Templates for Linear System Solvers and Eigenvalue Problems

Zhaojun Bai	U Kentucky	bai@ms.uky.edu
David Day	U Kentucky	dday@ms.uky.edu
James Demmel	UC Berkeley	demmel@cs.berkeley.edu
Jack Dongarra	U Tennessee	dongarra@cs.utk.edu
Ming Gu	UC Berkeley	minggu@math.berkeley.edu
W. Kahan	UC Berkeley	/dev/null
Beresford Parlett	UC Berkeley	parlett@math.berkeley.edu
Axel Ruhe	Chalmers U	ruhe@math.berkeley.edu
Dan Sorensen	Rice U	sorensen@rice.edu
Henk van der Vorst	U Utrecht	vorst@math.ruu.nl

Funding from ARPA, DOE, NSF

Outline

- Who needs numerical software? -
  Motivation for Templates
- Some existing “numerical advice” activities
- Templates for Linear Systems: Iterative Methods
- Proposal for Templates for Eigenvalue Problems
- Future work

Who needs numerical software?

- Kinds of users and their priorities:
  - Engineers and scientists generally, who make 12000 requests/day to netlib and keep NAG and IMSL in business:
    1. Easy interface, hidden details
    2. Reliability
    3. Speed
  - HPCC community, solving biggest and hardest problems:
    1. Speed
    2. Access to details for tuning
    3. Reliable enough for their application
  - Teachers and Students
    1. Ease of understanding
    2. Access to some details for learning
- Can we please everyone with one “medium”?

Related activities (not comprehensive!)

- Netlib (http://www.netlib.org/)
  - Code, tech reports, news ...
  - Organized by library name
  - Public-domain code, some commercial
  - Search by topic, routine, author, keyword ...
  - Small topic decision tree, leaves are libraries
- GAMS - Guide to Available Math Software (http://gams.nist.gov)
  - Access tree organized by problem
  - Pointers to netlib, some commercial
  - Tree dates to SHARE, being updated
- Optimization Software Guide (Moré, Wright)
  - Short mathematical description of common optimization problems, in book form
  - List of self-supplied short documentation of 75 packages, public and commercial

- **LAPACK/ScaLAPACK (http://www.netlib.org/{lapack,scalapack}/index.html)**
  - **Black boxes for dense/banded linear algebra problems;**
  - **On-line manual (latex2html)**
- **PETSc (http://www.mcs.anl.gov/petsc/petsc.html)**
  - **Fast prototyping of PDE, solver codes**
  - **Serial, parallel**
  - **Like C++ classes for preconditioners, Krylov space solvers, Newton solvers, grids, discretizations ...**
- **Numerical Recipes ...**

Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods

Richard Barrett, Michael Berry, Tony Chan,
James Demmel, June Donato, Jack Dongarra,
Victor Eijkhout, Roldan Pozo, Charles Romine,
and Henk van der Vorst

Contents of Templates for Linear Systems

- **Algorithm sketches for**
  - **Jacobi, Gauss-Seidel, SOR, SSOR, ...**
  - **PCG, Bi-CG, QMR, GMRES, ...**
  - **Preconditioners (Jacobi, SSOR, Incomplete Factorization, Polynomial, ...)**
  - **Intro to Domain Decomposition**
  - **Convergence testing**
  - **Hints for Parallelism**
- **Sparse data structures**
- **Code available in Matlab, Fortran (dense)**
- **Available from SIAM or http://www.netlib.org/templates/index.html**

Eigentemplates

Yes, we can please everyone!

- We propose that
  - NHSE - National High Performance Software Exchange
  - GAMS - Guide to Available Math Software
  - Netlib
  - Templates
  - NA textbooks
  - Annotated bibliographies
  - Computing environments á la Matlab or Maple
  - On-line compute servers (comp-bots) for bigger problems

  should all be part of one seamless whole.

- **We propose an E-book containing**
  - **Linearly ordered text**
  - **Multiple access modes via hypertext**
    - **Table of contents of book**
    - **Index of book**
    - **A decision-tree querying the user to pick the right algorithm**
    - **Pointers to other trees (netlib, GAMS)**
  - **Multiple levels of algorithmic description**
    - **"Template" in pseudo-code, further pointers to common notation, etc.**
    - **One-page description of algorithm**
    - **Matlab implementation**
    - **Fortran, C, C++ implementations**
    - **Detailed text (references to literature) ("Numerical Linear Algebra", by JD)**
    - **Pointers to literature**
- **C++ implementation of LinSolve or EigSolve incorporating parts of decision tree (Pozo)**
- **Open architecture for future growth**
- **Hard-nosed editing for most reliable parts**

Future Work

- Publically distribute detailed outline
  Superset of likely content
- Get (potential) user feedback
  (workshop planned)
- Keep linear version of book $\leq$ 200 pages
  (not all material)

- Q: Will a good design put NA faculty out of business?
  A: Better we do it than someone else.