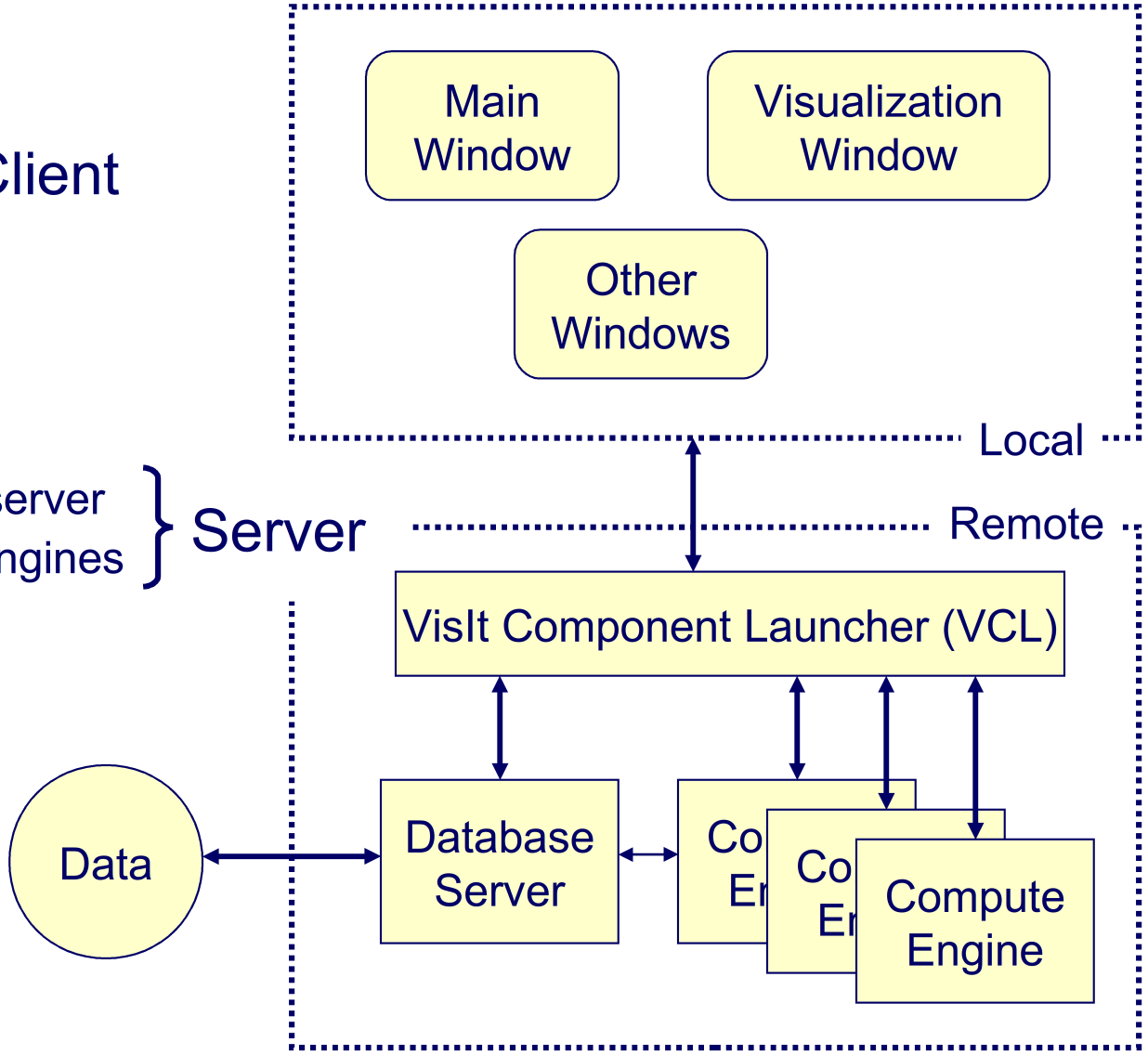
Client-server

- Local GUI
- Local viewer

} Client

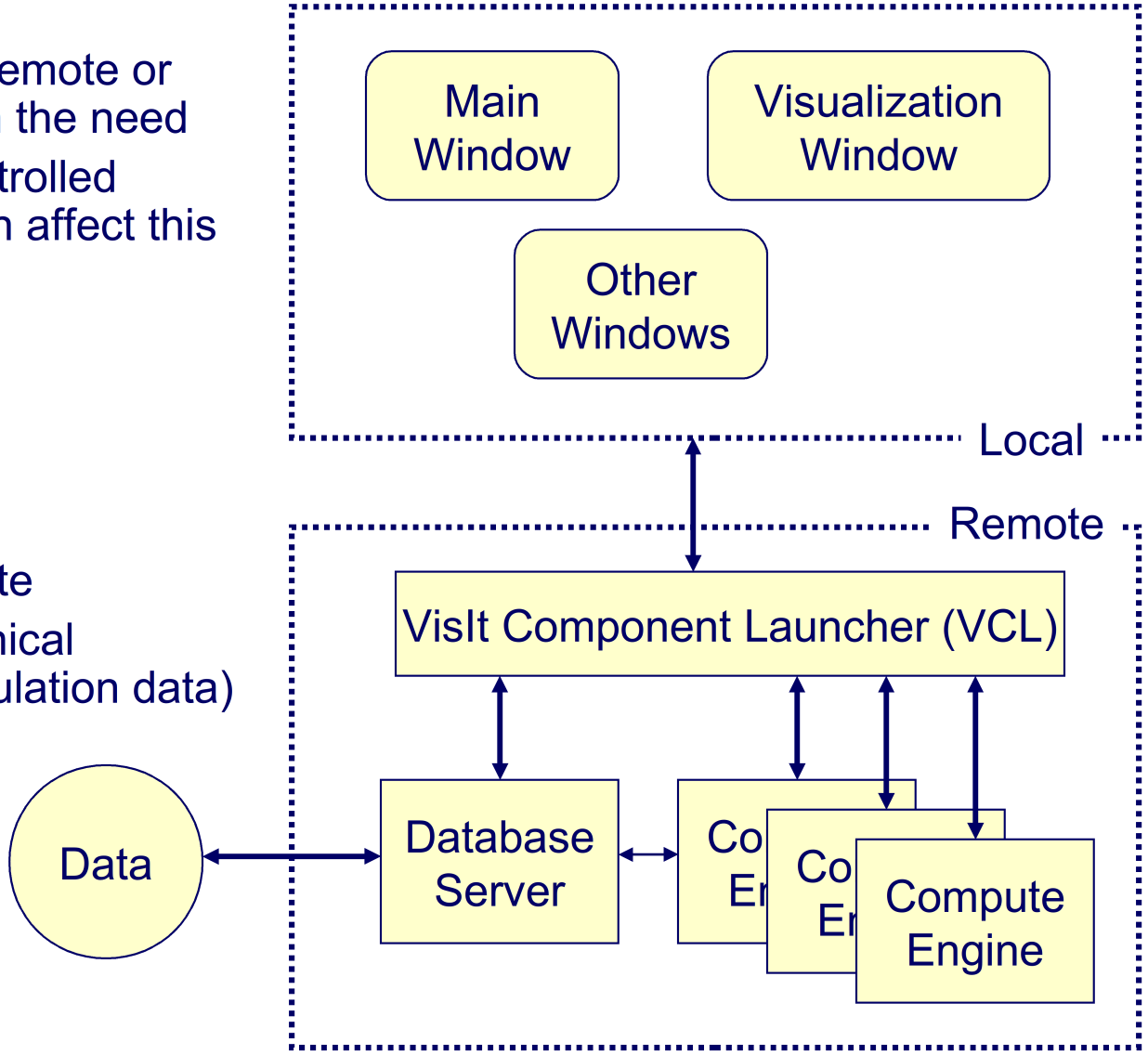
- Remote database server
- Remote compute engines

