A Case for Developing with a PDE Language

Andy R Terrel
Advisors: L Ridgway Scott and Robert C Kirby

Department of Computer Science
University of Chicago

Conference on
Computational Science and Engineering, 2007

Outline

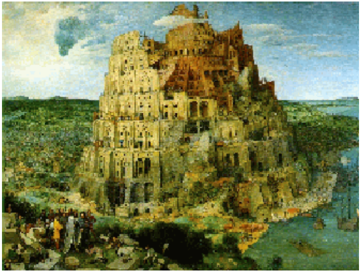
What Do You Mean Language?

Natural Languages

Business Interactions

Philosophy

- Epistemology
- Ethics
- Mathematics
  - Analysis
  - Geometry
  - Algebra



What Do You Mean Language?

Natural Languages

Business Interactions

Philosophy

- Epistemology
- Ethics
- Mathematics
  - Analysis
  - Geometry
  - Algebra

Computer Languages

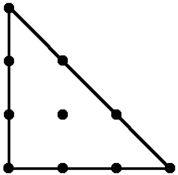
Turing Machine

- C - like languages
- Matlab

Church Lambda Calculus

- LISP
- Prolog

How Do You Want to Write Code?



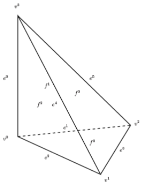
Simple Mesh

Points: 1,2,3
Edges: (1,2),(1,3),(2,3)
Face: (1,2,3)

Sieve Mesh

Points: 1,2,3
Edges: cone(Points)
Face: cone(Edges)

How Do You Want to Write Code?



Simple Mesh

Points: 1,2,3,4
Edges: (1,2),(1,3),
(1,4),(2,3),(2,4),(3,4)
Face: (1,2,3),(1,2,4),
(1,3,4),(2,3,4)

Sieve Mesh

Points: 1,2,3,4
Edges: cone(Points)
Faces: cone(Edges)

How Do You Want to Write Code?

$$
\begin{aligned}
-\Delta \mathbf{u} + \nabla \mathbf{p} &= f \\
\nabla \cdot \mathbf{u} &= 0
\end{aligned}
$$

```
for element in start ... finish
  for node in start ... finish
    a = integrate(ux*dx * vx*dx + uy*dy *vy*dy) ...
    L = integrate( v * f)
```

How Do You Want to Write Code?

$$
\begin{aligned}
-\Delta \mathbf{u} + \nabla \mathbf{p} &= f \\
\nabla \cdot \mathbf{u} &= 0
\end{aligned}
$$

```
a = dot(grad(v), grad(U))*dx - div(v)*P*dx
L = dot(v, f)*dx
```

How Do You Want to Write Code?

$$
\begin{aligned}
-\Delta \mathbf{u} + \nabla \mathbf{p} &= f \\
\nabla \cdot \mathbf{u} &= 0
\end{aligned}
$$

Nice GUI strong form.

How Do You Want to Write Code?

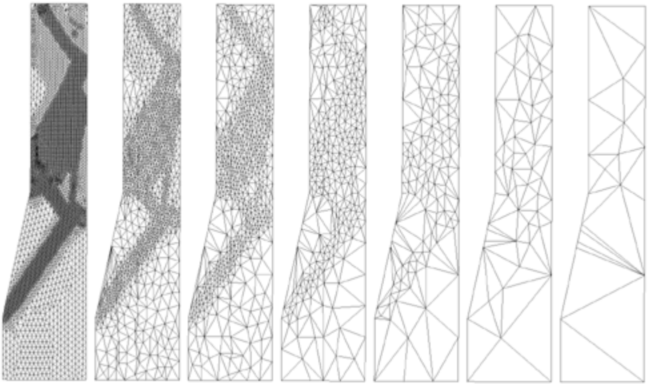
Different Elements

Look up in book

FIAT

Lagrange(triangle,k)
CrouzeixRaviart(triangle)

Separate Coding from Science



- courtesy Peter Brune.

