Introduction to CitcomS

Eh Tan

March 10, 2009

CIG Training Session, EarthScope Meeting

What is CitcomS

- Mantle convection code
  - 3D spherical global mesh
  - variable viscosity, tracers, compositions, phase transitions, dynamic topography
  - highly parallel (up to 4K CPUs) and fast
  - open source
  - good documentation and support
  - download from CIG www.geodynamics.org

Momentum Equations

$$\nabla \cdot v = 0$$

$$-\nabla p + \eta \nabla^2 v + \Delta\rho\, g = 0$$

- Material is incompressible
- Flow driven by density difference, impeded by viscosity

Energy Equation

$$\rho C_p \frac{\partial T}{\partial t} = \rho C_p v \cdot \nabla T + k \nabla^2 T + \rho H$$

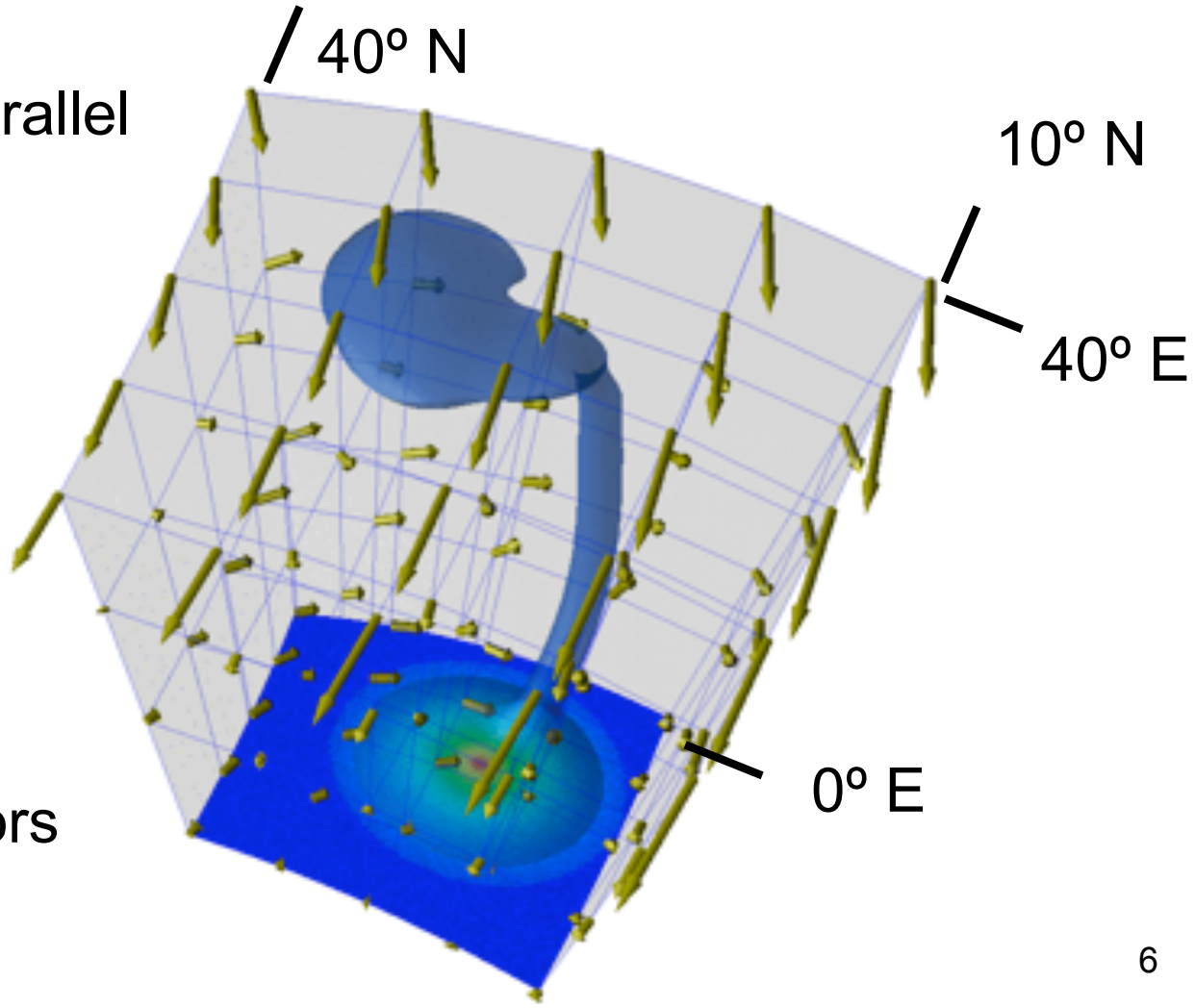
- Heat is carried by flow
- Conduction/diffusion
- Internal (radiogenic) heating

Meshes

Regional Mesh

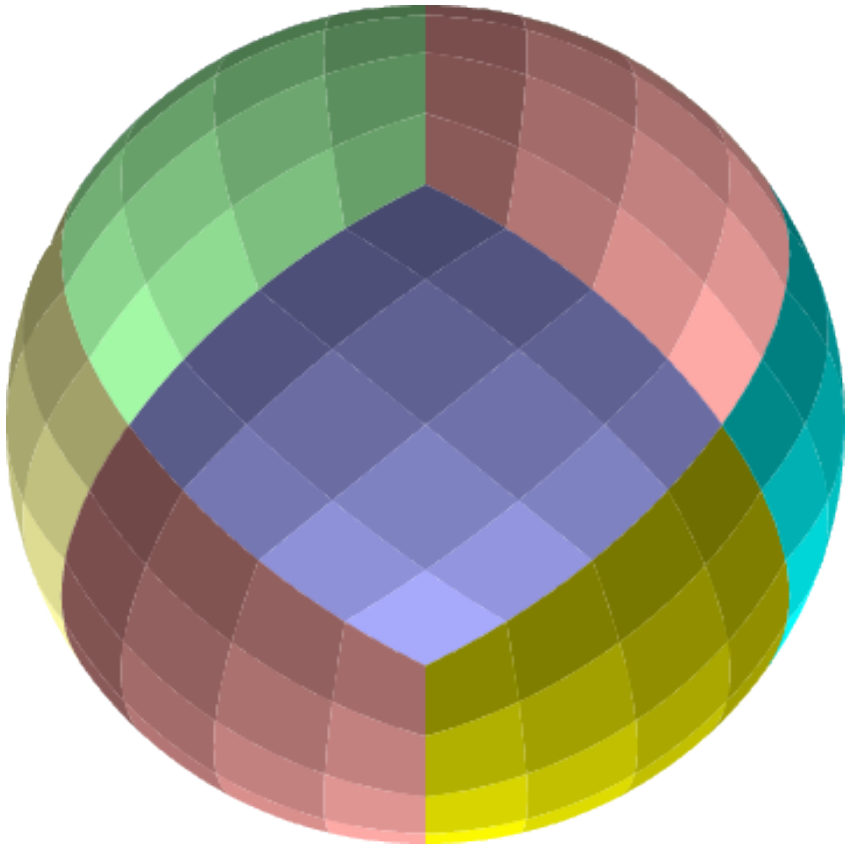
Grid lines are parallel to longitude and latitude

The whole domain can be partitioned into NxMxL processors

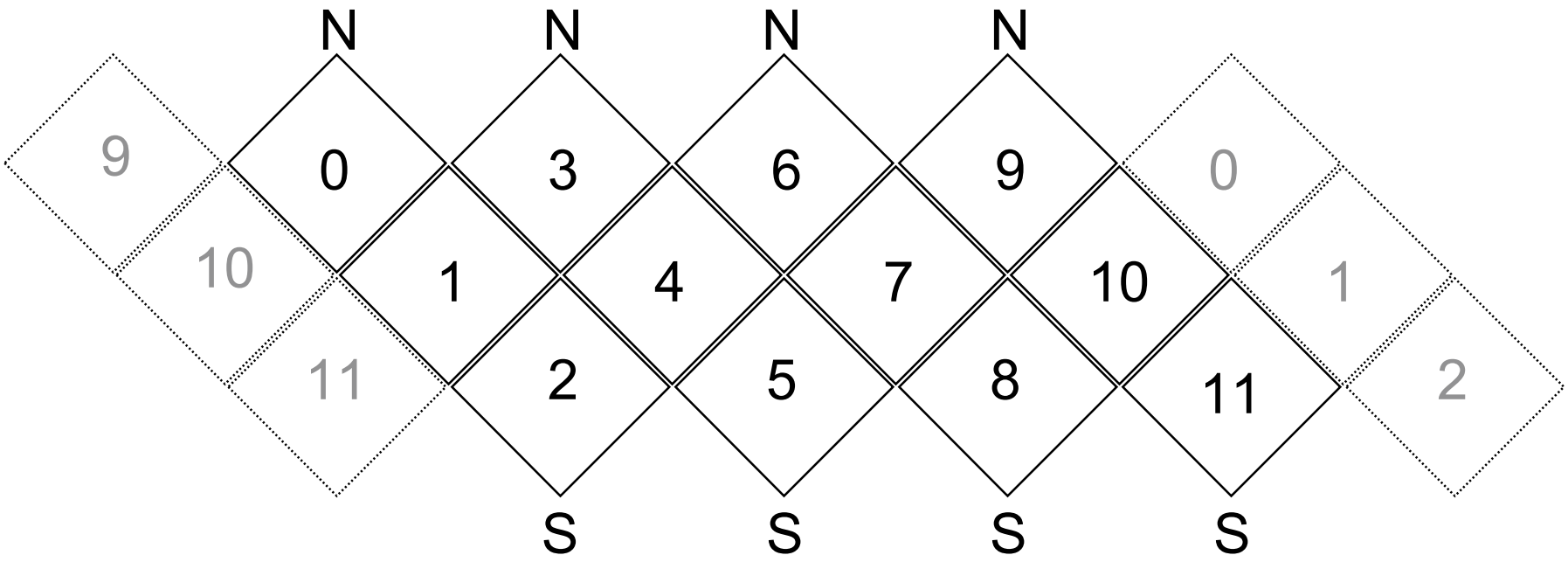


Global Mesh

- 12 caps
- Each cap extends from the surface to the CMB
- Each cap can be partitioned into NxNxM processors
- 12xNxNxM processors in total (N=4 in this figure)



