PyLith 1.3: A Finite-Element Code for Modeling Quasi-Static and Dynamic Crustal Deformation

Brad Aagaard baagaard@usgs.gov
Charles Williams willic3@rpi.edu
Matthew Knepley knepley@mcs.anl.gov
Leif Strand leif@geodynamics.org
Sue Kientz sue@geodynamics.org

Objectives

Create flexible, computationally efficient software for simulation of crustal deformation across spatial scales ranging from meters to hundreds of kilometers and temporal scales ranging from milliseconds to hundreds of years.

- **Modular**: Users can swap modules to run the problem of interest
- **Scalable**: Code runs on one to a thousand processors efficiently
- **Extensible**: Expert users can add functionality to solve their problem without polluting main code

Motivation

- Most available modeling codes
  - rarely solve the problem **you** want to solve
  - are often poorly documented
  - may not work correctly
- Current research demands larger, more complex simulations
- Want to avoid multiple, incompatible versions of the same code

Current Release: 1.3.0 (Aug 30, 2008)

PyLith 1.3 uses 20-30% less memory and is 25-30% faster than PyLith 1.1.

Features

- Cell types include triangles, quadrilaterals, hexahedra, and tetrahedra
- Kinematic fault interfaces using cohesive cells
- Multiple, potentially overlapping earthquake ruptures and aseismic creep
- Dirichlet (displacement and velocity) boundary conditions
- Neumann (traction) boundary conditions
- Absorbing boundary conditions
- Gravitational body forces
- Linear elastic, linear and generalized Maxwell viscoelastic materials
- Quasi-static and dynamic time-stepping
- Automatic or user-controlled time-stepping for quasi-static simulations
- Output of displacements, fault information, and state variables

Planned releases

- Release 1.4 (Dec 2008)
  Add support for spontaneous earthquake rupture and nonlinear material bulk rheologies.
  - Fault friction interface conditions
  - Include several popular fault constitutive models
  - Add a few popular nonlinear bulk rheologies
- Release 1.5 (Jun 2009)
  Add support for large deformations and more complex time dependent boundary conditions.
- Release 1.6
  Add support for automatic calculation of 4-D Green's functions.
- Release 1.7
  Support simulations coupling quasi-static and dynamic behavior.

Overview

PyLith 1.x combines the quasi-static modeling functionality of PyLith 0.8 and its predecessors (LithoMop and Tecton) and the dynamic modeling functionality of EqSim.

General features

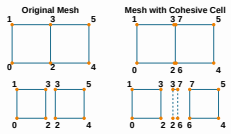
- 1-D, 2-D, and 3-D problems
- Quasi-static and dynamic time-stepping
- Support for several cell (element) types
- Easy specification of boundary conditions using spatial databases
- Fault implementation creates dislocations in mesh
- Seamless importing of meshes from CUBIT and LaGriT mesh generators
- Seamless use of SCEC CVM-H and USGS Bay Area seismic velocity models for elastic material properties
- Writes output using VTK files for seamless import into ParaView and MayaVi visualization tools

Target applications

- Pre- and post-seismic deformation with viscoelastic rheologies
- Ground motion simulations with kinematic or spontaneous ruptures
- Calculation of 3-D Green's functions
- Simulations of multiple earthquake cycles

Fault Implementation

We modify the topology of the finite-element mesh, inserting cohesive cells on the fault surface.



- System of equations without cohesive cells

$$\underline{A}\vec{u} = \vec{b}$$

- Use cohesive cells with Lagrange multipliers and conditioning to constrain slip

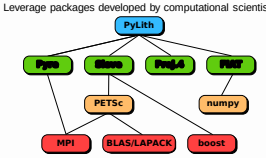
$$\begin{pmatrix} \underline{A} & \alpha\underline{C}^T \\ \underline{C} & 0 \end{pmatrix} \begin{pmatrix} \vec{u} \\ \frac{1}{\alpha}\vec{L} \end{pmatrix} = \begin{pmatrix} \vec{b} \\ \vec{D} \end{pmatrix}$$

- Advantages
  - Fault implementation is local to cohesive cell
  - Solution includes forces generating slip (Lagrange multipliers)
  - Retains block structure of matrix (same number of DOF per vertex)
  - Offsets in mesh mimic slip on natural faults
- Disadvantages
  - Conditioned matrix is asymmetric
  - Mixes displacements and forces in solution

Software Architecture

- Separate code into modules to encapsulate behavior and facilitate reuse
- Top-level code written in Python
  - Expressive, high-level, object-oriented language
  - Dynamic typing allows adding/replacing modules at runtime
  - Convenient scripting
- Low-level code written in C++
  - Compiled (fast execution), object-oriented language
- Bindings to glue Python & C++ together
  - Pyrex/pyrexembed generate C code for calling C++ from Python

Leverage packages developed by computational scientists



Major external packages

- **Pyre** is a science neutral simulation framework developed at Caltech.
- **Sieve** is a suite of general, parallel data structures for storing and manipulating finite-element meshes.
- **PETSc** is the Portable, Extensible Toolkit for Scientific Computation from the Argonne National Laboratory. It is used to perform operations on matrices and vectors in parallel.
- **Proj.4** is a library for converting between geographic projections.
- **FIAT** generates arbitrary order instances of Lagrange elements on lines, triangles, and tetrahedra.

