Matrix Computations - Math 221 - Fall 2010 - T Th 9:30 - 11am in 31 Evans Hall

Professor: Jim Demmel, 831 Evans Hall and 564 Soda Hall, 643-5386, demmel@cs.berkeley.edu.

Class Home Page: `www.cs.berkeley.edu/~demmel/ma221`

Prerequisites: Good knowledge of linear algebra, programming experience, numerical sophistication at level of Ma 128ab or equivalent.

Syllabus: The standard problems whose numerical solution we will study are systems of linear equations, least squares problems, eigenvalue problems, and singular value problems. Techniques for dense and sparse problems will be covered; it is impossible to cover these areas comprehensively, but students should still come to appreciate many state-of-the-art techniques and recognize when to consider applying them. We will also learn basic principles applicable to a variety of numerical problems, and apply them to the four standard problems. These principles include

1. matrix factorizations (also called "direct methods"),
2. iterative methods,
3. perturbation theory and condition numbers,
4. roundoff error and numerical stability
5. choosing the best (fastest and/or most accurate) algorithm based on the mathematical structure of your problem, and
6. designing the fastest algorithms for modern computer architectures

To elaborate on item (6), the computers on which everyone runs these (and other) algorithms are changing dramatically. The first change is that sequential computers are no longer becoming faster. Instead, any program that needs to run faster has to be rewritten to run in parallel, with the number of parallel processors, even in your laptop, increasing annually. The second change is that the cost of arithmetic (doing an addition, multiplication, etc.) has continued to get smaller and smaller than the cost of moving data within the computer system (for example, between main memory and cache, or between parallel processors connected over a network). New algorithms have emerged recently for most of the problems studied in this class, that are much faster than their predecessors, because they move much less data (in fact, they provably minimize the amount of data moved). Indeed, we have significant funding from the NSF to redesign the most widely used linear algebra libraries (LAPACK and ScaLAPACK, which have been developed in a collaboration between Berkeley, U. Tennessee and other research organizations over many years). A variety of class projects in this area are available. Recent papers on this topic may be found at bebop.cs.berkeley.edu.

In addition to discussing established solution techniques, other open problems will also be presented.

Grading: Grades will be based on weekly homework, as well as programs and a final project. Homework or programs turned in late will receive only half credit. You may work together on homework, but it should be turned in individually. It is all right to discuss programs with one another, but work should be done individually. Final projects should be done individually.

Programs will be of two kinds, Fortran, C or C++ (your choice), and Matlab. Assignments to be written in Fortran/C/C++ will use subroutines from libraries like LAPACK or CLAPACK. Matlab software related to the course is available on the class homepage. You may find it convenient to do mixed language programming (e.g. calling a Fortran routine from C or C++, or a C routine from Matlab); this is up to you.

Final project proposals are due Oct 14 in class: design a project related to the course material that will require about the same effort as 4 homework assignments, and write up a 1-page summary of what you plan to do, why it is related to numerical linear algebra, and how it relates to any larger scientific goals you have. There will be a poster session during the 9:30-11 on Thursday Dec 9 of RRR week. The final project writeup (5-10 pages) is due Monday Dec 13.

Text: *Applied Numerical Linear Algebra,* SIAM, 1997.

Recommended Texts:

- *Templates for the Solution of Linear Systems*, R. Barrett et al. SIAM, 1994. Describes standard Krylov-space iterative methods for solving $Ax = b$, with guidance on choosing and algorithm and software. Postscript and html versions of the book, and software, are available at the web site: `www.netlib.org/templates/`.

- *Templates for the Solution of Algebraic Eigenvalue Problems*, Z. Bai et al. SIAM, 2000. Gives a unified overview of theory, algorithms and practical software for eigenvalue problems. The web site `www.cs.ucdavis.edu/~bai/ET/contents.html`, has pointers to a software repository and on-line version of the book.

Other Reading

1. *Numerical Linear Algebra*, L. N. Trefether and D. Bau, SIAM, 1997. Also aimed at a first year graduate audience, but has a more pure mathematical flavor than the main text.

2. *Matrix Computations*, G. Golub and C. Van Loan, 3rd Ed. Johns Hopkins Press, 1996. Very complete, if not encyclopedic, book on matrix computations.

3. *Fundamentals of Matrix Computations*, David Watkins, Wiley, 1991. Very readable beginning graduate level textbook.

4. *LAPACK Users' Guide*, E. Anderson et al. SIAM 1999 (3rd Edition). Describes widely used library of dense numerical linear algebra software. Software and on-line text also available at `www.netlib.org/lapack`.

5. *ScaLAPACK Users' Guide*, L. S. Blackford et al, SIAM 1997. Describes version of LAPACK for parallel computers. See web site for software and on-line text: `www.netlib.org/scalapack`.

6. *The Algebraic Eigenvalue Problem*, J. Wilkinson, Clarendon Press. Classic, somewhat dated but still excellent comprehensive presentation of numerical linear algebra.

7. *Matrix Perturbation Theory*, G. W. Stewart and J.-G. Sun, Academic Press, 1990. Comprehensive account of perturbation theory for linear algebra problems.

8. *Perturbation Theory for Linear Operators*, T. Kato, Prentice Hall. Comprehensive account of analytic perturbation theory for eigenvalues and eigenvectors; chapter 2 covers the finite dimensional case, which is the subject of this course.

9. *Numerical Methods for Least Squares Problems*, Åke Björck, SIAM, 1996. Comprehensive work on methods for linear least squares problems.

10. *Solving Least Squares Problems*, C. Lawson and R. Hanson, Prentice Hall. Older classic, supplanted by Björck's book.

11. *The Symmetric Eigenvalue Problem*, B. Parlett, Prentice Hall. Algebraic and analytic theory of symmetric matrices and algorithms

12. *Accuracy and Stability of Numerical Algorithms*, N. J. Higham, SIAM, 2002 (2nd edition). Comprehensive rounding error analysis for linear equations and least squares problems.

13. *Iterative Methods for Sparse Linear Systems*, Yousef Saad, PWS Publishing, 1996. Expands on iterative methods in chapter 6 of the textbook.

14. *Iterative Methods for Solving Linear Systems*, Anne Greenbaum, SIAM. , 1997. Another good, modern account of iterative methods.

15. *A Multigrid Tutorial*, W. Briggs, SIAM, 1987. A good introduction to multigrid methods.

16. *MGNet*, or Multigrid Net, is a web page (`www.mgnet.org`) with pointers to books, software, and tutorial material.

Computer Resources: Since this is a graduate course, I will assume that students have access to computer accounts already; if this is not the case please contact me. Also, I will assume that students have access to Matlab (or its public domain look-alike, Octave). Again, please contact me if this is not the case.