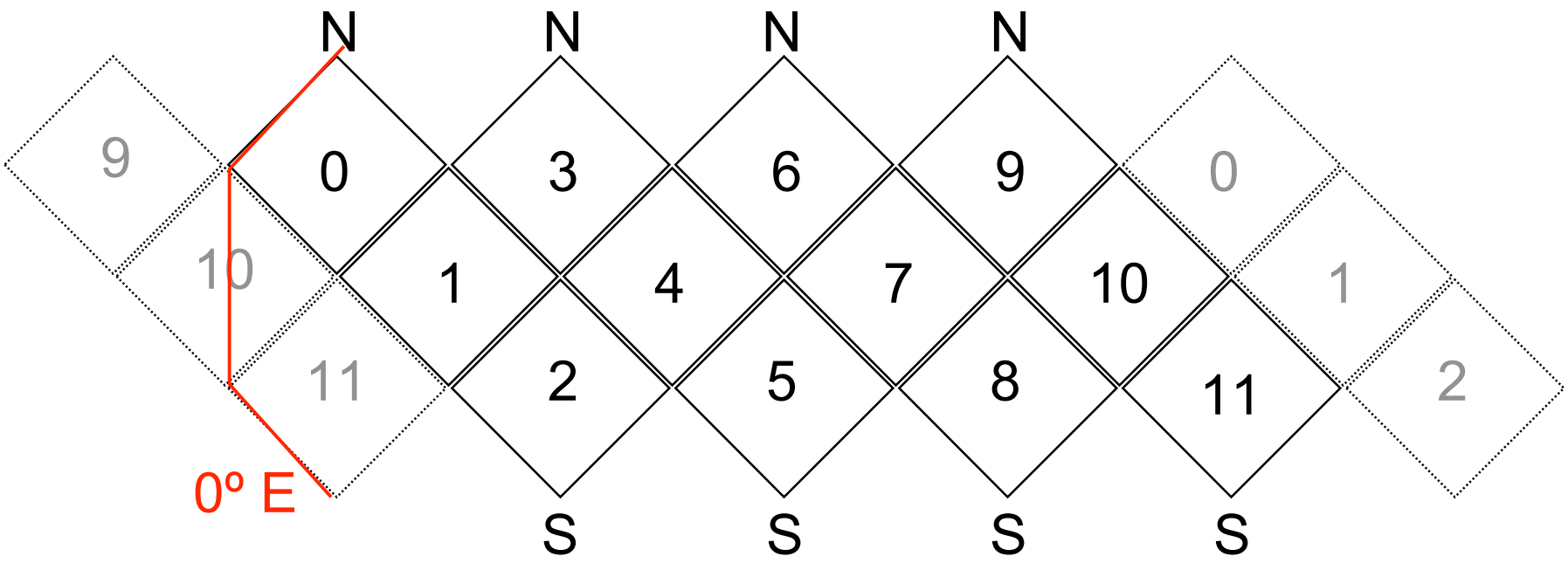
Topology of 12 Caps



N: North Pole S: South Pole

Computational Infrastructure for Geodynamics

Topology of 12 Caps



N: North Pole S: South Pole

Computational Infrastructure for Geodynamics

Solver

Physics

- Solve the viscous flow within a bounded, rigid mesh
  - regional mesh is bounded by all sides
  - global mesh is bounded by top/bottom
- The mesh is Eulerian, static, non-deforming
- Flow response is instantaneous
  - any local change will impact global flow

Solver

- Primary variables: V, P, T, C
- Velocity and pressure solver, requires ρ and η
  - ρ is a function of T, C, P
  - η can be a function of T, P, C, V
- Temperature solver, requires ρ, C_p, and κ
  - C_p and κ are constant in the code
- Composition solver, ratio tracer method

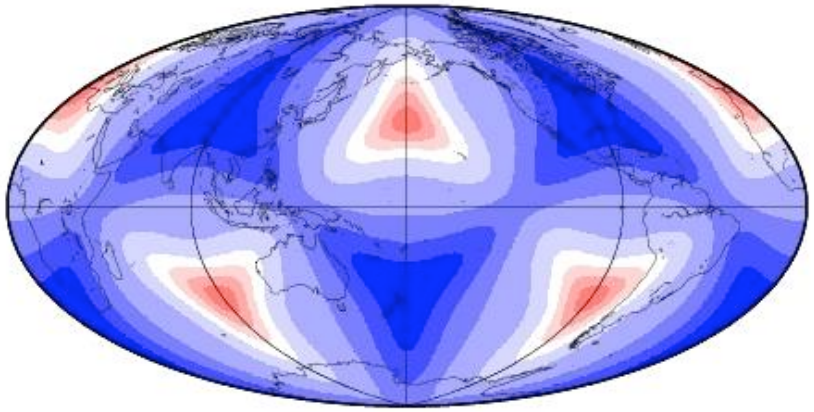
Timestepping

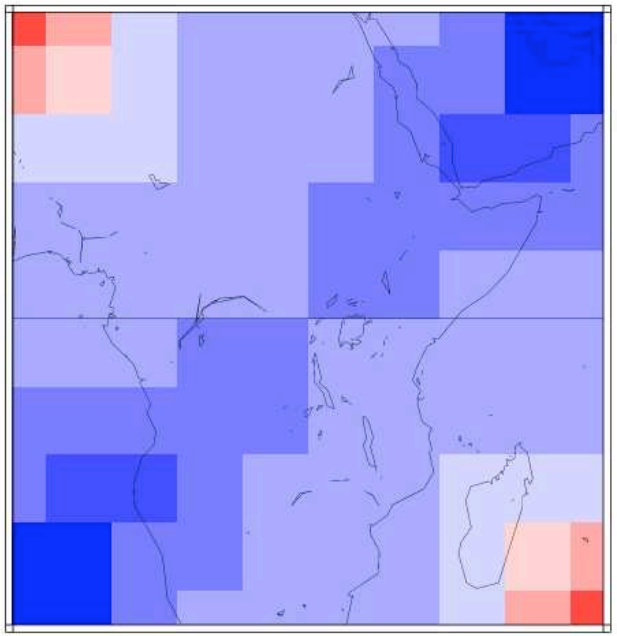
- Given V_{i-1} at t_{i-1}, advect T_{i-1} and C_{i-1} to T_i and C_i
- Compute ρ_i, according to T_i and C_i
- Compute η_i, according to T_i and C_i
- Compute V_i, according to ρ_i and η_i

Visualization in GMT

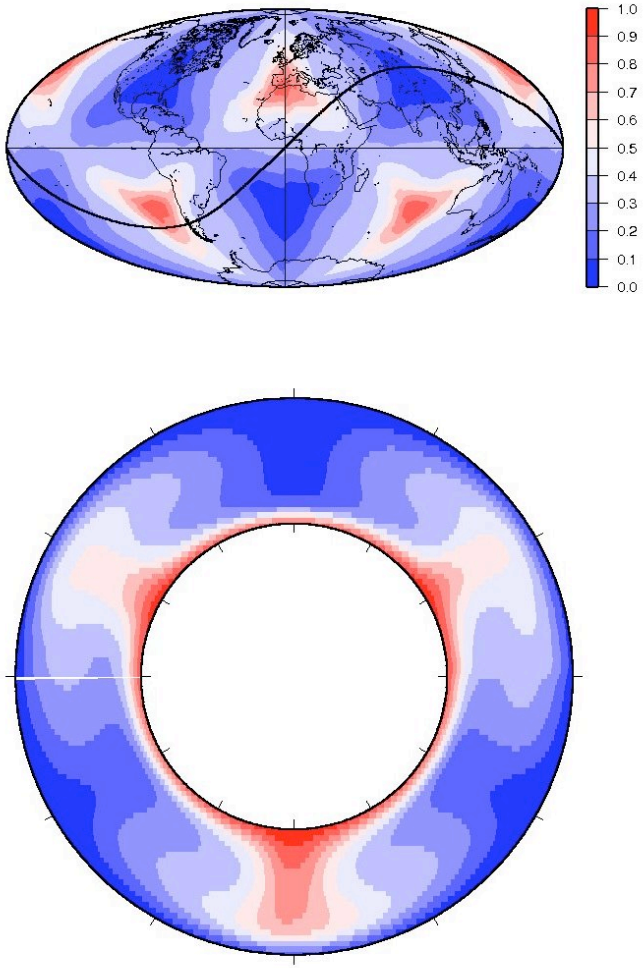
- plot_layer.py
  - plot horizontal cross section, for both regional and global versions
- plot_annulus.py
  - plot radial cross section, for global version only
- sample data files in *visual/samples/*