} Server



Client-server

- Rendering can be remote or local, depending on the need
- There are user-controlled parameters that can affect this
- Data remains remote
- Visualization (graphical primitives from simulation data) is remote



Client-server

- Rendering remotely ⇒ image pieces sent to client
- Rendering locally ⇒ graphical primitives sent to client (cut plane pieces, isosurface pieces, e.g.)

Client-server: conditions

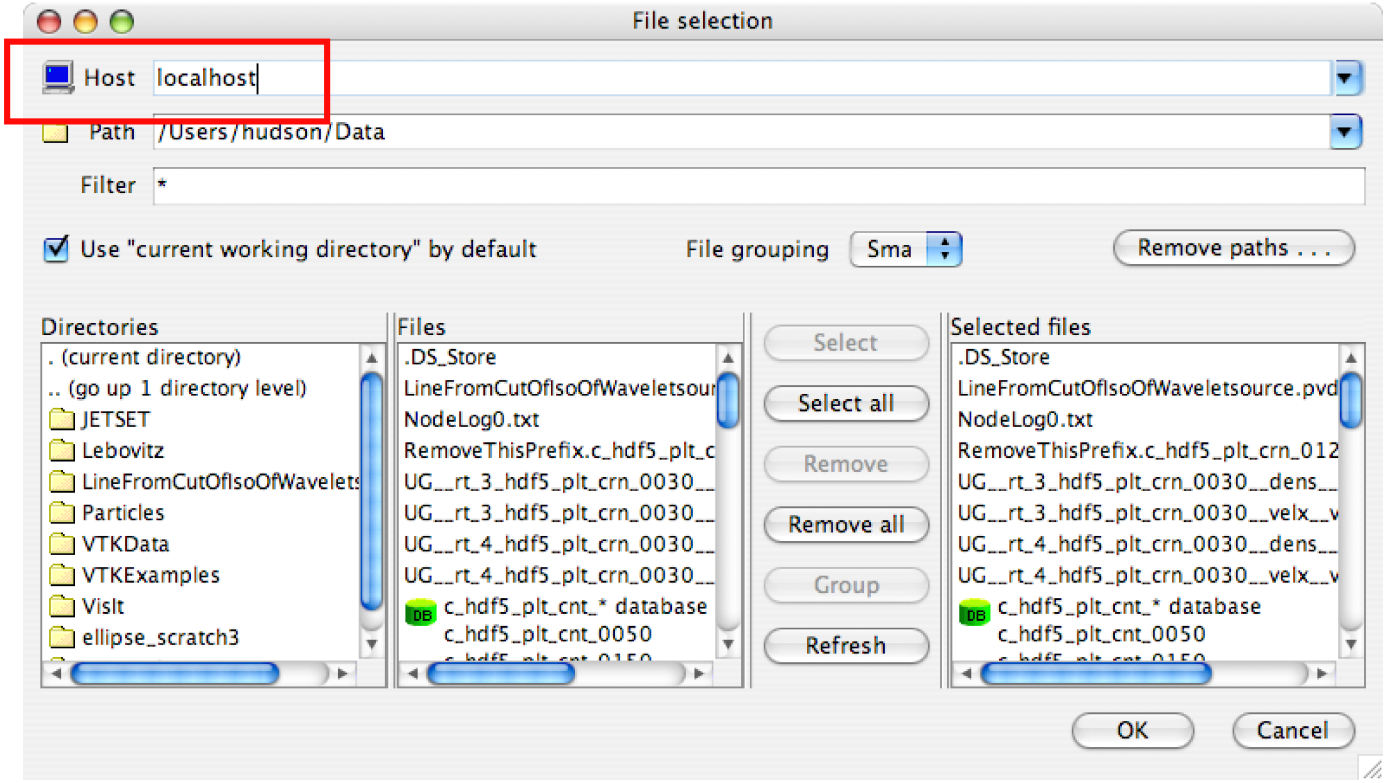
- *visit* installed on all machines
- *visit* in search path on all machines
- Compatible VisIt releases (not patches) on all machines
- (Passwordless *ssh*)

Client-server: open remote file

- Single-process, local VisIt is running
- Open *File selection* window with one of…
  - Key: ⌘ *F*
  - Menu: *File* → *Select file …*

