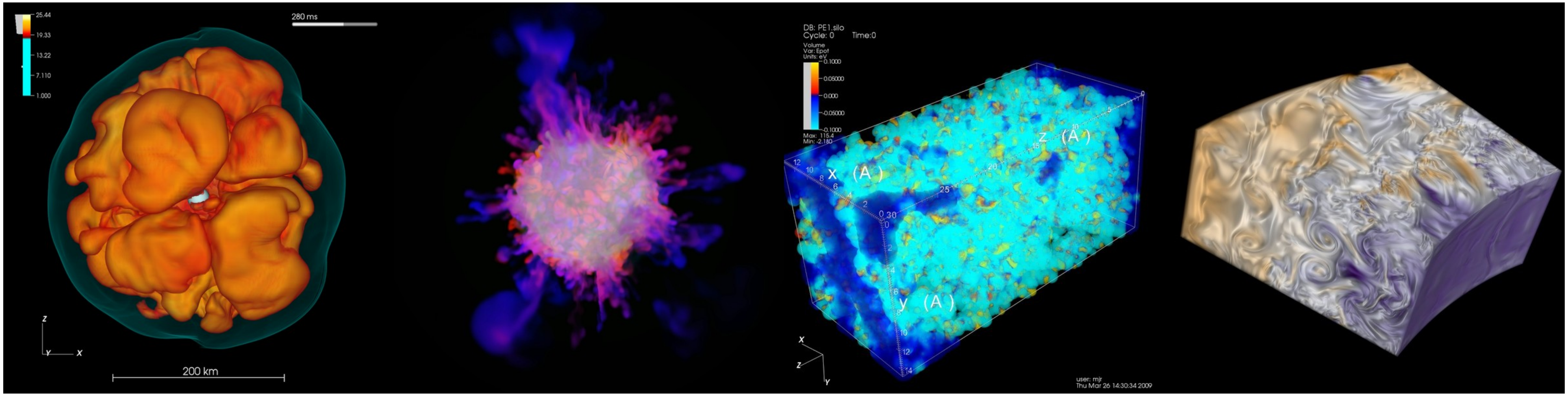
Visualization of molecular-simulation data
– tools overview and tutorial on octopus visualization –

Michele Compostella & Markus Rampp

Max Planck Computing and Data Facility (MPCDF)



Outline

Part I: overview & demo (M. Rampp)

1) overview of visualization methods and tools

   - visualization of (molecular) simulation data with VisIt (main focus) and ParaView (basics)
   - demo of the VisIt GUI → *get familiar with look and feel, basic commands*
   - a selection of advanced VisIt topics → *learn about further capabilities*
   - outlook on in-situ visualization techniques → *prepare discussion on implementation (Thu)*

[lunch break]

2) using the MPCDF remote visualization service (RVS)

Part II: hands-on (M. Compostella)

3) script-based visualization of octopus data

4) Q&A, special topics, “hackathon”, data formats, … (until Thu late morning)

Basic visualization methods

“Visualization” vs. “Rendering”:

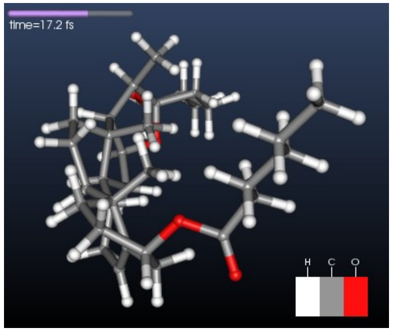
- visualization: visual representation of (simulation) data

  e.g. chemical structures: create a “balls and sticks” model from molecules' positions and render an image

- rendering: generation of an image from shapes

  e.g. 3D photo-realistic rendering of a “balls and sticks” model

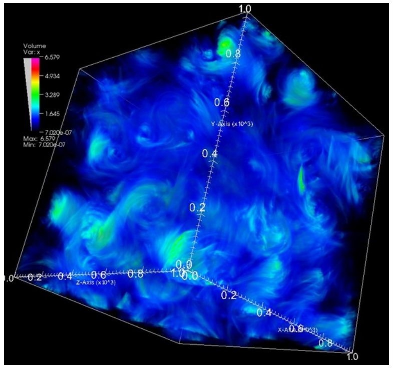
  (→ “information visualization”: creating a model for high-dimensional, unstructured data is highly nontrivial!)



(source: visitusers.org)

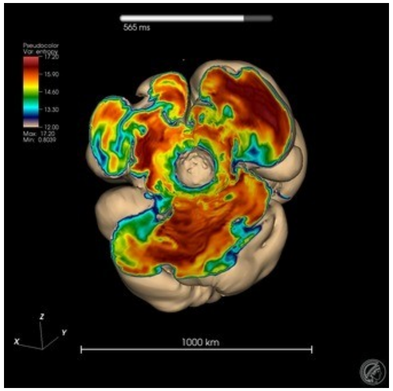
Popular techniques for scalar fields

- volume rendering: ray-casting, splatting
  - transfer function(1D, 2D) + colourtable maps from dataspace (pseudo-realistic: resembles opacity and emissivity of a gas)
  - qualitative (bulk structure and dynamics) + quantitative



volume rendering of 3D data (source: MPG)

- Pseudocolor plots (2D, 3D)
  - colour table provides mapping from dataspace
  - 2D: straightforward, 3D: requires “clipping”
  - most quantitative

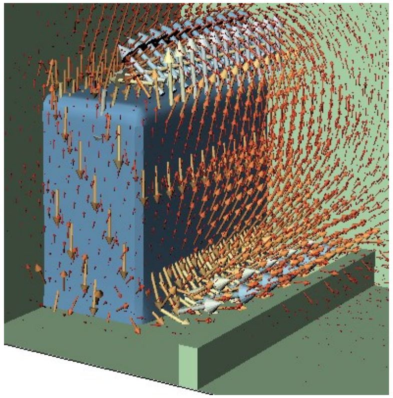


pseudocolor image of 3D data with clipping (source: MPG)

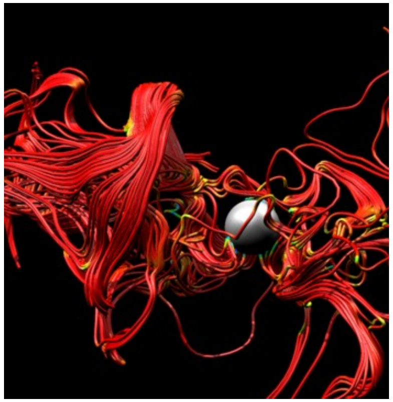
Basic visualization methods

Popular techniques for vector fields

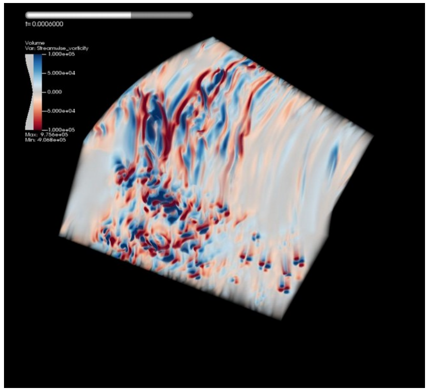
- arrow plots
- streamlines, streaklines, ...
- “contraction” to scalar field:
  - absolute magnitude
  - projected vorticity
  - ...



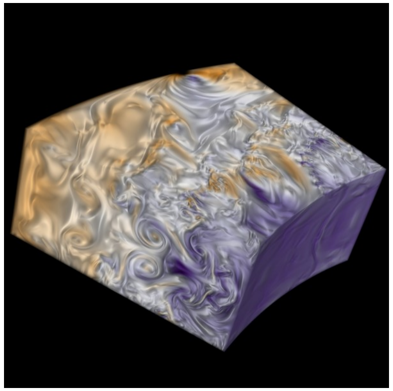
3D vector field
(source: visitusers.org)



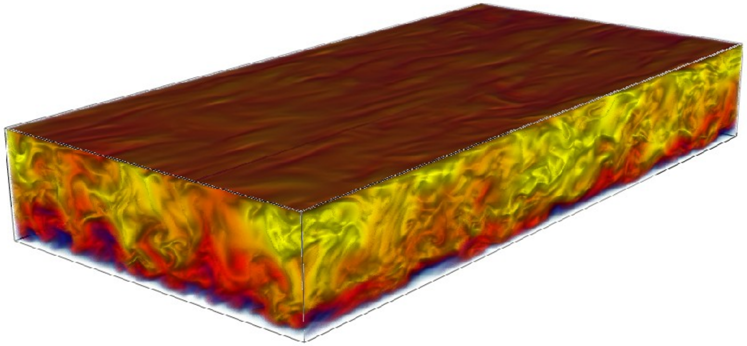
streamlines of 3D data
(source: visitusers.org)



streamwise vorticity
(source: MPG)



velocity magnitude
(source: MPG)



by courtesy of V. Avsarkisov (TU Darmstadt)

Visualization tools

Overview of software tools

- IDL, Matlab, octave, python/matplotlib, … for 1D and 2D plots (+time)

  → automated, quantitative analysis with lots of data processing (powerful languages)

VisIT, Paraview provide complementary functionality, not a complete replacement for IDL, Matlab, python/matplotlib, ...

- VisIt, Paraview for >2D data (+time)

  → interactive exploration

  → quantitative analysis

  → publication-quality plots, movies,

  → scriptable (Python)

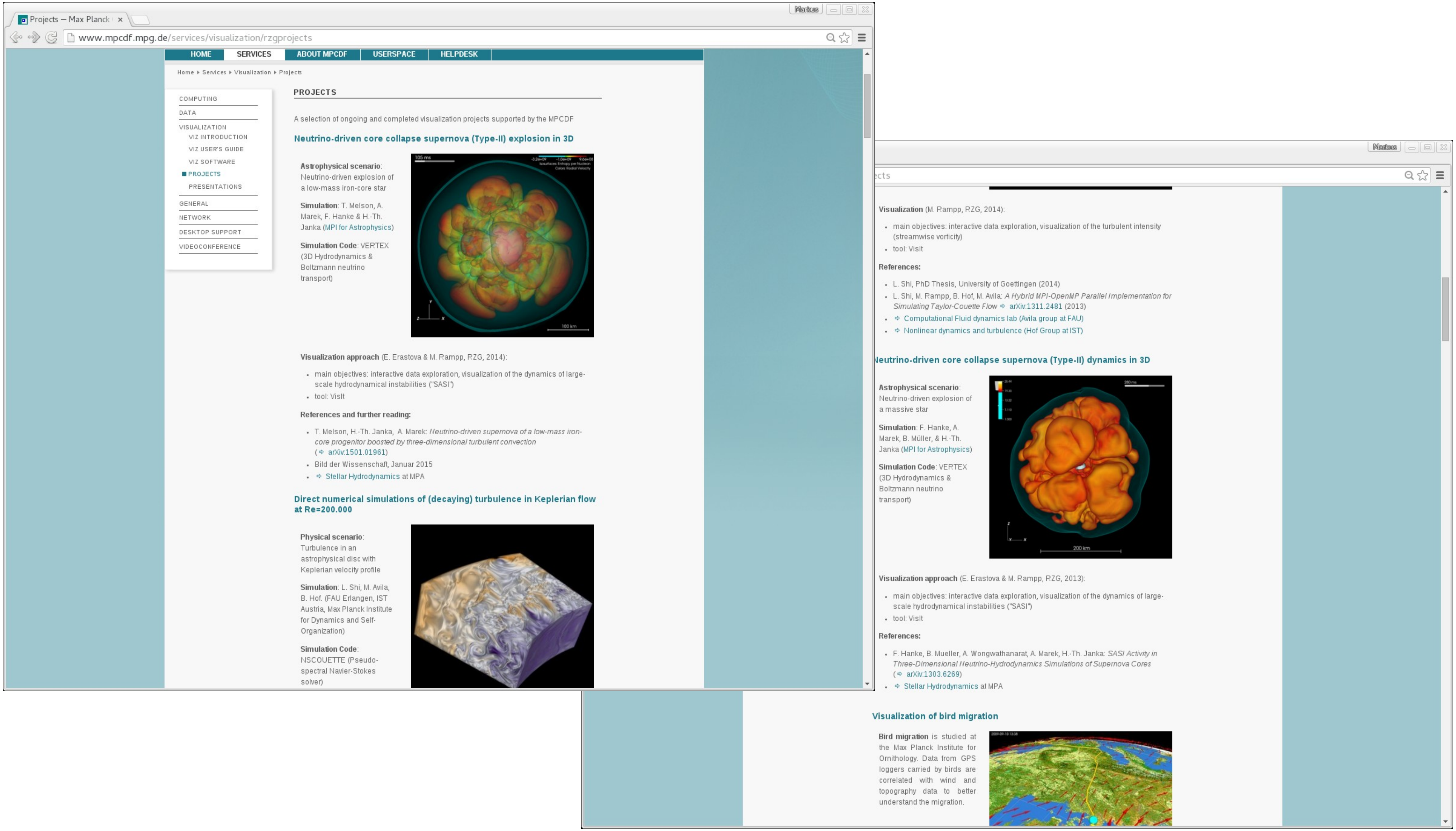
- Others?
  - VAPOR, VOREEN (optimized for special purposes), …, +commercial tools (AVIZO)
  - VMD, PyMOL, … (for – classical MD data): not considered in much detail so far, apparent weaknesses in handling scalar -DFT- data, solids)

Example projects

Example projects at the MPCDF (since 2006)

- scientific domains:
  - plasmaphysics, astrophysics, CFD, molecular dynamics, DFT, ...
- data structures/grids:
  - regular: cartesian, polar (2D, 3D), block-structured (”Yin-Yan”)
  - molecular data with replicated unit cells
  - irregular: (mapped) point clouds
- data sizes, dimensions:
  - up to 2048^3 (cartesian), 1000 × 180 × 360 (polar), 2048 × 769 × 1153 (cylindrical)
  - up to 10^7 particles in 3D, 10^7 nodes in 3D unstructured mesh
  - all: multi-variable (scalar, vector), time-dependent
  - see also: http://www.rzg.mpg.de/services/visualisation/scientificdata/projects
- tools: VisIT, Paraview, parallel HDF5 (+XDMF)

Example projects



PROJECTS

A selection of ongoing and completed visualization projects supported by the MPCDF

Neutrino-driven core collapse supernova (Type-II) explosion in 3D

Astrophysical scenario: Neutrino-driven explosion of a low-mass iron-core star

Simulation: T. Melson, A. Marek, F. Hanke & H.-Th. Janka (MPI for Astrophysics)

Simulation Code: VERTEX (3D Hydrodynamics & Boltzmann neutrino transport)

Visualization approach (E. Erastova & M. Rampp, RZG, 2014):

- main objectives: interactive data exploration, visualization of the dynamics of large-scale hydrodynamical instabilities ("SASI")
- tool: VisIt

References and further reading:

- T. Melson, H.-Th. Janka, A. Marek: *Neutrino-driven supernova of a low-mass iron-core progenitor boosted by three-dimensional turbulent convection* (➪ arXiv:1501.01961)
- Bild der Wissenschaft, Januar 2015
- ➪ Stellar Hydrodynamics at MPA

Direct numerical simulations of (decaying) turbulence in Keplerian flow at Re=200.000

Physical scenario: Turbulence in an astrophysical disc with Keplerian velocity profile

Simulation: L. Shi, M. Avila, B. Hof. (FAU Erlangen, IST Austria, Max Planck Institute for Dynamics and Self-Organization)

Simulation Code: NSCOUETTE (Pseudo-spectral Navier-Stokes solver)

Visualization (M. Rampp, RZG, 2014):

- main objectives: interactive data exploration, visualization of the turbulent intensity (streamwise vorticity)
- tool: VisIt

References:

- L. Shi, PhD Thesis, University of Goettingen (2014)
- L. Shi, M. Rampp, B. Hof, M. Avila: *A Hybrid MPI-OpenMP Parallel Implementation for Simulating Taylor-Couette Flow* ➪ arXiv:1311.2481 (2013)
- ➪ Computational Fluid dynamics lab (Avila group at FAU)
- ➪ Nonlinear dynamics and turbulence (Hof Group at IST)

Neutrino-driven core collapse supernova (Type-II) dynamics in 3D

Astrophysical scenario: Neutrino-driven explosion of a massive star

Simulation: F. Hanke, A. Marek, B. Müller, & H.-Th. Janka (MPI for Astrophysics)

Simulation Code: VERTEX (3D Hydrodynamics & Boltzmann neutrino transport)

Visualization approach (E. Erastova & M. Rampp, RZG, 2013):

- main objectives: interactive data exploration, visualization of the dynamics of large-scale hydrodynamical instabilities ("SASI")
- tool: VisIt

References:

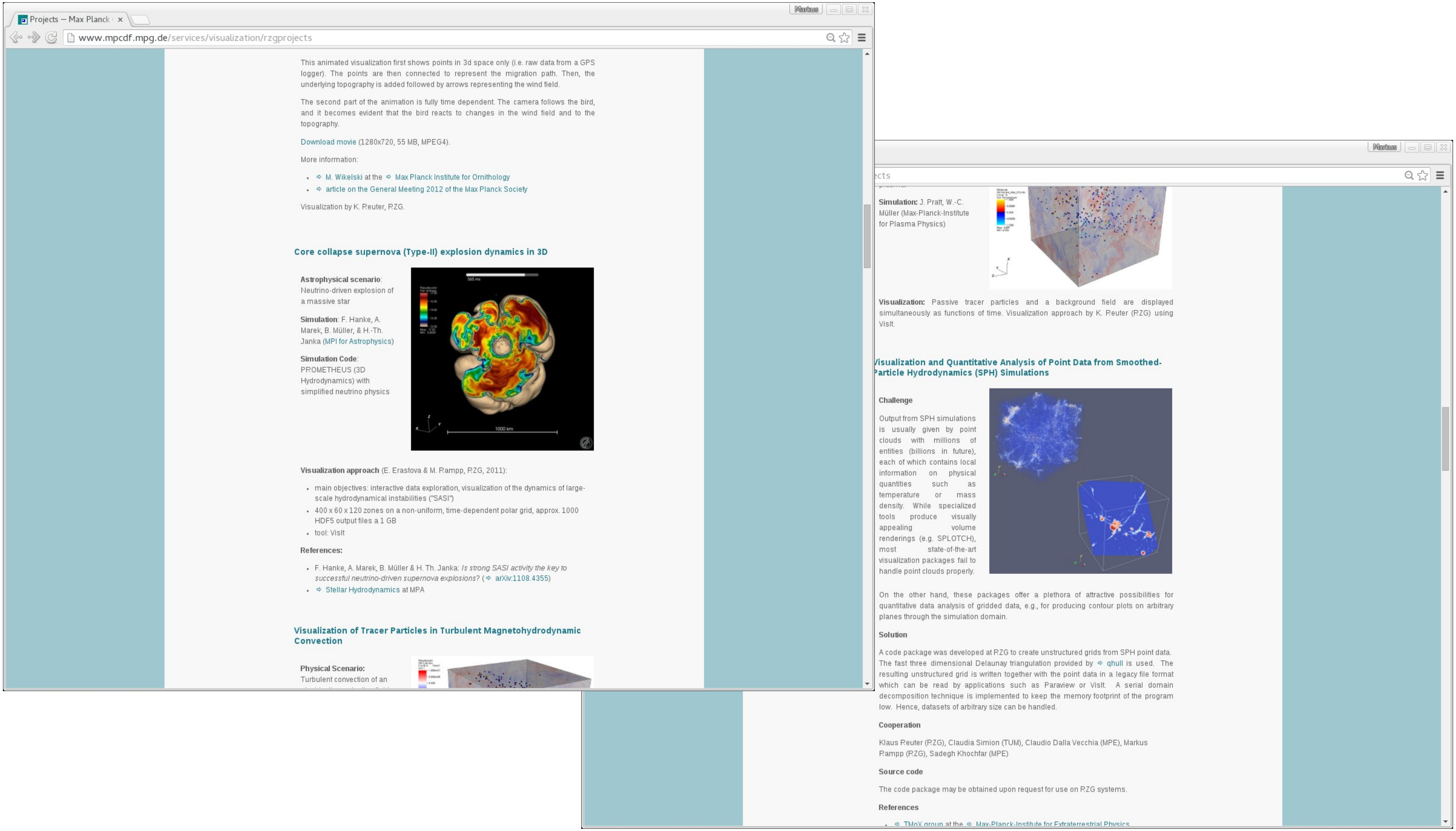
- F. Hanke, B. Mueller, A. Wongwathanarat, A. Marek, H.-Th. Janka: *SASI Activity in Three-Dimensional Neutrino-Hydrodynamics Simulations of Supernova Cores* (➪ arXiv:1303.6269)
- ➪ Stellar Hydrodynamics at MPA

Visualization of bird migration

Bird migration is studied at the Max Planck Institute for Ornithology. Data from GPS loggers carried by birds are correlated with wind and topography data to better understand the migration.

http://www.rzg.mpg.de/services/visualisation/scientificdata/projects

Example projects



This animated visualization first shows points in 3d space only (i.e. raw data from a GPS logger). The points are then connected to represent the migration path. Then, the underlying topography is added followed by arrows representing the wind field.

The second part of the animation is fully time dependent. The camera follows the bird, and it becomes evident that the bird reacts to changes in the wind field and to the topography.

Download movie (1280x720, 55 MB, MPEG4).

More information:

- ➪ M. Wikelski at the ➪ Max Planck Institute for Ornithology
- ➪ article on the General Meeting 2012 of the Max Planck Society

Visualization by K. Reuter, RZG.

Core collapse supernova (Type-II) explosion dynamics in 3D

Astrophysical scenario: Neutrino-driven explosion of a massive star

Simulation: F. Hanke, A. Marek, B. Müller, & H.-Th. Janka (MPI for Astrophysics)

Simulation Code: PROMETHEUS (3D Hydrodynamics) with simplified neutrino physics

Visualization approach (E. Erastova & M. Rampp, RZG, 2011):

- main objectives: interactive data exploration, visualization of the dynamics of large-scale hydrodynamical instabilities ("SASI")
- 400 x 60 x 120 zones on a non-uniform, time-dependent polar grid, approx. 1000 HDF5 output files a 1 GB
- tool: VisIt

References:

- F. Hanke, A. Marek, B. Müller & H. Th. Janka: *Is strong SASI activity the key to successful neutrino-driven supernova explosions?* (➪ arXiv:1108.4355)
- ➪ Stellar Hydrodynamics at MPA

Visualization of Tracer Particles in Turbulent Magnetohydrodynamic Convection

Physical Scenario: Turbulent convection of an

Simulation: J. Pratt, W.-C. Müller (Max-Planck-Institute for Plasma Physics)

Visualization: Passive tracer particles and a background field are displayed simultaneously as functions of time. Visualization approach by K. Reuter (RZG) using VisIt.

Visualization and Quantitative Analysis of Point Data from Smoothed-Particle Hydrodynamics (SPH) Simulations

Challenge

Output from SPH simulations is usually given by point clouds with millions of entities (billions in future), each of which contains local information on physical quantities such as temperature or mass density. While specialized tools produce visually appealing volume renderings (e.g. SPLOTCH), most state-of-the-art visualization packages fail to handle point clouds properly.

On the other hand, these packages offer a plethora of attractive possibilities for quantitative data analysis of gridded data, e.g., for producing contour plots on arbitrary planes through the simulation domain.

Solution

A code package was developed at RZG to create unstructured grids from SPH point data. The fast three dimensional Delaunay triangulation provided by ➪ qhull is used. The resulting unstructured grid is written together with the point data in a legacy file format which can be read by applications such as Paraview or VisIt. A serial domain decomposition technique is implemented to keep the memory footprint of the program low. Hence, datasets of arbitrary size can be handled.

Cooperation

Klaus Reuter (RZG), Claudia Simion (TUM), Claudio Dalla Vecchia (MPE), Markus Rampp (RZG), Sadegh Khochfar (MPE)

Source code

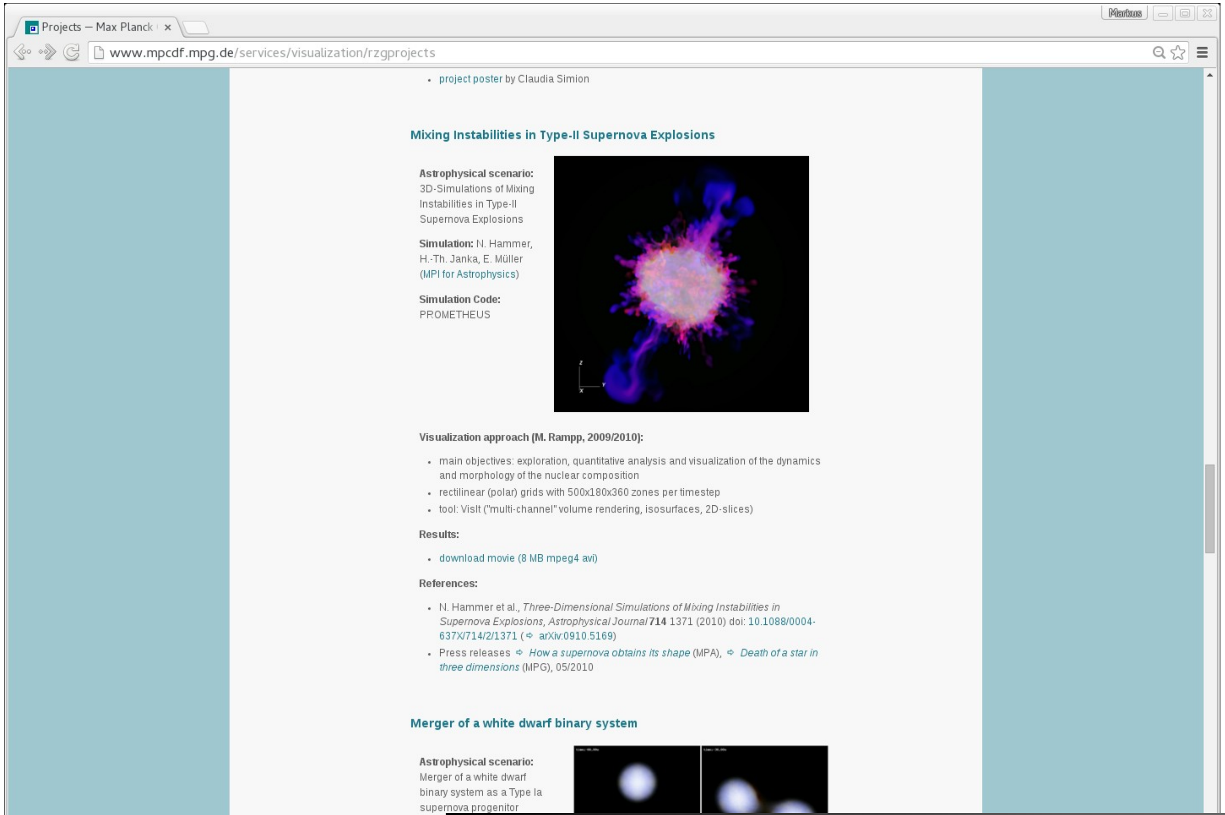
The code package may be obtained upon request for use on RZG systems.