plot_layer.py

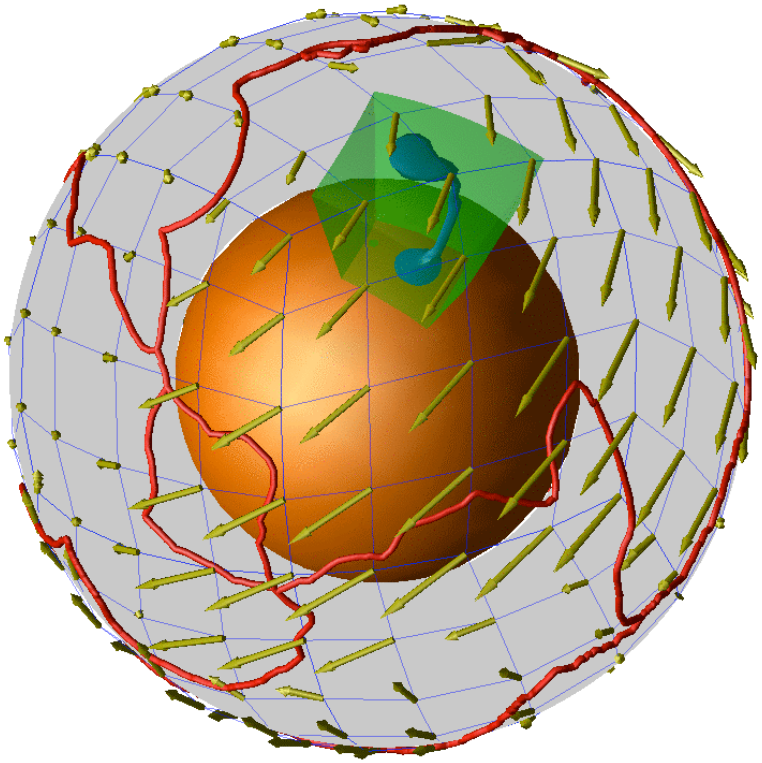


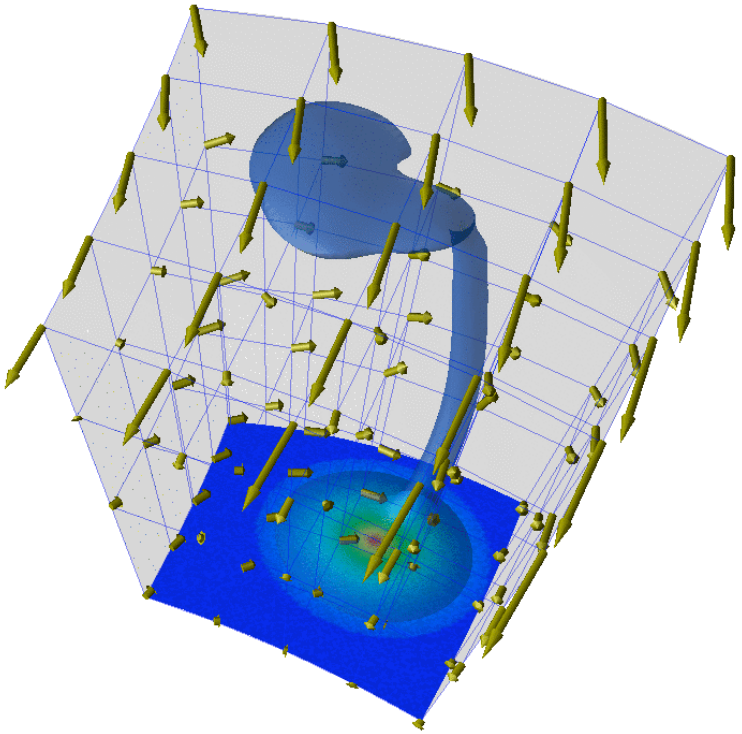


plot_annulus.py



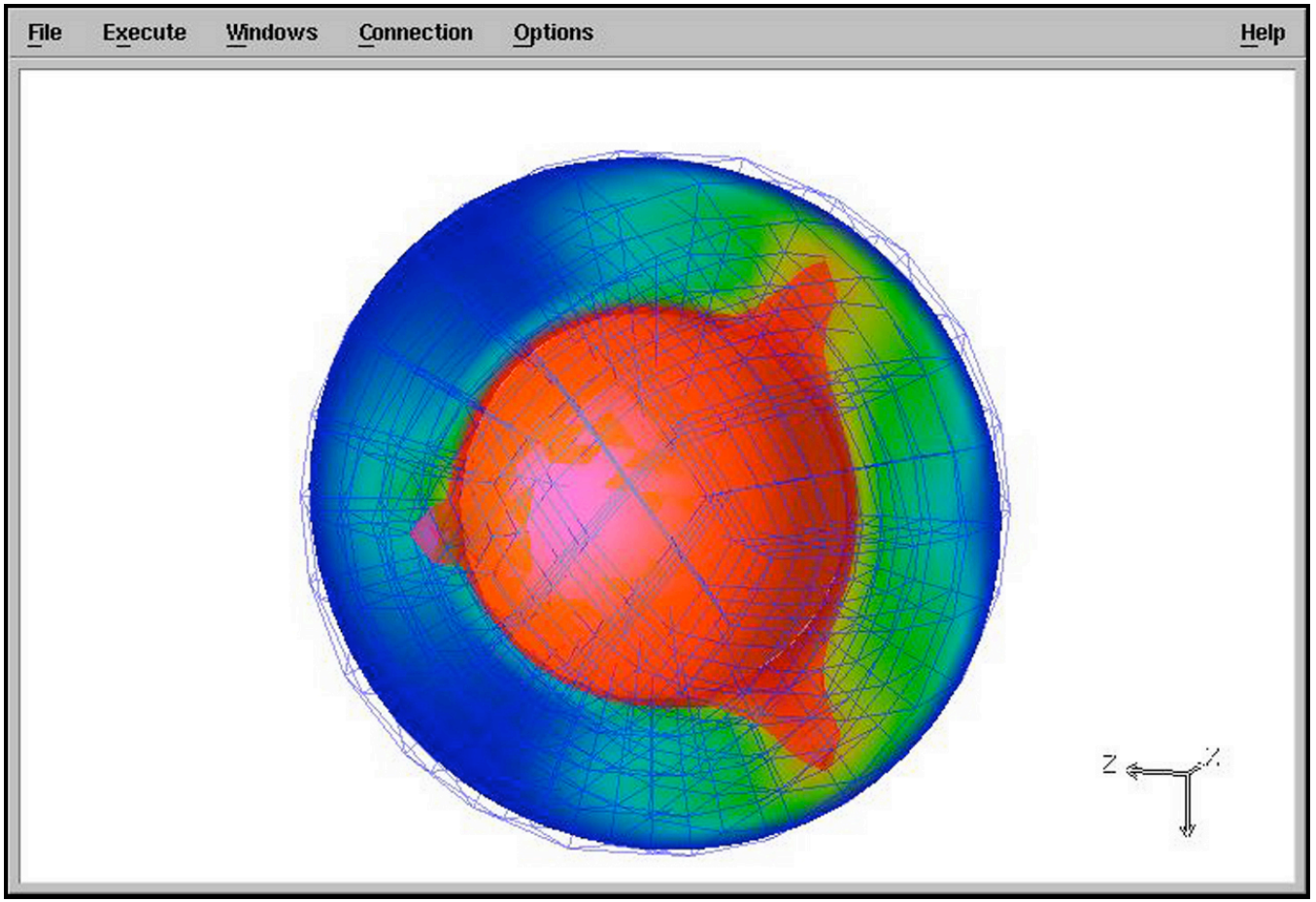
Visualization in OpenDX



