PyLith Troubleshooting Tips/Tricks

Brad Aagaard
Charles Williams
Matt Knepley

May 16, 2011

Outline

- General numerical modeling tips
- Mesh generation
- Running PyLith

General Numerical Modeling Tips

Start simple and progressively add complexity and increase resolution

- **Start in 2-D, if possible, and then go to 3-D**
  - Much smaller problems $\Rightarrow$ much faster turnaround
  - Experiment with meshing, boundary conditions, solvers, etc
  - Keep in mind how physics differs from 3-D
- **Start with coarse resolution and then increase resolution**
  - Much smaller problems $\Rightarrow$ much faster turnaround
  - Experiment with meshing, boundary conditions, solvers, etc.
  - Increase resolution until solution resolves features of interest
    - Resolution will depend on spatial scales in BC, initial conditions, deformation, and geologic structure
    - Is geometry of domain important? At what resolution?
    - Displacement field is integral of strains/stresses
    - Resolving stresses/strains requires fine resolution simulations
- **Use your intuition and analogous solutions to check your results!**

Mesh Generation Tips

There is no silver bullet in finite-element mesh generation

- Hex/Quad versus Tet/Tri
  - Hex/Quad are slightly more accurate and faster
  - Tet/Tri easily handle complex geometry
  - Easy to vary discretization size with Tet, Tri, and Quad cells
  - There is no easy answer
    For a given accuracy, a finer resolution Tet mesh that varies the discretization size in a more optimal way *might* run faster than a Hex mesh
- Check and double-check your mesh
  - Were there any errors when running the mesher?
  - Do all of the nodesets and blocks look correct?
  - Check mesh quality (aspect ratio should be close to 1)
- CUBIT
  - Name objects and use APREPRO or Python for robust scripts
  - Number of points in spline curves/surfaces has huge affect on mesh generation runtime

PyLith Tips

- **Read the PyLith User Manual**
- **Do not ignore error messages and warnings!**
- Use an example/benchmark as a starting point
- Quasi-static simulations
  - Start with a static simulation and then add time dependence
  - **Check that the solution converges at every time step**
- Dynamic simulations
  - Start with a static simulation
  - **Shortest wavelength seismic waves control cell size**
- CIG Short-Term Crustal Dynamics mailing list

  `cig-short@geodynamics.org`
- Short-Term Crustal Dynamics wiki (under construction)
- CIG bug tracking system

  `http://www.geodynamics.org/roundup`

PyLith Debugging Tools

- `pylithinfo [--verbose] [PyLith args]`
  Dumps all parameters with their current values to text file
- Command line arguments
  - `--help`
  - `--help-components`
  - `--help-properties`
  - `--petsc.start_in_debugger` (run in xterm)
  - `--nodes=N` (to run on N processors on local machine)
- Journal info flags turn on writing progress/inf

```
[pylithapp.journal.info]
timedependent = 1
```

  - Turns on/off info for each type of component independently
  - Examples turn on writing lots of info to stdout using journal flags