Client-server: open remote file

- Enter name of computer where data is

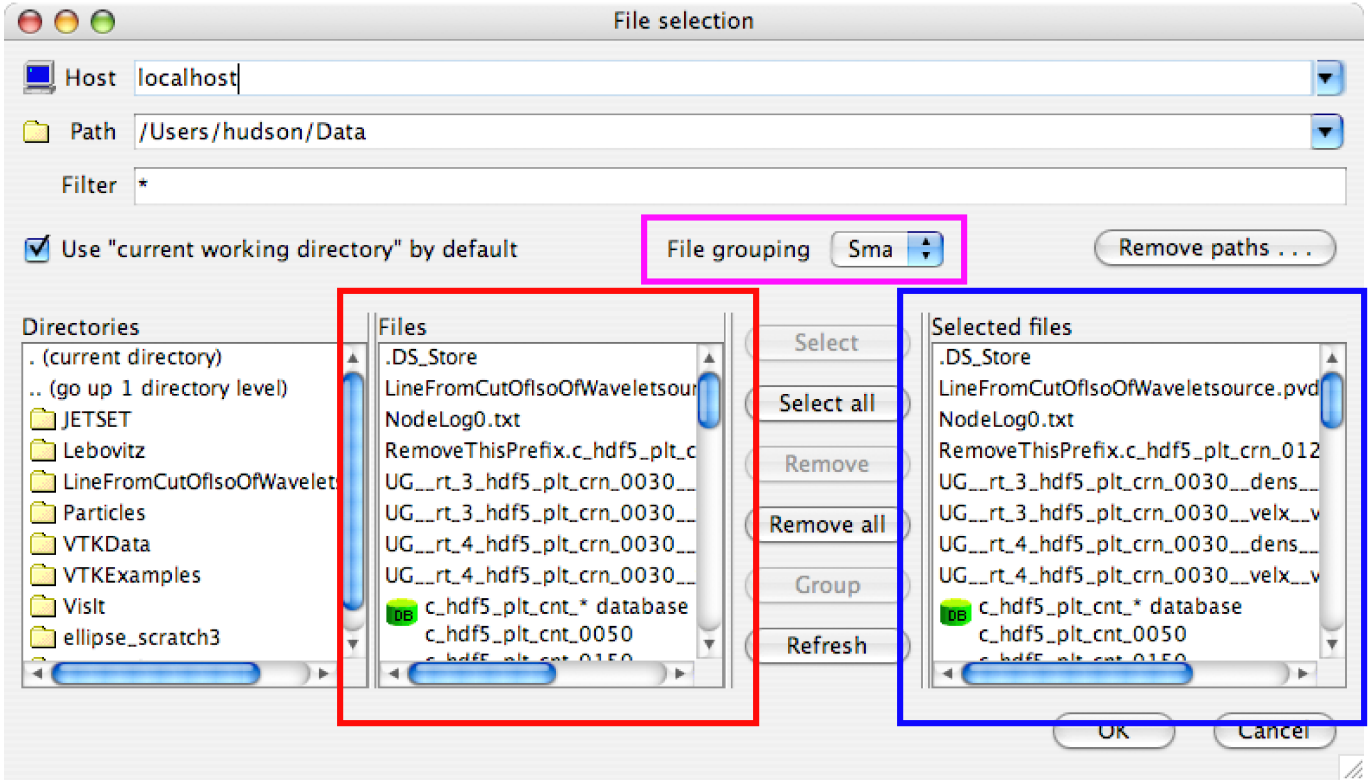


Client-server: open remote file

- VisIt starts VisIt Component Launcher (VCL) on remote machine
- VCL launches database server, compute engines, etc. and runs until you quit VisIt
- Remote files appear in *File selection* window

Client-server: open remote file

- Add files to *Selected files* list
- Can group files



Client-server: open remote file

- Dismiss *File selection* window
- Visualize the files' data

Animation

- Flipbook
  - Simple
  - Steps of a time-variant database
  - Only time changes
- Script
  - If many frames
  - Python or Java

Animation

- Keyframe
  - Complex behavior
  - Time + multiple other parameters change

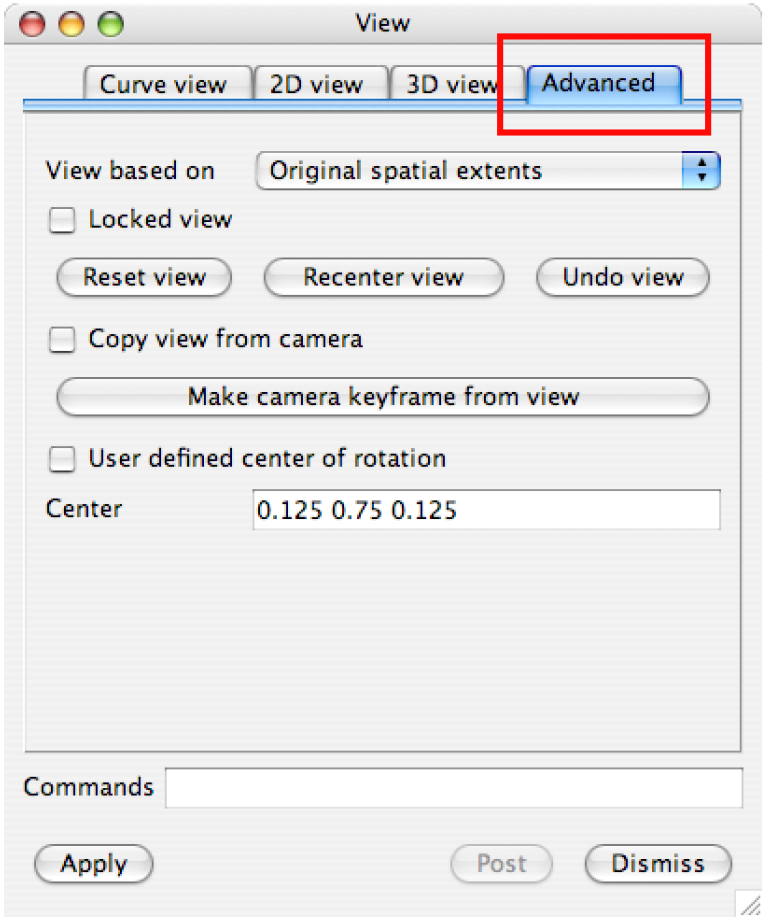
Animation - flipbook

- If file format has each time step in separate file
  - Create *.visit*
- Or
  - Let visit determine (might not be right)

Animation - keyframe

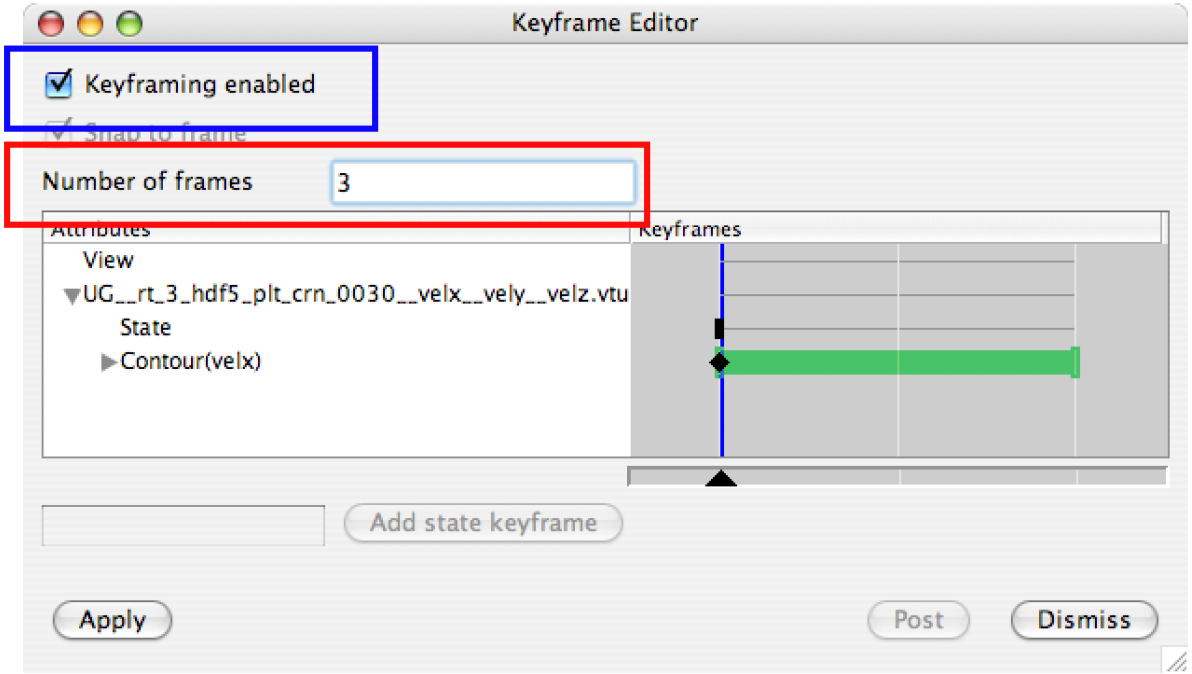
- These 11 slides show how to animate the viewpoint (camera location) of a contour plot over 3 time steps