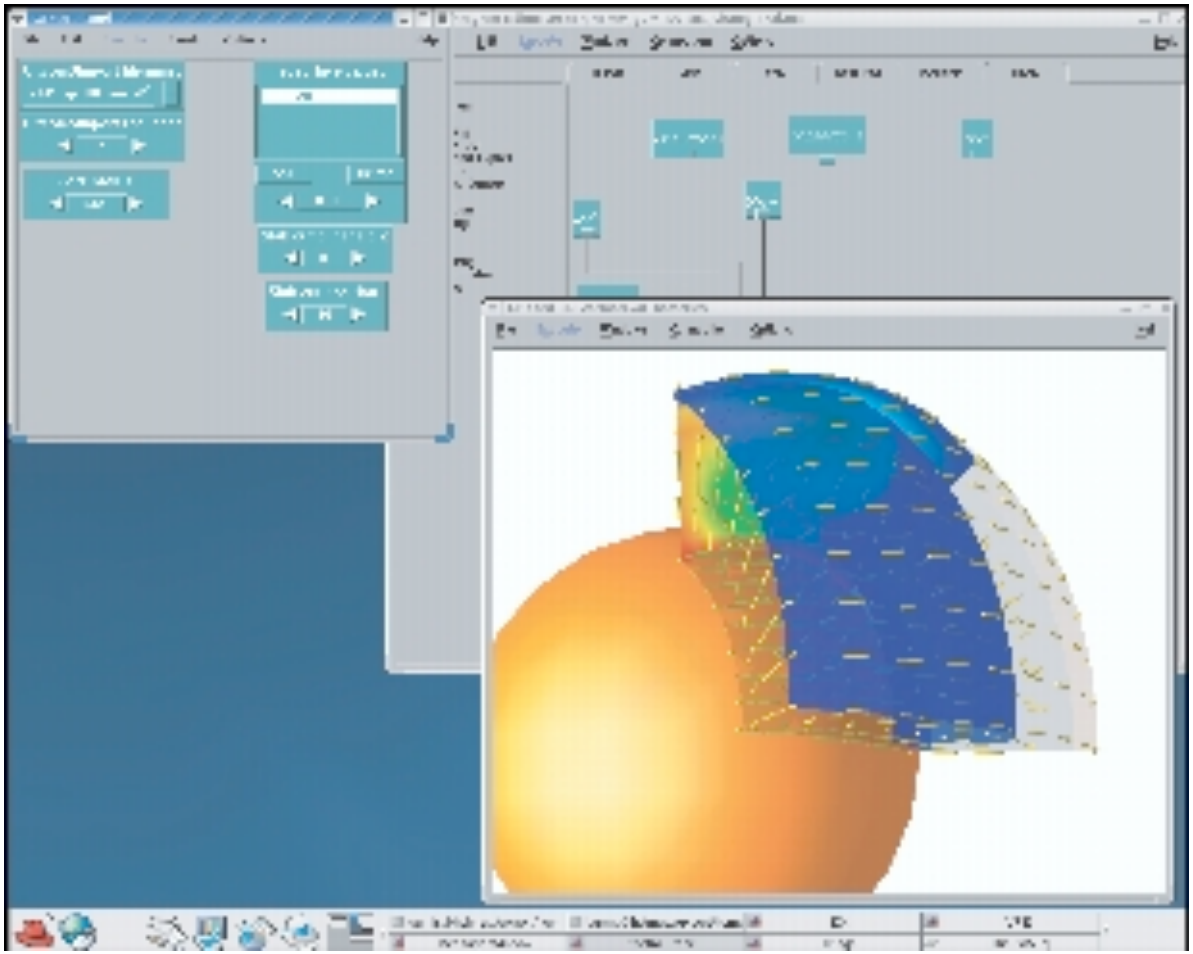


Cookbooks

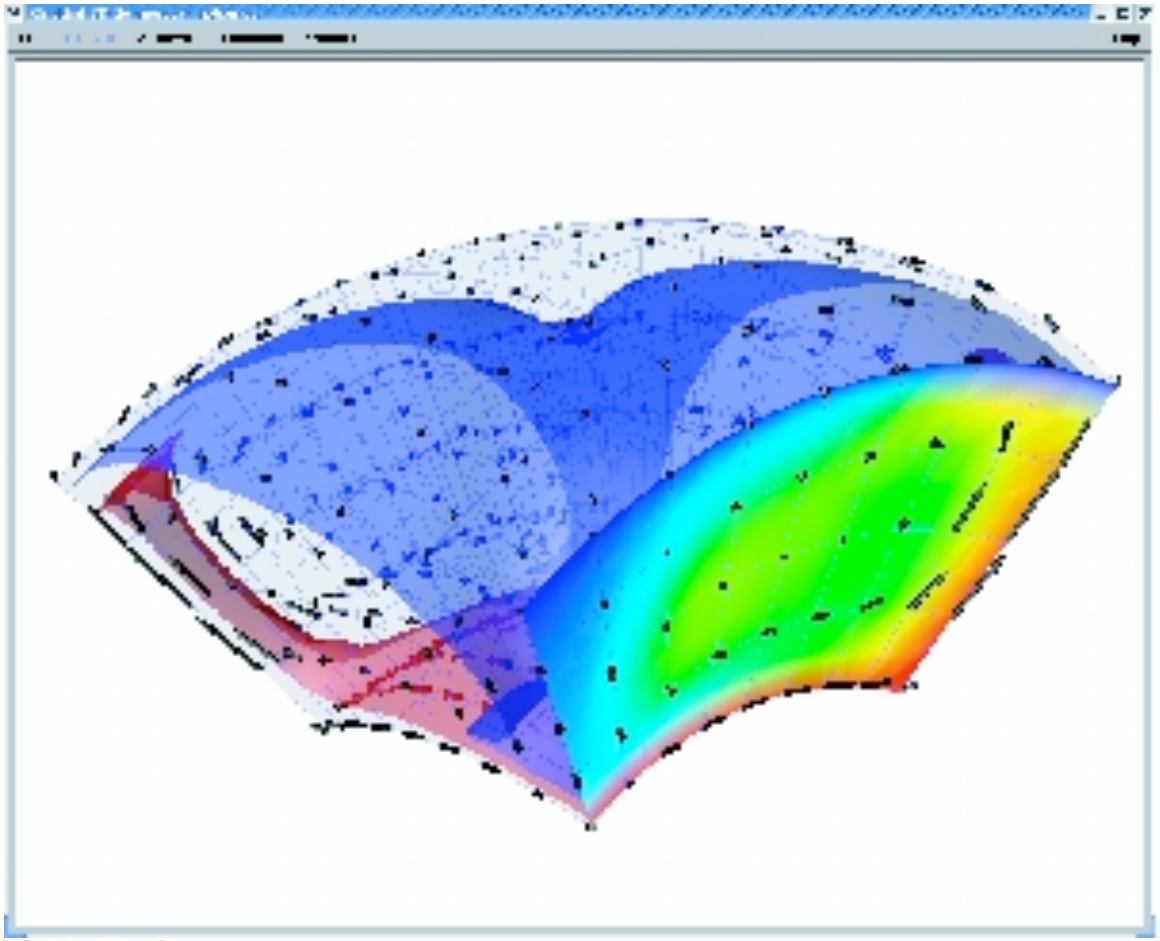
Cookbook 1 - Global Model



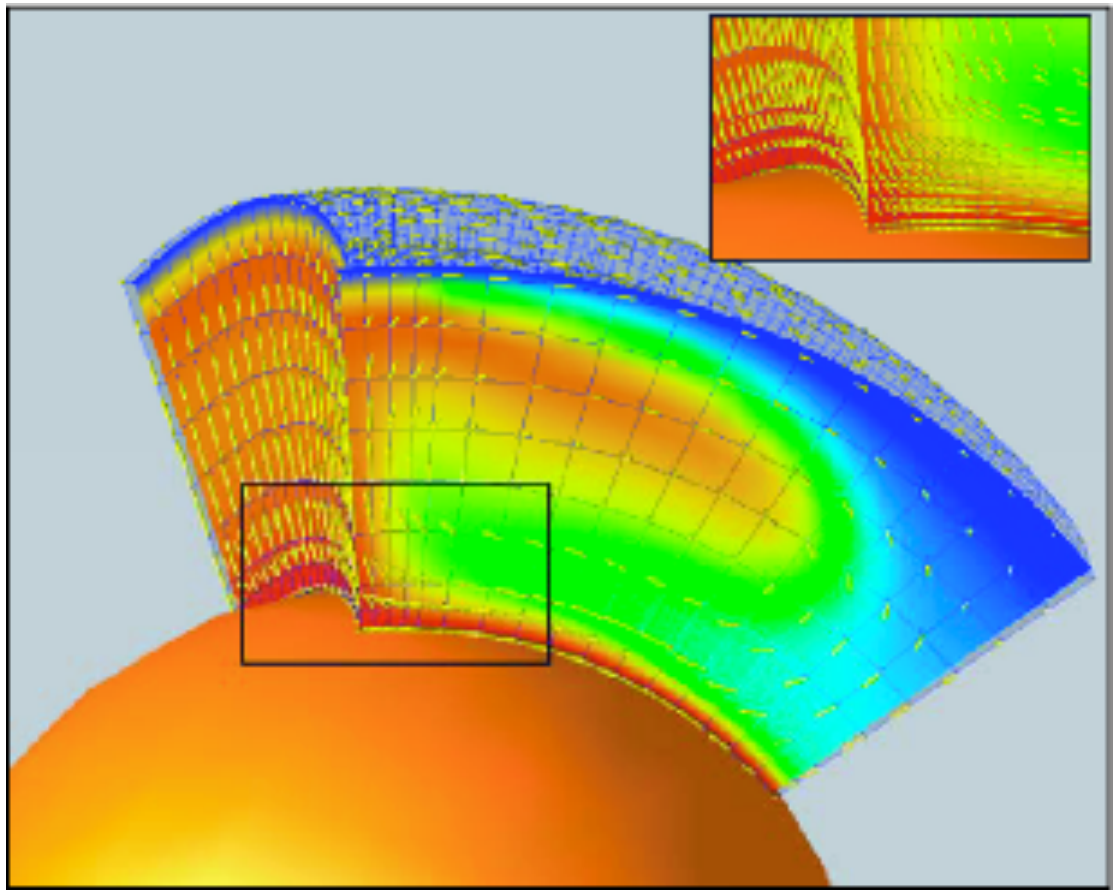
Cookbook 2 - Uniform Velocity Boundary Conditions