Example: SCEC Strike-Slip Benchmark

One of the benchmarks in the SCEC Crustal Deformation Modeling benchmark suite.

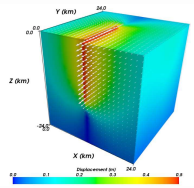
Description

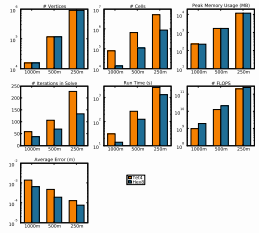
- Viscoelastic (Maxwell) relaxation from a strike-slip earthquake in 3-D without gravity.
- Elastic layer over a viscoelastic layer.
- Fault extends into viscoelastic layer.
- Slip is 1.0 m and tapers linearly over 4 km along buried edges of the fault.
- Displacements on boundaries set to semi-analytic elastic solution.
- **Results shown are for elastic solution only.**

Summary

- Cell types: Linear basis functions for hexahedron (Hex8) and tetrahedron (Tet4)
- Discretization sizes: 1000 m, 500 m, 250 m
- For linear basis functions, Hex8 cells outperform Tet4 cells.
- We can improve performance by switching from Krylov solver to multigrid solver.

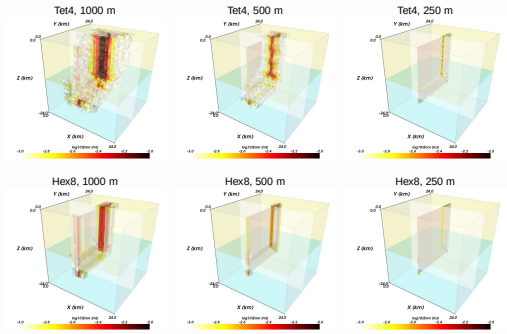
Elastic Solution





Comparison of Local Error

- Error is small and decreases with finer resolution. Solution is converging!
- Greatest error occurs where slip gradient is discontinuous and linear basis functions cannot match slip variation.



Availability

PyLith is open-source and aims to be a community code. It is distributed by CIG at `geodynamics.org`.

- Source code
  - SVN repository (development version)
  - Tarball (releases)
- Binary packages
  - Linux (32-bit)
  - OSX (Intel and PowerPC)
  - Windows (uses cygwin)
- User manual with tutorials

Inputs to PyLith

- Simulation parameters
- Finite-element mesh
  - Mesh exported from LaGriT
  - Mesh exported from CUBIT
  - Mesh constructed by hand (PyLith mesh ASCII format)
- Spatial databases for physical properties, boundary conditions, and rupture parameters
  - SCEC CVM-H, USGS Bay Area Velocity model, or simple ASCII files
  - Independent of discretization scheme and size

Unit and Regression Testing

Automatically run more than 875 tests on multiple platforms whenever code is checked into the source repository.

- Create tests for nearly every function in code during development
  - Remove most bugs during initial implementation
  - Isolate and expose bugs at origin
- Create new tests to expose reported bugs
  - Prevent bugs from reoccurring
- Rerun tests whenever code is changed
  - Code continually improves (permits optimization with quality control)
- Binary packages generated automatically upon successful completion of tests

Role of the CIG

The CIG is an NSF funded membership-governed organization that supports Earth science by developing and maintaining software for computational geophysics.

- PyLith is a fully-supported CIG code. CIG provides the source repository, web site for distribution, mailing lists, bug tracking system, and testing and benchmarking infrastructure.
- CIG provides developer time (Matt Knepley and Leif Strand) and help in writing the documentation (Sue Kientz)
- Development targets the needs of the CIG working groups, especially the Short-Term Crustal Dynamics group.

CIG long-term goals

- Develop reusable, well-documented, open-source geodynamics software
- Infrastructure to allow quick assembly of state-of-the-art modeling codes
- Extend existing software frameworks to interlink multiple codes and data
- Form strategic partnerships with the larger world of computational science
- Provide specialized training and workshops for both the geodynamics community

Interdependence of simulations & data