Matrix Computations - Math 221 - Fall 1999 - T Th 3:30 - 5 in 71 Evans Hall
Professor: Jim Demmel, 737 Soda Hall, 643-5396, demmel@cs. A number of lectures will also be
given by Prof. Beresford Parlett, 709 Evans Hall, 642-6655, parlett@math.
Teaching Assistant: Plamen Koev, 888 Evans Hall, plamen@math.
Class Home Page: http://www.cs.berkeley.edu/~demmel/ma221
Recommended Text: 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 http://www.netlib.org/templates/.
Prerequisites: Good knowledge of linear algebra, programming experience, numerical sophis-
tication at level of Ma 128ab or equivalent.
Other Reading

at a first year graduate audience, but has a more purer mathematical flavor than the main
text.

complete, if not encyclopedic, book on matrix computations.

graduate level textbook.

used library of dense numerical linear algebra software. Software and text also available at
http://www.netlib.org/lapack

excellent comprehensive presentation of numerical linear algebra.

hensive account of perturbation theory for linear algebra problems.

of analytic perturbation theory for eigenvalues and eigenvectors; chapter 2 covers the finite
dimensional case, which is the subject of this course.

work on methods for linear least squares problems.

for linear unconstrained and constrained least squares problems. Older classic, supplanted by
Björck's book.

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

presentation of rounding error analysis for linear equations and least squares problems.


**Grading:** Final: 50%, Programs: 30%, Homework: 20%.

You may work together on homework, but it should be turned in individually. On the other hand, it is all right to discuss programs with one another, but work should be done individually. Programs will be of two kinds, Fortran, C or C++ (your choice), and Matlab. Matlab software related to the course is available on the class homepage. Assignments to be written in Fortran/C/C++ will use subroutines from libraries like LAPACK or CLAPACK. Matlab is an interactive, user-friendly interface to a large body of numerical and graphics software, including linear algebra, and is widely used for testing and prototyping algorithms. 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).

**Computer Resources:** There is a campus site license for Matlab, so it should run everywhere. You will also need a WWW browser like Netscape; this should also be widely available. If you have difficulty finding appropriate resources, please see me and I will give you an account on socrates; you may prefer the resources in your own department. Math grad students must use socrates accounts. Accounts, manuals, and documentation standards will be handed out later.

Matlab documentation is available from at least three sources. First, there is an extensive online help facility (just type ‘help commandname’ or ‘help’ in Matlab). Second, a brief manual is available on the class homepage. Third, comprehensive manuals can be inspected or purchased in the Computer Center in Evans Hall.

**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 three standard problems. These principles include (1) matrix factorizations, (2) perturbation theory and condition numbers (3) effects of roundoff error on algorithms, including properties of floating point arithmetic (4) analyzing the speed of an algorithm, and (5) choosing the best algorithm for the mathematical structure of your problem, and (6) engineering numerical software.

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