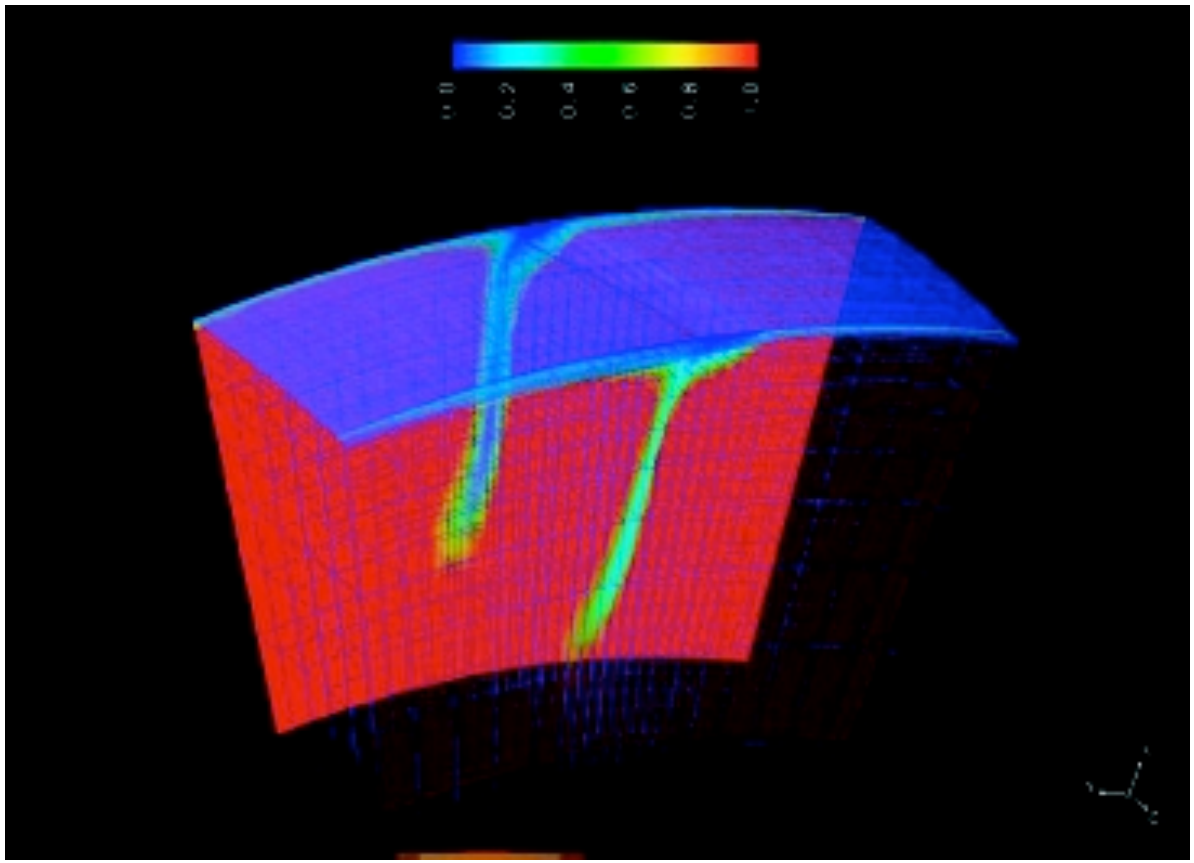
Cookbook 3 - Temperature-Dependent Viscosity



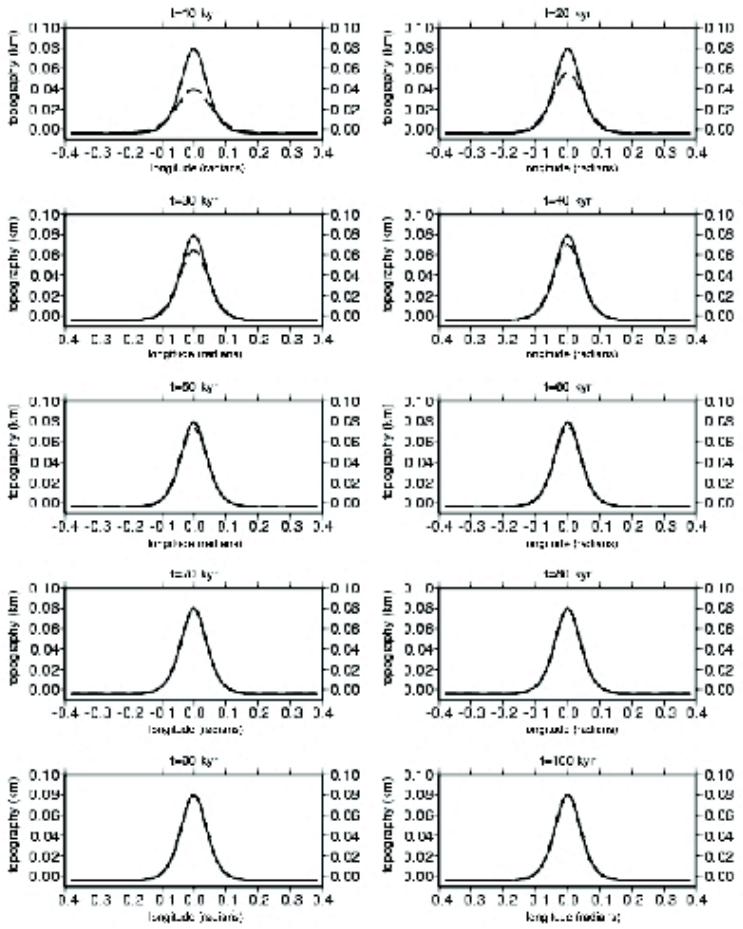
Cookbook 4 - Mesh Refinement



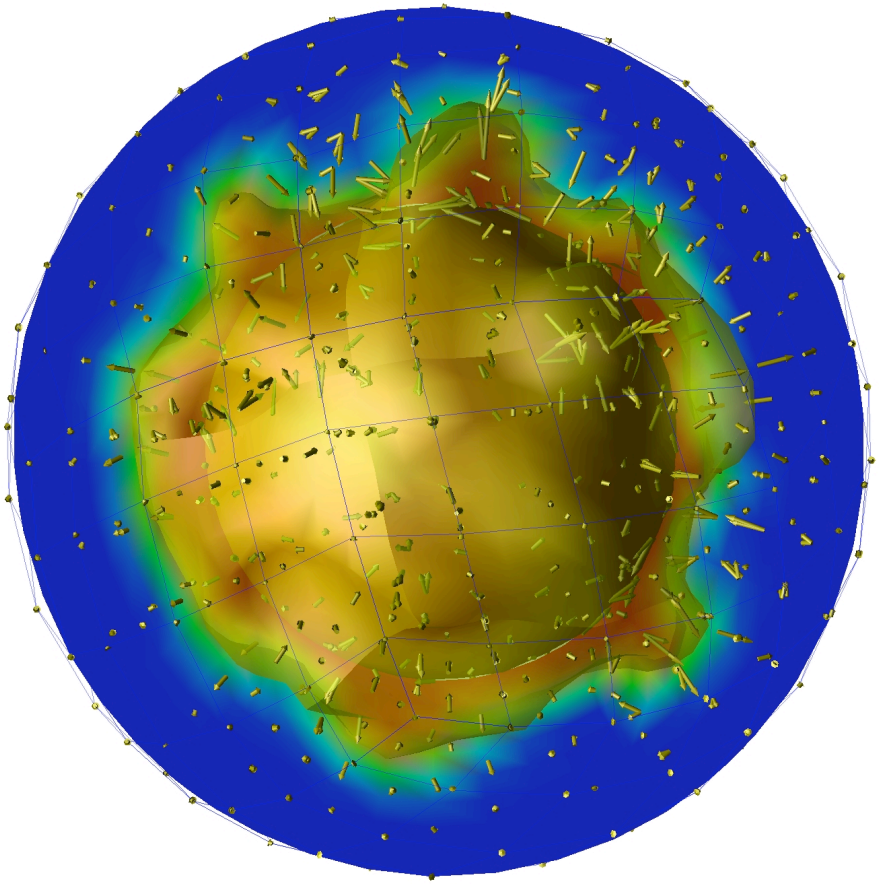
Cookbook 5 - Time-Dependent Velocity Boundary Conditions