- Open file
- Create plot
- *View* panel
  - Menu: *Controls* → *View…*
  - Select *Advanced* tab



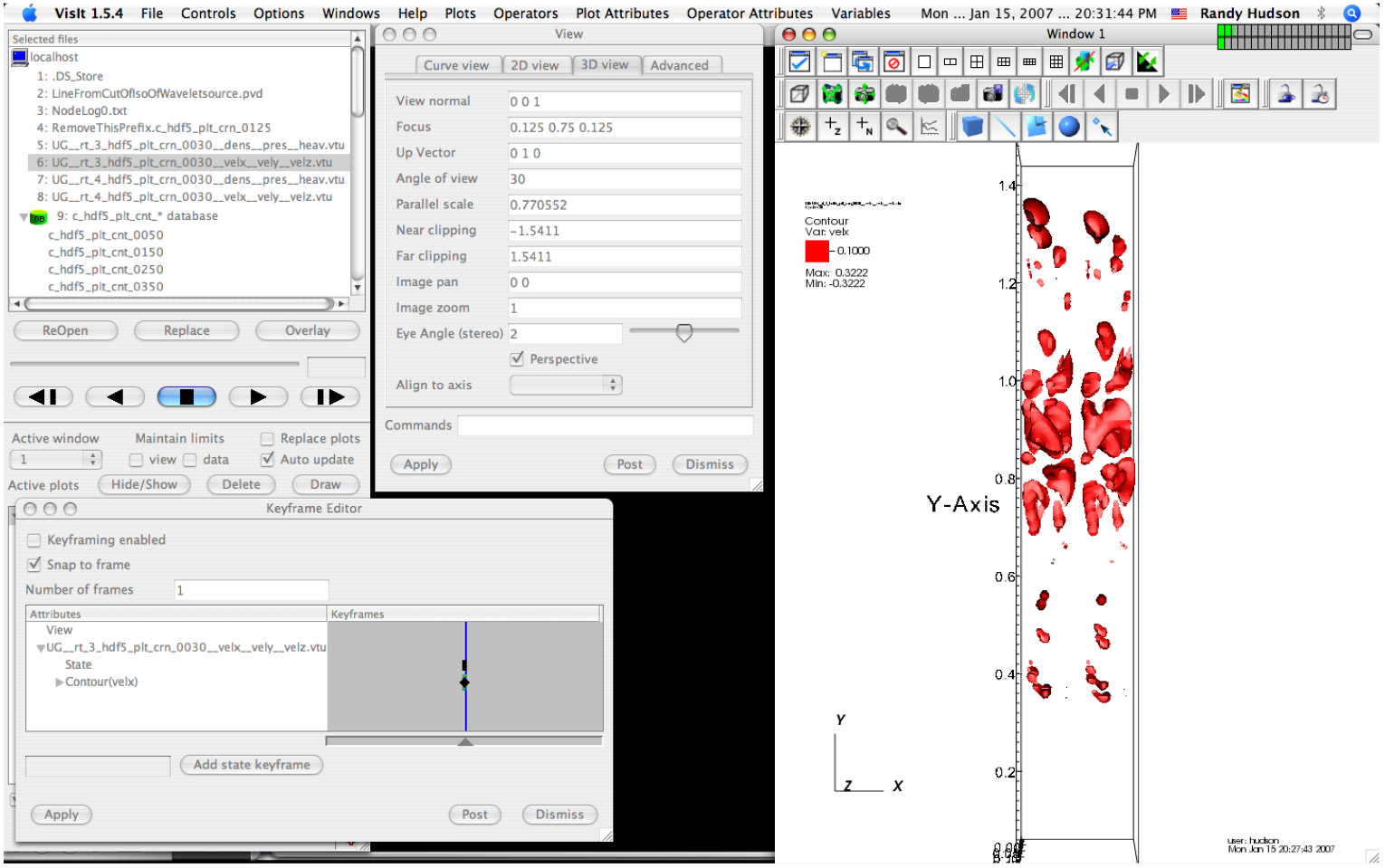
Animation - keyframe

- *Keyframe Editor* panel
  - Menu: *Controls* → *Keyframing…*
  - Set *Number of frames* to 3
  - Click *Keyframing enabled*



