

Abstractions in FEM Software

Andy R Terrel
aterrel@uchicago.edu

Department of Computer Science
University of Chicago

November 8-9, 2006
FEniCS'06
Delft University of Technology

Outline

- 1 Motivation and Goals
- 2 Experiences and Observations
- 3 Outlook

Motivation and
Goals

Experiences and
Observations

Outlook

Appendix

- Success of the Finite Element Method(FEM) has led to a proliferation of FEM simulation software.
 - The FEniCS Project, Sundance, DEAL
 - Others: FreeFEM, FEMLab, ...
- No single package meets everyone's needs ... yet.
 - Sundance handles optimization well, but is limited in kinds of elements,
 - FEniCS gives a good smaller bundle that efficiently generates code,
 - DEAL handles a more elements and has a larger user community,
 - Both Sundance and FEniCS use easily readable code for user input.

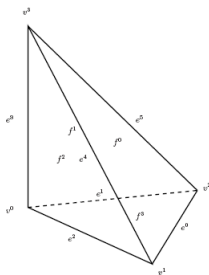
- There seem to still be disjunctions between the math and the software.
 - Domains and meshes are not always identical.
 - Small variations of methods are hard or not possible.
 - ... other issues
- Some goals for this talk
 - Point out some rough spots in current software through my experiences.
 - Get feedback on validity and feasibility of ideas.
 - Challenge the next generation of FEM software to be more mathematically rigorous
 - Rigorous in code correctness.
 - Rigorous in correct mathematical abstractions from problems.

Mathematically a finite element method is simply:

- A reference element, K
- A space of shape functions, \mathcal{P}
- A basis, \mathcal{N}

But it seems there is something missing:

- Links between elements,
- How the elements affect the solvers.



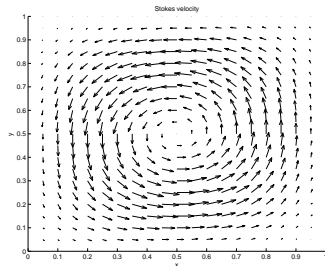
- It seems there is always someone who wants to do something different.
- How much control of you software to give the user, to play with new methods?
- Example: Mixed methods, do we just put the formulation in the software or give the user the matrices.

Example: Stokes Equations

$$\begin{aligned} -\Delta u + \nabla p &= f \\ \nabla \cdot u &= 0 \end{aligned}, \quad u = \begin{bmatrix} \sin(\pi x) \cos(\pi y) \\ -\cos(\pi x) \sin(\pi y) \end{bmatrix}$$

Using Taylor-Hood elements with mixed formulation,

Number of Iterations		
mesh	P1/P2	P2/P3
4x4	14	22
8x8	24	54
16x16	83	283
32x32	328	1319

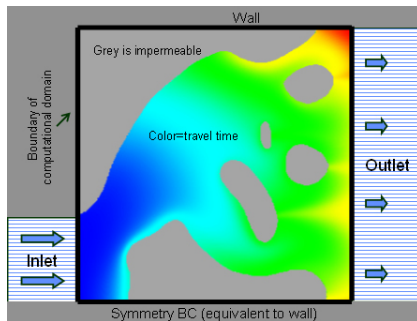


How about optimization problems?

- Use Automatic Differentiation tools on produced code - expensive on user side
- Create a symbolics engine that can give derivatives - expensive on developer side

Example Problem in Microfluidic Devices

- To optimize flow, change channel geometry
- Most effective methods, use level set methods



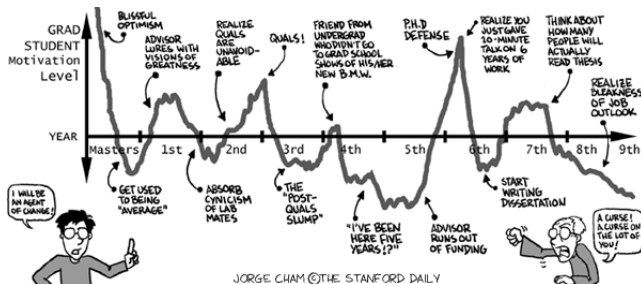
Why is Mathematical Software Hard?

The design space for mathematical software is multi-dimensional and not orthogonal.

- Mesh: uniform meshes, general geometry, adaptive meshes, unstructured meshes
- Function Space: linears, menu of options, arbitrary order, FIAT, exterior calculus
- Equation Description: menu, language, derived forms, error estimators
- Solver algorithms: menu, language
- Parallel Computing Support
- Boundary Conditions and embedded geometries.

Future

- Good mathematical abstractions have gotten us this far, where else can we go?
- How do these issues play well with software design principles?
- Is there a single solution to automated mathematical modelling?



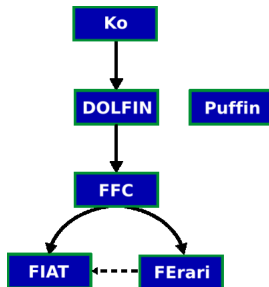
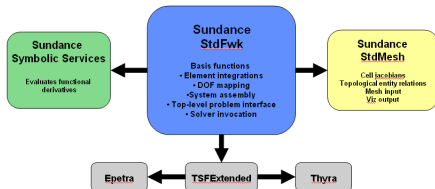
Questions or Comments?
aterrel@uchicago.edu

What about code complexity?

One bad estimate is lines of code:

- Dolfin + FFC \sim 50K lines
- Sundance \sim 100K lines
- DEAL \sim 400K lines

Some Organization Charts



More Detailed Dependencies

An example of Dependencies for Sundance (not especially different from others)