References

- ➪ TMoX group at the ➪ Max-Planck-Institute for Extraterrestrial Physics

http://www.rzg.mpg.de/services/visualisation/scientificdata/projects

Example projects



- project poster by Claudia Simion

Mixing Instabilities in Type-II Supernova Explosions

Astrophysical scenario: 3D-Simulations of Mixing Instabilities in Type-II Supernova Explosions

Simulation: N. Hammer, H.-Th. Janka, E. Müller (MPI for Astrophysics)

Simulation Code: PROMETHEUS

Visualization approach (M. Rampp, 2009/2010):

- main objectives: exploration, quantitative analysis and visualization of the dynamics and morphology of the nuclear composition
- rectilinear (polar) grids with 500x180x360 zones per timestep
- tool: VisIt ("multi-channel" volume rendering, isosurfaces, 2D-slices)

Results:

- download movie (8 MB mpeg4 avi)

References:

- N. Hammer et al., *Three-Dimensional Simulations of Mixing Instabilities in Supernova Explosions, Astrophysical Journal* **714** 1371 (2010) doi: 10.1088/0004-637X/714/2/1371 (➪ arXiv:0910.5169)
- Press releases ➪ *How a supernova obtains its shape* (MPA), ➪ *Death of a star in three dimensions* (MPG), 05/2010

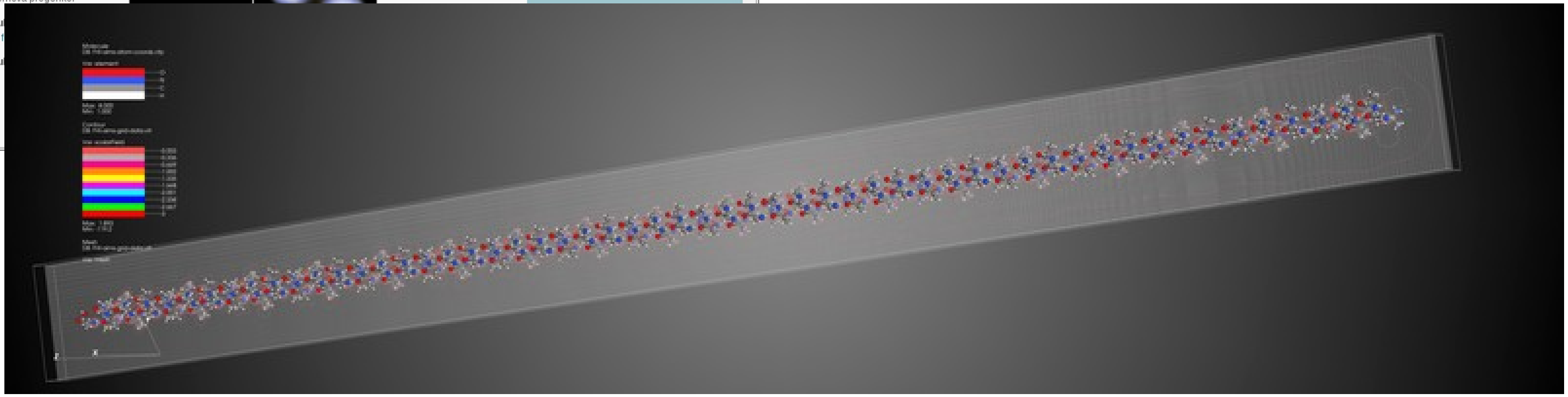
Merger of a white dwarf binary system

Astrophysical scenario: Merger of a white dwarf binary system as a Type Ia supernova progenitor

Simu

(MPI f

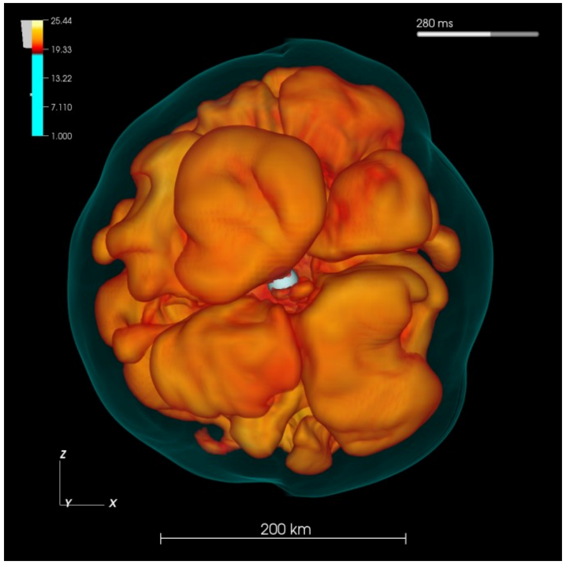
Simu



http://www.rzg.mpg.de/services/visualisation/scientificdata/projects

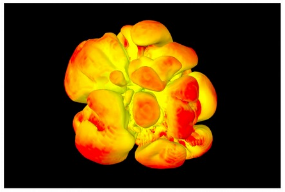
Visualization projects

Visualization highlights (focus on dissemination)



3D Supernova simulations

Simulation: H.-Th. Janka et al. (Max-Planck-Inst. f. Astrophysics)
Visualization: E. Erastova & M. Rampp (RZG)
Tool: VisIt

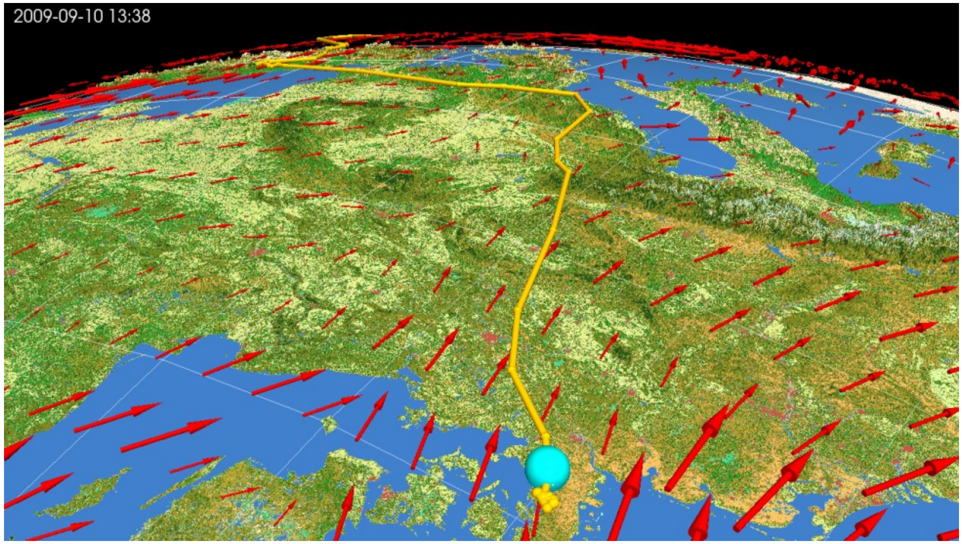


„Max Planck Award 2013 – Hidden Treasures ?“ (2nd prize)
Bild der Wissenschaft, Jan 2015

Bird migration & geospatial data

GPS-Data: M. Wikelski et al. (Max-Planck-Inst. f. Ornithology)
Visualization: K. Reuter (RZG)
Tool: Paraview

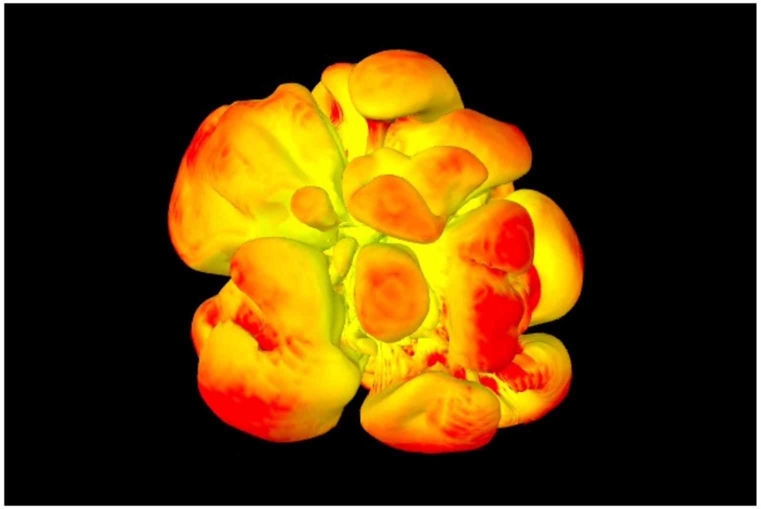
presented at general assembly of the MPG, 2012 and installed at the “Hennhouse” (wall projection at the Visitor Center in Radolfzell)



Example projects: Interactive graphics

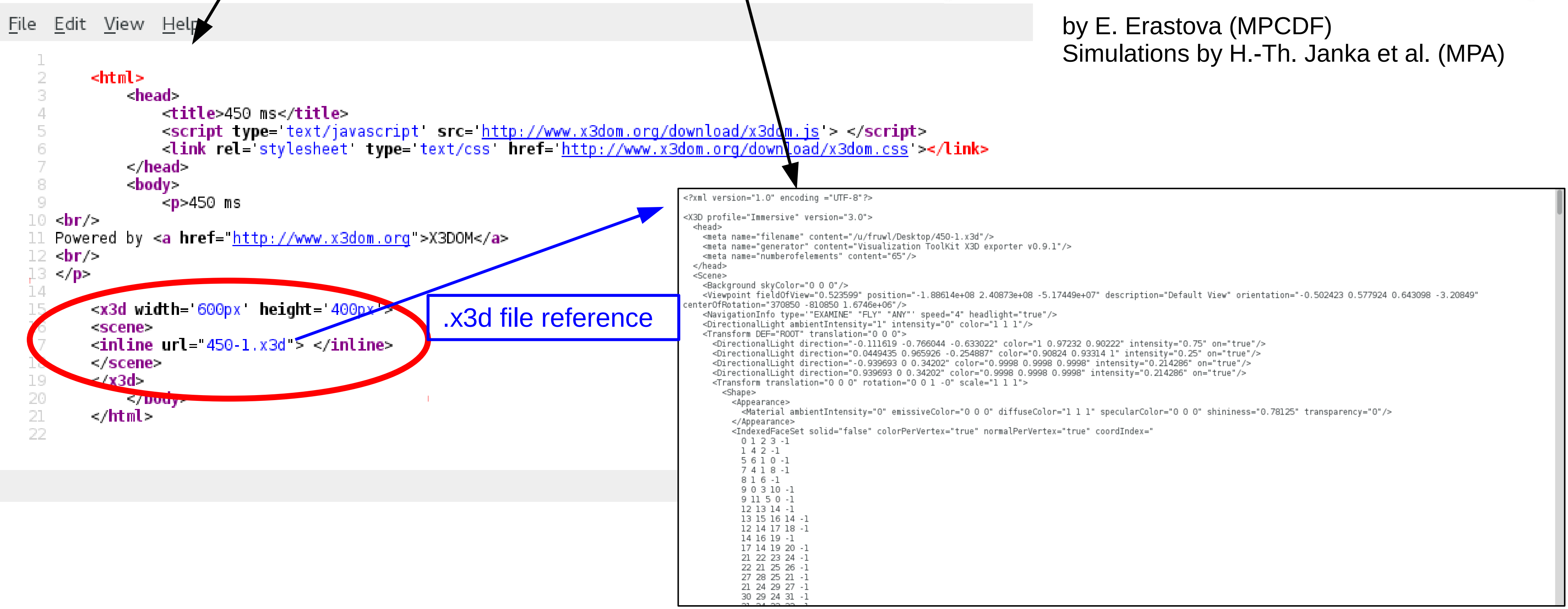
Interactive graphics with X3DOM

- supplements publishing of simulation results, e.g., by APJ (http://iopscience.iop.org/0004-637X/793/2/127/media)
- 3D data format and object model (http://www.x3dom.org/)
- X3D(OM) file export supported by Paraview, VisIT (2.10)
- controls: mouse, zoom, +custom interaction
- HTML5, no browser plugin required



by E. Erastova (MPCDF)
Simulations by H.-Th. Janka et al. (MPA)

.x3d file export



```
    <html>
        <head>
            <title>450 ms</title>
            <script type='text/javascript' src='http://www.x3dom.org/download/x3dom.js'> </script>
            <link rel='stylesheet' type='text/css' href='http://www.x3dom.org/download/x3dom.css'></link>
        </head>
        <body>
            <p>450 ms
<br/>
Powered by <a href="http://www.x3dom.org">X3DOM</a>
<br/>
</p>

    <x3d width='600px' height='400px'>
    <scene>
    <inline url="450-1.x3d"> </inline>
    </scene>
    </x3d>
        </body>
    </html>
```

.x3d file reference

```
<?xml version="1.0" encoding ="UTF-8"?>

<X3D profile="Immersive" version="3.0">
  <head>
    <meta name="filename" content="/u/fruwl/Desktop/450-1.x3d"/>
    <meta name="generator" content="Visualization ToolKit X3D exporter v0.9.1"/>
    <meta name="numberofelements" content="65"/>
  </head>
  <Scene>
    <Background skyColor="0 0 0"/>
    <Viewpoint fieldOfView="0.523599" position="-1.88614e+08 2.40873e+08 -5.17449e+07" description="Default View" orientation="-0.502423 0.577924 0.643098 -3.20849"
centerOfRotation="370850 -810850 1.6746e+06"/>
    <NavigationInfo type='"EXAMINE" "FLY" "ANY"' speed="4" headlight="true"/>
    <DirectionalLight ambientIntensity="1" intensity="0" color="1 1 1"/>
    <Transform DEF="ROOT" translation="0 0 0">
      <DirectionalLight direction="-0.111619 -0.766044 -0.633022" color="1 0.97232 0.90222" intensity="0.75" on="true"/>
      <DirectionalLight direction="0.0449435 0.965926 -0.254887" color="0.90824 0.93314 1" intensity="0.25" on="true"/>
      <DirectionalLight direction="-0.939693 0 0.34202" color="0.9998 0.9998 0.9998" intensity="0.214286" on="true"/>
      <DirectionalLight direction="0.939693 0 0.34202" color="0.9998 0.9998 0.9998" intensity="0.214286" on="true"/>
      <Transform translation="0 0 0" rotation="0 0 1 -0" scale="1 1 1">
        <Shape>
          <Appearance>
            <Material ambientIntensity="0" emissiveColor="0 0 0" diffuseColor="1 1 1" specularColor="0 0 0" shininess="0.78125" transparency="0"/>
          </Appearance>
          <IndexedFaceSet solid="false" colorPerVertex="true" normalPerVertex="true" coordIndex="
            0 1 2 3 -1
            1 4 2 -1
            5 6 1 0 -1
            7 4 1 8 -1
            8 1 6 -1
            9 0 3 10 -1
            9 11 5 0 -1
            12 13 14 -1
            13 15 16 14 -1
            12 14 17 18 -1
            14 16 19 -1
            17 14 19 20 -1
            21 22 23 24 -1
            22 21 25 26 -1
            27 28 25 21 -1
            21 24 29 27 -1
            30 29 24 31 -1
```

Visualization projects

Developed and implemented 2011 (K. Reuter, A. Marek, H. Appel):

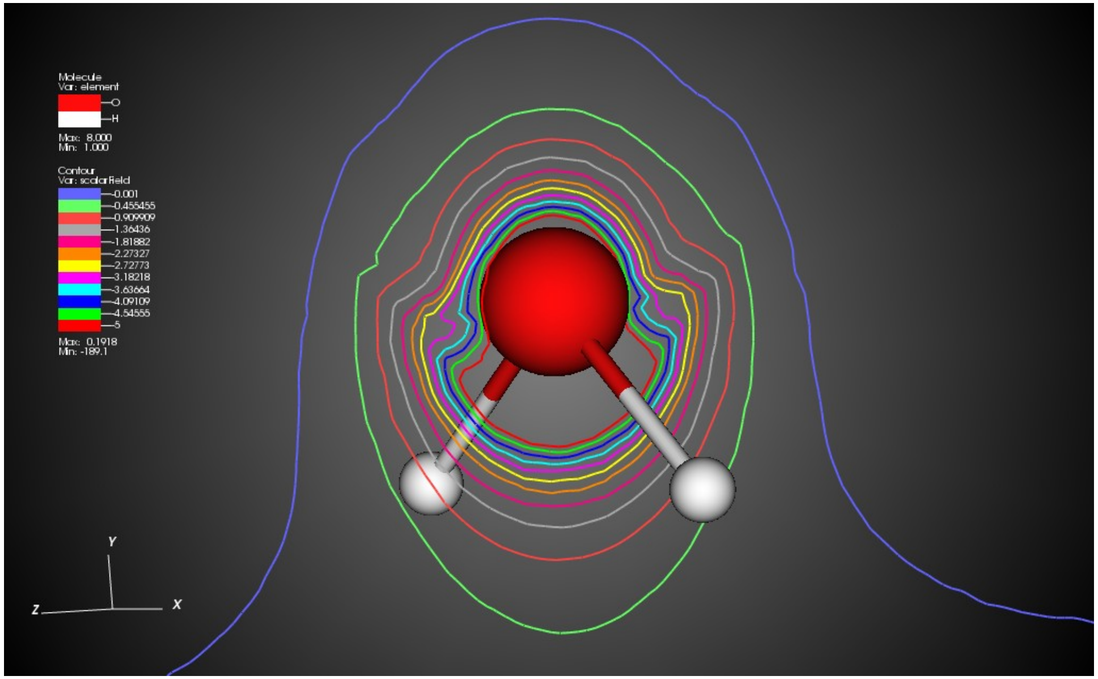
- file format for task-local output of scalar fields in FHI-aims (default off)
- parallel post-processing tool a2rg (mapping of atom-centered grid to rectilinear grid by triangulation)

→ avoids memory and runtime overhead in FHI-aims simulations (cf. CUBE output)
→ enables interactive visualization and high-quality rendering with VisIT, Paraview, ...

Some test results taken from the user guide:

contours of the Hartree potential for H_2O

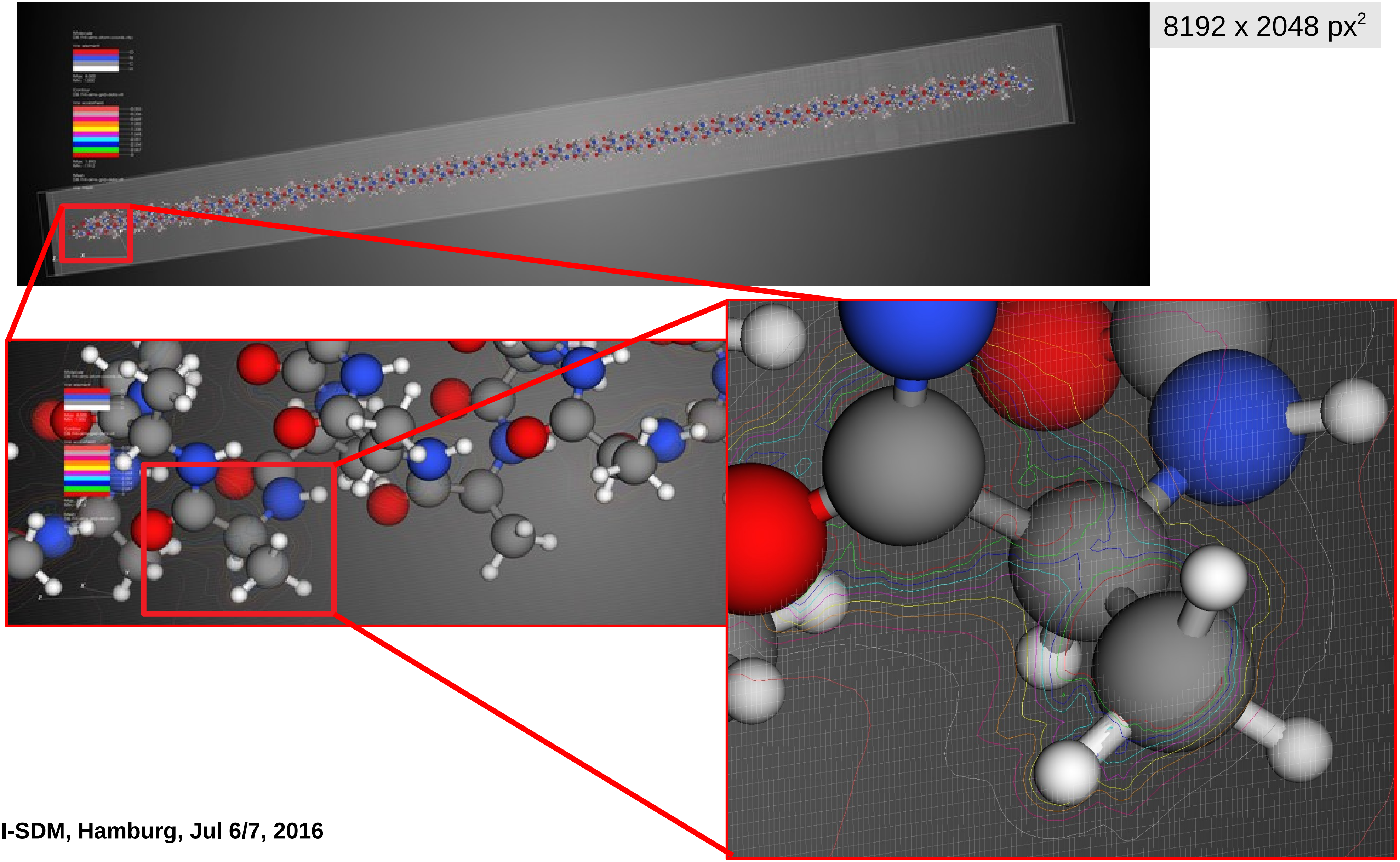
- 2d projection onto 512x512 grid



Visualization projects

contours of the Hartree potential for Alanine

- 2d projection onto 256x8192 grid



8192 x 2048 px^2

Visualization tools

VisIT and Paraview (according to our own experience in the MPG) are:

- *comprehensive* visualization tools
- *well documented*: web pages, manuals, WiKi
- *well supported*: responsive mailing lists, feedback from developers, user community: www.visitusers.org, www.paraview.org
- *extensible*: e.g. data-reader plugins
- *widely used* in the scientific community, installed at many computing centres
- *easy to install* (pre-built executables for Linux x86_32/x86_64, Windows XP/Vista/7, Mac OS X)
- *mature* but also still under active development:
  - VisIt 2.10.2 (Mar 2016) ... VisIt 2.9.1 (May 2015) …VisIt 2.0 (May 2010) … VisIt 1.5 (Dec 2005) ...
  - Paraview 5.1 (Mar 2016) … Paraview 4.3 (Jan 2015) ... Paraview 3.2 (Nov 2007) … Paraview 0.6 (Oct 2002)

VisIT or Paraview is not ...

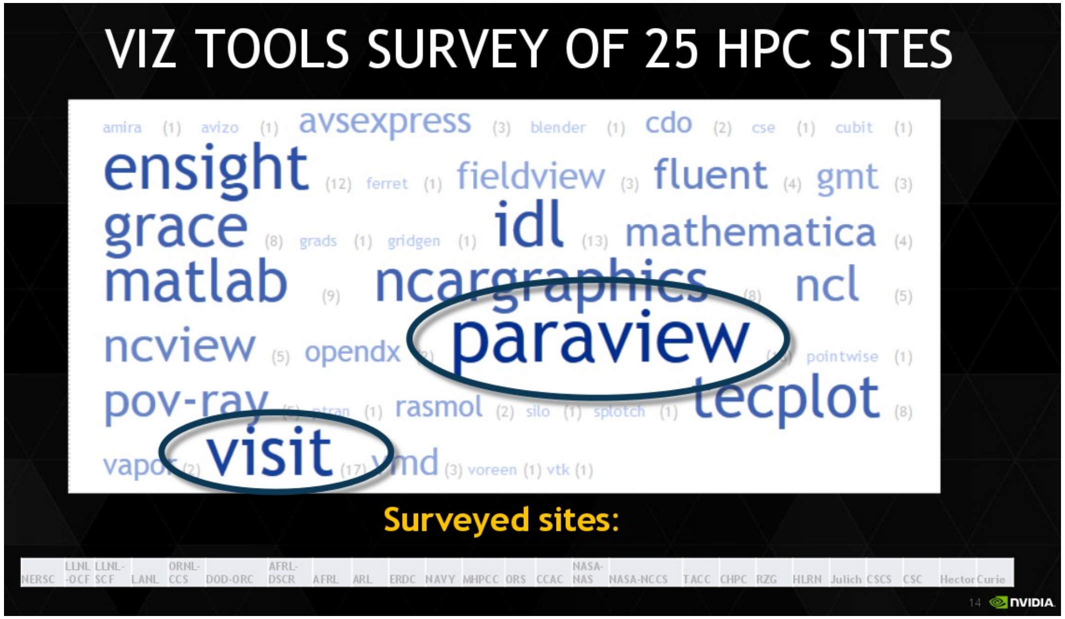
- necessary (sufficient?) for analyzing low-dimensional data (< 3D) → python, idl, matlab, . . .
- a complete replacement for comprehensive quantitative analysis → python, idl, matlab, …
- a world-wide community effort for developing comprehensive repositories → python, GNU-R, ...

Visualization tools

VisIT or Paraview ?