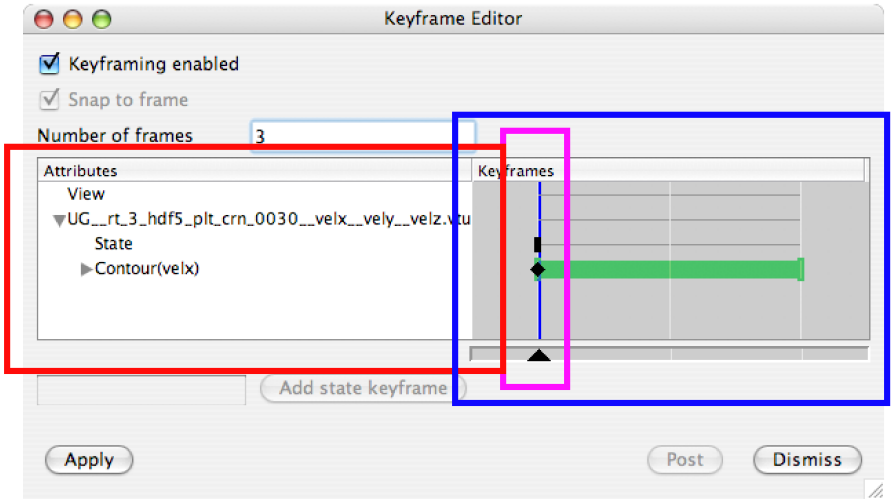
Animation - keyframe

- Suggestion: resize & move VisIt windows so they're accessible



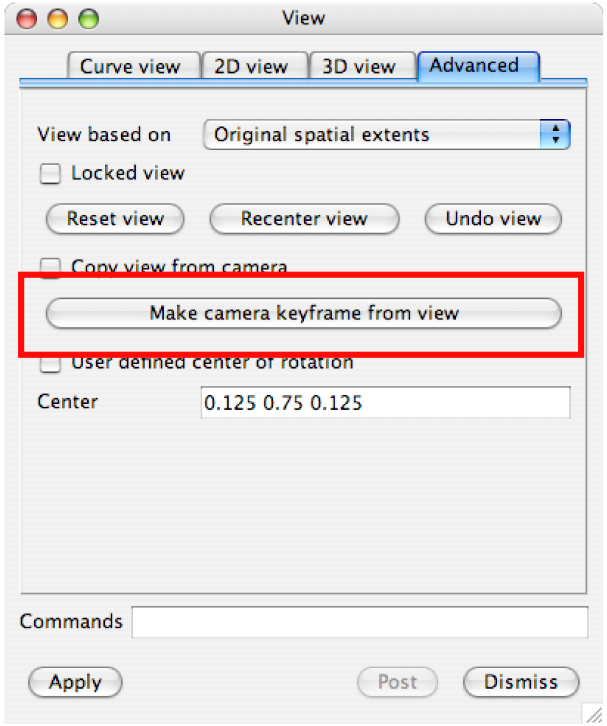
Animation - keyframe

- About the *Keyframe editor* panel
  - *Attributes* area
    - A horizontal entry for each parameter that can change across frames
  - Grey *Keyframes* area
    - A vertical line for each keyframe
    - Green bar for *Contour* attribute means contour will be drawn in visualization window for all keyframes
    - Current keyframe is first keyframe, indicated by blue vertical line and black triangle



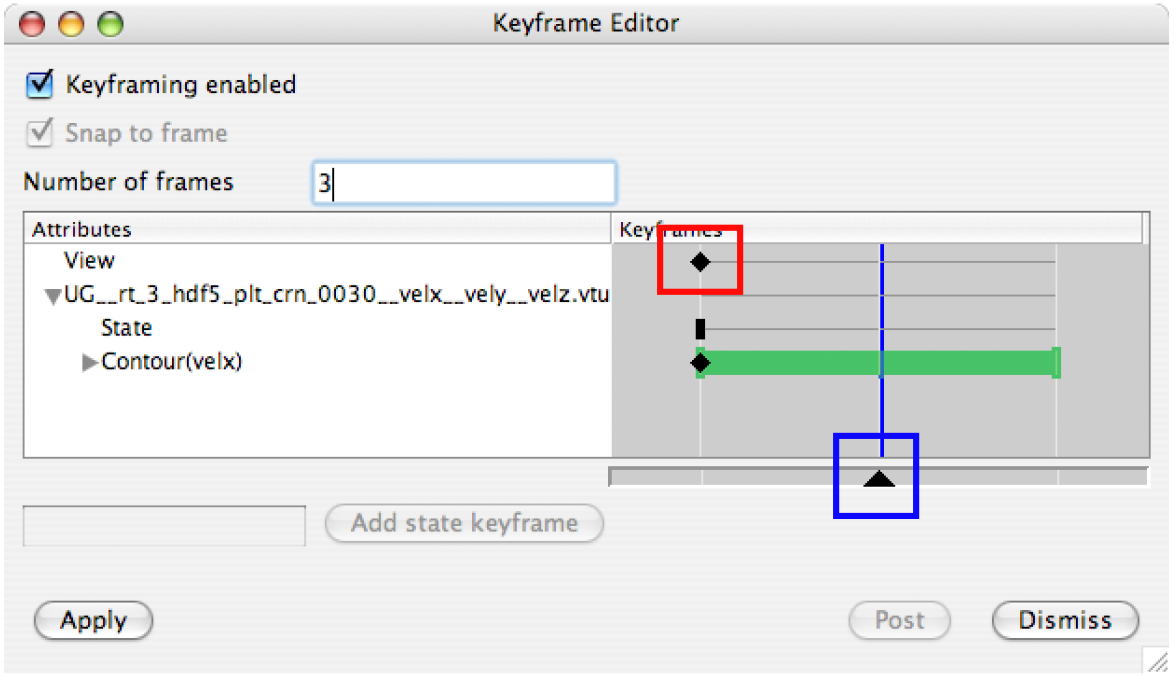
Animation - keyframe

- Make *View* window active and click on *Make camera keyframe from view*



Animation - keyframe

- Current view of plot, in visualization window, becomes first animation frame, as indicated by black diamond
- Click below second keyframe's vertical ine to move black triangle and make *that* keyframe current

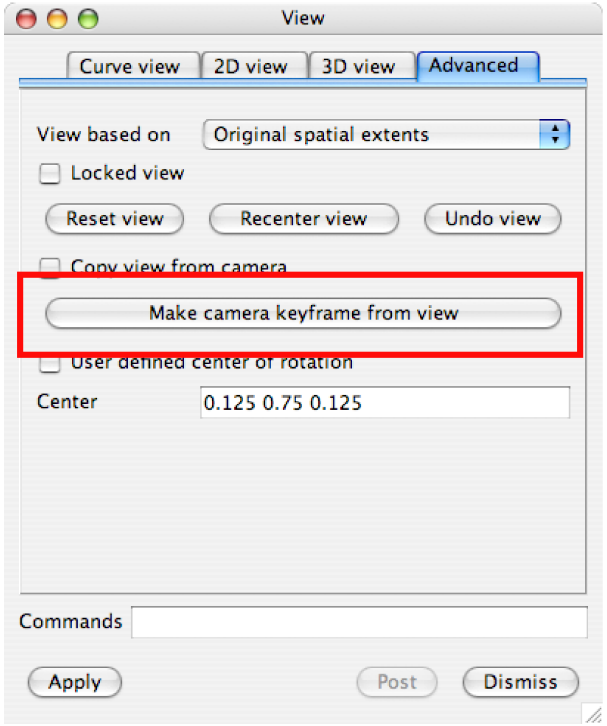


Animation - keyframe

- Make visualization window (where plot Is drawn) active, and rotate plot a significant amount (e.g., 45 degrees)