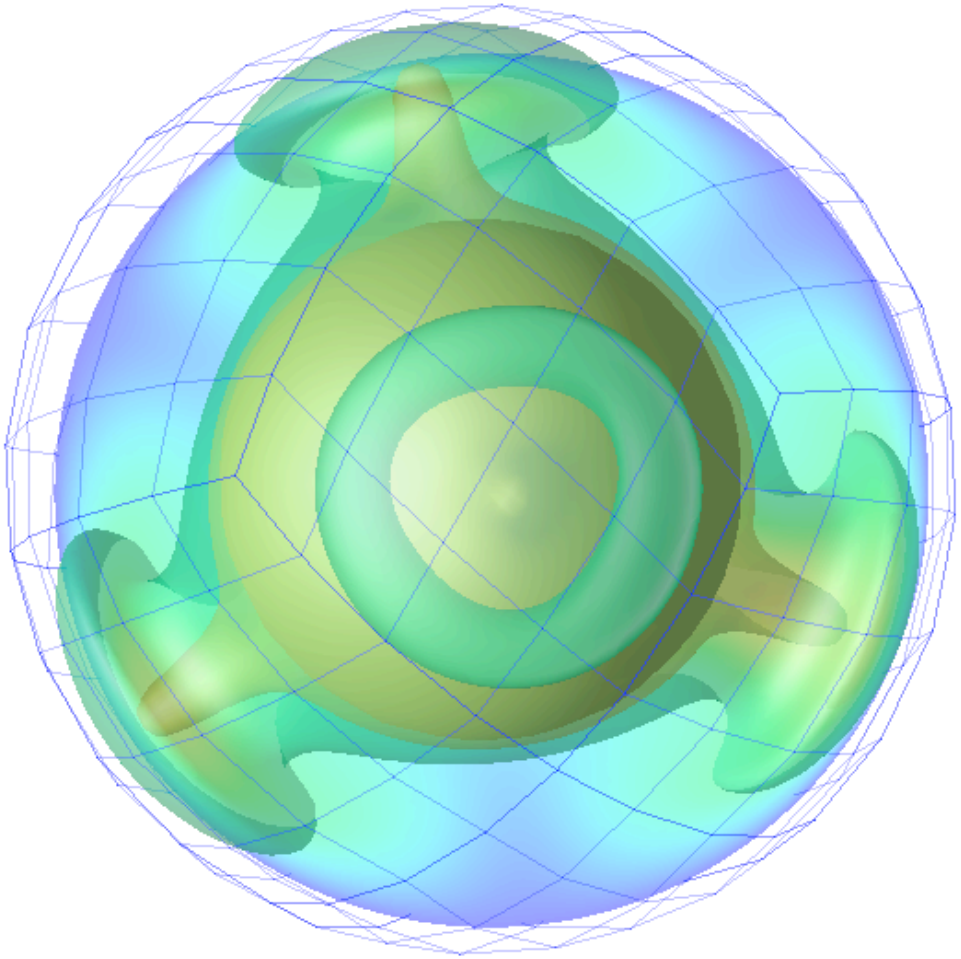
Cookbook 6 - Pseudo-Free-Surface Topography



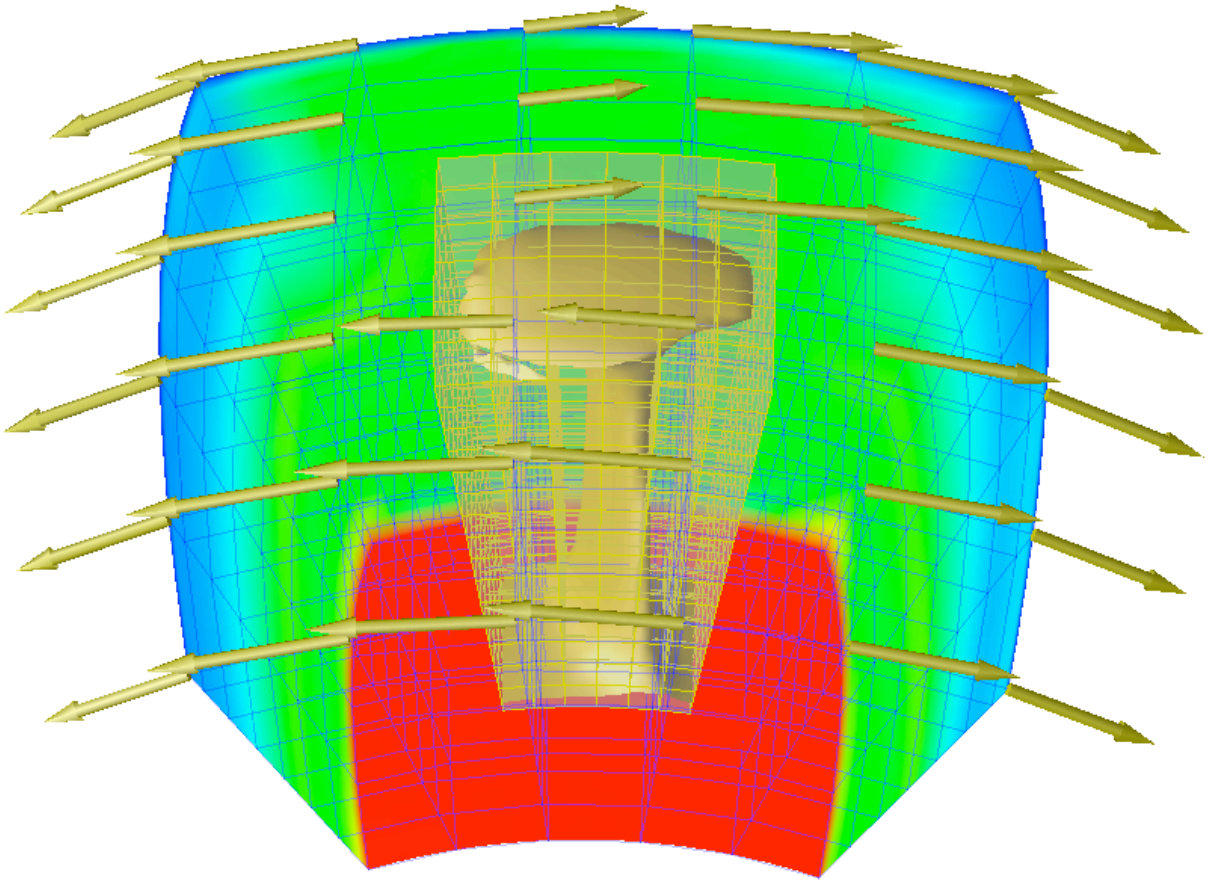
Cookbook 7 - Thermo-Chemical Convection



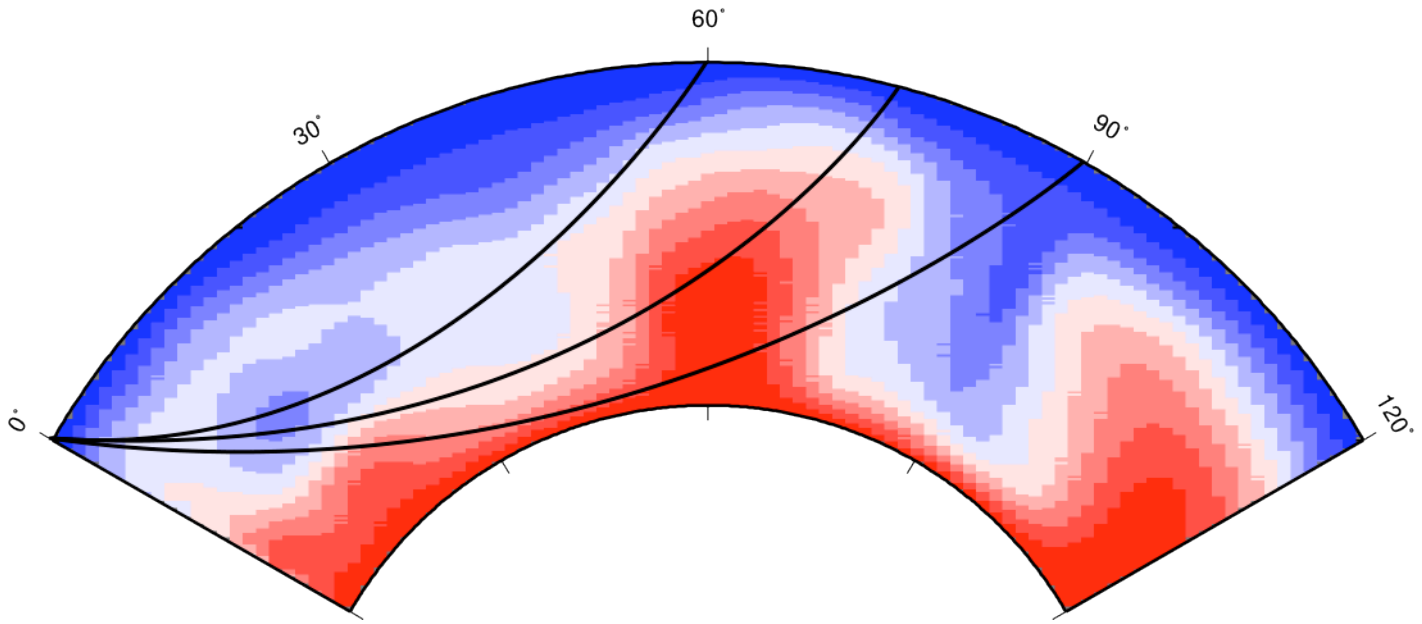
Cookbook 8 - Compressible Convection



Cookbook 9 - Nested Solver Coupling



Cookbook 10 - Synthetic Seismograms from Mantle Convection Models



Cookbook 10

- Converting convection models to seismic velocities

$$V = a \times \left(T - \overline{T}(r)\right) + b \times \left(C - \overline{C}(r)\right)$$

