The Programming Language Python
In Earth System Simulations

L Gross, I Azeezullah, P Mora, E Saez, J Smillie, C Wang, and Paul Cochrane

Earth Systems Science Computational Centre, and Australian Computational Earth Systems Simulator, The University of Queensland, Brisbane, Australia
Fault simulation

- Hazard assessment
  - determine areas of high risk
- Earthquake prediction
  - try to determine “when”
- Typical Fault Model
- Wave propagation:

\[
0 = \rho \frac{\partial u_i^2}{\partial t^2} - \sum_j \frac{\partial \sigma_{ij}}{\partial x_j}
\]

\[
\sigma_{ij} = \lambda \sum_k \frac{\partial u_k}{\partial x_k} \delta_{ij} + \mu \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

- on faults: \( n_j \sigma_{ij} = f_n n_i + f_\tau \tau_i \); restoring force: \( f_n = \max (E_n[u]_j n_j, 0) \)
- tangential stress: \( f_\tau = \max (E_\tau[u]_i \tau_i, \mu f_n) \)
Simulation Approach

- Simulations scripted with escript
  - High-level Python-based modular simulation scripting language
  - Independent of underlying PDE solving software
  - Our simulations use Finley:
    - flexible, high-performance PDE solver for escript

- Spatial discretisation
  - Within PDE solving software
    - use the finite element method

- Time discretisation
  - Handled at the escript level
    - linear PDE is solved at each step
**escript objects**

- **escript.Data**
  - Represents data with spatial distribution
    - stored on sample points
    - tensors up to order 4
  - Data manipulation
    - eg: +, -, *, /, cos, sin, exp, sqrt, ...
    - without spatial dependency
    - data parallel: OpenMP and MPI
- **escript.linearPDE**
  - Interface to a general linear PDE
```python
text  escript example

```from escript import *
myPDE = linearPDE(mydomain)
myPDE.setMatrixType(LUMPED)
while <something true>:
g = grad(u)
stress = l*trace(g) + m/2*(g+transpose(g))
jump = u.interpolate(faultface1) - u.interpolate(faultface0)
fn = max(En*inner(jump,n), 0)
ftau = max(Etau*inner(jump,tau), mu*fn)
myPDE.setCoefficients(X=stress, y_contact=fn*n+ftau*tau)
# velocity-verlet scheme:
anew = myPDE.getSolution()
v += h/2*(a+anew)
u += h*v + h**2/2*a
a = anew
```
Typical simulation results

Pyvisi

- Project to have a consistent interface to many visualisation and rendering backends
- For example:
  - vtk—the visualisation toolkit
  - OpenDX
  - gnuplot
  - povray
  - ...
- Independent of underlying visualisation software
- Project is in its infancy
Inserting visualisation code into previous example

```python
from pyvisi import *
scene = Scene(renderer='vtk')
plot = ContourPlot(scene)
plot.title = "Fault system simulation"
scene.add(plot)
myPDE = linearPDE(mydomain)
while <something true>:
    <set up pde>
    myPDE.setCoefficients(X=stress, y_contact=fn*n+ftau*tau)
    # velocity-verlet scheme:
    anew = myPDE.getSolution()
    plot.setData(anew)
    scene.render()
    u += h*v + h**2/2*a
    v += h/2*(a+anew)
    a = anew
```
Closing comments

- **escript**
  - High performance, modular scriptable simulation system
  - Rapid development of high-level solution algorithms

- **Pyvisi**
  - Clean, easy-to-use interface to visualisation packages
  - Project page:
    - [http://pyvisi.sourceforge.net](http://pyvisi.sourceforge.net)
Supporting Institutions

- Australian Commonwealth Government
- Australian Computational Earth Systems Simulator Major National Research Facility
- Queensland State Government
- Smart State Research Facility Fund
- The University of Queensland
- SGI