(MPCDF has no self-interests in any of the projects nor any conflict of interest)

- choice is primarily a matter of taste: very similar standard functionality, free software, well supported, ...
- Paraview may look and feel a bit more “modern” (coming from the world of AVIZO/AMIRA)
- Paraview might be somewhat easier to install (part of some Linux distributions)
- VisIt has its roots in astrophysics (and some built-in strengths in this area)
- VisIt provides rich(er) functionality for handling data from molecular simulations (http://www.visitusers.org/index.php?title=Molecular_data_features)

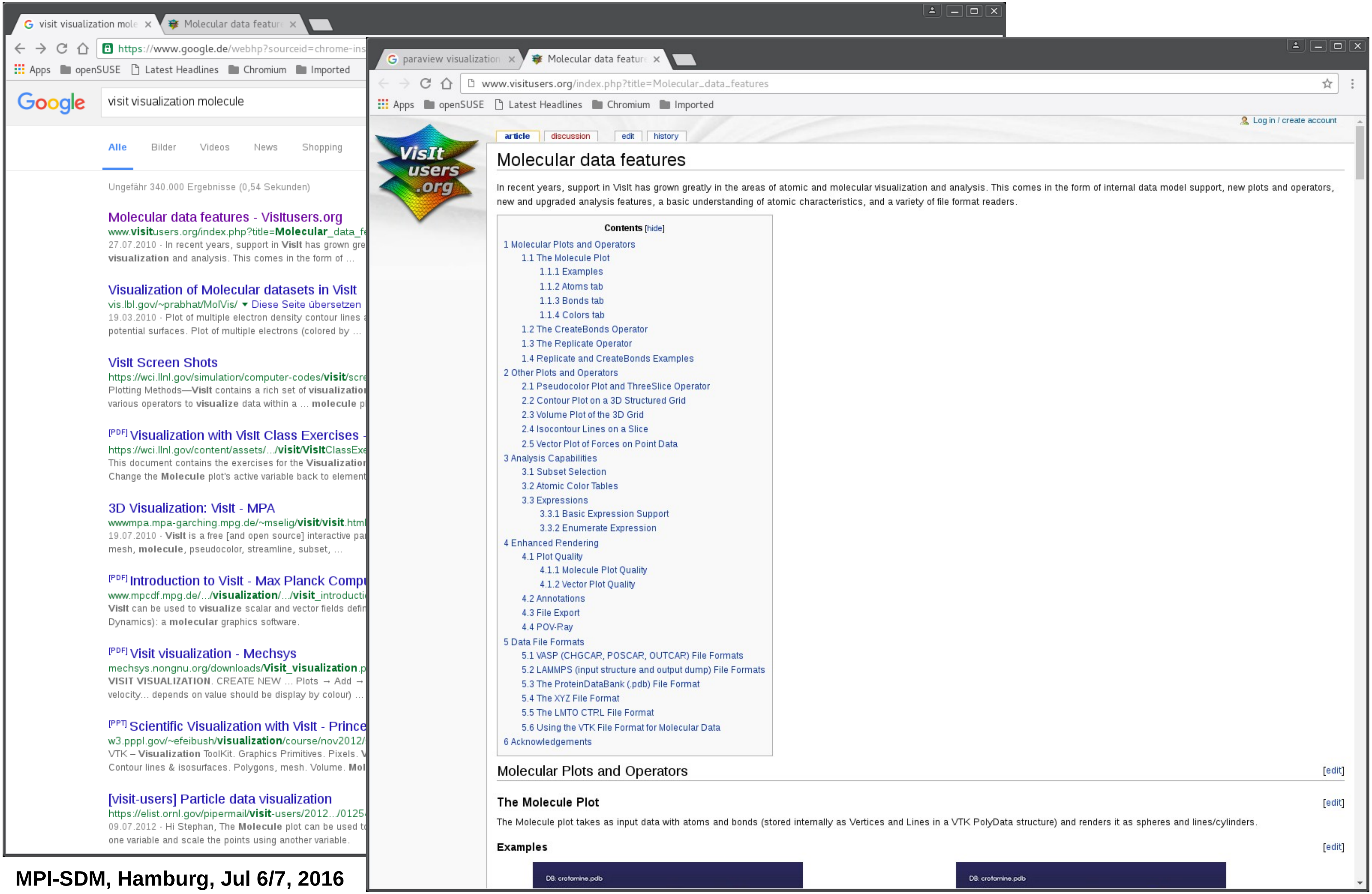

VIZ TOOLS SURVEY OF 25 HPC SITES

Surveyed sites:

NERSC	LLNL -OCF	LLNL- SCF	LANL	ORNL- CCS	DOD-ORC	AFRL- DSCR	AFRL	ARL	ERDC	NAVY	MHPCC	ORS	CCAC	NASA- NAS	NASA-NCCS	TACC	CHPC	RZG	HLRN	Julich	CSCS	CSC	Hector	Curie

non-authoritative survey by P. Messmer (Nvidia)

Molecular data: VisIt



visit visualization mole ×
Molecular data featur ×
https://www.google.de/webhp?sourceid=chrome-ins
Apps openSUSE Latest Headlines Chromium Imported
Google
visit visualization molecule
Alle Bilder Videos News Shopping

Ungefähr 340.000 Ergebnisse (0,54 Sekunden)

Molecular data features - VisItusers.org
www.visitusers.org/index.php?title=Molecular_data_fe
27.07.2010 - In recent years, support in VisIt has grown gre
visualization and analysis. This comes in the form of ...

Visualization of Molecular datasets in VisIt
vis.lbl.gov/~prabhat/MolVis/ ▾ Diese Seite übersetzen
19.03.2010 - Plot of multiple electron density contour lines a
potential surfaces. Plot of multiple electrons (colored by ...

VisIt Screen Shots
https://wci.llnl.gov/simulation/computer-codes/visit/scre
Plotting Methods—VisIt contains a rich set of visualizatio
various operators to visualize data within a ... molecule p

[PDF] Visualization with VisIt Class Exercises -
https://wci.llnl.gov/content/assets/.../visit/VisItClassExe
This document contains the exercises for the Visualizatior
Change the Molecule plot's active variable back to element

3D Visualization: VisIt - MPA
wwwmpa.mpa-garching.mpg.de/~mselig/visit/visit.html
19.07.2010 - VisIt is a free [and open source] interactive par
mesh, molecule, pseudocolor, streamline, subset, ...

[PDF] Introduction to VisIt - Max Planck Compu
www.mpcdf.mpg.de/.../visualization/.../visit_introducti
VisIt can be used to visualize scalar and vector fields defir
Dynamics): a molecular graphics software.

[PDF] Visit visualization - Mechsys
mechsys.nongnu.org/downloads/Visit_visualization.p
VISIT VISUALIZATION. CREATE NEW ... Plots → Add →
velocity... depends on value should be display by colour) ...

[PPT] Scientific Visualization with VisIt - Prince
w3.pppl.gov/~efeibush/visualization/course/nov2012/
VTK – Visualization ToolKit. Graphics Primitives. Pixels. V
Contour lines & isosurfaces. Polygons, mesh. Volume. Mol

[visit-users] Particle data visualization
https://elist.ornl.gov/pipermail/visit-users/2012.../0125
09.07.2012 - Hi Stephan, The Molecule plot can be used to
one variable and scale the points using another variable.

paraview visualization ×
Molecular data featur ×
www.visitusers.org/index.php?title=Molecular_data_features
Apps openSUSE Latest Headlines Chromium Imported
VisIt users .org
Log in / create account
article discussion edit history

Molecular data features

In recent years, support in VisIt has grown greatly in the areas of atomic and molecular visualization and analysis. This comes in the form of internal data model support, new plots and operators, new and upgraded analysis features, a basic understanding of atomic characteristics, and a variety of file format readers.

Contents [hide]

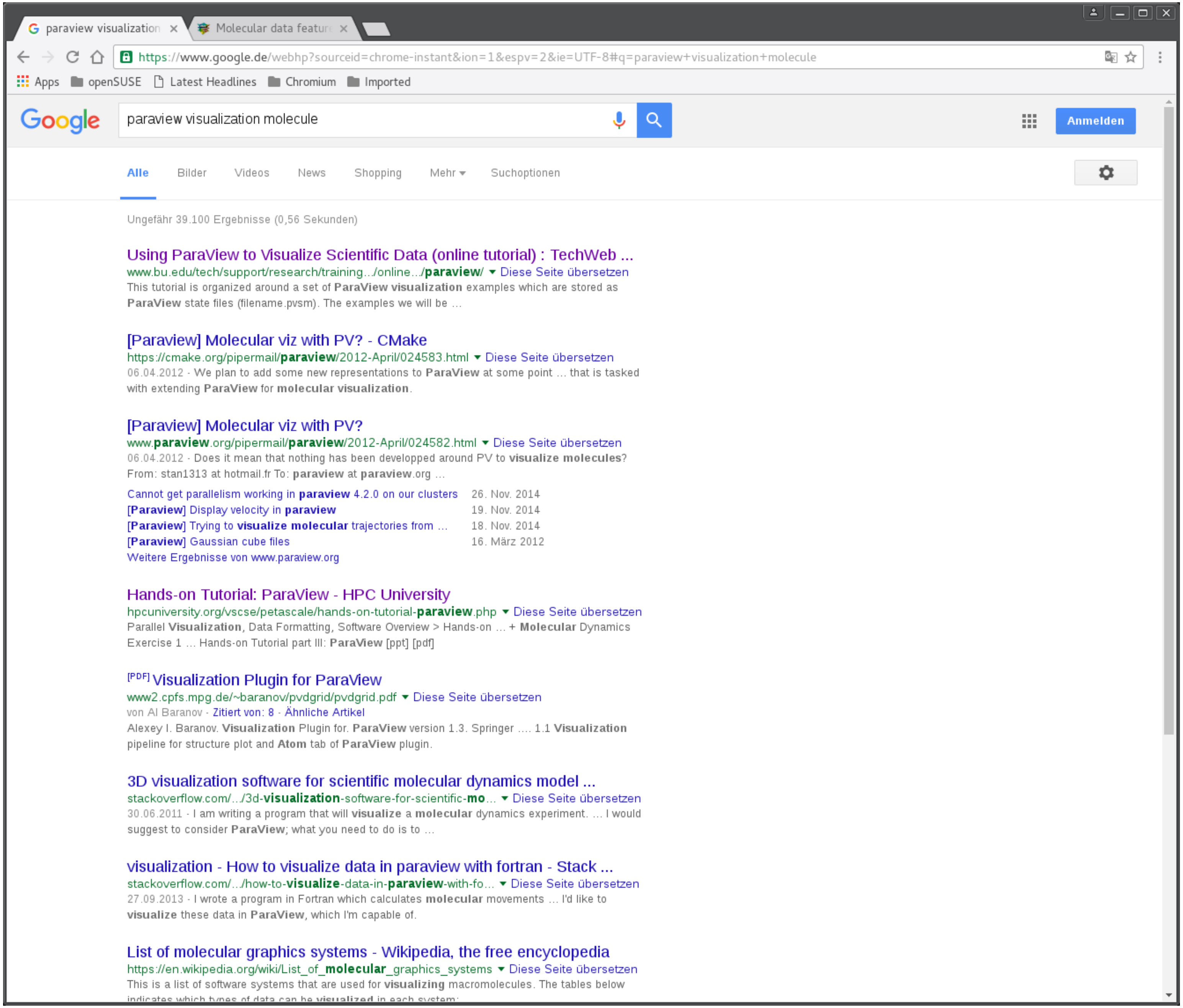
Molecular Plots and Operators [edit]

The Molecule Plot [edit]

The Molecule plot takes as input data with atoms and bonds (stored internally as Vertices and Lines in a VTK PolyData structure) and renders it as spheres and lines/cylinders.

Examples [edit]

Molecular data: Paraview



Ungefähr 39.100 Ergebnisse (0,56 Sekunden)

Using ParaView to Visualize Scientific Data (online tutorial) : TechWeb ...
www.bu.edu/tech/support/research/training.../online.../**paraview**/ ▾ Diese Seite übersetzen
This tutorial is organized around a set of **ParaView visualization** examples which are stored as **ParaView** state files (filename.pvsm). The examples we will be ...

[Paraview] Molecular viz with PV? - CMake
https://cmake.org/pipermail/**paraview**/2012-April/024583.html ▾ Diese Seite übersetzen
06.04.2012 - We plan to add some new representations to **ParaView** at some point ... that is tasked with extending **ParaView** for **molecular visualization**.

[Paraview] Molecular viz with PV?
www.**paraview**.org/pipermail/**paraview**/2012-April/024582.html ▾ Diese Seite übersetzen
06.04.2012 - Does it mean that nothing has been developped around PV to **visualize molecules**? From: stan1313 at hotmail.fr To: **paraview** at **paraview**.org ...

Cannot get parallelism working in **paraview** 4.2.0 on our clusters	26. Nov. 2014
[**Paraview**] Display velocity in **paraview**	19. Nov. 2014
[**Paraview**] Trying to **visualize molecular** trajectories from ...	18. Nov. 2014
[**Paraview**] Gaussian cube files	16. März 2012

Weitere Ergebnisse von www.paraview.org

Hands-on Tutorial: ParaView - HPC University
hpcuniversity.org/vscse/petascale/hands-on-tutorial-**paraview**.php ▾ Diese Seite übersetzen
Parallel **Visualization**, Data Formatting, Software Overview > Hands-on ... + **Molecular** Dynamics Exercise 1 ... Hands-on Tutorial part III: **ParaView** [ppt] [pdf]

[PDF] **Visualization Plugin for ParaView**
www2.cpfs.mpg.de/~baranov/pvdgrid/pvdgrid.pdf ▾ Diese Seite übersetzen
von AI Baranov - Zitiert von: 8 - Ähnliche Artikel
Alexey I. Baranov. **Visualization** Plugin for. **ParaView** version 1.3. Springer .... 1.1 **Visualization** pipeline for structure plot and **Atom** tab of **ParaView** plugin.

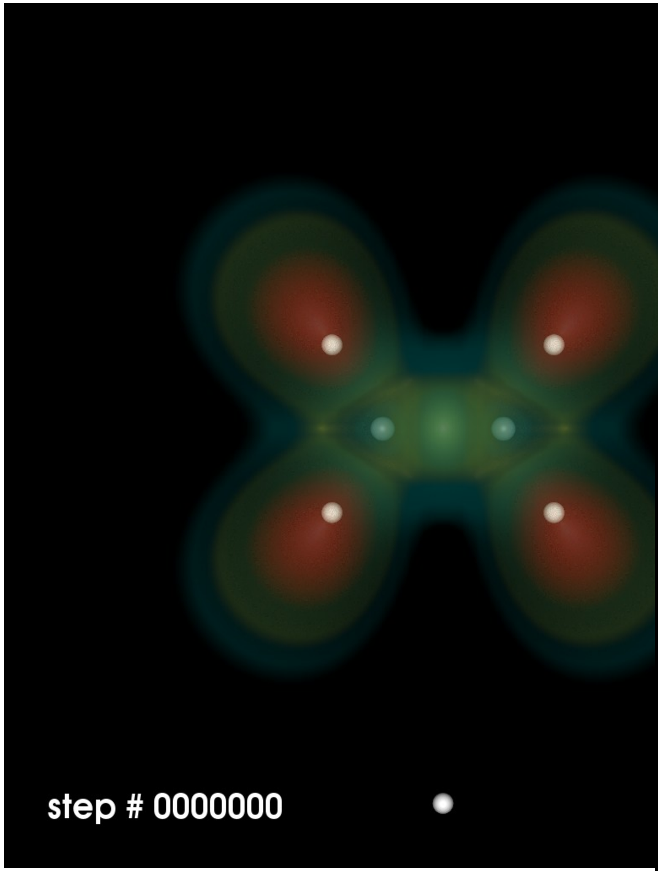
3D visualization software for scientific molecular dynamics model ...
stackoverflow.com/.../3d-**visualization**-software-for-scientific-**mo**... ▾ Diese Seite übersetzen
30.06.2011 - I am writing a program that will **visualize** a **molecular** dynamics experiment. ... I would suggest to consider **ParaView**; what you need to do is to ...

visualization - How to visualize data in paraview with fortran - Stack ...
stackoverflow.com/.../how-to-**visualize**-data-in-**paraview**-with-fo... ▾ Diese Seite übersetzen
27.09.2013 - I wrote a program in Fortran which calculates **molecular** movements ... I'd like to **visualize** these data in **ParaView**, which I'm capable of.

List of molecular graphics systems - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/List_of_**molecular**_graphics_systems ▾ Diese Seite übersetzen
This is a list of software systems that are used for **visualizing** macromolecules. The tables below indicates which types of data can be **visualized** in each system:

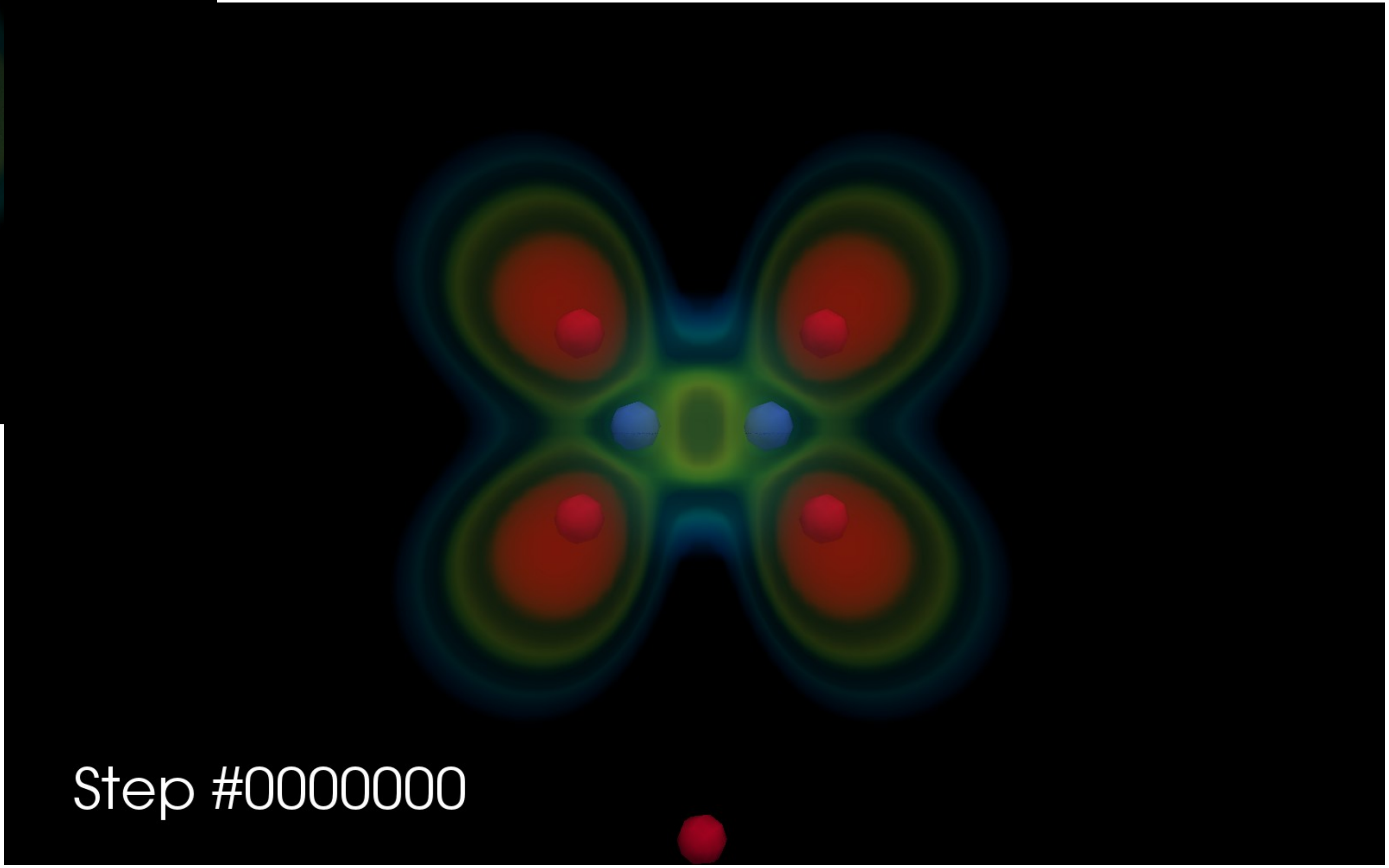
Molecular data

VisIt (see hands-on by M. Compostella)

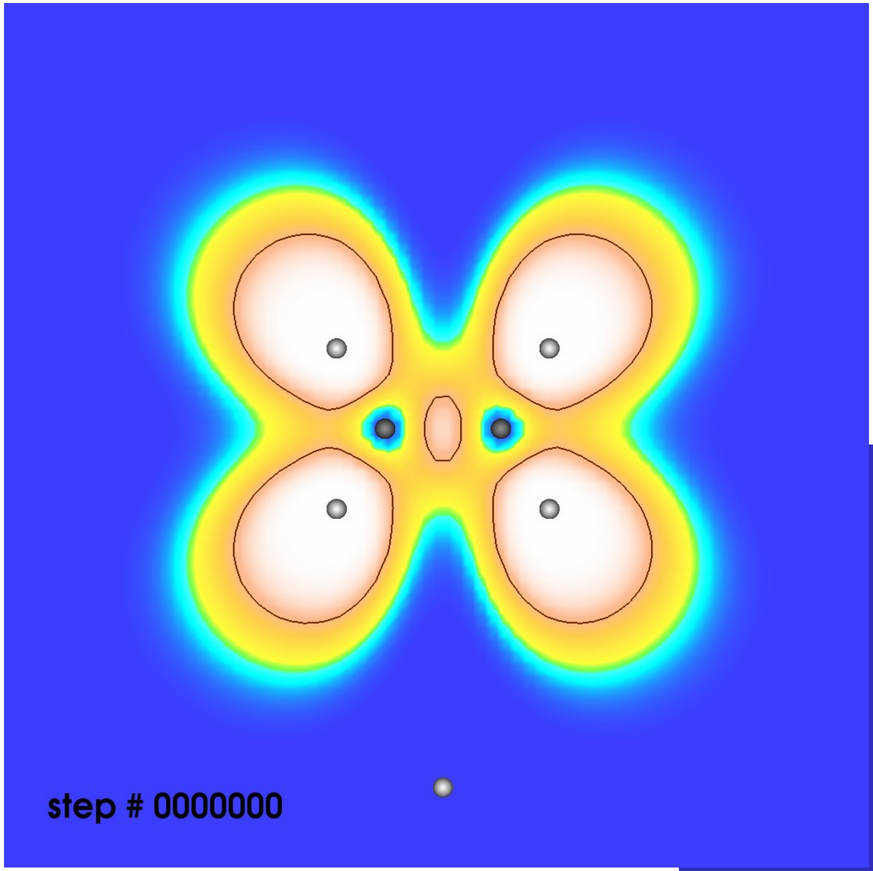


Paraview (scripts by K. Duglué)

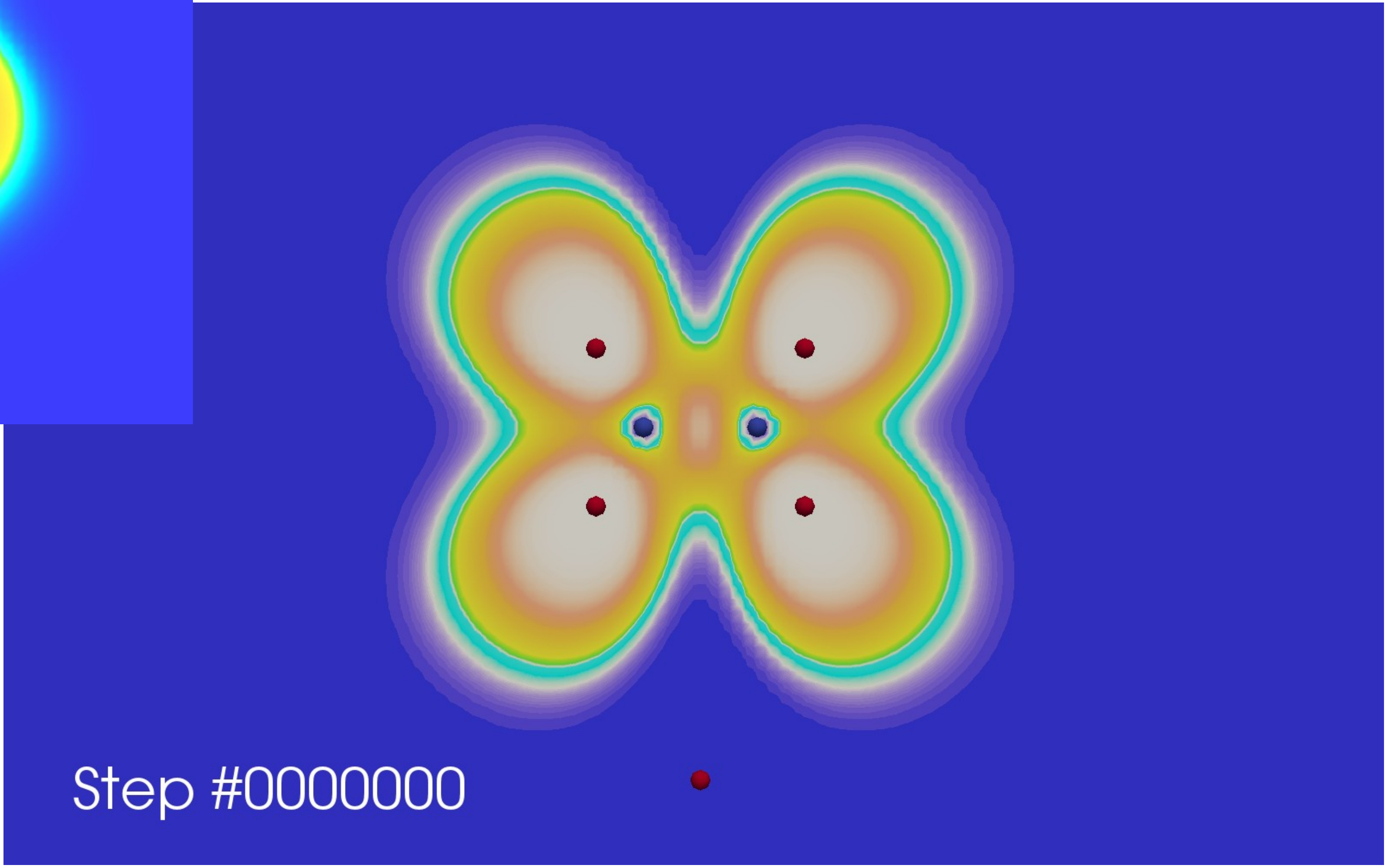
- standard ray-casting
- molecules can be rendered as simple glyphs (sphere)
- only .cube format (VTK + xyz does not work)
- no interactive time series



Molecular data



VisIt



Paraview

VisIT Basics

VisIT is … not easy to google:

- homepage: http://visit.llnl.gov/
- do not confound with VISIT - a Visualisation Toolkit (FZ Jülich)

VisIT, (according to the VisIt homepage) is ...

