

PyLith Modeling Tutorial

Computing Static Green's Functions

Charles Williams
Brad Aagaard
Matthew Knepley



June 17, 2016

Concepts Covered in this Session

- Generation of a simple graded mesh in 2D (optional)
- Generation of Green's functions in 2D and 3D
- Solution output at a specified set of points (`OutputSolnPoints`)
- Postprocessing of HDF5 output using `h5py`
- Simple linear inversion using `numpy`
- Plotting of inversion results using `matplotlib`

- Compute deformation due to unit (i.e., 1 m) slip at fault vertex for use in an inversion for fault slip
 - Slip decreases **linearly** to 0 at surrounding vertices
 - Similar but not equivalent to uniform slip over a patch (Okada dislocation)
- Provides ability to compute Green's functions with arbitrarily complex elastic structure

Green's Functions Examples

- 2-D examples: [examples/2d/greensfns](#)
 - Example components
 - 1 Compute synthetic (fake) observations for an earthquake
 - 2 Compute displacements at sites for Green's functions
 - 3 Invert for fault slip
 - See Section 7.15 of the PyLith User Manual
- 3-D example: [examples/3d/hex8/step21](#)
 - Limited to computing displacements at sites for Green's functions
 - No inversion

2-D Green's Functions Example

Invert for slip on reverse fault

Files are in `examples/2d/greensfns/reverse`

- 1 Generate mesh using CUBIT
- 2 Compute synthetic (fake) observations for an earthquake
`pylith eqsim.cfg`
- 3 Compute displacements at sites for Green's functions
`pylith --problem=pylith.problems.GreensFns`
- 4 Invert for fault slip
See README in `examples/2d/greensfns`
- 5 Visualize inversion results using matplotlib Python package
See README in `examples/2d/greensfns`

Python Packages Needed

numpy Arrays plus scientific computing tools in Python

- Similar to numerical functions in Matlab
- Included in PyLith binary distribution

h5py Python wrappers for HDF5 library

- Provides high-level access to HDF5 files
- Included in PyLith binary distribution

matplotlib 2-D plotting for Python

- Designed to be very similar to Matlab
- **Not** included in PyLith binary distribution
- Available from matplotlib.org

Simple Linear Inversion

Parameters

G Green's function matrix

d Unknown fault slip

$d_{apriori}$ A priori estimate of fault slip

u_{obs} Observed displacement

D Penalty matrix

θ Penalty parameter

The matrix G_{ij} gives displacement component i due to a unit of slip from component j .

Simple Linear Inversion

Equations

- Original system of equations:

$$Gd = u_{obs} \quad (1)$$

- Augmented system of equations:

$$G_a d = u_a, \text{ where } G_a = \begin{bmatrix} G \\ \theta D \end{bmatrix} \text{ and } u_a = \begin{bmatrix} u_{obs} \\ d_{apriori} \end{bmatrix} \quad (2)$$

- Generalized inverse:

$$G^{-g} = \left(G_a^T G_a \right)^{-1} G_a^T \quad (3)$$

$$d_{est} = G^{-g} u_a \quad (4)$$

3-D Green's Functions Example

Demonstrate computing Green's functions; no inversion

Files are in `examples/3d/hex8`

- Compute responses due to strike-slip and reverse slip separately
- Parameters are distributed across multiple `.cfg` files
 - `pylithapp.cfg` General parameters for this mesh
 - `greensfns.cfg` General Green's function problem settings
 - `step21.cfg` Parameters specific to this example

```
pylith --problem=pylith.problems.GreensFns step21.cfg
```