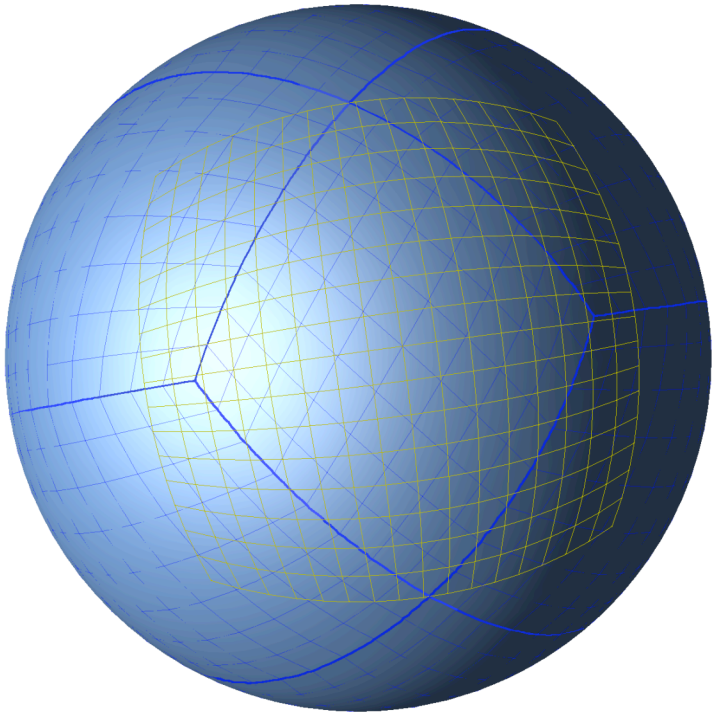
- The values of a and b are taken from Trampert et al. [2003].
- Modify lib/Mineral_physics_model.c for your need

Cookbook 10

- Running synthetics from CitcomS velocity model on CIG Portal
- A custom Earth model which reads CitcomS velocity model and interpolates it to SPECFEM3D grid
- Read Cookbook 10 in CitcomS manual for step-by-step instructions

Cookbook 10

- CitcomS vs. SPECFEM3D
- Different grids
- Different domain partitions

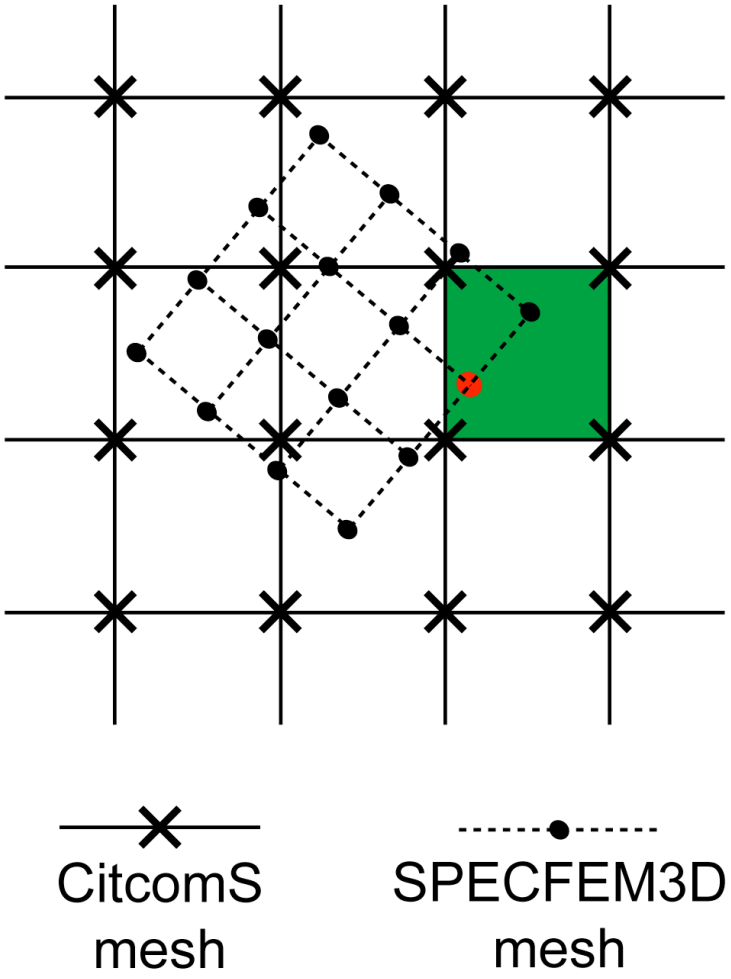


Blue: CitcomS grid

Yello: SPECFEM3D grid

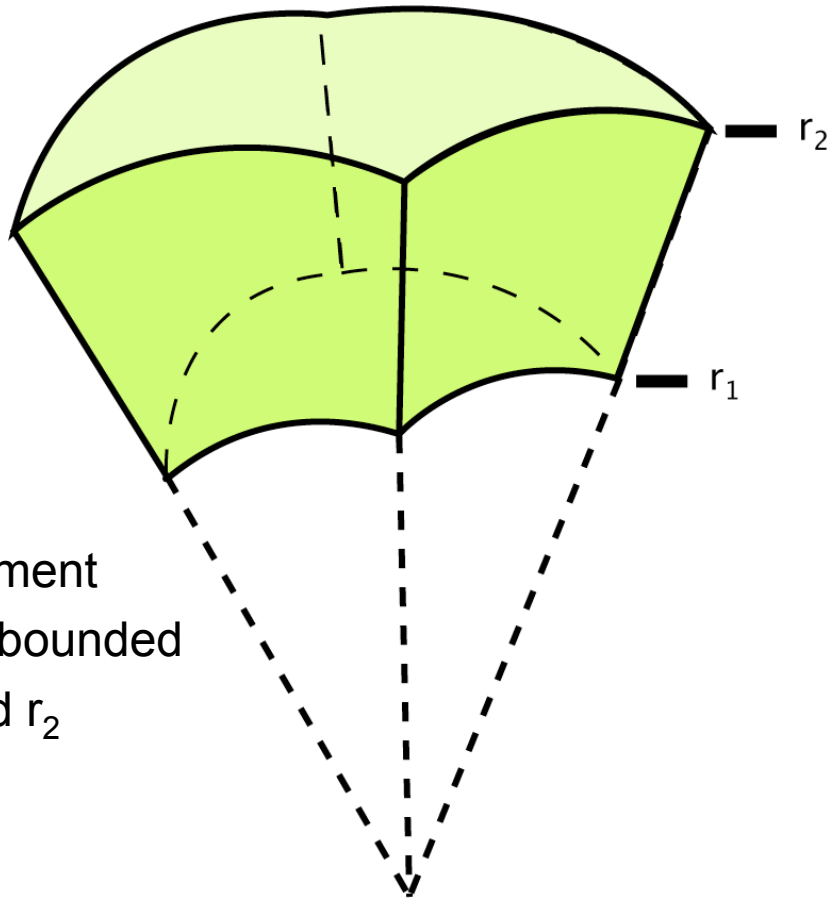
Cookbook 10

- Interpolation



Cookbook 10

- Interpolation



a CitcomS element
is a “pyramid” bounded
between r_1 and r_2

Getting Help

- Contact me
  tan2@geodynamics.org
- Post on the mailing list
  cig-mc@geodynamics.org
- Report a bug
  http://www.geodynamics.org/bugs

Installing CitcomS

Installation

- Required libraries
  - MPI library
  - Python 2.3 or higher (2.4 if using 64-bit machine), including header files
- Visualization packages
  - OpenDX
  - GMT

Networkless Installation

- In the USB drives, Earthscope09/ directory
  - untar CitcomS-3.1.0a.tar.gz to your disk
  - ./configure
  - make

Troubleshooting 1

- Problem 1: python is too old
- Solution: install newer python from your OS vendor
  - sudo apt-get install python2.5 python2.5-dev
  - sudo yum install python2.5 python2.5-devel
  - ./configure PYTHON=/usr/bin/python2.5

Troubleshooting 2

- Problem 2: Python.h not found
- Solution: install the *python-dev* or *python-devel* package from your OS vendor