- a free [and open-source], interactive parallel visualization and graphical analysis tool for viewing scientific data on Unix and PC platforms [Windows, Mac OS].
- users can quickly generate visualizations from their data, animate them through time, manipulate them, and save the resulting images for presentations
- VisIt can be used to visualize scalar and vector fields defined on two- and three-dimensional (2D and 3D) structured and unstructured meshes.
- VisIt was designed to handle very large data set sizes in the terascale range and yet can also handle small data sets in the kilobyte range
- originated from Lawrence Livermore National Laboratory (ASC/DOE)
- distributed project, developed by several groups: VACET (SciDAC), ASC, GNEP

VisIT Basics

Why VisIT ? – experiences at the MPCDF with MPG projects

- Computing Centre’s point of view:
  - 2007: survey of freely available tools: VisIt, (ParaView)
  - VisIt produced good results in short time
  - flexible client-server architecture: allows running GUI, and rendering, data access on different machines
  - continuing positive feedback from a (experienced and critical) scientific users from different domains
  - a main workhorse employed by the MPCDF visualisation team
- Scientific user’s point of view:
  - steep (i.e. efficient) learning curve (knowledge transfer: visualisation team → users)
  - promotes gradual transition from GUI-based, interactive work to Python scripting
  - comprehensive set of standard functionalities
  - data handling: many supported formats
  - allows to produce publication-quality plots along the way

Remarks

Experiences with VisIt & lessons learned

- VisIt is a tool definitely worth considering for multi-D data analysis & visualization
- highly competitive with Paraview, VAPOR, and also commercial tools
- VisIt has its roots clearly in visualization of massive (astrophysics) data ...

  … but has matured to a comprehensive, general-purpose analysis & visualisation tool
- VisIt employs intuitive concepts for user interaction
- *personal experience:* quite often, the right tool (operator, plot, ...) is *just there*
- *observation:* many plots in scientific publications (astro and related) apparently produced with by VisIt
- parallel capabilities allow to utilize CPU, RAM of a cluster (multiple nodes) for a single visualization session
- movies: Python scripting superior (in flexibility and work efficiency) to keyframing
- parallel version not trivial to install and operate *(system administrator's perspective)*
- parallel efficiency not easy to achieve *(user's perspective)*
- long-running renderings: more control required (start/stop, pause)
- progress bar sometimes useless (shows percentage of individual rendering phases)
- many other deficiencies already resolved (VisIT 1.x → VisIT 2.x)

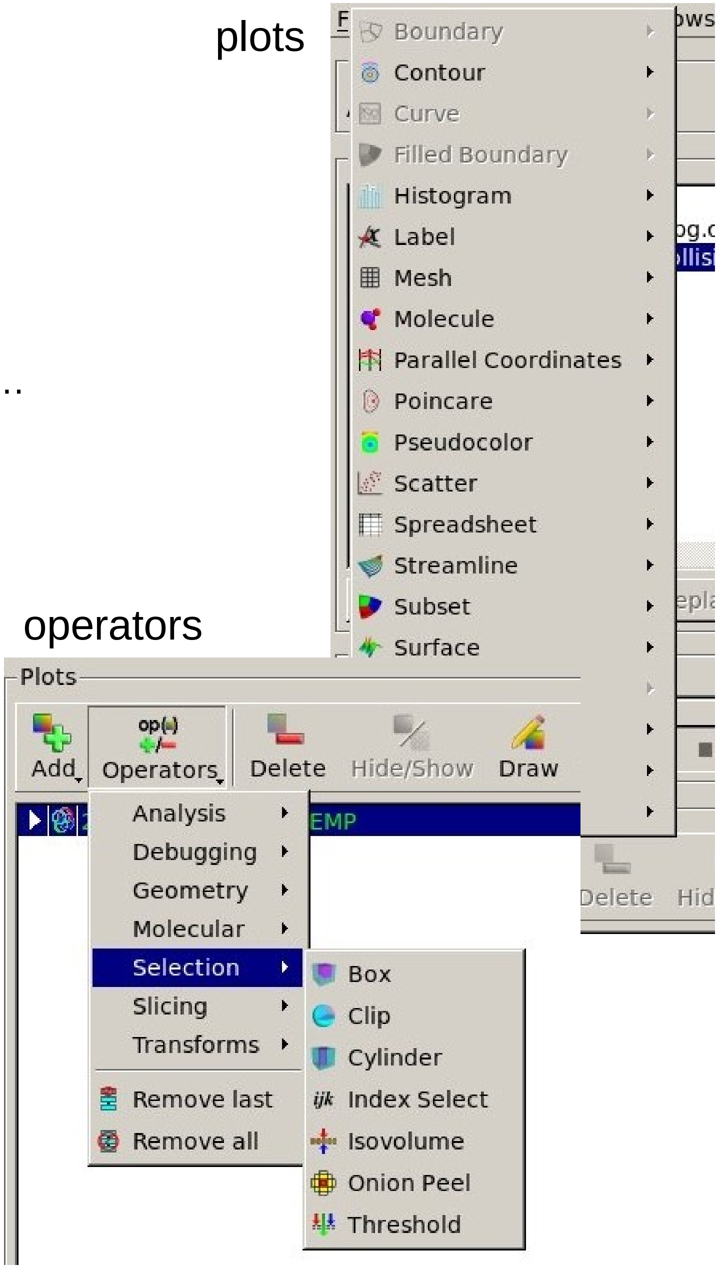
VisIT Overview

Basic concepts

- *plots (ways to render data)*: > 20
  - pseudocolor, volume-rendering, contour, vector, scatter, ...
- *operators (ways to manipulate data, ”filters”)*: > 40
  - isosurface, isovolume, clip, slice, project, (coordinate-)transforms, …
- *file format readers (ways to import data)*: > 80
- *queries (ways to extract quantitative information)*: > 50
  - HDF5/XDMF, NetCDF, (common CFD fmts), VTK, BOV, PDB, ...
- *expressions (ways to create derived quantities)*
  - simple expression syntax: math, logical, relational, mesh, …
  - Python syntax
- *other features/utilities* (not fully covered here)
  - movie generation: simple, keyframing
  - session management: save/restore status of interactive session
  - it-situ visualisation: instrumentation of simulation codes

plots

operators



VisIT Overview

Intuitive concept for visualization “pipeline”

- Plots: pseudocolor, volume, vector, …
- and Operators: isosurface, transform (e.g. coordinate), clip, box, revolve, …
- options for plots and operators (double-click or pull down menu)
- multiple (successive) operators per plot
- hint: for performance reduce as early as possible, e.g.:1: box, 2: transform (not vice versa)
- multiple plots per window

Example

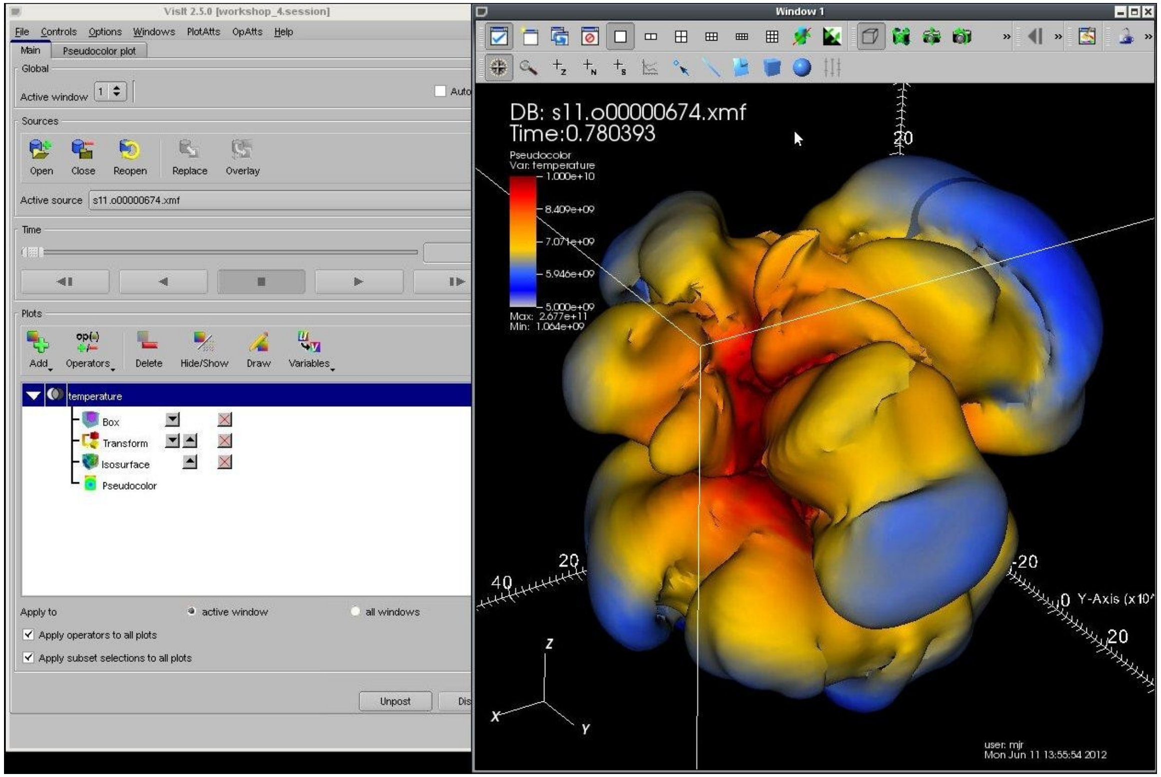
isosurface of a scalar variable F(x,y,z) colored

by the value of another scalar variable G(x,y,z)

(F, G are defined on the same mesh):

1) create pseudocolor plot for G
2) apply isosurface operator for F

   (instead of default: G)



VisIt 2.5.0 [workshop_4.session]

Active window 1

Active source s11.o00000674.xmf

DB: s11.o00000674.xmf
Time:0.780393

Pseudocolor
Var: temperature

Max: 2.677e+11
Min: 1.064e+09

user: mjr
Mon Jun 11 13:55:54 2012

VisIT GUI

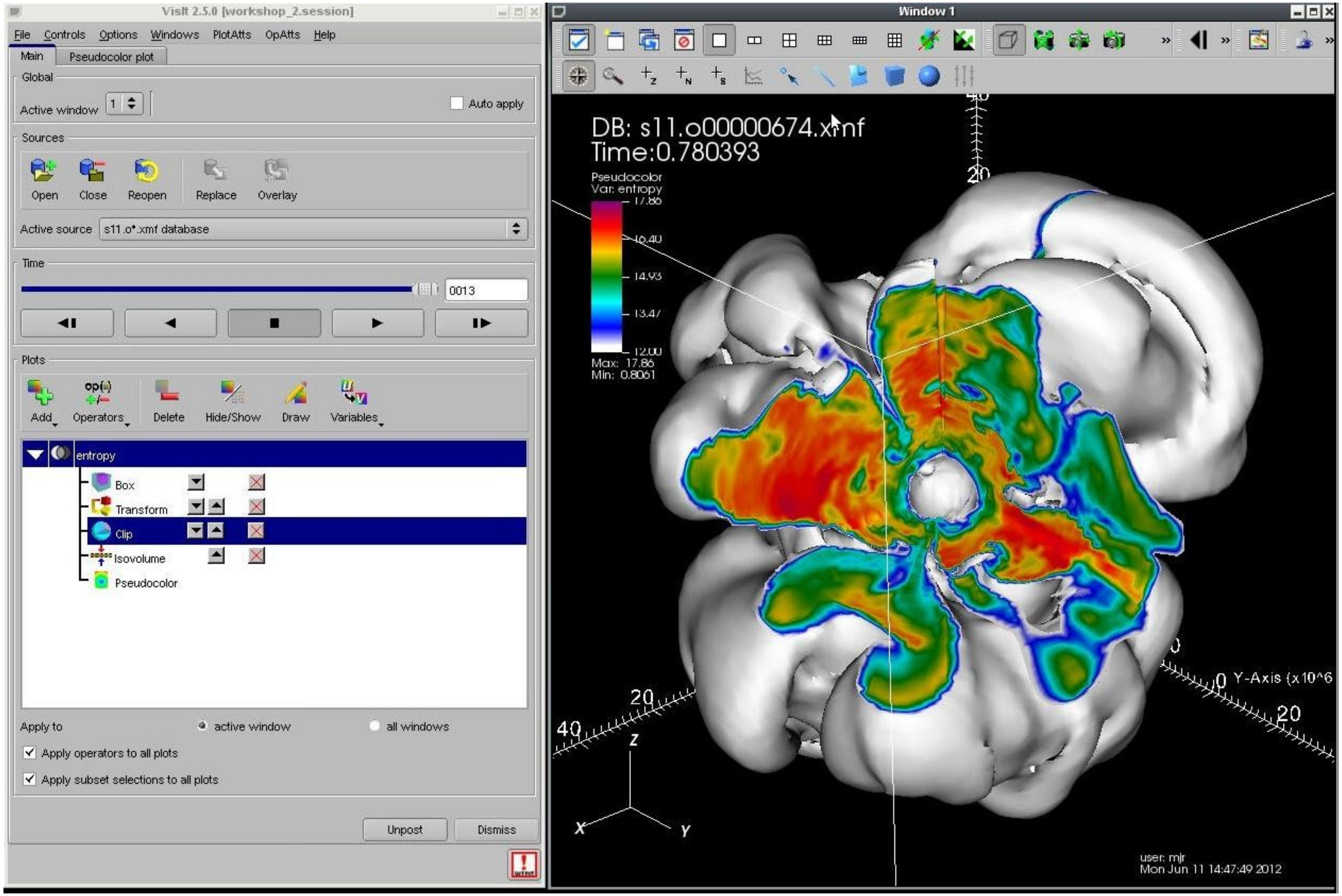
Main menu

file controls

animation controls

plot controls

pipeline



Active window 1

☐ Auto apply

Active source s11.o*.xmf database

Apply to ● active window ○ all windows

☑ Apply operators to all plots

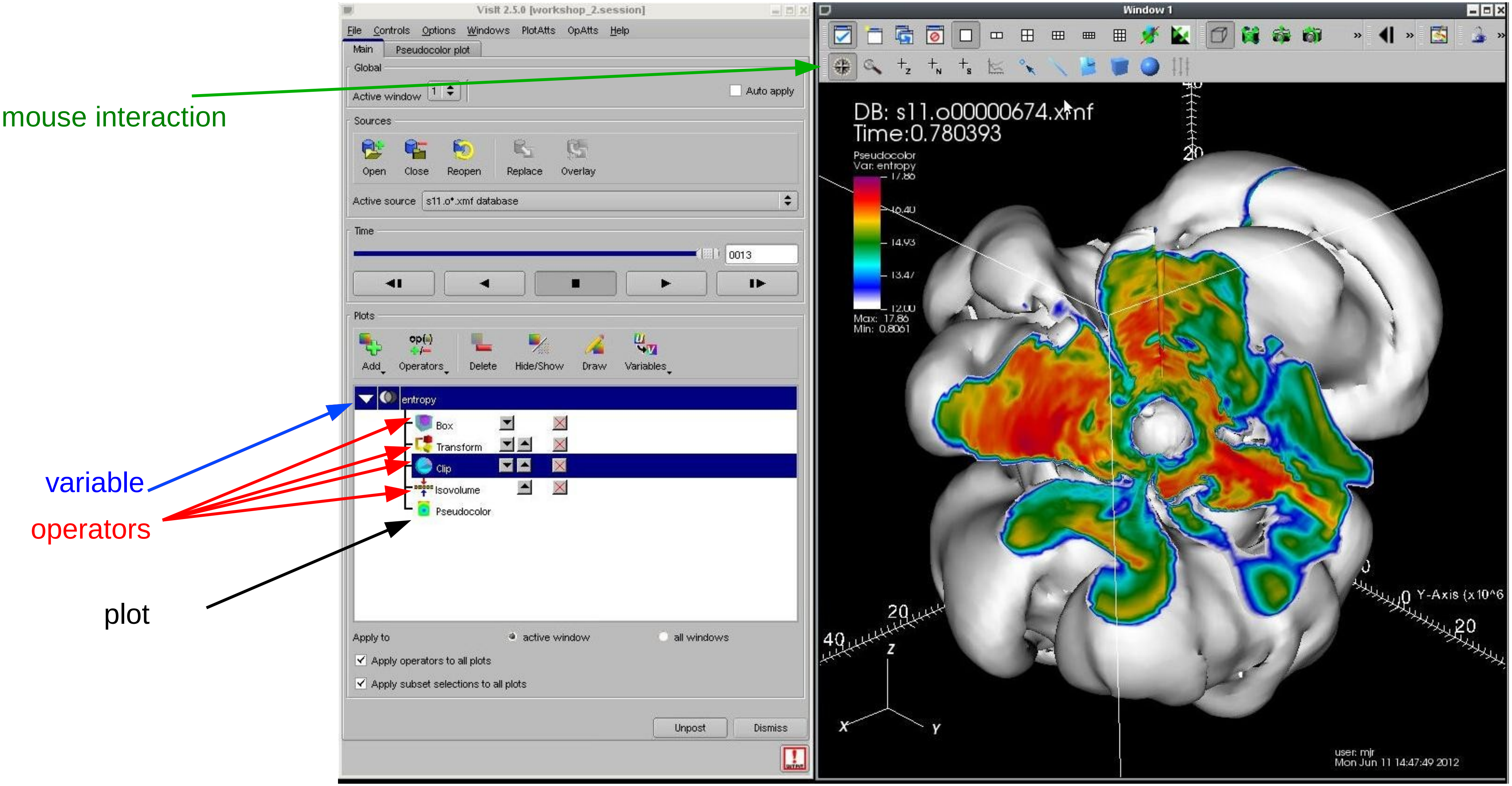
☑ Apply subset selections to all plots

DB: s11.o00000674.xmf
Time:0.780393

single main window

multiple visualization windows

VisIT GUI



Active window 1

□ Auto apply

Active source s11.o*.xmf database

0013

Apply to ● active window ○ all windows

☑ Apply operators to all plots

☑ Apply subset selections to all plots

mouse interaction

variable

operators

plot

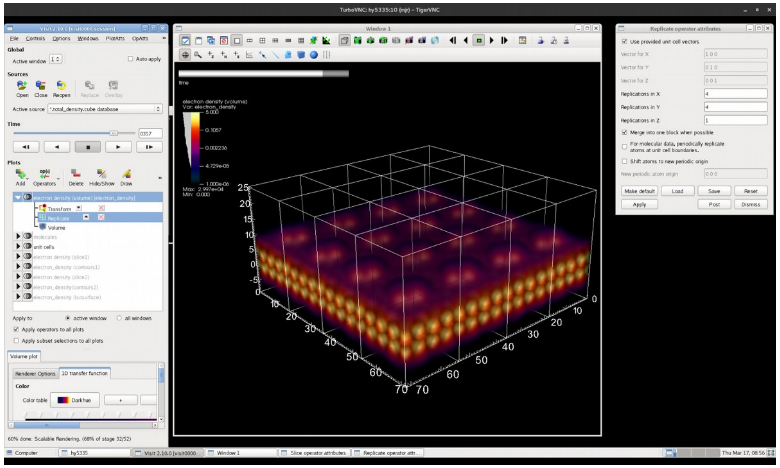
VisIT demo

Example dataset (NOMAD CoE)

context: adsorption of a CO_2 molecule on CaO(001) surface

data:

- 400 time steps x 3 MB (Gaussian cube) from ab-initio MD simulation with FHI-aims
- 50x50x100 grid, 35 atoms, 2x2x1 replication of unit cell → 100^3 grid, 140 atoms


TurboVNC: hy5335:10 (mjr) - TigerVNC
Window 1
time
electron density (volume)
Var: electron_density
Max: 2.997e+04
Min: 0.000

Replicate operator attributes

☑ Use provided unit cell vectors

Vector for X: 1 0 0

Vector for Y: 0 1 0

Vector for Z: 0 0 1

Replications in X: 4

Replications in Y: 4

Replications in Z: 1

☑ Merge into one block when possible

☐ For molecular data, periodically replicate atoms at unit cell boundaries.

☐ Shift atoms to new periodic origin

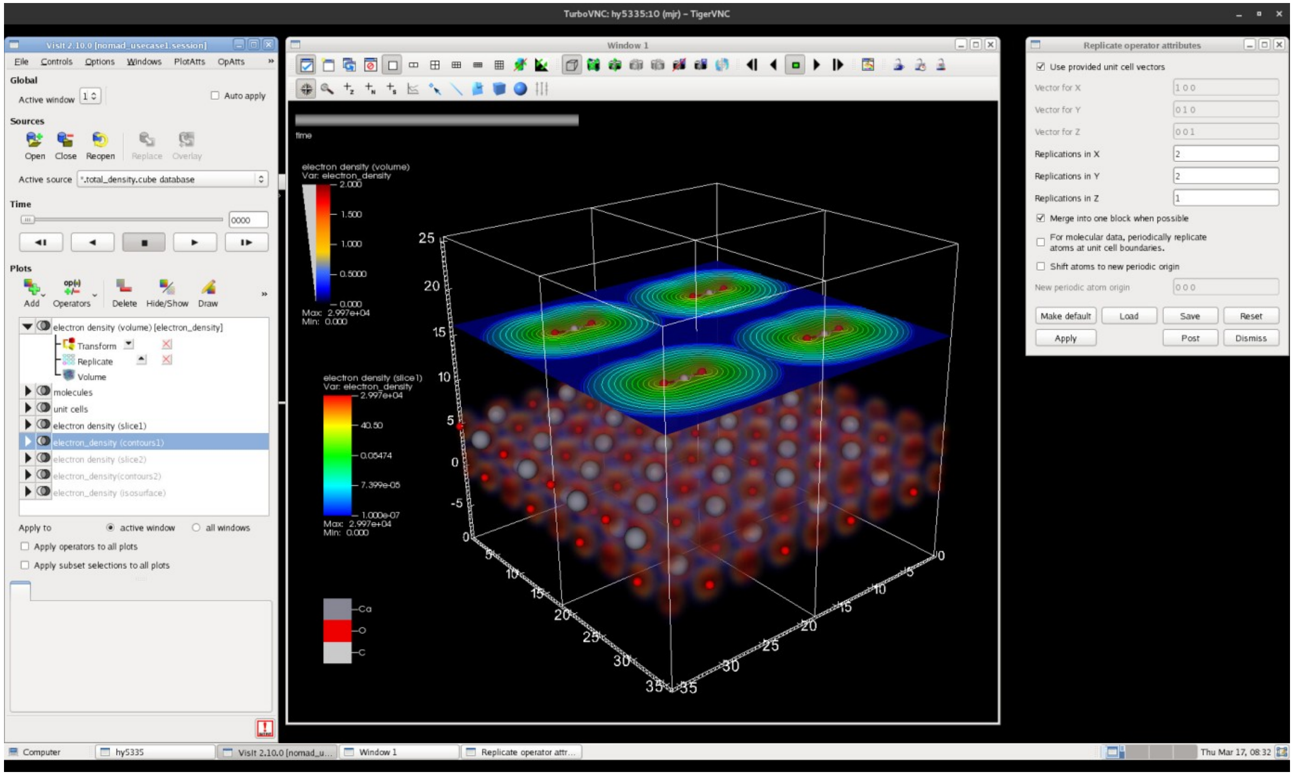
New periodic atom origin: 0 0 0

Make default | Load | Save | Reset | Apply | Post | Dismiss

documentation: https://gitlab.rzg.mpg.de/nomad-lab/public-wiki/wikis/remoteviz/Home

- data and pre-configured sessions

  (VisIt, Paraview)
- publicly accessible on Hydra



Active window: 1

☐ Auto apply

Active source: *.total_density.cube database

Time: 0000

electron density (volume) [electron_density]

Apply to: ● active window ○ all windows

☐ Apply operators to all plots

☐ Apply subset selections to all plots

Replicate operator attributes

☑ Use provided unit cell vectors

Vector for X: 1 0 0

Vector for Y: 0 1 0

Vector for Z: 0 0 1

Replications in X: 2

Replications in Y: 2

Replications in Z: 1

☑ Merge into one block when possible

☐ For molecular data, periodically replicate atoms at unit cell boundaries.

