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

**Objective**
- Code runs on one to a thousand processors efficiently
- Developer: Brad Aagaard
- Code continually improves (permits optimization with quality control)
- Isolate and expose bugs at origin
- Solution includes forces generating slip (Lagrange multipliers)
- System of equations without cohesive cells
- Output of displacements, fault information, and state variables

**Current Release:** 1.3.0 (Aug 30, 2008)

**Features**
- **Cell types** include triangles, quadrilaterals, hexahedra, and tetrahedra
- **Magma** to **Orogenesis**
- **Fault implementation** is local to cohesive cell
- **Deformation** across spatial scales ranging from meters to hundreds of kilometers and temporal scales ranging from milliseconds to hundreds of years

**Planned releases**
- **Release 1.5 (Jun 2009)** Add support for spontaneous earthquake rupture and nonlinear material bulk rheologies
- **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.

**Availability**
- PyLith is open-source and aims to be a community code. It is distributed by CIG.
- Code runs on one to a thousand processors efficiently
- Development targets the needs of the CIG working groups, especially the Short-Term Crustal Dynamics group.
- PyLith is open-source and aims to be a community code.

**Inputs to PyLith**
- **Simulation parameters**
- **Finite-element mesh**
- **Mesh exported from LaGiT**
- **Mesh exported from CUBIT**
- **Mesh constructed by hand (PyLith mesh ASCII format)**
- **Spatial databases for physical properties, boundary conditions, and rupture parameters**
- **JAVA, C++, UGS 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 compiled into the source repository.
- Create bugs for nearly every function in code during development
- Inspect and expose bugs at origin
- Isolate and expose bugs at origin
- Develop new tests to expose reported bugs
- Prevent bugs from recurring
- **Role of the CIG**

**Overview**
- PyLith is a finite-element code for modeling quasistatic and dynamic deformations across spatial scales ranging from meters to kilometers and temporal scales ranging from milliseconds to centuries. It is designed to support research in earth science by developing and maintaining software for computational geodynamics.
- 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 (Leif Strand and Sue Kienzle) and help in writing the documentation (Sue Kienzle)
- Development targets the needs of the CIG working groups, especially the Short-Term Crustal Dynamics group.

**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/removing modules at runtime
- Conventient scripting
- Low-level code written in C++
- Compiled (fast execution), object-oriented language
- Bindings to glue Python & C++ together
- Pyrex/rewindable generate C code for calling C++ from Python

**Example: SCEC Strike-Slip Benchmark**
- **One of the benchmarks in the SCEC Crustal Deformation Modeling benchmark suite.**
- **Description**
  - Viscouslastic (Maxwell) relaxation from an initial earthquake in 3-D without gravity
  - Elastic: layer over a viscouslastic layer
  - Fault extends into viscouslastic layer
  - Slip is 1.0 m and stops (linearly over 4 km along fault edges of the fault
  - Displacement on boundaries set to semi-analytic elastic solution
- **Results shown are for elastic solution only.**

**Elastic Solution**

**Comparison of Local Error**
- Error is small and decreases with finer resolution. Solution is converging!
- Greatest error sources are slope gradient is discontinuous and linear basis functions cannot match slip variation.
- For linear basis functions, Heaviside cells support 400 test cells.
- We can improve performance by switching from Python to C++ implementation.

**Role of the CIG**
- **Geodynamics**
- The CIG is an NSF funded membership-owned organization that supports Earth science by developing and maintaining software for computational geodynamics.
- 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 (Leif Strand and Sue Kienzle) and help in writing the documentation (Sue Kienzle)
- 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
- External scientific 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

**Target applications**
- Pre- and post-seismic deformation with viscouslastic rheologies
- Ground-motion simulations with kinematic or spontaneous ruptures
- Calculation of 3-D Green’s functions
- Simulations of multiple earthquake cycles

**Target applications**
- Pre- and post-seismic deformation with viscouslastic rheologies
- Ground-motion simulations with kinematic or spontaneous ruptures
- Calculation of 3-D Green’s functions
- Simulations of multiple earthquake cycles

**PyLith 1.3 uses 30-35% less memory and is 25-30% faster than PyLith 1.1.**

**Major external packages**
- PETSc is a Portable, Extensible Toolkit for Scientific Computation (PETS). It is used to perform operations on a parallel system.
- Proj.4 is a library for converting between geographic projections.
- PAM generate arbitrary order instances of Lagrange elements on lines, triangles, and tetrahedra.

**Unit and Regression Testing**
- Automatically run more than 875 tests on multiple platforms whenever code is compiled into the source repository.
- Create tests for nearly every function in code during development
- Inspect and expose bugs at origin
- Isolate and expose bugs at origin
- Develop new tests to expose reported bugs
- Prevent bugs from recurring
- Return tests whenever code is changed
- Code continually improves (permits optimization with quality control)
- Binary packages generated automatically upon successful completion of tests.