Troubleshooting 3

- Problem 3: no MPI library
- Solution: install MPI
  - sudo apt-get install openmpi openmpi-dev
  - sudo yum install openmpi openmpi-libs openmpi-devel

Configuring CitcomS

- ./configure [options]
  - Options include:
    - --without-pyre: if you don't have python or don't like to use Pyre framework (not recommended)
    - CC=*your_c_compiler*
    - CFLAGS=*compiler_options* (eg: CFLAGS=-g for a debugging build)
    - LDFLAGS=*linker_options* (eg: LDFLAGS=-L/opt/mpich/lib for non-standard MPI library location)
  - Try to configure without any options first

Your Configuration

- If ./configure finished without problems, you should see a summary:

```
================ Configuration Summary ================
         PYTHON:  /usr/bin/python
         PYTHONPATH:
         CC:  mpicc
         CFLAGS:  -g -O2
         CPPFLAGS:
         LDFLAGS:
         with-pyre:  yes
         with-hdf5:  no
```

- If ./configure finished with errors, see config.log for the detailed error information

Building CitcomS

- Compile the code
  - make
- bin/citcoms -- main program
- visual/autocombine.py -- combine results

Visualization Packages

- OpenDX (Data Explorer)
  - install the *dx* package from your OS vendor
- GMT (Generic Mapping Tools)
  - isn't it already on your system?
  - http://gmt.soest.hawaii.edu

Running CitcomS

Running serial jobs

- bin/citcoms examples/example0.cfg
  - simple 1-processor job
  - confirm your installation is working

Basic Format of the Input File

```
[CitcomS]
steps = 5

[CitcomS.controller]
monitoringFrequency = 1

[CitcomS.solver]
datafile = example0

[CitcomS.solver.mesher]
nodex  = 17
nodey  = 17
```

```
[CitcomS.section.subsection]
parameter = value
```

```
[CitcomS]
steps = 5

[CitcomS.controller]
monitoringFrequency = 1

[CitcomS.solver]
datafile = example0

[CitcomS.solver.mesher]
nodex  = 17
nodey  = 17
```

# of time step

interval of output

prefix of the output filename

# of grid points

Changing Parameters

- from the system default file
  - ~/.pyre/CitcomS/CitcomS.cfg
  - useful for system-wide configuration, eg: configuration for your cluster
- from the input file
  - `bin/citcoms common.cfg case1.cfg`
  - `bin/citcoms common.cfg case2.cfg`
- from the command line
  - `bin/citcoms --section.subsection.parameter=value`

Screen Output

```
[tan2@ifox tracer]$ bin/citcoms examples/example0.cfg
Problem has 17 x 9 x 17 nodes per cap, 2601 nodes and 2048
elements in total
...
Initial temperature perturbation:  layer=5  mag=0.05  l=1
m=1
Momentum equation force 5.784737881e+00
(000)   0.1 s v=3.623420e+00 p=0.000000e+00 div/v=4.82e+00
dv/v=1.00e+00 dp/p=1.00e+00 step 0
(001)   0.1 s v=3.179374e+00 p=3.822932e+01 div/v=1.34e+00
dv/v=5.32e-01 dp/p=1.00e+00 step 0
(002)   0.2 s v=3.161851e+00 p=3.938087e+01 div/v=3.07e-01
dv/v=7.62e-02 dp/p=2.47e-01 step 0
...
```

accuracy of continuity eqn

relative size of velocity correction

relative size of pressure correction

```
[tan2@ifox tracer]$ bin/citcoms examples/example0.cfg
Problem has 17 x 9 x 17 nodes per cap, 2601 nodes and 2048
elements in total
...
Initial temperature perturbation:  layer=5  mag=0.05  l=1
m=1
Momentum equation force 5.784737881e+00
(000)    0.1 s v=3.623420e+00 p=0.000000e+00 div/v=4.82e+00
dv/v=1.00e+00 dp/p=1.00e+00 step 0
(001)    0.1 s v=3.179374e+00 p=3.822932e+01 div/v=1.34e+00
dv/v=5.32e-01 dp/p=1.00e+00 step 0
(002)    0.2 s v=3.161851e+00 p=3.938087e+01 div/v=3.07e-01
dv/v=7.62e-02 dp/p=2.47e-01 step 0
...
```

Accuracy Settings

- Parameters affecting the accuracy of the velocity solver
  - under [CitcomS.solver.vsolver]
  - *accuracy=1e-4*
  - *piterations=375*
- The default values are conservative

Running parallel jobs

- bin/citcoms examples/example1.cfg
  - simple 4-processor job
  - confirm your parallelism setup is correct

```
[CitcomS]
steps = 70

[CitcomS.controller]
monitoringFrequency = 10

[CitcomS.solver]
datafile = example1

[CitcomS.solver.mesher]
nprocx =  2
nprocy =  2
nodex  = 17
nodey  = 17
nodez  =  9
```

# of processors

Launching Parallel Jobs

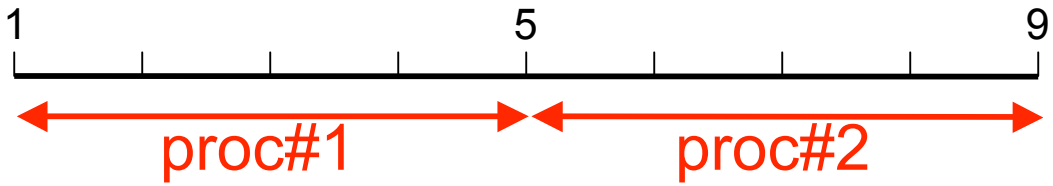
- Case 1: a cluster without scheduler
  - under [CitcomS.launcher]
  - *nodegen*: printf-style string to generate the machine hostnames
  - *nodelist*: comma-seperated list of machine name
- Ex: nodegen=m%03d; nodelist=[1,3-5]
  ==> m001, m003, m004, m005
- Try putting *nodegen* in a system default file (~/.pyre/CitcomS/CitcomS.cfg)

Launching Parallel Jobs

- Case 2: a cluster with scheduler
  - supported schedulers: LSF, PBS, Globus, SGE
  - case-by-case
- Case 3: single workstation
  - similar to serial jobs

Parallel Partitioning

- Within each cap, the nodes are logically Cartesian
- Number of nodes within each cap specified by (*nodex* * *nodey* * *nodez*)
- Number of processors within each cap specified by (*nprocx* * *nprocy* * *nprocz*)
- Each processor has the same number of nodes.
  - x-direction: ((*nodex* - 1) / *nprocx*) + 1



Ordering of Nodes

- In a regional mesh, the nodes are ordered by ϕ-θ-r
  - r-direction increases first, then θ-direction, then ϕ direction
- In a global mesh, the nodes are ordered in a similar way
  - the grid lines are not parallel to longitude nor latitude, so θ and ϕ both changes

Basic Troubleshooting

- The error message is usually helpful, but could be hard to find…
- Several places to look for error messages
  - Look at the screen output
  - Look at the end of the log file: *datafile*.log
  - If using tracers, look at the end of all tracer log files: *datafile*.tracer_log.*rank*

Understanding the Output

- Output format can be either ASCII, Compressed ASCII or HDF5 (will always use ASCII in this tutorial)
- Output directory specified by *datadir*
- Special strings in *datadir*
  - *%HOSTNAME*: replaced by the hostname of the computer
  - *%RANK*: replaced by the MPI rank of the computer
  - *%DATADIR*: replaced by the returning string of a script

Output Files

- Plenty of output files
- Each has a name like: *datafile*.xxxx.*rank*.*step*
  - *datafile*: prefix
  - *rank*: MPI rank
  - *step*: time step, the interval is controlled by the parameter "controller.monitoringFrequency"
- Coordinate output: *datafile*.coord.*rank*
- Timing output: *datafile*.time
- Formats of the output files can be found in the manual (Appendix C)

Output Files

- *datafile*.xxxx.*rank*.*step*, xxxx can be:
  - velo: velocity and temperature
  - visc: viscosity
  - surf/botm: topography and heatflux
  - stress: deviatoric stress tensor
  - pressure: pressure
  - geoid: geoid (in spherical harmonics coefficients)
  - tracer: tracer information
  - comp_el/comp_nd: composition (on elements or nodes)
  - horiz_avg: radially averaged profile

Output Optional Data

- Except coord, velo, and visc, other data output can be turned on/off
- surf and botm data are output by default (but can be turned off)
- Eg: this turned on pressure and stress, but turned off surf and bottm

```
[CitcomS.solver.output]
output_optional = pressure,stress
```

Combining the Data

- Plenty of data files, scattered over the cluster nodes
- Combining the data into one file per cap per time step
- visual/autocombine.py
  - see the usage by running autocombine.py without any options

Autocombine.py

- usage: *autocombine.py machinefile inputfile step1 [step2 [...]]*
  - *machinefile*: filename of the MPI machinefile, or *localhost* if the data are in a local file system
  - *inputfile*: the original input file, or the pidfile (*pid12345.cfg*)

Combined Results

- Each cap has two files, eg:
  - prefix.cap05.100, this is the data file
  - prefix.cap05.100.general, this is the header file for OpenDX