☐ Shift atoms to new periodic origin

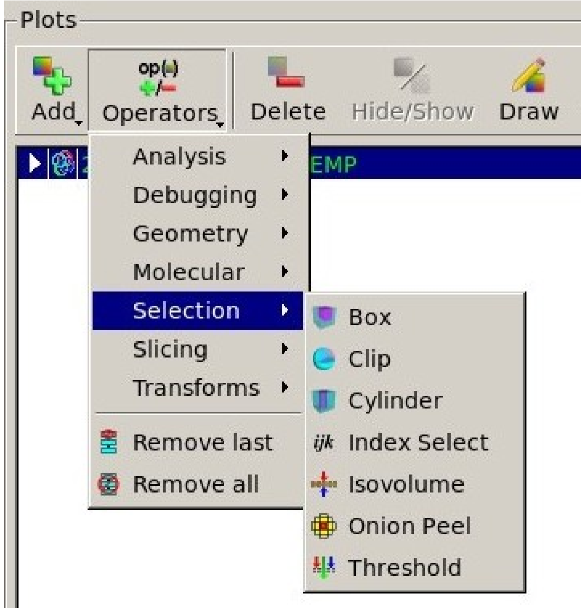
New periodic atom origin: 0 0 0

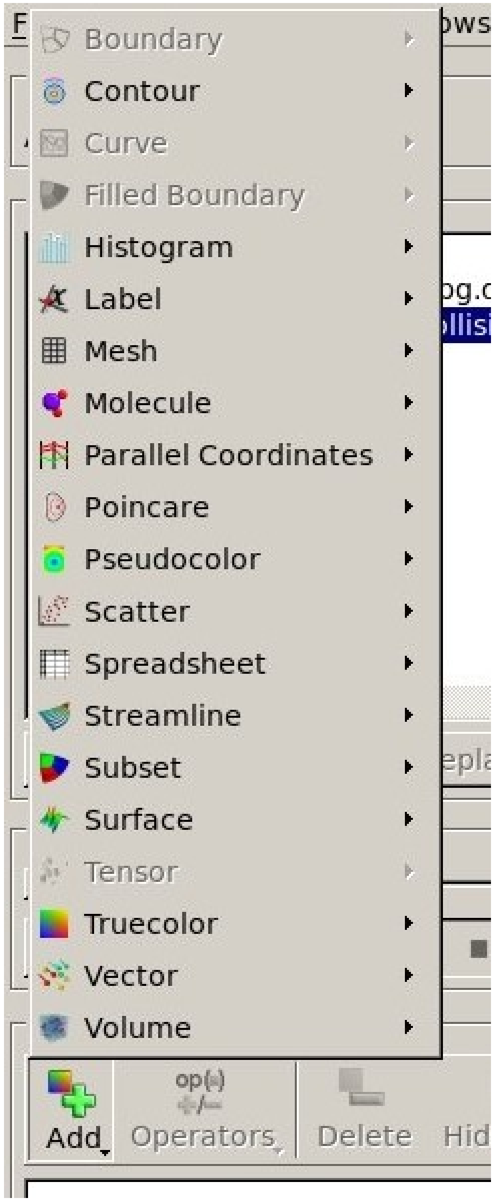
Make default | Load | Save | Reset | Apply | Post | Dismiss

VisIT demo

Basic functionalities of the GUI: a brief demo covering:

- *plots:*
  - pseudocolor
  - volume: splatting, ray-casting
- *operators:*
  - transform: replicate (unit cell)
  - selection: isovolume,slice,clip,box, ...
- *interaction tools*
  - navigate (default)
  - zoom
  - plane
  - line
  - pick (zone node)





Advanced VisIT: Colour tables

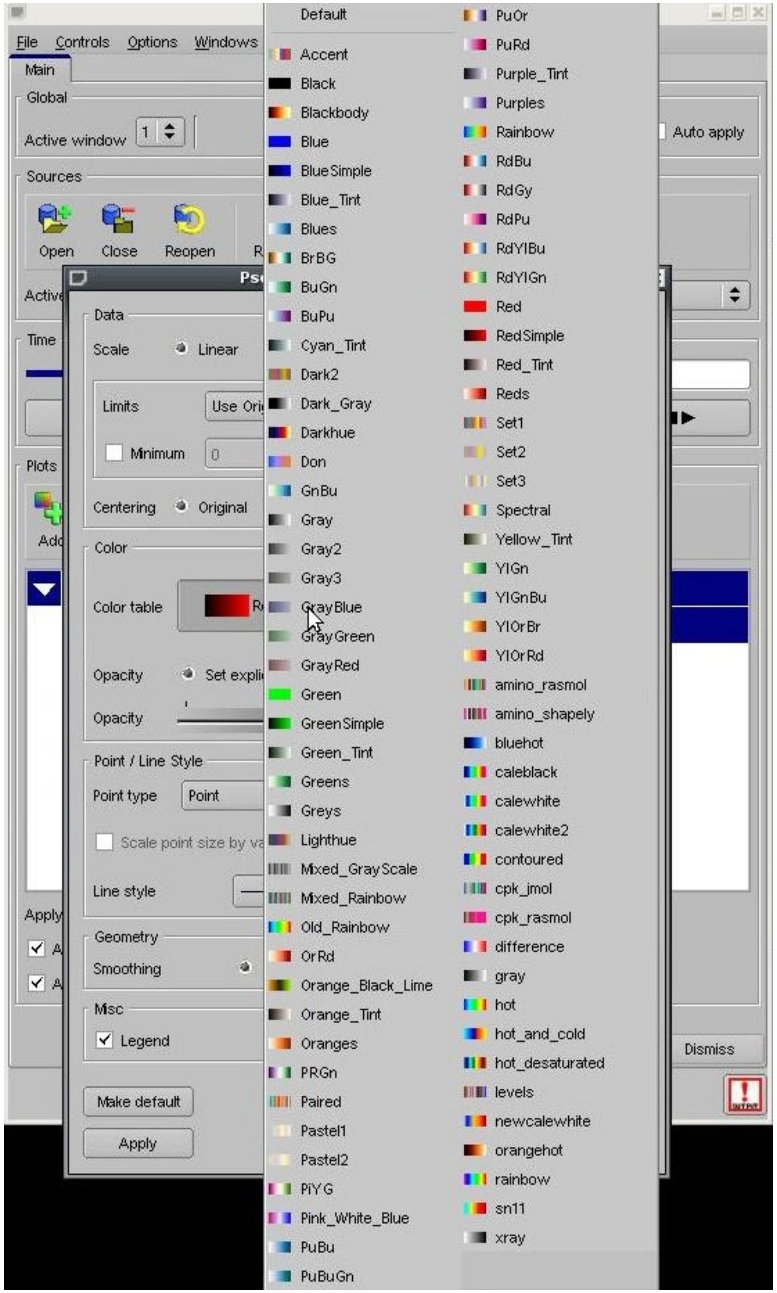
Notes of colour tables

- VisIt 2.5 added many new colour tables

- preview icons for choosing colour table in plot options

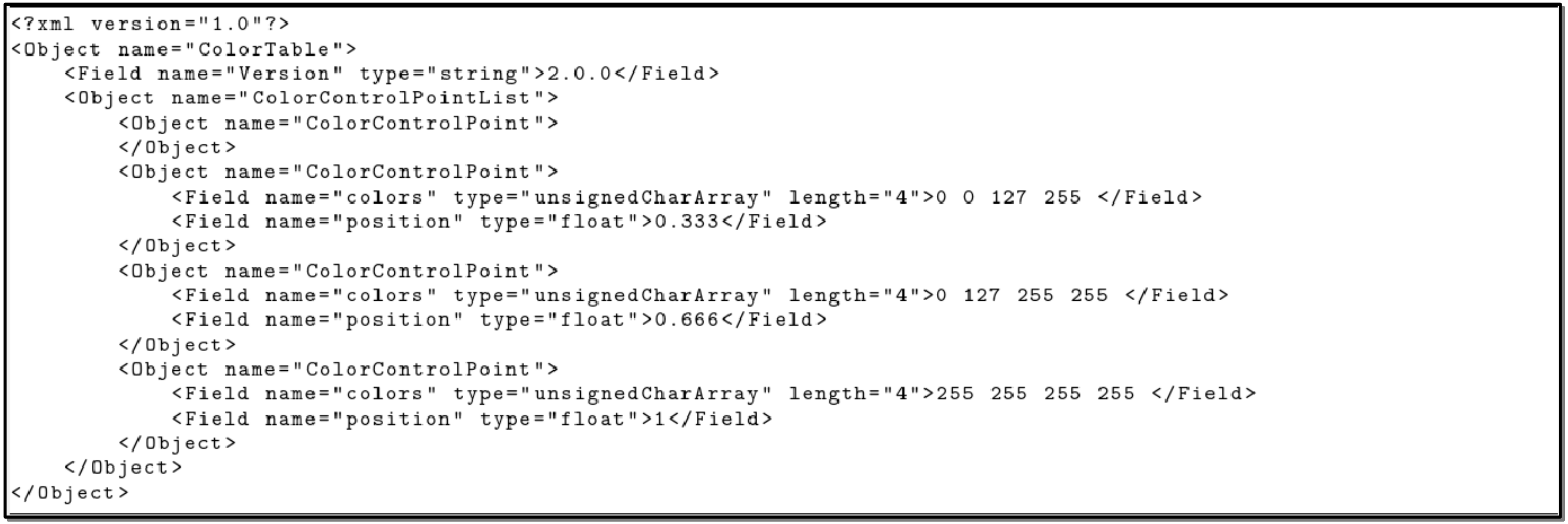
- VisIt allows to create/manipulate colour tables:
  - *interactively*: by manipulating existing colour tables
  - *programmatically*: via Python interface
  - *externally*: via editing colour table files

- additional colour tables can be stored in userspace and shared with others



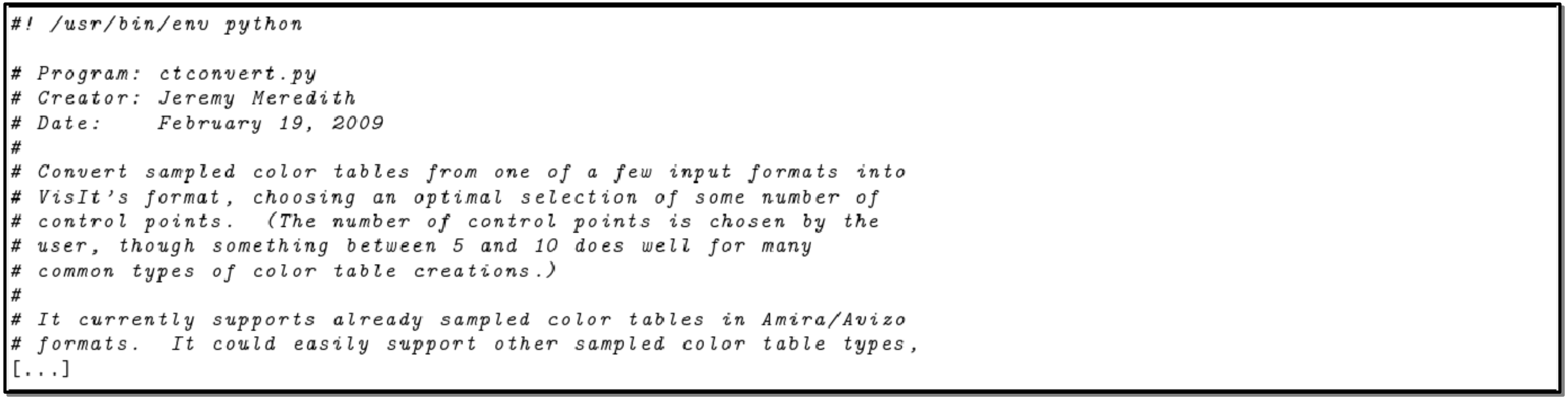
Advanced VisIT: Colour tables

→ colour tables in XML format defined by control points Є [0,1] and (RGB,alpha) Є[256^3,256]



```
<?xml version="1.0"?>
<Object name="ColorTable">
    <Field name="Version" type="string">2.0.0</Field>
    <Object name="ColorControlPointList">
        <Object name="ColorControlPoint">
        </Object>
        <Object name="ColorControlPoint">
            <Field name="colors" type="unsignedCharArray" length="4">0 0 127 255 </Field>
            <Field name="position" type="float">0.333</Field>
        </Object>
        <Object name="ColorControlPoint">
            <Field name="colors" type="unsignedCharArray" length="4">0 127 255 255 </Field>
            <Field name="position" type="float">0.666</Field>
        </Object>
        <Object name="ColorControlPoint">
            <Field name="colors" type="unsignedCharArray" length="4">255 255 255 255 </Field>
            <Field name="position" type="float">1</Field>
        </Object>
    </Object>
</Object>
```

→ simple XML format facilitates conversion or creation (e.g. Python script for converting Amira/Avizo tables)

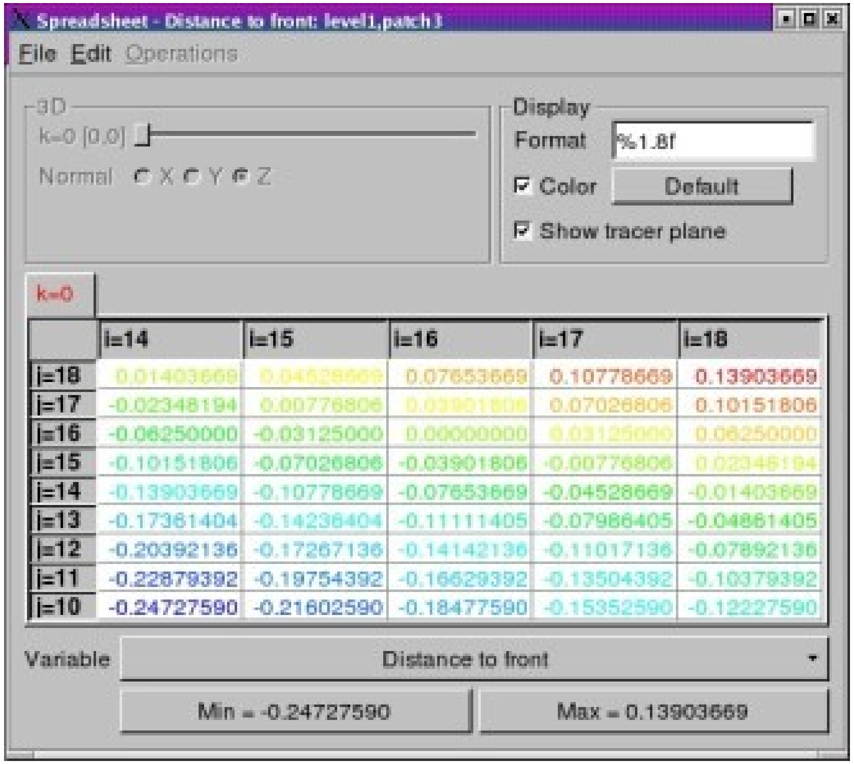


```
#! /usr/bin/env python

# Program: ctconvert.py
# Creator: Jeremy Meredith
# Date:    February 19, 2009
#
# Convert sampled color tables from one of a few input formats into
# VisIt's format, choosing an optimal selection of some number of
# control points.  (The number of control points is chosen by the
# user, though something between 5 and 10 does well for many
# common types of color table creations.)
#
# It currently supports already sampled color tables in Amira/Avizo
# formats.  It could easily support other sampled color table types,
[...]
```

Advanced VisIt: GUI utilities

Advanced features:

- expressions: expression language
- animation: simple movie generation
- sessions: save/restore session state
- movies: keyframing
- quantitative analysis:
  - spreadsheet
  - zone pick
  - queries
  - correlations of multivariate data



Spreadsheet - Distance to front: level1,patch1

Format: %1.8f

☑ Color

☑ Show tracer plane

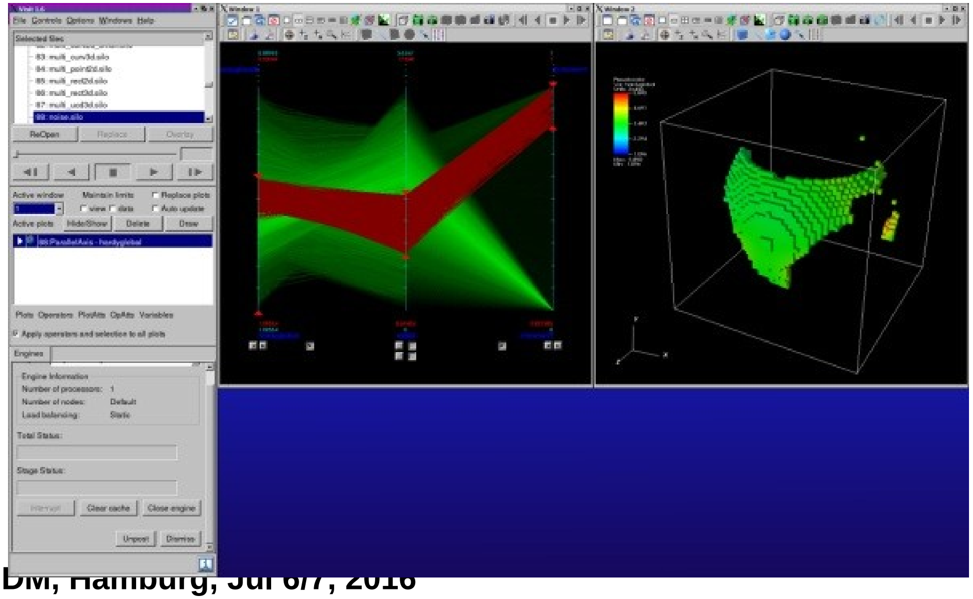
k=0

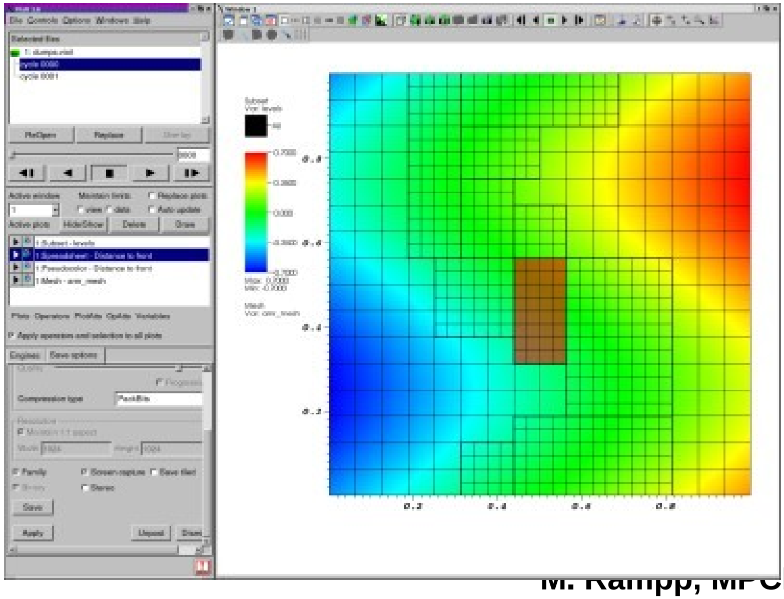
	i=14	i=15	i=16	i=17	i=18
j=18	0.01403669	0.04528669	0.07653669	0.10778669	0.13903669
j=17	-0.02346194	0.00776806	0.03901806	0.07026806	0.10151806
j=16	-0.06250000	-0.03125000	0.00000000	0.03125000	0.06250000
j=15	-0.10151806	-0.07026806	-0.03901806	-0.00776806	0.02346194
j=14	-0.13903669	-0.10778669	-0.07653669	-0.04528669	-0.01403669
j=13	-0.17361404	-0.14236404	-0.11111405	-0.07986405	-0.04861405
j=12	-0.20392136	-0.17267136	-0.14142136	-0.11017136	-0.07892136
j=11	-0.22879392	-0.19754392	-0.16629392	-0.13504392	-0.10379392
j=10	-0.24727590	-0.21602590	-0.18477590	-0.15352590	-0.12227590

Variable: Distance to front

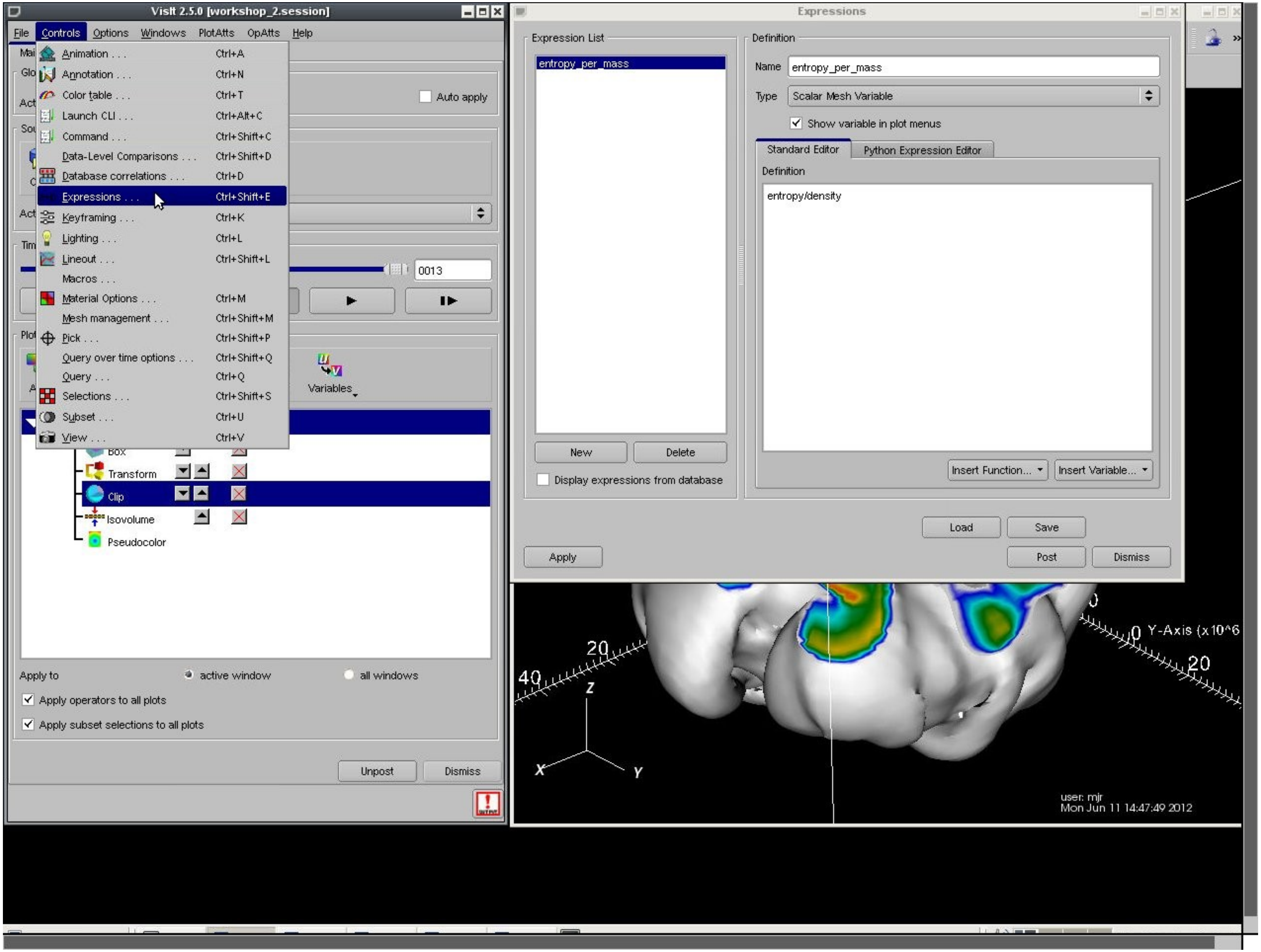
Min = -0.24727590

Max = 0.13903669





Advanced VisIT: expressions



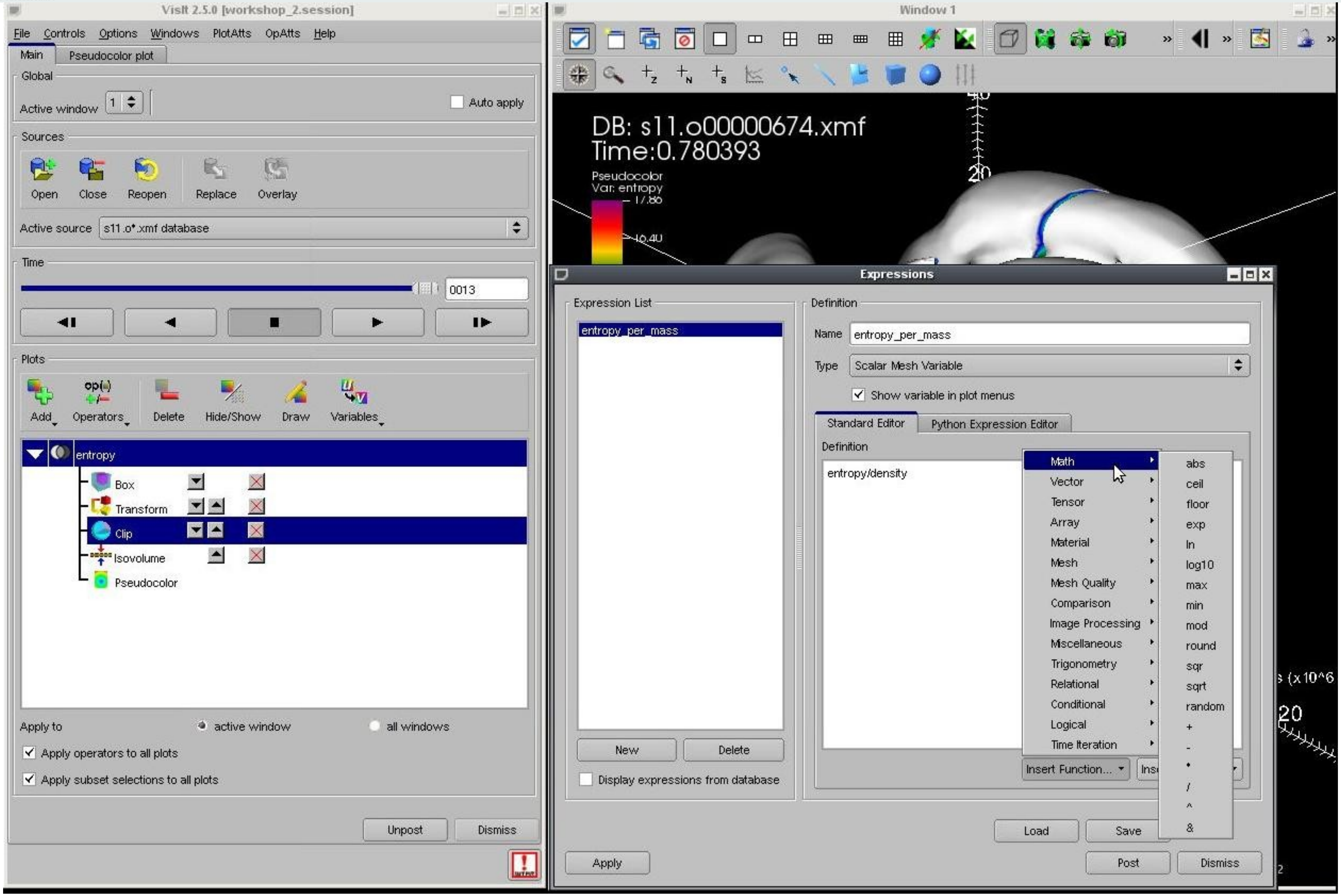
Name: entropy_per_mass

Type: Scalar Mesh Variable

☑ Show variable in plot menus

Definition: entropy/density

Advanced VisIT: expressions



VisIt 2.5.0 [workshop_2.session]

Active window 1

□ Auto apply

Active source s11.o*.xmf database

Time 0013

Plots:

- entropy
  - Box
  - Transform
  - Clip
  - Isovolume
  - Pseudocolor

Apply to ● active window ○ all windows

☑ Apply operators to all plots

☑ Apply subset selections to all plots

Window 1

DB: s11.o00000674.xmf
Time:0.780393

Pseudocolor
Var: entropy

Expressions

Expression List: entropy_per_mass

Name: entropy_per_mass

Type: Scalar Mesh Variable

☑ Show variable in plot menus

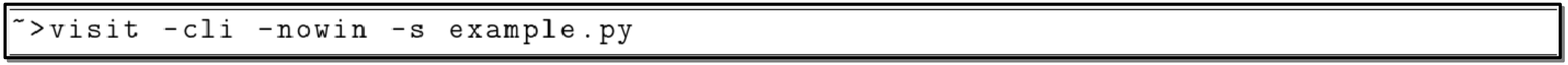
Definition: entropy/density

□ Display expressions from database

- allows simulation codes to dump only non-redundant quantities
- basic expression syntax: math, logical, relational, mesh, ...
- new: Python expression editor

