TEACHING STATEMENT

JED BROWN

I am an advocate of incorporating frequent shifts between presentation, live demos, and interactive sessions and challenges incorporating peer instruction. In the past several years of teaching tutorials on numerical methods with PETSc and working with students and domain scientists with variable backgrounds in computational science and mathematics, I have noticed certain recurring misconceptions that once corrected, enable significant scientific progress and renewed clarity of thought. I look forward to an opportunity to catch these misconceptions early and to nurture early development of practical intuition.

I make extensive use of examples and counter-examples; there is no substitute for seeing *how* a method or result fails when assumptions are violated. When teaching numerical methods, I like to tour misrepresentations (many of which appear in the literature), with students interactively stressing the methods in various ways and seeing how seemingly subtle differences in presentation can drastically change perceived robustness and performance.

I regularly incorporate interdisciplinary connections and applications when presenting mathematical and computational tools. As a student, there were few things that I appreciated more than five-minute boiled-down explanation of a connection to a subject area with which I was either unfamiliar or had never considered in that light. Although full context cannot be presented in such little time, the imprecise context is invaluable for establishing relevance of the tool as well as future interdisciplinary research.

My tutorials are rife with live demos that students get a chance to interact with further during the hands-on session. Due to venue constraints, I have not yet been able to merge the tutorial and hands-on sessions, but I would like to do so because latency between presentation and interactive exploration kills retention. I am intrigued by the more radical "flipped classroom" concept that my colleague Lorena Barba has been using recently[1]. She provides recorded lectures for students to view in advance, then uses class time for discussion and friendly competition. I would expect to choose format for each course separately, informed by discussion with colleagues and results of empirical studies.

I encourage an informal atmosphere so that students feel comfortable asking questions and do not hesitate to hold their ground if an explanation seems contradictory. The ensuing discussion is far more educational than an appeal to authority and also helps me gauge comprehension.

Software and programming are becoming essential components of computational mathematics. To develop these skills, I would use individual and group projects involving implementing methods within the context of some carefully-chosen interfaces. The fixed interfaces are intended to provide enough structure to prevent

Date: Fall 2012.

[1] http://www.bu.edu/me/2012/03/13/flipped-classroom-energizes-computational-fluid-dynamics-course/

students from getting too far off-track, yet leave enough opportunity for students to make their own design decisions and see the consequences of those decisions pan out in follow-on assignments. Group collaboration and project submission would use modern version control and automated testing such as one would encounter when interacting with real software projects.

1. Course outline: Parallel multilevel solvers and multiscale simulation

This proposed course, which I would love to introduce, would target students at the graduate level or advanced computationally-aware undergrads. The topic is highly relevant to many scientific and engineering applications since it is all too frequently the bottleneck preventing the next step. The course would be relatively unique since its subject matter is far broader than that of multigrid courses taught at other universities. Its purpose is to prepare students for research-level investigation and the context necessary to apply these methods to diverse application areas.

- introduction to partial differential equations and local discretizations
- lower bounds for convergence rates of one-level methods
- multigrid convergence "with regularity"
- local Fourier analysis for additive, multiplicative, block, and composite smoothers
- h-ellipticity and stability of spatial discretizations
- non-symmetric and saddle point problems
- what goes wrong in classical multigrid?
- principles of Krylov methods
- nonlinear multigrid and the τ formulation of Full Approximation Scheme
- adaptive mesh refinement and adaptive visits
- principles of multigrid "without regularity"
- classical algebraic multigrid and smoothed aggregation
- energy-minimizing basis functions and multiscale finite elements
- coupled and multiphysics problems
- multilevel methods for optimization and uncertainty quantification
- n-body problems and integral equations
- systematic upscaling for stochastic multiscale models

I would also be especially interested in teaching courses on discretization techniques for partial differential equations and on high-performance scientific computing.

2. Mentoring

I co-advised PhD summer student Abraham Taicher (from Todd Arbogast at UT Austin) on development of a PETSc-based code for Darcy-Stokes (melt migration in geodynamics) using a new finite element discretization. He learned about parallel unstructured finite element implementation, designing for software reuse, and field-split preconditioning.

I co-advised PhD summer student Lulu Liu (from David Keyes at KAUST) on development of a PETSc-based code for reservoir simulation using novel implicit solvers. She learned about finite volume methods, designing for software reuse, field-split preconditioners, the relative cost and benefit of algorithmic components such as an assembled Jacobian, and performance analysis. Her code is now a test-bed for ongoing research on multilevel nonlinear domain decomposition methods.

I co-advised undergraduate summer student Xuan Zhou (from IIT) in developing a PETSc interface to the dense linear algebra library Elemental and the sparse direct library Clique. He learned about software concerns for libraries, stable interfaces, mixed-language programming, parallel distributions and logical remapping. The code is now in production use within the PETSc library.