Separate Coding from Science

Stokes Equation

- Taylor-Hood
- Crouzeix-Raviart
- Iterated Penalty

$$\begin{aligned} -\Delta \mathbf{u} + \nabla \mathbf{p} &= f \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned}$$

Separate Coding from Science

Stokes Equation
Taylor-Hood
Crouzeix-Raviart
Iterated Penalty

Navier-Stokes

- Stokes Solver
- Nonlinear Solver
- Time Stepping

$$\frac{du}{dt} + u \cdot \nabla u = -\frac{\nabla \mathbf{p}}{\rho} + \nu \Delta \mathbf{u}$$

Separate Coding from Science

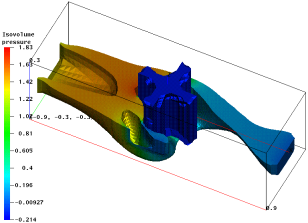
Stokes Equation
Taylor-Hood
Crouzeix-Raviart
Iterated Penalty

Navier-Stokes
Stokes Solver
Nonlinear Solver
Time Stepping

Non-Newtonian Flow

- Oldroyd-B
- Grade 2

Separate Coding from Science



Stokes Equation
Taylor-Hood
Crouzeix-Raviart
Iterated Penalty

Navier-Stokes
Stokes Solver
Nonlinear Solver
Time Stepping

Non-Newtonian
Odroyd-B
Grade 2
...

Fluid Solid Interfaces

- Free Boundary Problems
- Couple to legacy Codes

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

- uniform meshes,
- general geometry,
- adaptive meshes,
- unstructured meshes

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

- linears,
- menu of options,
- arbitrary order,
- tabulator

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

- menu,
- language,
- derived forms,
- error estimators

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

- menu,
- language

What Are the Parts Needed for FEM Software?

- Mesh Generation
- Function Spaces
- Equation Description
- Discrete Equation Solver
- Parallel Computing Support

- parallel linear solve,
- parallel assembly,
- load balancing

Why Automate FEM?

- Ensure Correctness:
  Complicated error prone `mathematical` process
  Complicated error prone `programming` process
- Reduce Programming Hours:
  Gives ability to quickly change `models`
  Gives ability to quickly change `elements`
  Gives ability to quickly change `methods`
- Optimize Computation:
  Allow a non-expert programmer to make efficent calculations

Why Automate FEM?

- Ensure Correctness:
  Complicated error prone `mathematical` process
  Complicated error prone `programming` process
- Reduce Programming Hours:
  Gives ability to quickly change `models`
  Gives ability to quickly change `elements`
  Gives ability to quickly change `methods`
- Optimize Computation:
  Allow a non-expert programmer to make efficent calculations

Why Automate FEM?

- Ensure Correctness:
  Complicated error prone `mathematical` process
  Complicated error prone `programming` process
- Reduce Programming Hours:
  Gives ability to quickly change `models`
  Gives ability to quickly change `elements`
  Gives ability to quickly change `methods`
- Optimize Computation:
  Allow a non-expert programmer to make efficent calculations

The Software and Mathematics.

"Mathematical Software should be Mathematical"

but

- Directly mapping mathematics to code is flawed.
- Automation requires more mathematical understanding
  - Global - local interactions
  -

Some Major Projects

Simulation Engines

- Sundance
- FFC/Dolfin
- Deal.II
- Analysa
- FreeFEM
- GetDP

Tabulators

- FIAT
- SyFi

Linear Solvers

- UMFPack
- PETSc
- Trilinos

Why are we NOT Using Automated?

- Different mathematical and algorithmic abstractions
- Hand coding is very attractive (“If you want it done right...”)
- Quite difficult to switch between elements, solvers, and methods.

The Stokes Equation.

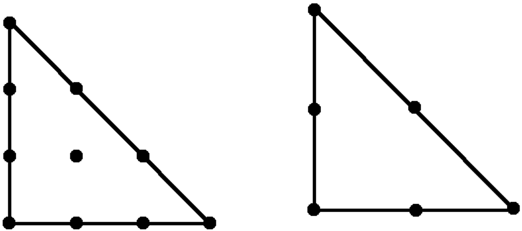
The Stokes equations are a model for steady incompressible flow:

$$\begin{aligned} -\Delta \mathbf{u} + \nabla \mathbf{p} &= f \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned}$$

Important Features:

- Coupling of pressure and velocity
- Numerous methods for solving

Taylor - Hood Elements

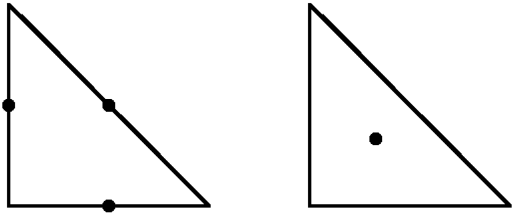


(a) P_3 for V (b) P_2 for Π

Important Features:

- Available using any P_k elements,
- Built from standard Lagrange elements,
- Easily extendable to arbitrary order

Crouzeix - Raviart Elements



(c) Crouzeix-Raviart for V

(d) P_0 for Π

Important Features:

- Non Conforming
- Low Order
- Divergence-free Pressure Space

$C^0 P_i C^{-1} P_{i-1}$ Elements

Use a Continuous Lagrange element P_i for V and a Discontinuous Lagrange element P_{i-1} for Π

Important Features:

- May not satisfy $\inf\sup$ condition
- Divergence-free Pressure Space
- Still use Lagrange elements

Iterated Penalty

Let $r \in \mathbb{R}$ and $\rho > 0$ define u^n and $p = w^n$ by

$$
\begin{aligned}
a(\mathbf{u^n}, \mathbf{v}) + r(\nabla \cdot \mathbf{u^n}, \nabla \cdot \mathbf{v}) &= F(\mathbf{v}) - (\nabla \cdot \mathbf{v}, \nabla \cdot \mathbf{w^n}) \\
\mathbf{w}^{n+1} &= \mathbf{w}^n + \rho \mathbf{u}^n
\end{aligned}
$$

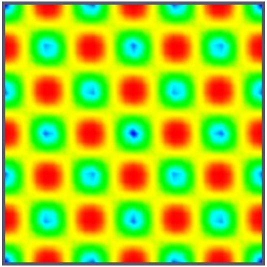
Important Features

- One discrete spaces, V,
- Similar to $C^0 P_i C^{-1} P_{i-1}$ Elements but different implementation

Problem statement

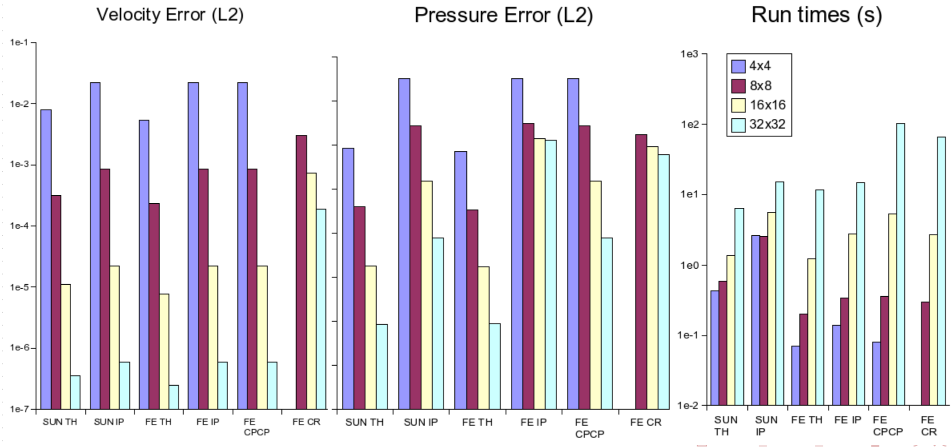
$$\mathbf{u} = \begin{bmatrix} \sin(3\pi x)\cos(3\pi y) \\ -\cos(3\pi x)\sin(3\pi y) \end{bmatrix}$$

$$p = \sin(3\pi x)\sin(3\pi y)$$



The Numbers.

Comparison of Fourth Order



User Experience of Testing

- Debugging is entirely different
- Getting something working is easy, getting it right ...
- Everyone can read your code

Comparisons between Software Packages

- Sundance and FEniCS program very similarly
- FIAT a common interface for defining elements
- Coding time almost identical
- Both still very active development

Summary

- Domain Specific Languages separate Science from Programming
- Mathematics $\Leftrightarrow$ Software Abstractions
- Meaningful test simulations (not just Poisson)

- Outlook
  - Explore mathematical abstractions for global-local interactions
  - Compare Grade 2 and Oldroyd-B fluid model

Do It Yourself

Where to get the code:

Sundance - http://software.sandia.gov/sundance/
FEniCS Project (FIAT, FFC, DOLFIN) - www.fenics.org
Masters Thesis - email me

Any Questions

aterrel@uchicago.edu