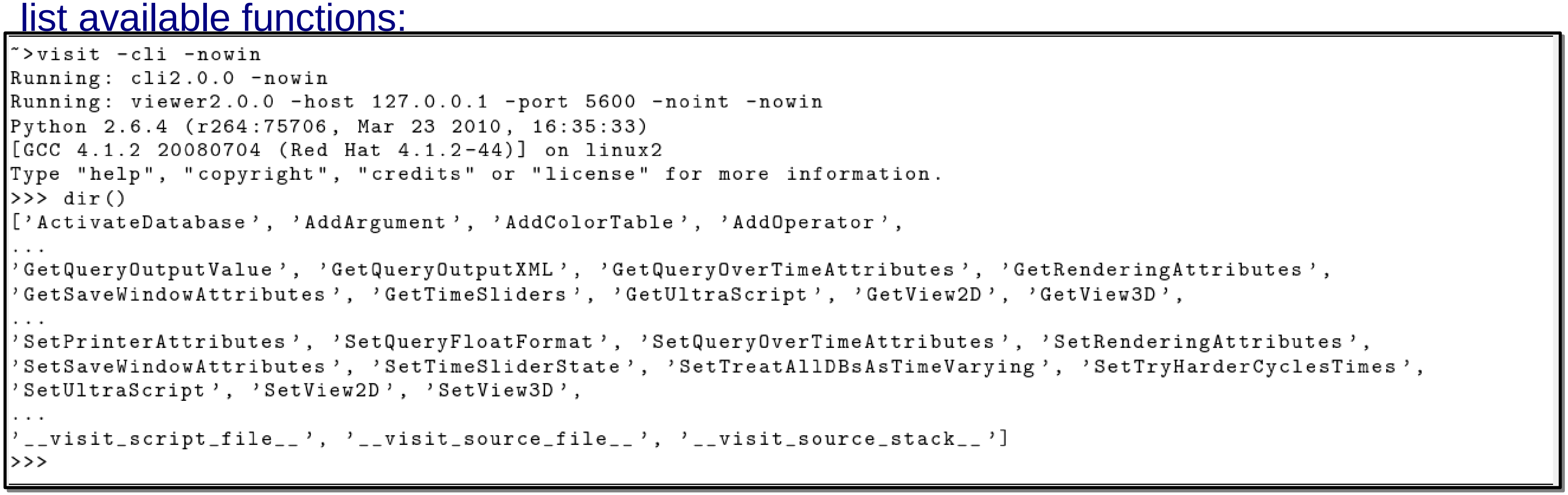
Advanced VisIT: Python scripting

startup:



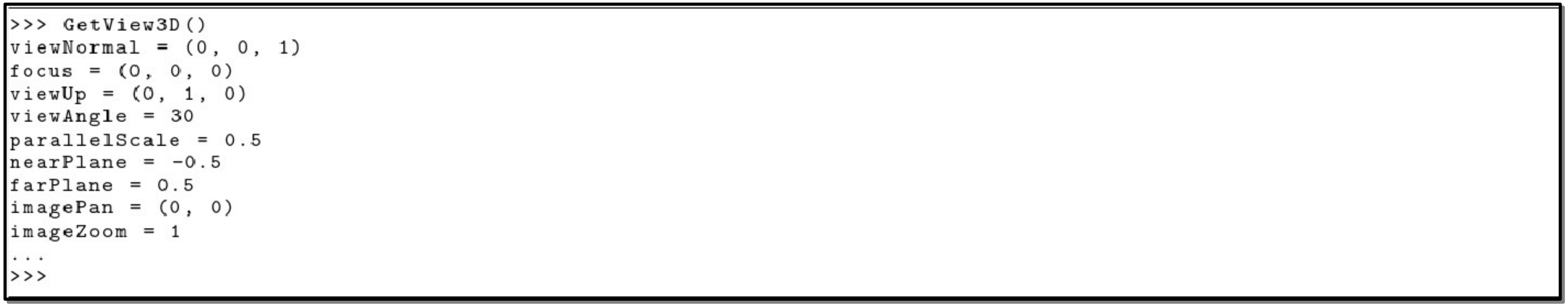
```
~>visit -cli -nowin -s example.py
```

list available functions:



```
~>visit -cli -nowin
Running: cli2.0.0 -nowin
Running: viewer2.0.0 -host 127.0.0.1 -port 5600 -noint -nowin
Python 2.6.4 (r264:75706, Mar 23 2010, 16:35:33)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-44)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> dir()
[’ActivateDatabase’, ’AddArgument’, ’AddColorTable’, ’AddOperator’,
...
’GetQueryOutputValue’, ’GetQueryOutputXML’, ’GetQueryOverTimeAttributes’, ’GetRenderingAttributes’,
’GetSaveWindowAttributes’, ’GetTimeSliders’, ’GetUltraScript’, ’GetView2D’, ’GetView3D’,
...
’SetPrinterAttributes’, ’SetQueryFloatFormat’, ’SetQueryOverTimeAttributes’, ’SetRenderingAttributes’,
’SetSaveWindowAttributes’, ’SetTimeSliderState’, ’SetTreatAllDBsAsTimeVarying’, ’SetTryHarderCyclesTimes’,
’SetUltraScript’, ’SetView2D’, ’SetView3D’,
...
’__visit_script_file__’, ’__visit_source_file__’, ’__visit_source_stack__’]
>>>
```

query/modify visit objects with accessor methods :GetXXX(), SetXXX()

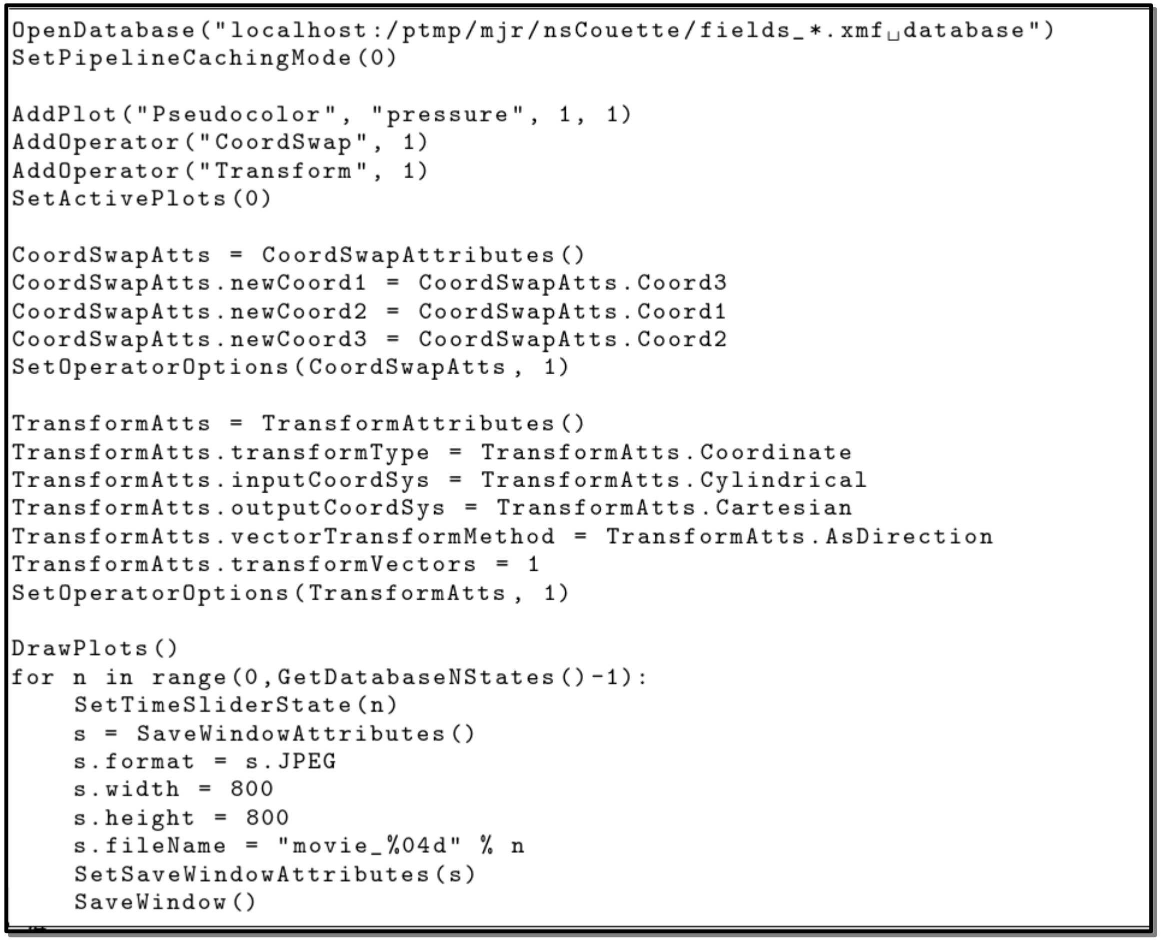


```
>>> GetView3D()
viewNormal = (0, 0, 1)
focus = (0, 0, 0)
viewUp = (0, 1, 0)
viewAngle = 30
parallelScale = 0.5
nearPlane = -0.5
farPlane = 0.5
imagePan = (0, 0)
imageZoom = 1
...
>>>
```

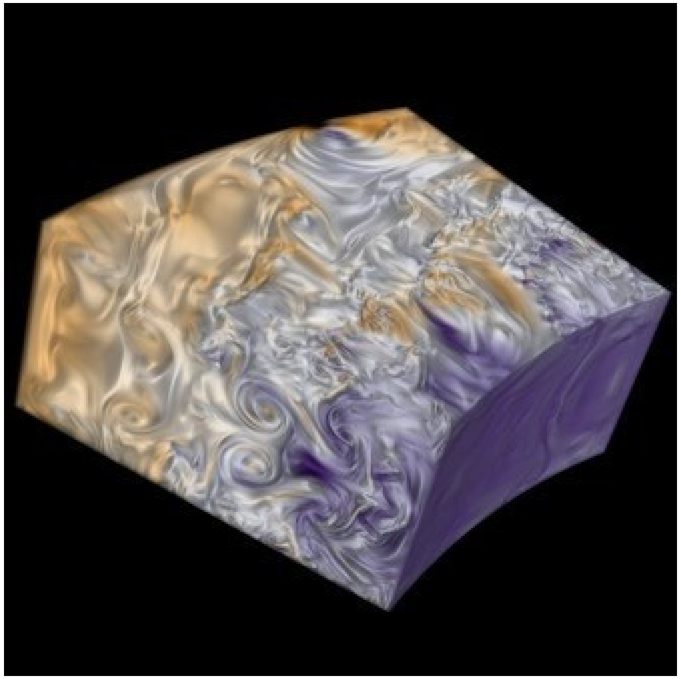
Advanced VisIT: Python scripting

Prototypical example: “movie generation” (with some non-standard transforms)

- expressions: vorticity($u_{r,}u_\theta,u_z$)= $\partial u_r/\partial z - \partial u_z/\partial r$
- operator ”swap coordinates”: (θ,z,r) → (r,θ,z)
- operator ”transform coordinates”: (r,θ,z)→(x,y,z)



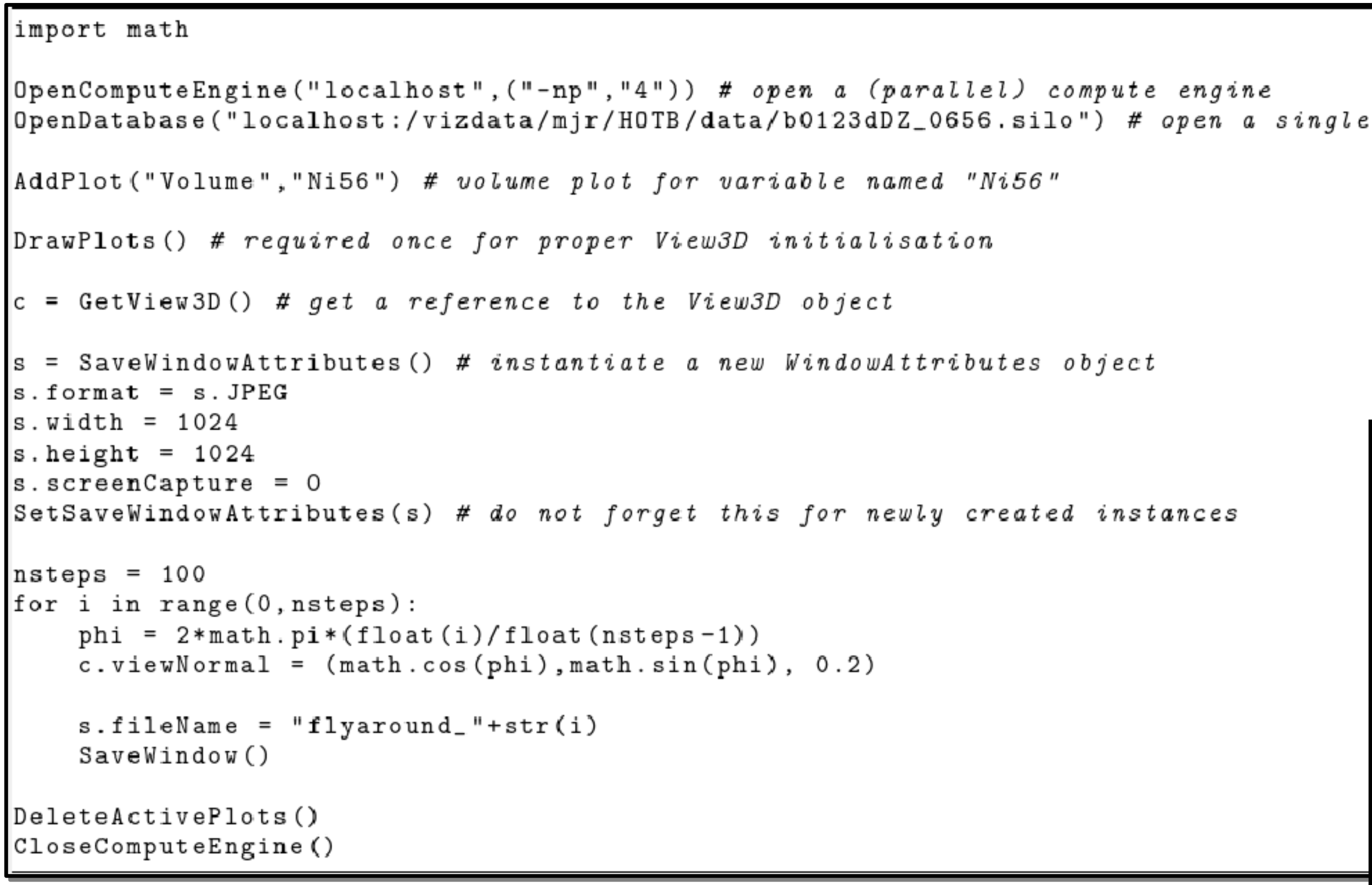
```
OpenDatabase("localhost:/ptmp/mjr/nsCouette/fields_*.xmf␣database")
SetPipelineCachingMode(0)

AddPlot("Pseudocolor", "pressure", 1, 1)
AddOperator("CoordSwap", 1)
AddOperator("Transform", 1)
SetActivePlots(0)

CoordSwapAtts = CoordSwapAttributes()
CoordSwapAtts.newCoord1 = CoordSwapAtts.Coord3
CoordSwapAtts.newCoord2 = CoordSwapAtts.Coord1
CoordSwapAtts.newCoord3 = CoordSwapAtts.Coord2
SetOperatorOptions(CoordSwapAtts, 1)

TransformAtts = TransformAttributes()
TransformAtts.transformType = TransformAtts.Coordinate
TransformAtts.inputCoordSys = TransformAtts.Cylindrical
TransformAtts.outputCoordSys = TransformAtts.Cartesian
TransformAtts.vectorTransformMethod = TransformAtts.AsDirection
TransformAtts.transformVectors = 1
SetOperatorOptions(TransformAtts, 1)

DrawPlots()
for n in range(0,GetDatabaseNStates()-1):
    SetTimeSliderState(n)
    s = SaveWindowAttributes()
    s.format = s.JPEG
    s.width = 800
    s.height = 800
    s.fileName = "movie_%04d" % n
    SetSaveWindowAttributes(s)
    SaveWindow()
```



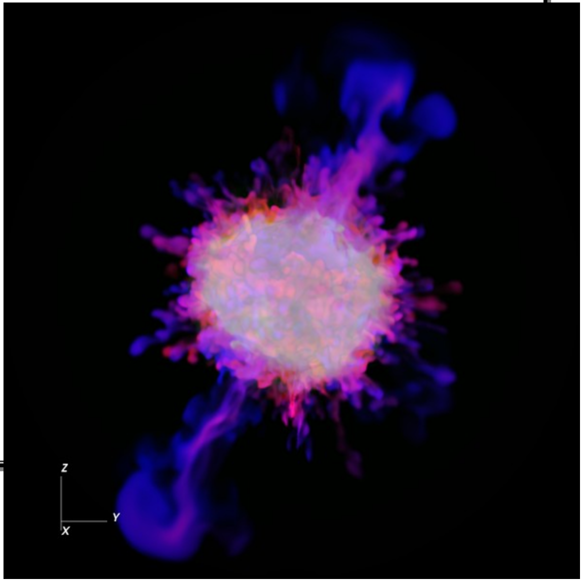
Advanced VisIT: Python scripting

Prototypical example: “flyaround”

Python fragment for rotating an object



```
import math

OpenComputeEngine("localhost",("-np","4")) # open a (parallel) compute engine
OpenDatabase("localhost:/vizdata/mjr/HOTB/data/b0123dDZ_0656.silo") # open a single data file

AddPlot("Volume","Ni56") # volume plot for variable named "Ni56"

DrawPlots() # required once for proper View3D initialisation

c = GetView3D() # get a reference to the View3D object

s = SaveWindowAttributes() # instantiate a new WindowAttributes object
s.format = s.JPEG
s.width = 1024
s.height = 1024
s.screenCapture = 0
SetSaveWindowAttributes(s) # do not forget this for newly created instances

nsteps = 100
for i in range(0,nsteps):
    phi = 2*math.pi*(float(i)/float(nsteps-1))
    c.viewNormal = (math.cos(phi),math.sin(phi), 0.2)

    s.fileName = "flyaround_"+str(i)
    SaveWindow()

DeleteActivePlots()
CloseComputeEngine()
```



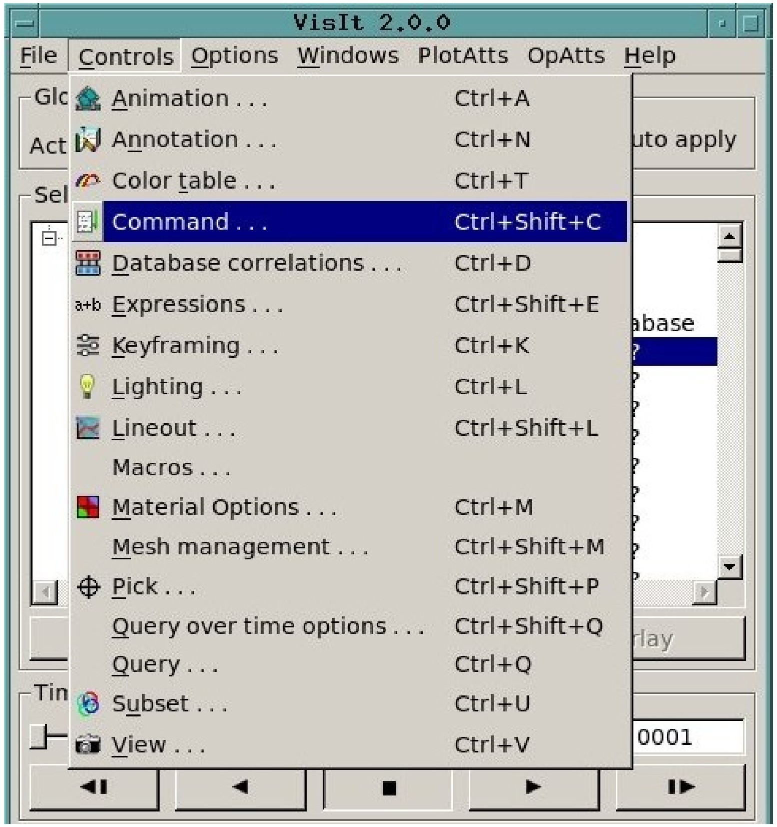
see also examples at http://visitusers.org/index.php?title=Category:Scripting

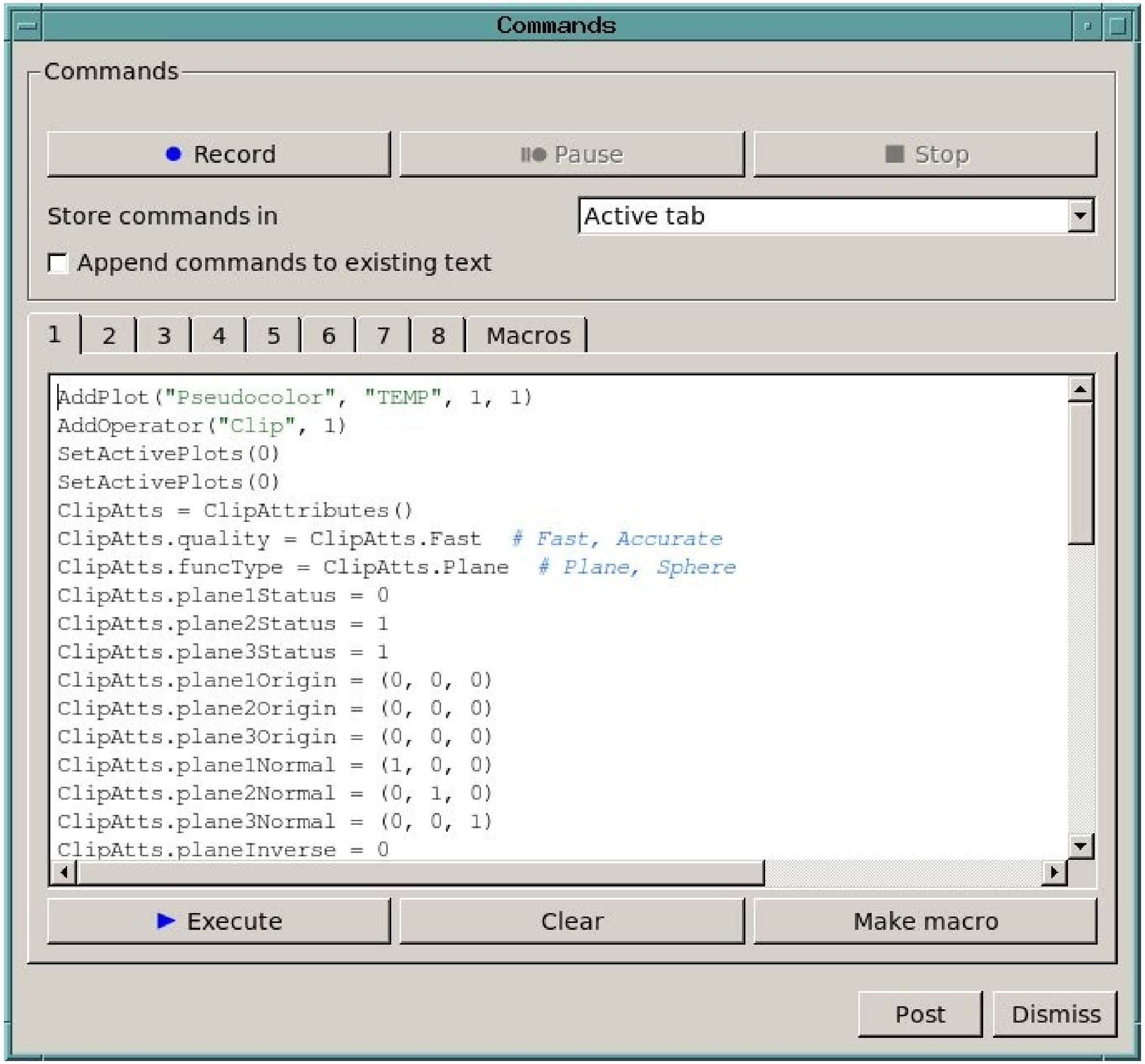
→ Visit-tutorial-python-fly

→ fly through

Advanced VisIT: Python scripting

getting started with Python scripting





```
AddPlot("Pseudocolor", "TEMP", 1, 1)
AddOperator("Clip", 1)
SetActivePlots(0)
SetActivePlots(0)
ClipAtts = ClipAttributes()
ClipAtts.quality = ClipAtts.Fast  # Fast, Accurate
ClipAtts.funcType = ClipAtts.Plane  # Plane, Sphere
ClipAtts.plane1Status = 0
ClipAtts.plane2Status = 1
ClipAtts.plane3Status = 1
ClipAtts.plane1Origin = (0, 0, 0)
ClipAtts.plane2Origin = (0, 0, 0)
ClipAtts.plane3Origin = (0, 0, 0)
ClipAtts.plane1Normal = (1, 0, 0)
ClipAtts.plane2Normal = (0, 1, 0)
ClipAtts.plane3Normal = (0, 0, 1)
ClipAtts.planeInverse = 0
```

1) use the dialog `Controls` → `Command` to automatically create Python code
2) paste code into your favourite editor and save as a python script
3) polish and extend Python code
4) consult the *“VisIT Python Interface Manual”*

Paraview Basics

Paraview:

- homepage: www.paraview.org
- tutorial: http://www.paraview.org/Wiki/The_ParaView_Tutorial

Paraview, (according to the Paraview homepage) is ...