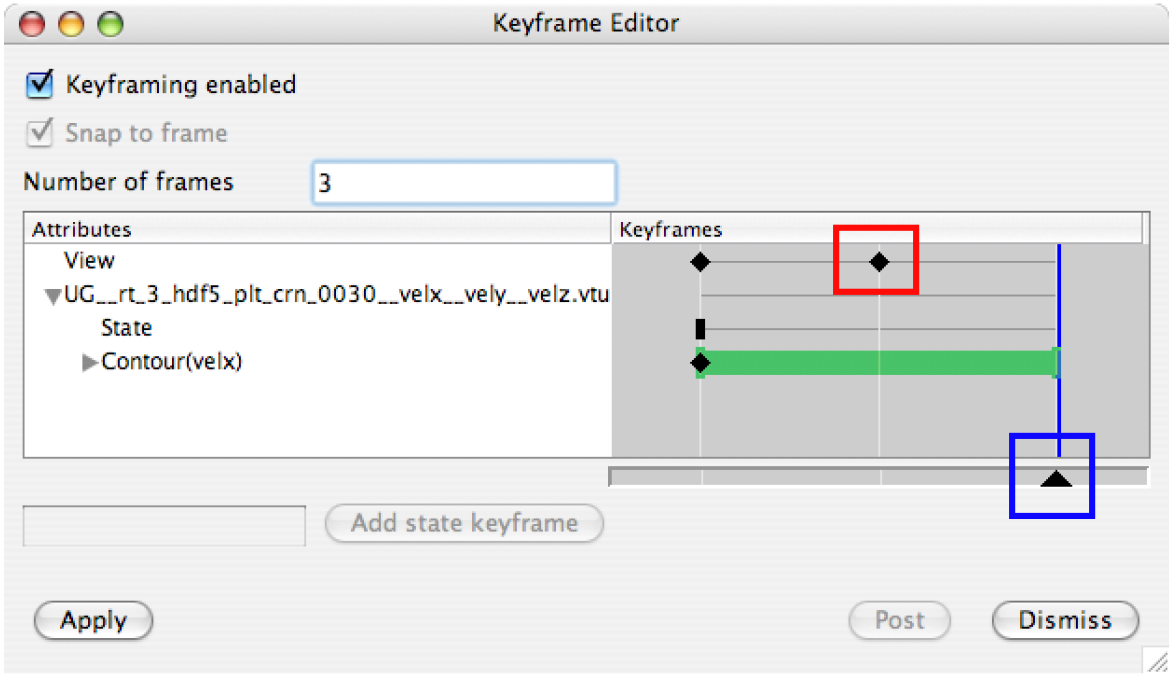
Animation - keyframe

- Make *View* window active and click on *Make camera keyframe from view*



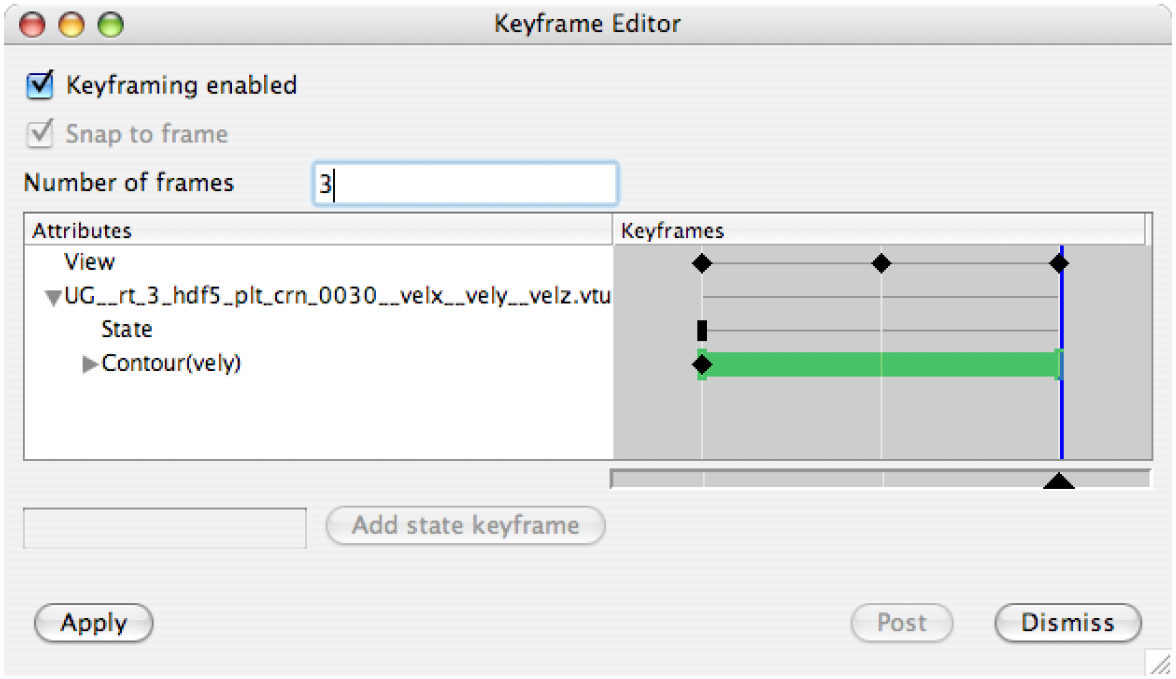
Animation - keyframe

- Current view of plot, in visualization window, becomes second animation frame, as indicated by black diamond
- Click below third keyframe's vertical line to move black triangle and make *that* keyframe current



