Matrix Computations - Math 221 - Fall 2024 - T Th 12:30-2pm in 241 Cory Hall

Professor: Jim Demmel, demmel@berkeley.edu.

Class Home Page: `people.eecs.berkeley.edu/~demmel/ma221_Fall24`

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

Instructional mode: All lectures will be in-person, and we will record and post them on bcourses; see Announcements on bcourses.berkeley.edu for information on how to connect. We will also prepost typed notes before each lecture.

We will also use piazza and bcourses for class announcements, along with the class home page above.

Syllabus: The standard problems whose numerical solution we will study are systems of linear equations, least squares problems, eigenvalue problems, singular value problems, and some of their generalizations and applications. Techniques for dense, sparse and "structured" 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. In other words, the classical metric that says that Algorithm A is faster than Algorithm B if it does fewer arithmetic operations is no longer always true. 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).

We will also discuss randomized algorithms, that "probably" give an answer with "reasonable" accuracy, a topic with much recent work.

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

Given the span of possible topics, it is important to fill out the survey on the class webpage, so that I can determine what priorities the students have.

Grading: Grades will be based on weekly homework, as well as programs and a final project. All homework should be turned in on gradescope, while projects should be turned in using bcourses. 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. Programs for homework will be written in Matlab. Matlab software related to the course is available on the class homepage.

Final projects should also be team projects this semester, unlike previous semesters. (This is to accommodate the growing number of students taking the class this semester.) These should mostly be teams of 2 students (if an odd number of students end up enrolling in the class, we will make an exception!).

To help students find teammates with common interests, we ask that you write a 1 or 2 page "preproposal" describing your possible interests, that we will post on bcourses, so that (only) your classmates can read them. We will also provide short descriptions of a few previously submitted projects as examples. Feel free to use piazza and other media to find partners too. Projects can be a mixture of literature review or survey, implementing and evaluating the accuracy and/or performance of an algorithm on selected test problems, or theoretical analysis of the properties of selected problems and algorithms.

Final project proposals are due Oct 15 (on bcourses). The project should take about the same effort as 4 homework assignments. Your 1-page proposal (1 per team) should outline what you plan to do, why it is related to numerical linear algebra, and how it relates to any larger scientific goals you have. Feel free to talk to me well before this deadline about possible projects. There will hopefully be a poster session during RRR week (to be scheduled). The final project writeup (5-10 pages) is due Monday of finals week, Dec 16.

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 (more recent papers will be pointed out during lectures)

1. *Scribed Class Notes for Ma221.* Thanks to the generous efforts of students in recent classes, all the class notes have been typed up in latex. These notes, which may turn into a new edition of the textbook, have only been partly proofread (suggested corrections welcome!), and so

are only posted on bcourses, under files, as ma221_scribed_notes.pdf (173 pages). Suggested comments or corrections are welcome, in any class materials!

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

3. *Matrix Computations*, G. Golub and C. Van Loan, 4th Ed., Johns Hopkins Press, 2013. Very comprehensive, if not encyclopedic, book on matrix computations.

4. *Fundamentals of Matrix Computations*, David Watkins, 3rd Ed., Wiley, 2010. Very readable beginning graduate level textbook.

5. *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`.

6. *Randomized Numerical Linear Algebra: A Perspective on the Field With an Eye to Software*, R. Murray, J. Demmel, and many others. 2023. arxiv.org/abs/2302.11474; A recent description of a proposed extension of LAPACK to include randomized algorithms.

7. *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`.

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

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

10. *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 a subject of this course.

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

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

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

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

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

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

17. *An Introduction to the Conjugate Gradient Method without the Agonizing Pain*, Jonathan Shewchuk, 1994, `www.cs.cmu.edu/~jrs/jrspapers.html#cg`, In contrast to the terse treatment in the course text book, you might want to see Shewchuk's answer to the question "How could fifteen lines of code take fifty pages to explain?"

18. *Communication lower bounds and optimal algorithms for numerical linear algebra*, Grey Ballard, Erin Carson, James Demmel, Mark Hoemmen, Nick Knight, and Oded Schwartz. An overview of recent work on communication-optimal algorithms. Acta Numerica (invited), Cambridge University Press, 23, pp 1-155, 2014. Available at bebop.cs.berkeley.edu.

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

20. *Multigrid Overview*, J. Demmel, 35 power point slides,
`www.cs.berkeley.edu/~demmel/ma221_Fall04/Multigrid.ppt`

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 a public domain look-alike, such as Octave). Again, please contact me if this is not the case.