- an open-source, multi-platform data analysis and visualization application. ParaView users can quickly build visualizations to analyze their data using qualitative and quantitative techniques. The data exploration can be done interactively in 3D or programmatically using ParaView's batch processing capabilities.
- ParaView was developed to analyze extremely large datasets using distributed memory computing resources. It can be run on supercomputers to analyze datasets of petascale size as well as on laptops for smaller data, has become an integral tool in many national laboratories, universities and industry, and has won several awards related to high performance computation.
- developed by Kitware Inc. and academic/US government agency partners (LANL, Sandia)

Paraview, (according to *my personal bias*) is …

- certainly a great tool ! slightly less intuitive than VisIt
- slightly less tailored towards typical (astrophysics, CFD) visualization tasks than VisIt
- lacks genuine molecular visualization capabilities

Paraview Overview

Visualization “pipeline”

- filters are the central concept of manipulation
- options for display

Example

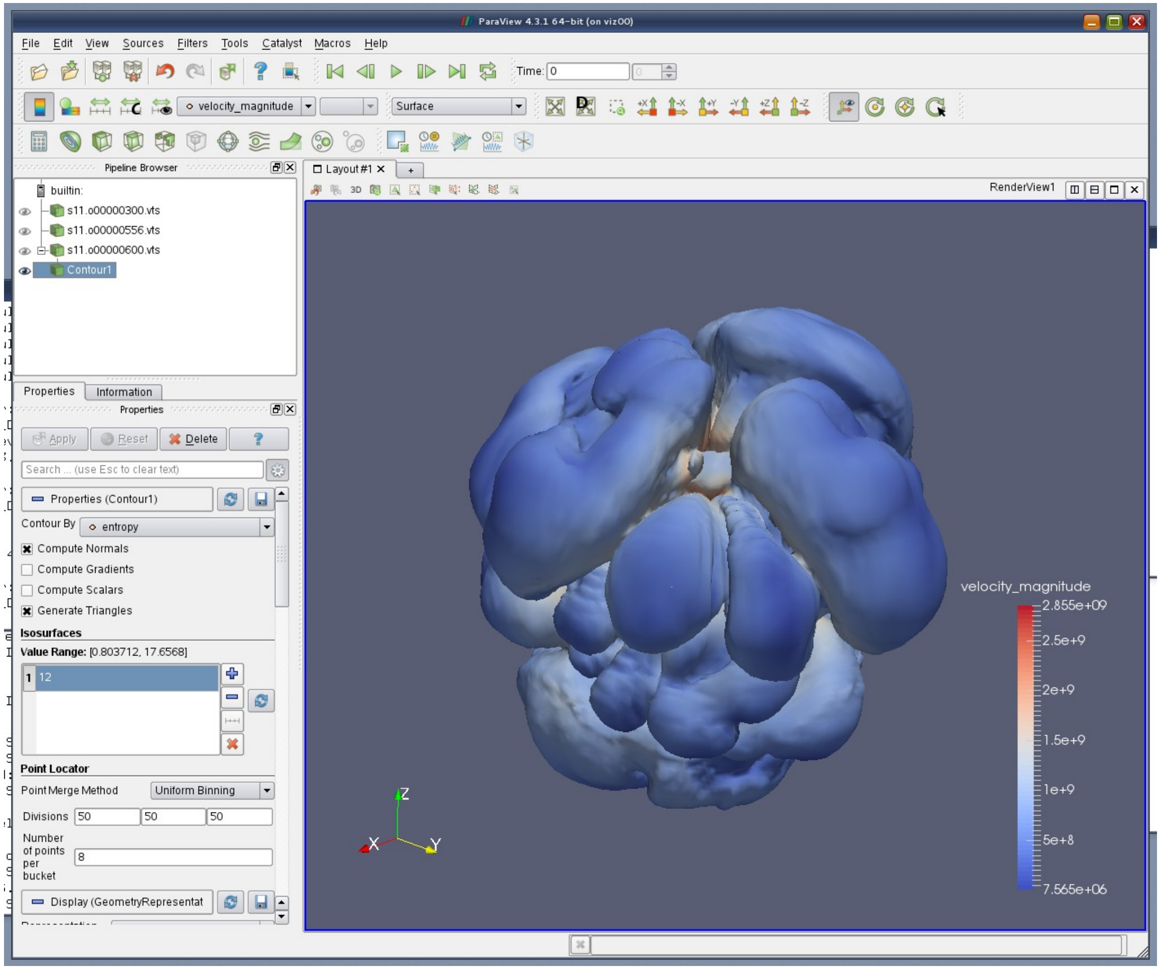
isosurface of a scalar variable F(x,y,z) colored

by the value of another scalar variable G(x,y,z)

(F, G are defined on the same mesh):

1) create a contour filter for F
2) apply colouring for G

   (instead of default: F)



Contour By: entropy

☒ Compute Normals

☐ Compute Gradients

☐ Compute Scalars

☒ Generate Triangles

Isosurfaces

Value Range: [0.803712, 17.6568]

1 12

Point Locator

Point Merge Method: Uniform Binning

Divisions: 50 50 50

Number of points per bucket: 8

Paraview GUI

main menu

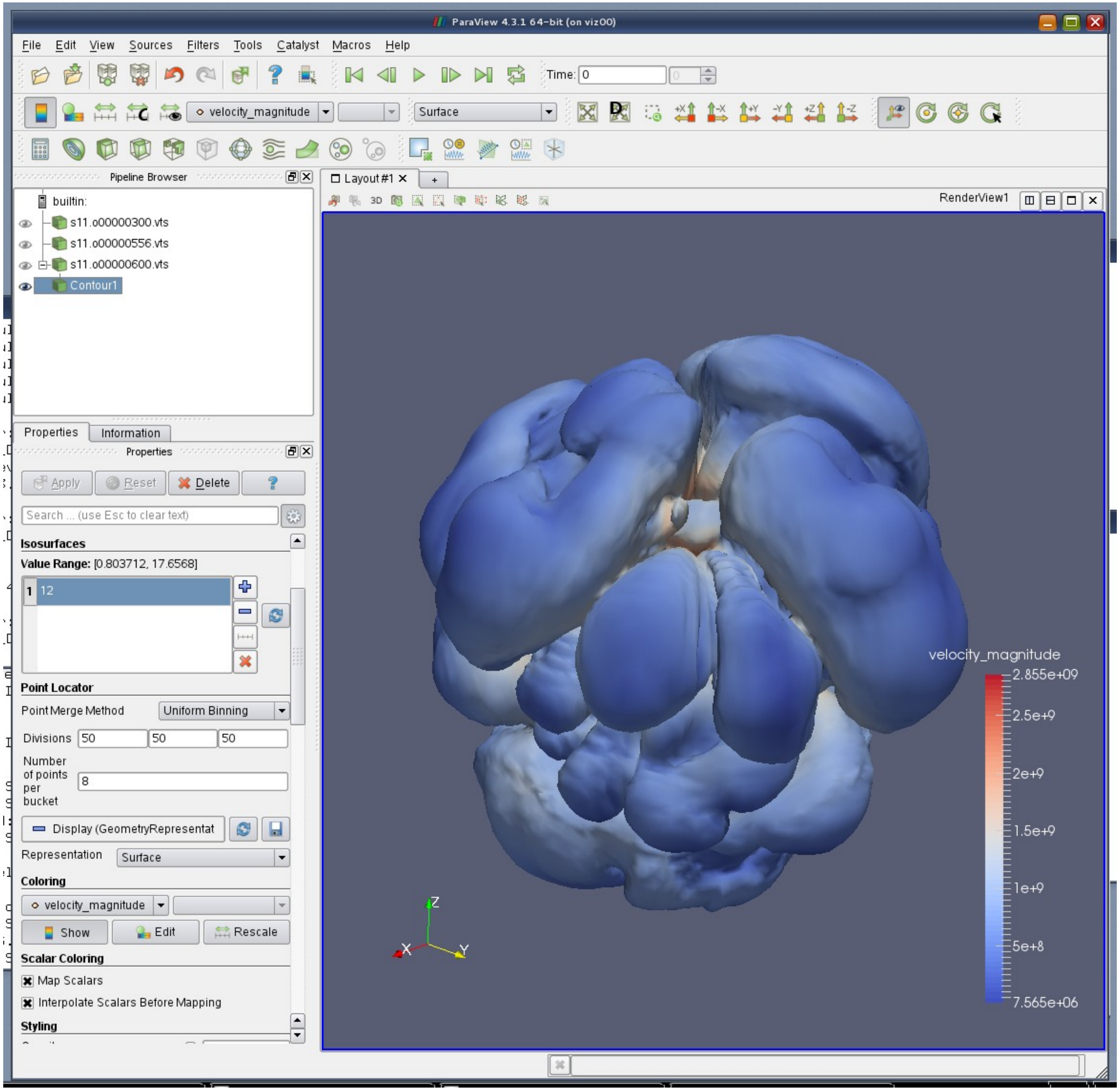
file and animation controls

common options

common filters

pipeline

options



Time: 0

velocity_magnitude | Surface

builtin:
- s11.000000300.vts
- s11.000000556.vts
- s11.000000600.vts
- Contour1

Search ... (use Esc to clear text)

Isosurfaces

Value Range: [0.803712, 17.6568]

1 12

Point Locator

Point Merge Method: Uniform Binning

Divisions: 50 50 50

Number of points per bucket: 8

Display (GeometryRepresentat

Representation: Surface

Coloring

velocity_magnitude

Scalar Coloring

☒ Map Scalars

☒ Interpolate Scalars Before Mapping

Styling

Paraview demo

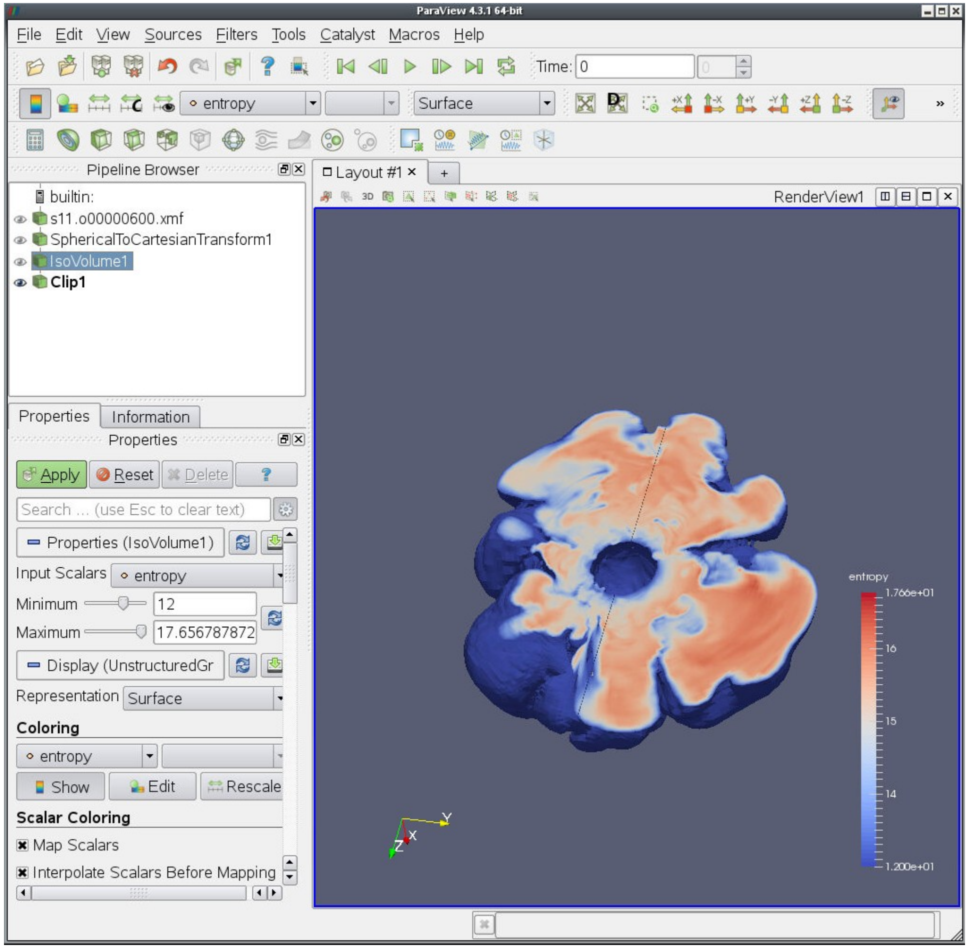
The demo dataset

- example data taken from 3D supernova simulations of Hanke et al. (arXiv:1108.4355)
- data and grid:
  - spherical coordinates *(r,Θ,Φ)*
  - scalar variable *s(r,Θ,Φ)*
  - a subset (3 time steps, few variables) of the complete dataset

    (400x60x120), ca. 20 variables, 1000 dumps

  (→ meanwhile: 10x increase in resolution)

  - Xmdf format: .xmf, .h5
  - **goals:**
    1) **follow the demo**
    2) **experiment yourself later on (Q&A w/ Jan and Markus at the workplaces)**



Time: 0

entropy | Surface

builtin:
s11.000000600.xmf
SphericalToCartesianTransform1
IsoVolume1
Clip1

Properties (IsoVolume1)
Input Scalars: entropy
Minimum: 12
Maximum: 17.656787872

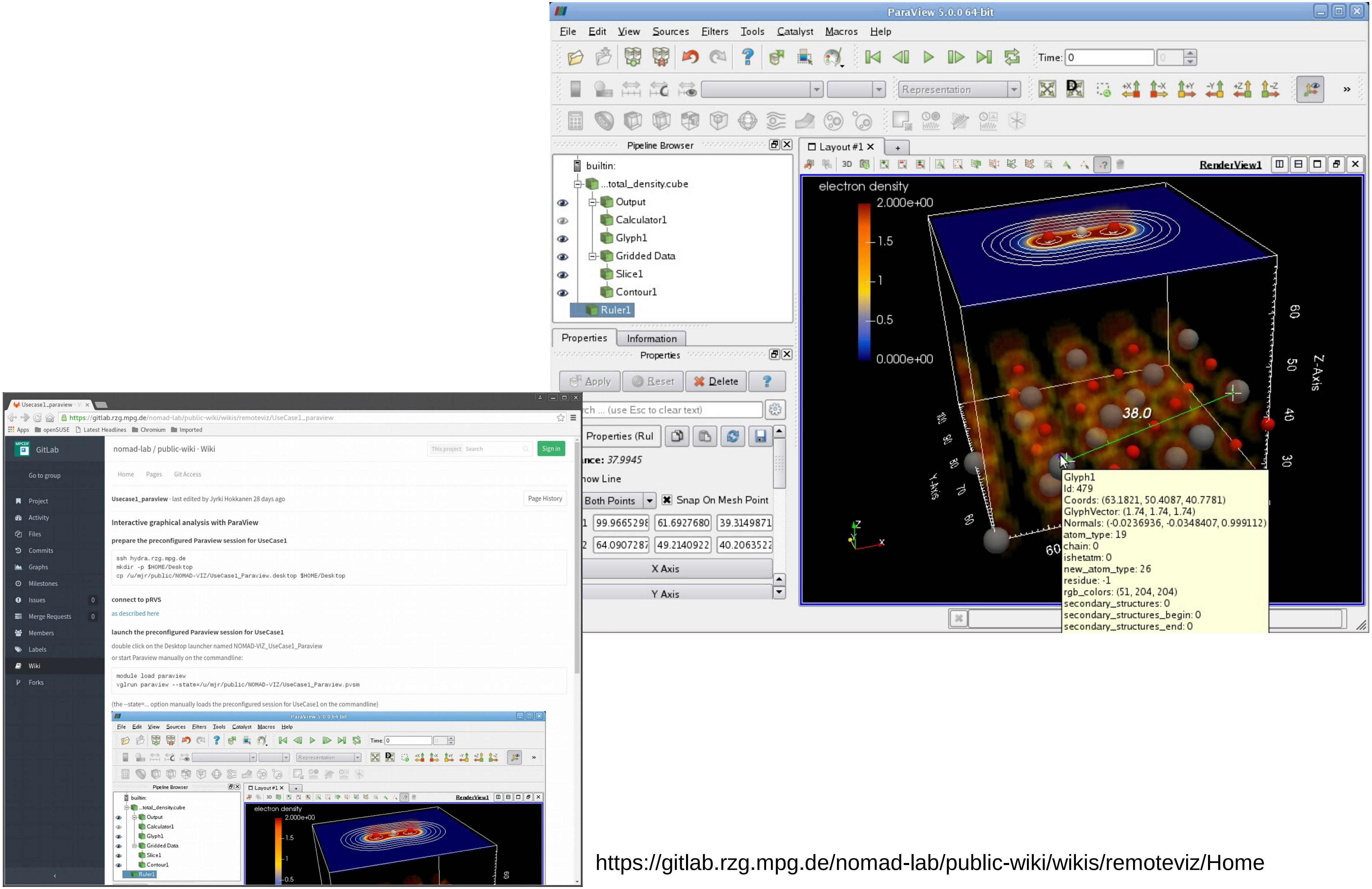
Display (UnstructuredGr
Representation: Surface

Coloring
entropy

Scalar Coloring
☒ Map Scalars
☒ Interpolate Scalars Before Mapping

- **files:**
  `~/ISSS12/Rampp_Skala/VIZ/visit/*.xmf`
  `~/ISSS12/Rampp_Skala/VIZ/paraview/SphericalToCartesianTransform.cpd`
  (special thanks to E. Erastova for developing this "programmable filter" - prototype, a generic solution will be contributed)

NOMAD visualization use case



Glyph1
Id: 479
Coords: (63.1821, 50.4087, 40.7781)
GlyphVector: (1.74, 1.74, 1.74)
Normals: (-0.0236936, -0.0348407, 0.999112)
atom_type: 19
chain: 0
ishetatm: 0
new_atom_type: 26
residue: -1
rgb_colors: (51, 204, 204)
secondary_structures: 0
secondary_structures_begin: 0
secondary_structures_end: 0

nomad-lab / public-wiki · Wiki

Usecase1_paraview · last edited by Jyrki Hokkanen 28 days ago

Interactive graphical analysis with ParaView

prepare the preconfigured Paraview session for UseCase1

```
ssh hydra.rzg.mpg.de
mkdir -p $HOME/Desktop
cp /u/mjr/public/NOMAD-VIZ/UseCase1_Paraview.desktop $HOME/Desktop
```

connect to pRVS

as described here

launch the preconfigured Paraview session for UseCase1

double click on the Desktop launcher named NOMAD-VIZ_UseCase1_Paraview

or start Paraview manually on the commandline:

```
module load paraview
vglrun paraview --state=/u/mjr/public/NOMAD-VIZ/UseCase1_Paraview.pvsm
```

(the --state=... option manually loads the preconfigured session for UseCase1 on the commandline)

https://gitlab.rzg.mpg.de/nomad-lab/public-wiki/wikis/remoteviz/Home

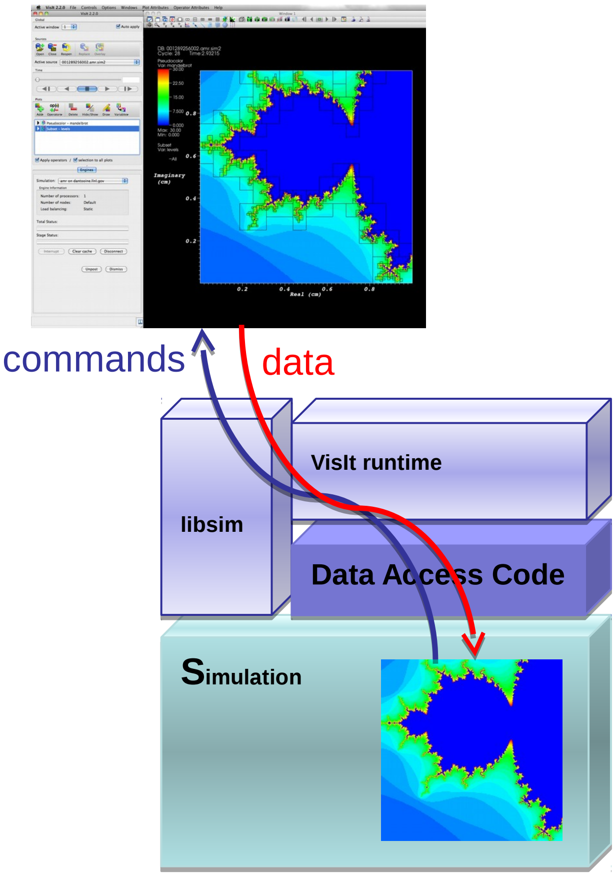
M. Rampp, MPCDF

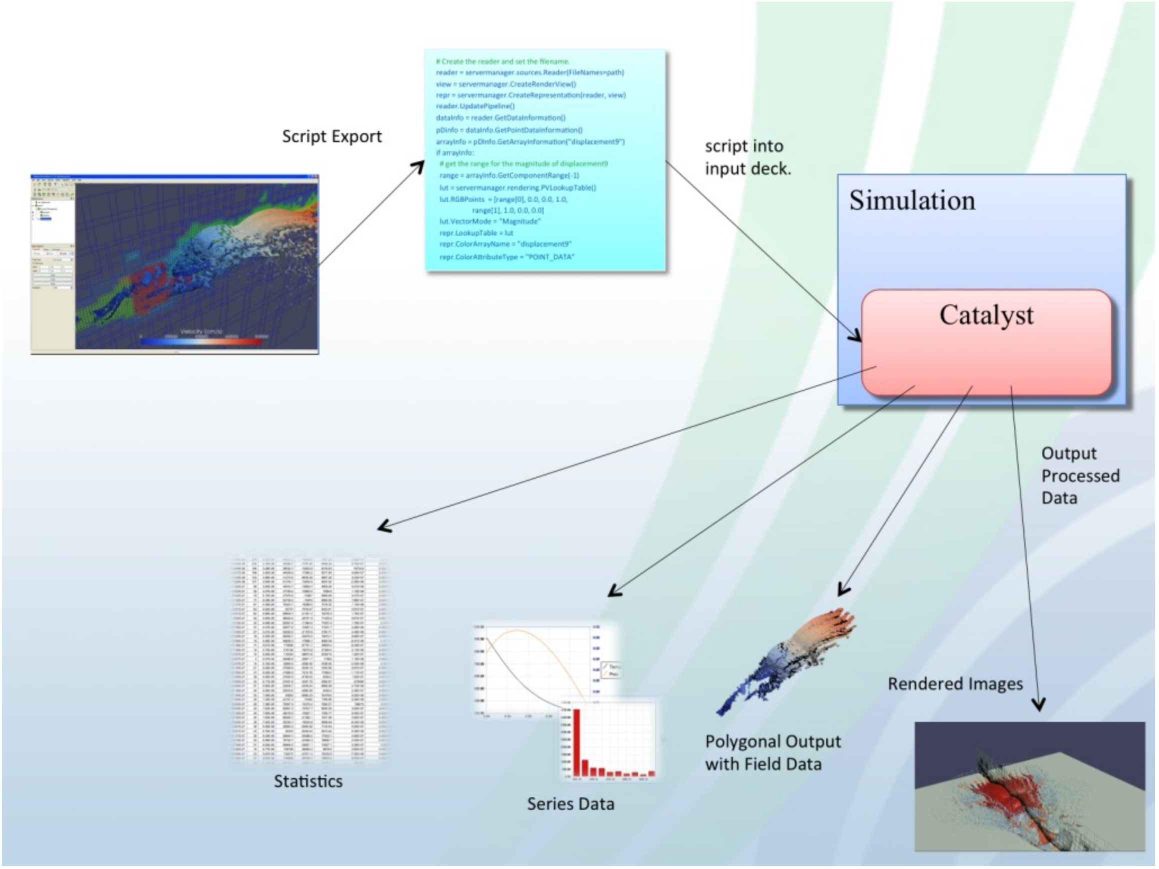
In-situ visualization

- supported by Paraview (*Catalyst*), VisIt (*libsim*)

- basic technique:
  implement library calls in simulation code (C, Fortran API)
  mediates callbacks to visualization tool

a big buzz or something interesting to watch ???





```
# Create the reader and set the filename.
reader = servermanager.sources.Reader(FileNames=path)
view = servermanager.CreateRenderView()
repr = servermanager.CreateRepresentation(reader, view)
reader.UpdatePipeline()
dataInfo = reader.GetDataInformation()
pDinfo = dataInfo.GetPointDataInformation()
arrayInfo = pDInfo.GetArrayInformation("displacement9")
if arrayInfo:
 # get the range for the magnitude of displacement9
 range = arrayInfo.GetComponentRange(-1)
 lut = servermanager.rendering.PVLookupTable()
 lut.RGBPoints  = [range[0], 0.0, 0.0, 1.0,
                range[1], 1.0, 0.0, 0.0]
 lut.VectorMode = "Magnitude"
 repr.LookupTable = lut
 repr.ColorArrayName = "displacement9"
 repr.ColorAttributeType = "POINT_DATA"
```

In-situ visualization

Motivation

- GPUs of a supercomputer can do graphics (*sic!*)
- watch the simulation running, reduce latency to first results ? *… Hmmm, well, …*
- enable real-time/interactive simulations (exploration)?

- **avoid large-scale postprocessing runs**
- **reduce I/O traffic and volume (!)**
- write “data products” (e.g. iso-surfaces in vtk format) on the fly rather than full scale dumps

  → **for later postprocessing**
- flexible “instrumentation” of the code:

  Catalyst: python script decides at runtime (job startup) about the data to be dumped

  (vs. implementation of data-reduction in the I/O section of the code → x-plane, y-plane, or z-plane, … ?)

  → **for later postprocessing**

HPC Visualization infrastructure

Remote visualization

why centralizing visualization ?