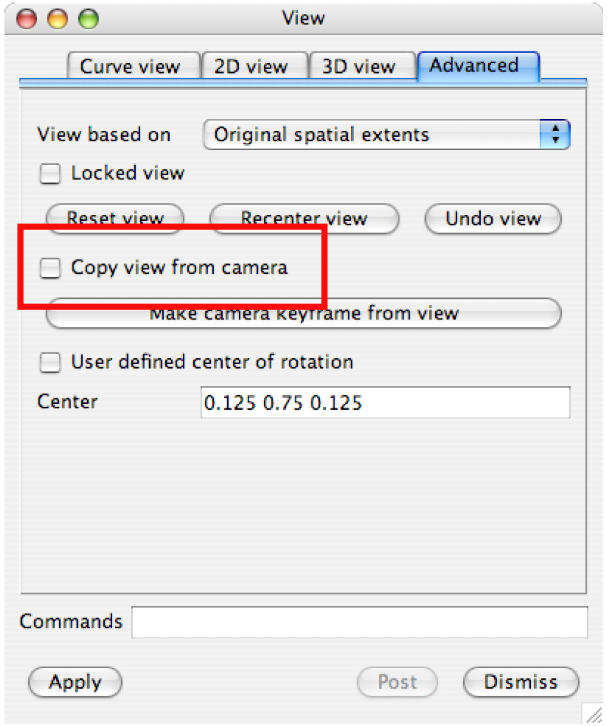
Animation - keyframe

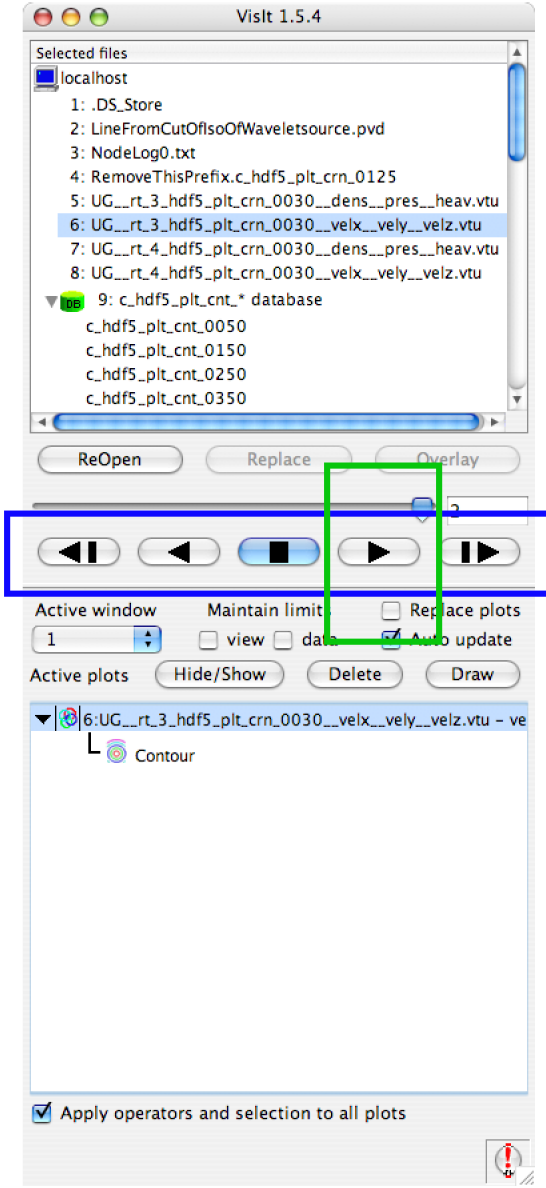
- Rotate plot
- Click on *Make camera keyframe from view* button of *View* window
- This is the resultant *Keyframe Editor* panel



Animation - keyframe

- Click *Copy view from camera* to enable the Time slider of the main window to animate the viewpoint
- A rough, 3-frame animation of the viewpoint can now be run in the visualization window by clicking the play button

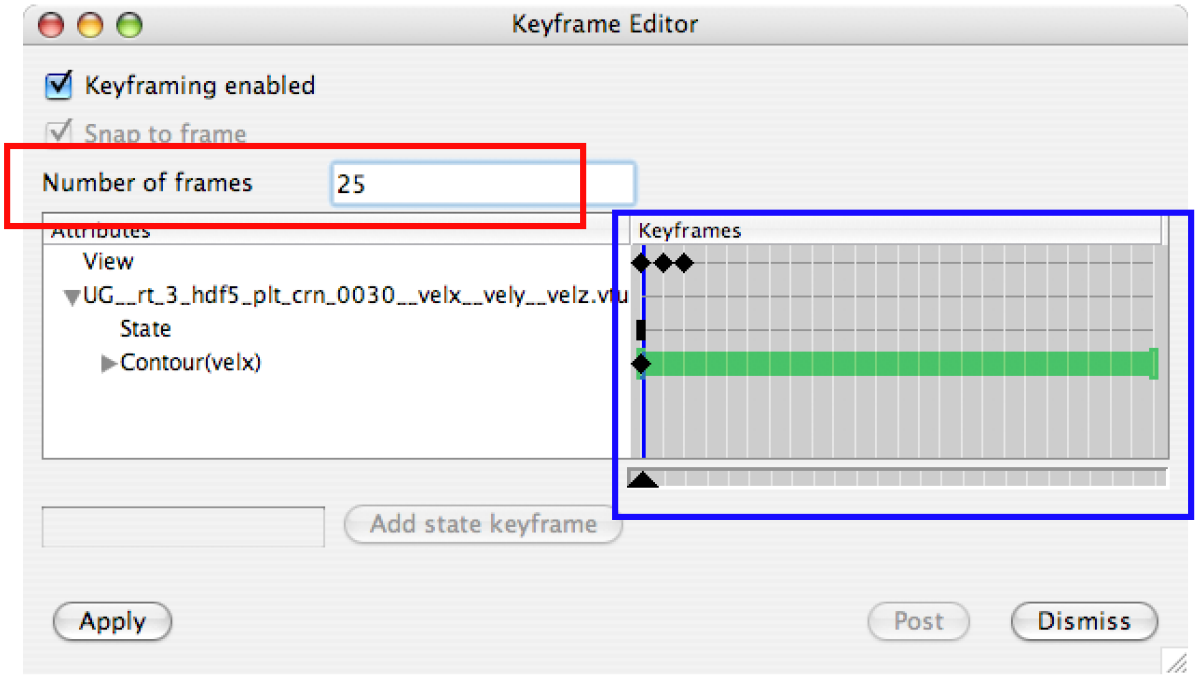




Animate - keyframe

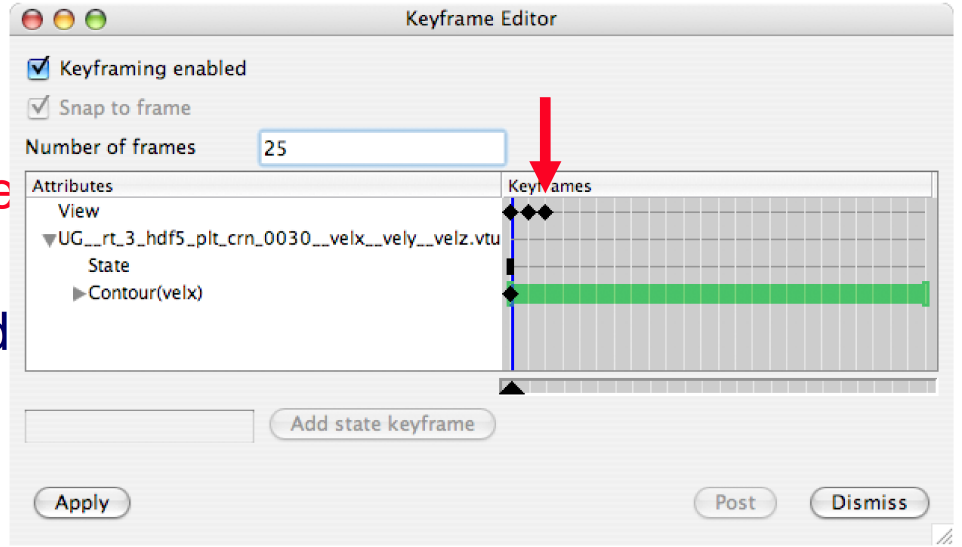
- These 4 slides show how to refine the 3-step animation to one of 25 steps, thus smoothing it

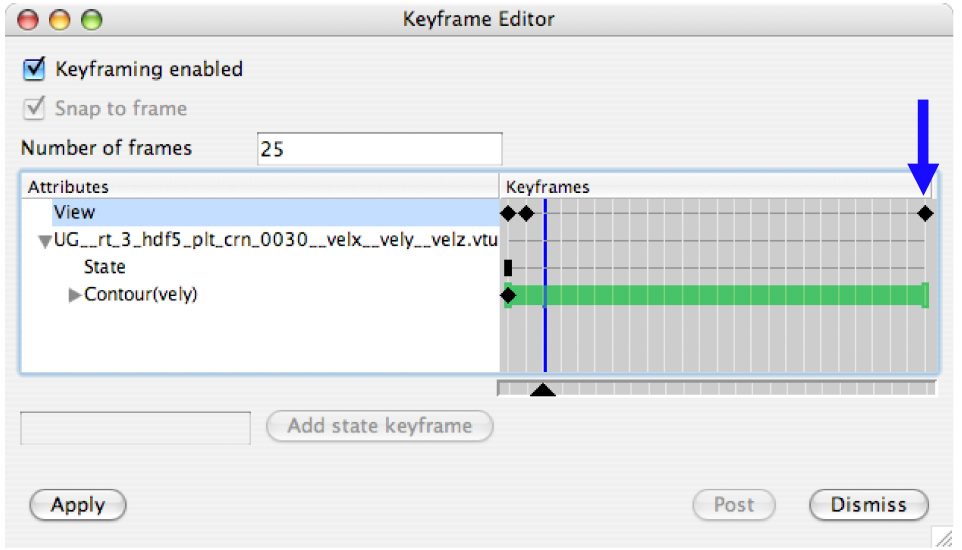
- Change *Number of frames* to 25
  - New (empty) frames are added to *Keyframes* area
  - Green bar reaches keyframe 25



Animate - keyframe

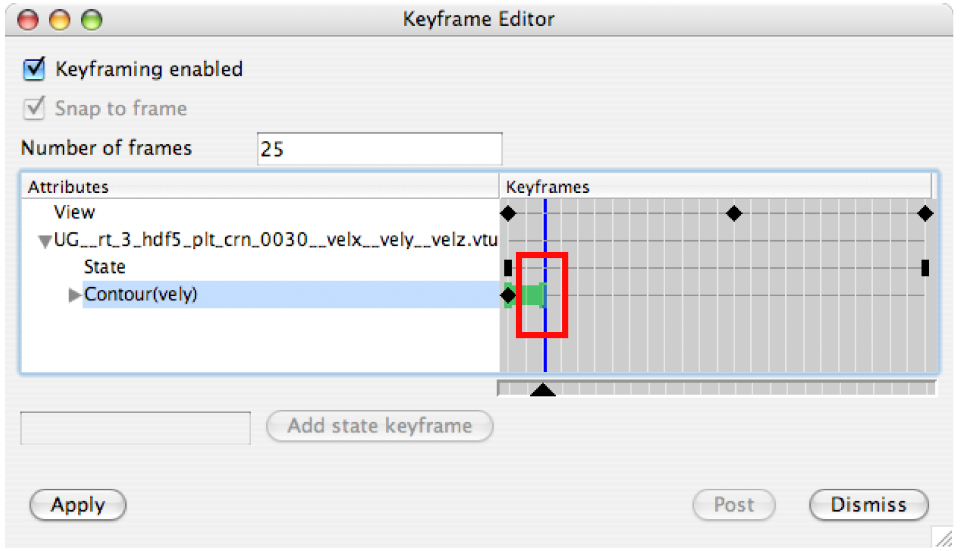
- With the mouse, drag the black diamond for keyframe 3 to keyframe 25
- Likewise, drag the diamond for keyframe 2 to some keyframe close to the middle (result is on next slide)





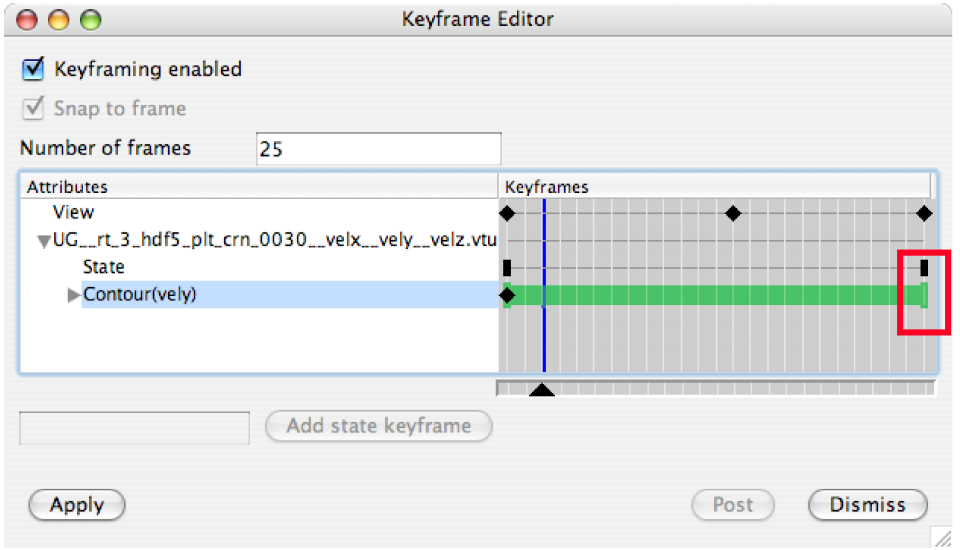
Animate - keyframe

- Click the play button of the main window's time slider
- Hmm… Result not as expected:
  - Only the first 3 frames show the plot
  - Reason: the green bar for the contour plot, which had been extended to keyframe 25 when the number of frames was changed, has reverted to keyframe 3



Animate - keyframe

- Use the mouse to drag the end of the green bar back to keyframe 25
- Run the animation
- Good:
  - Other frames now show views interpolated between original 3



Python CLI

- Run *visit -cli*
  - Starts interactive Python session
  - Loads Python visit module
- To visualize a Pseudocolor plot of variable *var* from data file *dfile*
  - At Python prompt, enter

    OpenDatabase("<*path_to_file*>/*dfile*")
    AddPlot("Pseudocolor", "*var*")
    DrawPlots()

Python script

- Run *visit -cli -s <pythonscript_name>*
  - Loads Python visit module
  - Script calls VisIt commands

Python + GUI

- Menu: *Controls* → *Command…*
  - Record widget clicks and see Python
  - Write Python
  - Execute written or recorded Python