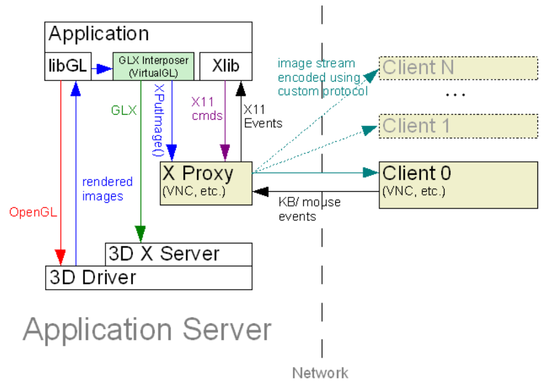
- huge amounts of output data produced by HPC simulations
- transfer of raw data for local analysis & visualisation no more possible
- even dumping the RAM is becoming prohibitive due to I/O constraints
  - in-situ visualisation (not covered here)
- visualisation requires HPC-like resources (specialized hardware, housing, . . .)
- requires substantial expertise on methods, software, . . .   sustainability

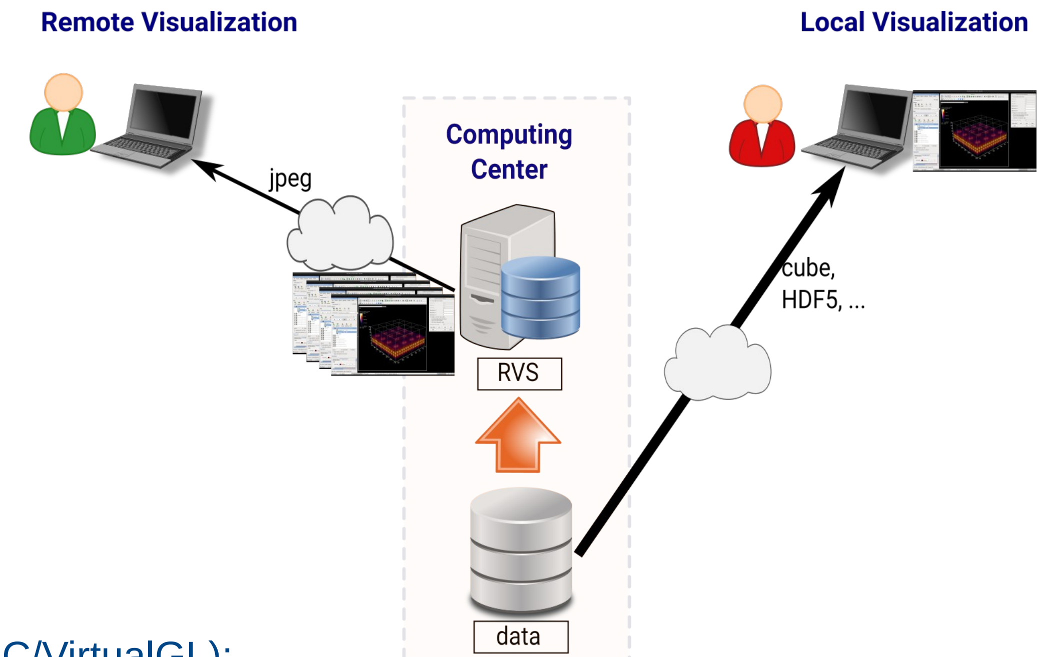
→ a necessity for a HPC centre rather than an optional service

enabling technology for operating a remote visualization service (RVS)

- “server-side” rendering ← naïve approach (“ssh+X”) does not work!
- efficient and *transparent* remote rendering solution via WAN: VirtualGL/TurboVNC
- issues: trans-continental latency

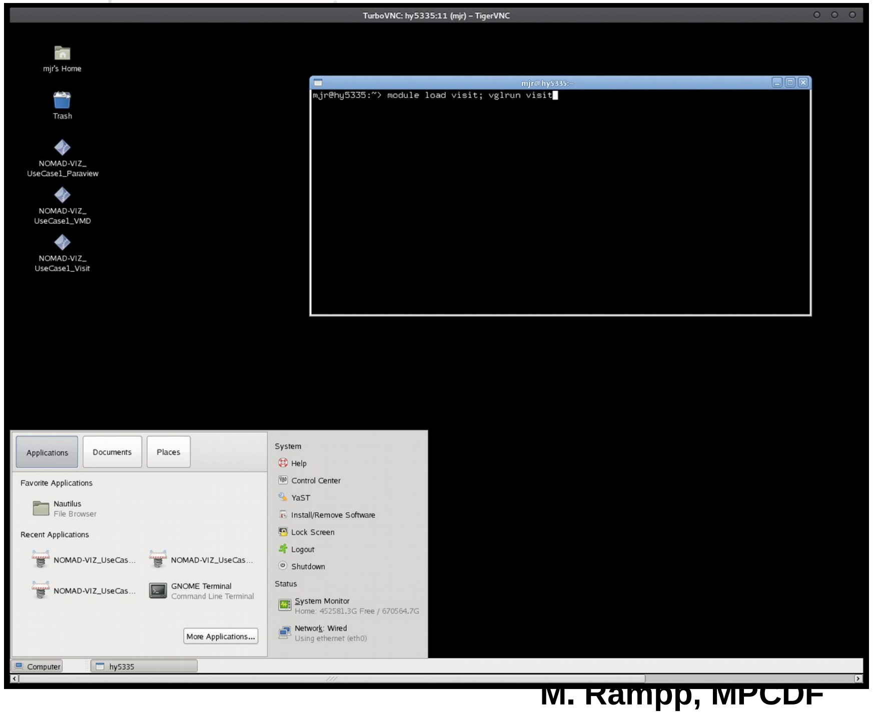
Remote visualiztion basics





Technology for remote visualisation (TurboVNC/VirtualGL):

- proven open-source solution, deployed by many HPC centres (e.g. BSC, CSC, LRZ, MPCDF)
- application-agnostic “remote desktop” (with optimizations for 3D graphics)
- user's experience (example MPCDF viz service, linux desktop): remote desktop with optimization options: network bandwidth, latency, quality of rendering
- transparent use of visualization resources and applications (look-and-feel like local desktop):
  ~>vglrun <executable>


TurboVNC: hy5335:11 (mjr) – TigerVNC

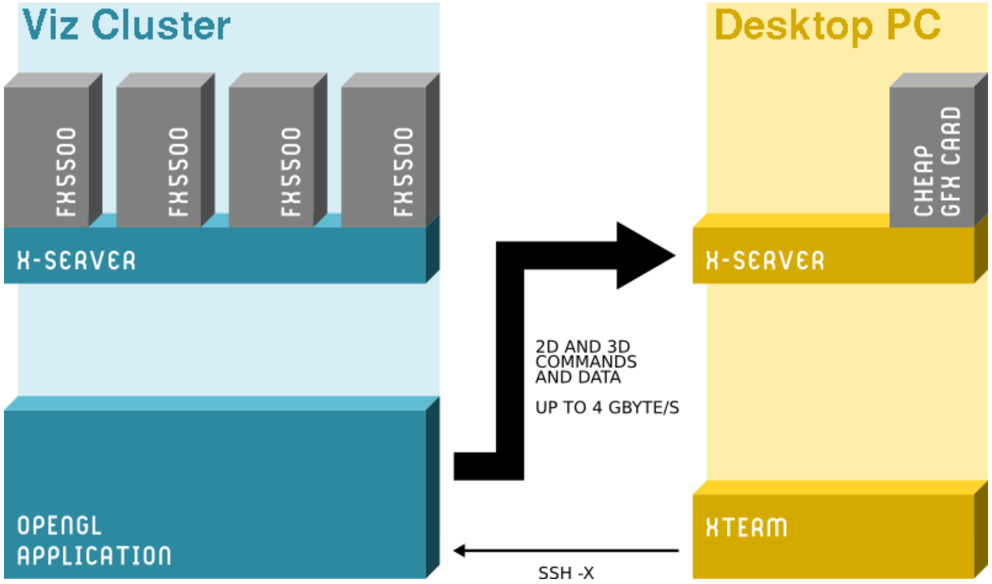
mjr@hy5335:~

```
mjr@hy5335:~> module load visit; vglrun visit
```

Visualization infrastructure

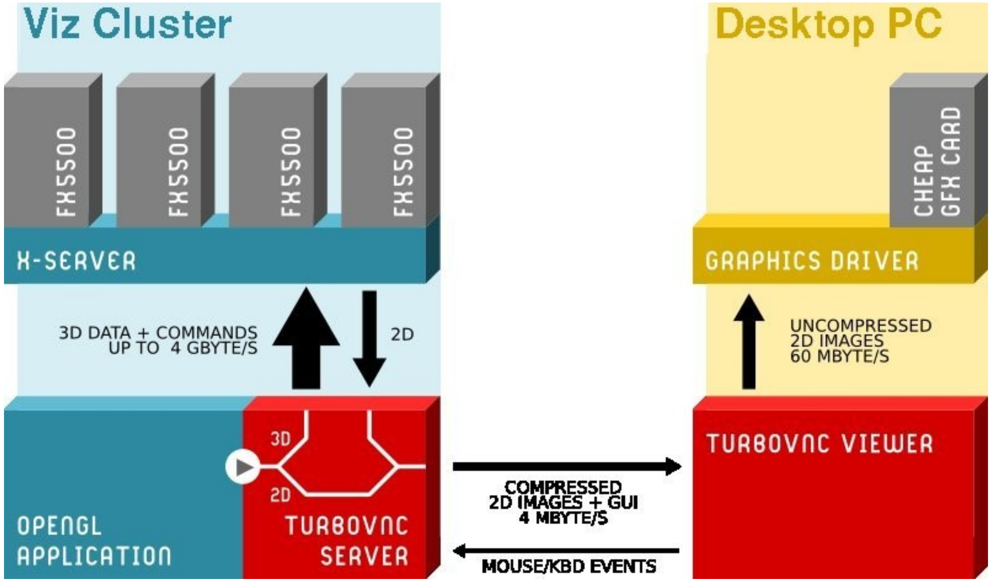
Traditional (“X forwarding over ssh”)

- 3D geometry is transferred over network
- fails to deliver interactive frame rates
- uses X-server/graphics card of the client
- not suited for 3D applications



Server-side rendering (“VNC”-like)

- only (compressed) images are transferred
- interactive frame rates with moderate WAN bandwidth
- uses X-server/graphics card(s) of the server
- generic and transparent solution (OpenGL)
- mature software solutions/products:
  - VirtualGL/TurboVNC (Open Source, ex SUN)



(illustrations by L. Scheck, with permission by LRZ)

RVS at the MPCDF

Focus MPCDF:

- enable our (geographically dispersed) scientific users to perform complex visualization tasks *without special technical prerequisites* (software, hardware)
- remote visualization via WAN, LAN (VirtualGL/TurboVNC)

Hardware: Hydra-viz (+Draco-viz)

4 visualization nodes, each with:

- 2 Intel Xeon CPUs (E5-2680v2 @2.80GHz), 10 cores each
- 2 NVIDIA Tesla GPUs (K20x), 6 GB GPU memory each
- 2 nodes with 128 GB CPU RAM
- 2 nodes with 256 GB CPU RAM

=> 8 concurrent sessions (8+ on draco)

=> use view-only VNC password to (passively) share screen (~> `vncpasswd`)

Software/Workflow

- visualization applications: VisIt, Paraview, pymol, VMD, …

setup once:

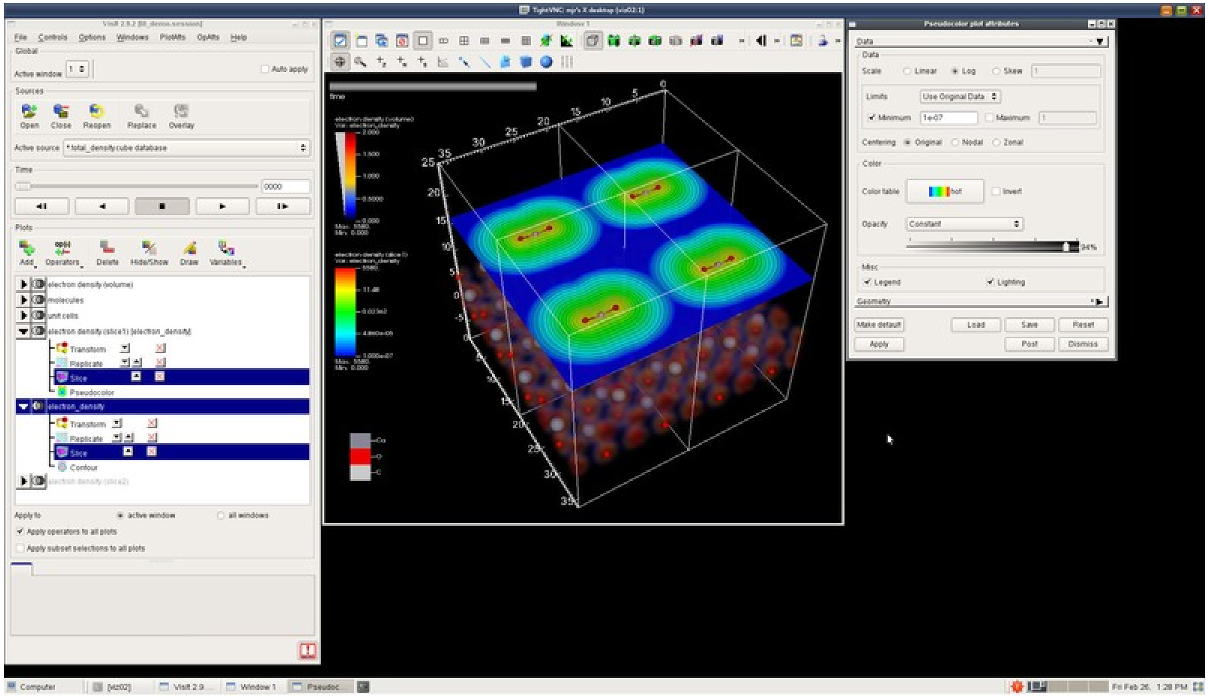
```
mjr@hydra> module load remotevis
mjr@hydra> setup_remotevis
```

submit visualization job:

```
mjr@hydra> vi $HOME/remotevis/remotevis.ll
mjr@hydra> llsubmit $HOME/remotevis/remotevis.ll
```

receive E-mail with connection info and connect with TurboVNC

```
mjr@myPC> vncviewer -via gatezero.rzg.mpg.de hydra:55910
```



Outlook

next steps (octopus):

- finalize data format for octopus viz: likely VTK (or cube)
- plan implementation of libsim and/or catalyst (in-situ viz) into octopus
- help streamlining standard visualization workflows with octopus (et al.)
- support viz applications with custom python scripts → M. Compostella
- keep an eye on Paraview and other tools

next steps (MPCDF infrastructure):

- RVS on draco (8+ visualization slots, like hydra)
- GPU virtualization and convenience of RVS access ? → NOMAD CoE
- VR ? → LRZ, NOMAD

- Backup material

Data handling strategies

Main selection criteria for data format (resp. I/O library)

portability

- platforms: different HPC machines, hardware generations, software stacks
- runs: distribution of data to parallel processors
- software support: available tools, libraries, community experience, ...

performance

- parallel I/O (parallel file systems: LUSTRE, GPFS, BeeGFS, ...)
- data volumes (compression, archiving, ...)

usability/human efforts

- data “handling”: copying, bundling, archiving, debugging, .

... avoid sacrificing portability and usability for performance

Challenges

- typically there *is* an I/O bottleneck in HPC → performance can be a real showstopper
- balance/good compromise?

Data handling

General recommendations

- design and implement a clear hierarchy for output data (dimensionality, frequency of dumps)
    - control variables (scalars, 1D vectors) → usability, negligible I/O (stdout or alike)
    - data to be *routinely* analyzed/visualized (2D, 3D, 4D, …) → optimize for usability keep an eye on performance
    - checkpoint/restart: full precision, usually never analyzed → optimize for performance
    - in-situ visualization (?): full precision, full time resolution
- if usability vs. performance does not work out → think of a post-processing pipeline

  caveat: memory requirements → an HPC job in its own right => gains?

Software solutions for HPC

- MPI-IO (low-level), HDF5 (ecosystem of specifications, APIs, tools), NETCDF, ADIOS, …
- popular strategy: HDF5 for data to be visualized, raw MPI-IO or HDF5 for checkpoints
    - HDF5: open source, widespread use (blame your local computing center!), integrates with parallel applications and file systems, high-performance I/O (implemented on top of MPI-IO), many utilities and tools, …
    - **supported by many visualization tools (VisIt, Paraview ,...)**

HDF5

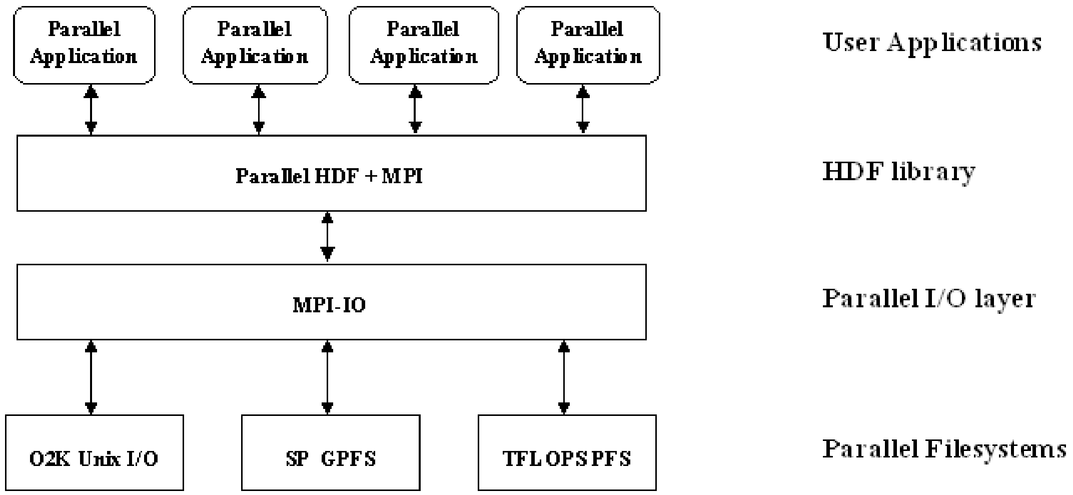
HDF5 in a nutshell

- **H**ierarchical **D**ata **F**ormat
- documentation www.hdfgroup.org
- known issues and pitfalls:

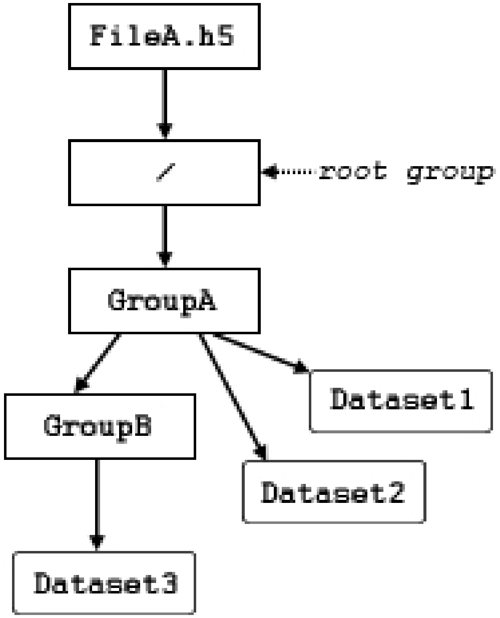
  →performance (tiny block sizes)

  →2GB limit per MPI task (will be fixed)

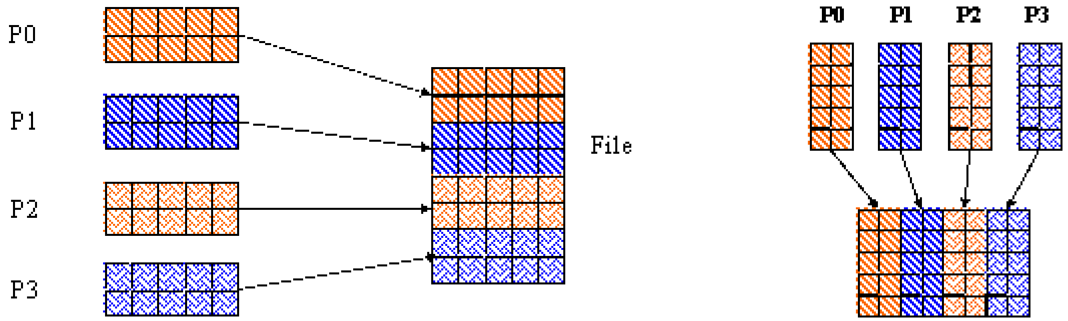
architecture



hierarchical data format



example data distribution: “contiguous hyperslab”



memory-to-file mapping (C)

memory-to-file mapping (Fortran)

(images taken from : www.hdfgroup.org)

XDMF: metadata for HDF5

XMDF (eXtensible Data Model and Format) in a nutshell

Purpose: meta data description for HDF5 → required to read (non-trivial) HDF5 data into VisIT, Paraview

(why? → there is, no concept of, e.g., a “grid” in a hdf5 file)

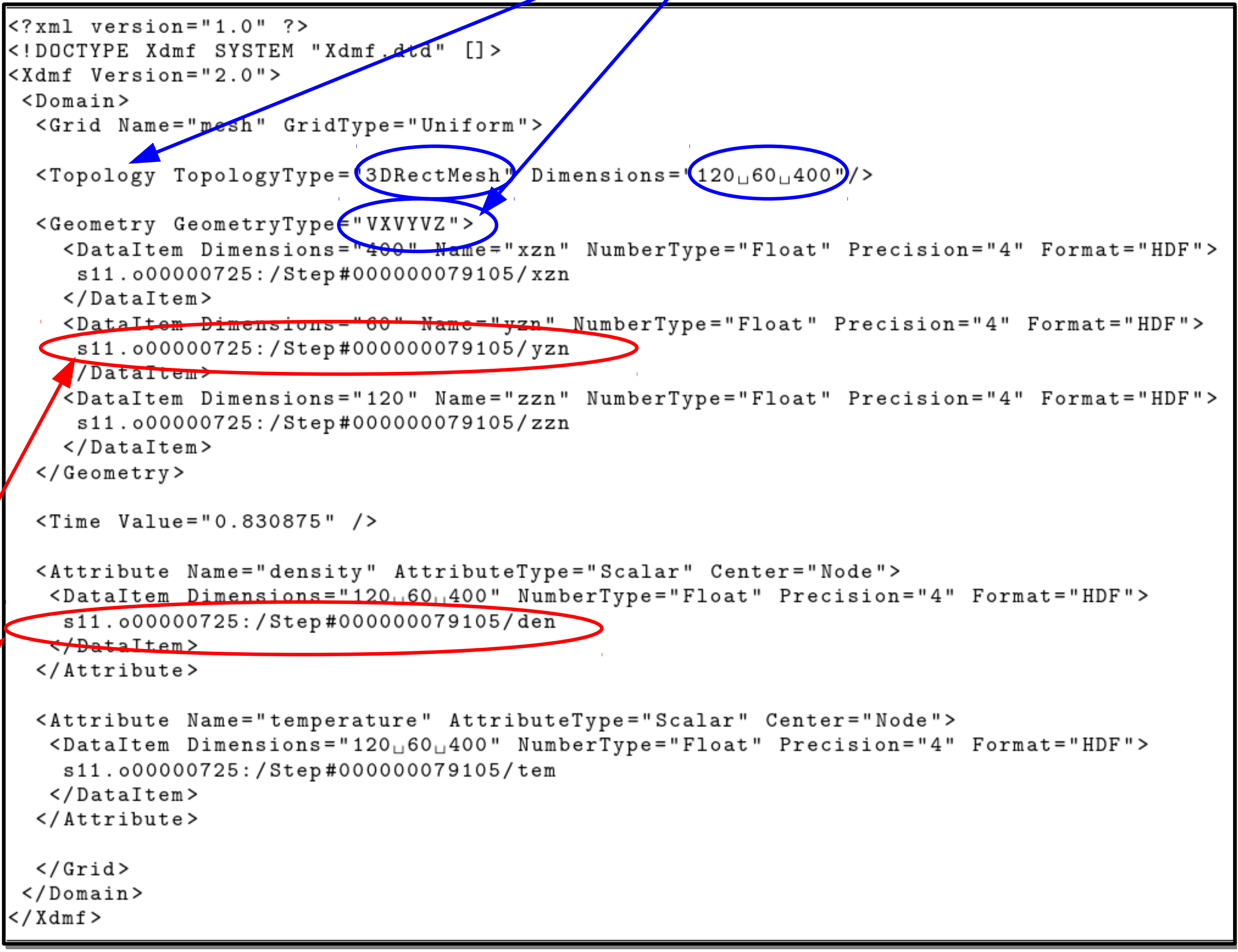
concept:

- XML file with “light” meta data
- references to “heavy” data stored in hdf5 file

- APIs for FORTRAN, C++, …, but can easily be generated *a-posteriori* (→ *no installation required*)

documentation: www.xdmf.org (poor)

grid metadata



```
<?xml version="1.0" ?>
<!DOCTYPE Xdmf SYSTEM "Xdmf.dtd" []>
<Xdmf Version="2.0">
 <Domain>
  <Grid Name="mesh" GridType="Uniform">

  <Topology TopologyType="3DRectMesh" Dimensions="120␣60␣400"/>

  <Geometry GeometryType="VXVYVZ">
    <DataItem Dimensions="400" Name="xzn" NumberType="Float" Precision="4" Format="HDF">
     s11.000000725:/Step#000000079105/xzn
    </DataItem>
    <DataItem Dimensions="60" Name="yzn" NumberType="Float" Precision="4" Format="HDF">
     s11.000000725:/Step#000000079105/yzn
    </DataItem>
    <DataItem Dimensions="120" Name="zzn" NumberType="Float" Precision="4" Format="HDF">
     s11.000000725:/Step#000000079105/zzn
    </DataItem>
  </Geometry>

  <Time Value="0.830875" />

  <Attribute Name="density" AttributeType="Scalar" Center="Node">
   <DataItem Dimensions="120␣60␣400" NumberType="Float" Precision="4" Format="HDF">
    s11.000000725:/Step#000000079105/den
   </DataItem>
  </Attribute>

  <Attribute Name="temperature" AttributeType="Scalar" Center="Node">
   <DataItem Dimensions="120␣60␣400" NumberType="Float" Precision="4" Format="HDF">
    s11.000000725:/Step#000000079105/tem
   </DataItem>
  </Attribute>

  </Grid>
 </Domain>
</Xdmf>
```

links to data in hdf5 file

Data handling strategies

Explicit data conversion

- 🟢 allows some basic post-processing and/or data reduction of simulation output
- 🟠 quick (& dirty) programming: copy/paste from I/O statements in simulation code
- 🔴 duplication of data

→ which format? Silo (VisIt's "proprietary" data format), HDF5, VTK, ...

Development of a plugin (VisIt, Paraview)

- 🟢 no data duplication, no additional pre-processing step
- 🟢 plugin is dynamically loaded (code can reside under `$HOME`)
- 🟠 development requires C programming and compilation against a VisIt installation
- 🔴 not portable to other tools

Adaptation of I/O in simulation code

- 🟢 no data duplication, no additional preprocessing step
- 🟢 can promote interoperability with other tools (depending on chosen format, e.g. HDFVIEW)
- 🟠 implications for software management (code policies, access to source code, …)

→ which format? HDF5 (requires XDMF or alike for metadata), VTK, ...