V22.0421: Numerical Computing
Computer Science Department

New York University, Spring 2010
Instructor: Margaret H. Wright

Course Project
Assigned March 2, 2010; due April 13, 2010

Purpose of the project. A Numerical Computing project is intended to represent your own individual original work, including consideration of related previous work, significant numerical computation devised by you, and the associated insights gained from the project.

Form of the project. A project should have the form of a scientific paper, i.e., typed pages, beginning with the project title, your name (as author), an abstract, the project body, and a bibliography. The project body is likely to consist of at least 5–7 pages that describe the topic, summarize your investigations, and state what you learned, based on a combination of mathematical analysis and computational experiments. (However, there are no rigid rules about length.) A supplement containing your code (or code from others, properly attributed) that was used for the computations should be provided.

Advance approval. Each student must arrange an individual meeting with me to discuss his/her project before starting serious work on it. As part of the discussion (possibly afterwards), I will ask for a "prospectus" in which you describe your project in general terms. *Submitted projects whose topics I have not explicitly approved in advance will receive no points.*

Grading. As discussed in class, your project will count as at least 30%, but no more than 40%, of your grade in the course (depending on which result is more favorable to you).

Your project grade will be based on (1) knowledge and understanding of numerical computing, (2) creativity in applying this knowledge and understanding to a new problem, (3) clarity and correctness of explanations and examples, (4) insightfulness of the computational testing, and (5) originality. The exposition of the paper as well as the quality of its content will count in the grade.

Citing the work of others. I expect everyone to seek references and information on the Web and in the library beyond the initial pointers mentioned below. *It is essential* *for your project to cite any and all reference material used in the project, including material from the Web. A failure to cite work by someone else that you use in your project will be considered plagiarism.*

The following list provides several suggestions. If you wish to do your project on some other topic, that is fine as long as it meets the criteria described above and I have approved the project in advance following our individual meeting.

1. **Condition estimators.**

   Various procedures have been suggested for obtaining an estimate of the condition of a matrix without forming an explicit inverse. For background, see Chapter 14 of Nick Higham's book *Accuracy and stability of numerical algorithms* (first edition, 1996; second edition, 2002), published by the Society for Industrial and Applied Mathematics (SIAM; `www.siam.org`). This project could (for example) describe the more popular condition estimators, including their strong and weak points, trying them out on your own examples using Matlab, and commenting on the overall usefulness of such techniques.

2. **Linear algebra in Web search and data mining.**

   The 2007 Research Experiences for Undergraduates (REU) project supervised by Carl Meyer, Professor of Mathematics at North Carolina State University, focused on mathematical and computational techniques used in data mining, search, and information retrieval systems. There is a link to the final report for the 2007 REU project at

   `http://meyer.math.ncsu.edu`

   An excellent survey (as of 2006) is "Information Retrieval and Web Search", by Amy Langville and Carl Meyer, in the *Handbook of Linear Algebra* (2007), Chapman and Hall. A pdf version of this paper is on Amy Langville's web site at the College of Charleston:

   `math.cofc.edu/langvillea`

3. **Smoothed analysis applied to Gaussian elimination.**

   In recent years, Dan Spielman and Shang-Hua Teng have developed a remarkable approach, *smoothed analysis*, that tries to explain the observed behavior of algorithms in practice. See their recent paper, "Smoothed analysis", in the Review Article section of *Communications of the ACM*, October 2009. The last section of that paper discusses algorithm design based on perturbations and smoothed analysis, mentioning applications to Gaussian elimination.

   Some related papers are available on Spielman's website:

   `www.cs.yale.edu/homes/spielman/SmoothedAnalysis`

4. **Numerical computing in finance.**

   The User's Guide for the Mathworks™ Financial Toolbox discusses common financial tasks, and the chapter "Solving Sample Problems" of that guide explains how the toolbox solves many real-world problems. Depending on your interests, several possible projects could be defined in which you would choose a particular problem and try different numerical methods for different scenarios/data, explaining strategies that work well and situations when difficulties can arise.

5. **Applications of linear algebra.**

   Linear algebra plays an important role in biomolecular modeling, one of the many areas in modern computational biology. The paper "Linear algebra in biomolecular modeling", by Zhijun Wu (2006),

   `orion.math.iastate.edu/wu/modeling06.pdf`

   discusses several of the main applications of linear algebra. (This paper is a chapter in *Handbook of Linear Algebra*, mentioned in project 2). A key idea is that "off the shelf" techniques such as Gaussian elimination need to be adapted to take advantage of the special structure of the biology problems.

6. **Wavelets and their applications.**

   The Mathworks™ Wavelet Toolbox provides a good introduction to the powerful tool of wavelets, mathematical functions that allow data to be analyzed at widely different scales. Material from Gil Strang's MIT course on wavelets (18.327) is available through MIT OpenCourseware (Google "MIT OCW"). Wavelets are related to Fourier analysis, the subject of project 9.

7. **Deblurring images.**

   The book *Deblurring Images: Matrices, Spectra, and Filtering* (2006), by Per Christian Hansen, James Nagy, and Dianne O'Leary, published by SIAM, shows how numerical computing techniques are applied in deblurring images. (The book is in the Courant Library.) Related material is also available at Per Christian's website:

   `www2.imm.dtu.dk/∼pch/HNO`.

8. **Possibly misleading image reconstruction.**

   In class, we will discuss techniques sometimes described as "regularization", intended to deal with ill-conditioned or ill-posed problems. For project ideas, you might consult the Wikipedia article about Tikhonov regularization, and then look at a recent talk (October 2009) by Dianne O'Leary, "Confidence and Misplaced Confidence in Image Reconstruction",

   `www.cs.umd.edu/∼oleary/talkview.pdf`.

9. **Fourier analysis (cell phones, DVDs, compact disks, etc.).**

   Fourier analysis is part of our daily lives, in touch-tone dialing, cell phones, and DVDs. A chapter of *Numerical Computing with Matlab* (2004) by Cleve Moler (founder of the Mathworks) discusses several aspects of Fourier analysis; this is available online at

   `www.mathworks.com/moler/fourier.pdf`.

   Moler's book can be purchased through SIAM.

10. **Strong words by Velvel Kahan on roundoff error.**

    As discussed in class, Professor William ("Velvel") Kahan, University of California, Berkeley, was the principal mover in creation of the IEEE floating-point arithmetic standard. His pungent commentary on some of the difficulties with performing assessments of roundoff errors can be found at

    `www.eecs.berkeley.edu/~wkahan/Mindless.pdf`

    Sections 2 (examples from Excel) and 8 suggest several projects.

11. **The IEEE Standard for Floating-Point Arithmetic (IEEE 754).**

    A project on this topic could start with the Wikipedia article, a 2-page 1998 interview with Velvel Kahan, and a 2008 paper by David Monniaux (in the context of software verification), "The pitfalls of verifying floating-point computations".

    `www.freecollab.com/dr-chuck/papers/columns/r3114.pdf`

    `hal.archives-ouvertes.fr/hal-00128124/en/`

    A nice paper (from 1991) by David Goldberg, *What every computer scientist should know about floating-point arithmetic*,

    `www.validlab.com/goldberg/paper.pdf`,

    contains a detailed discussion of IEEE arithmetic (and much more; see project 12).

12. **What computer scientists should know about floating-point arithmetic.**

    The 1991 paper by David Goldberg, *What every computer scientist should know about floating-point arithmetic*,

    `www.validlab.com/goldberg/paper.pdf`,

    presents an array of issues, including a discussion of the IEEE standard, but it also considers the effects of floating-point arithmetic on those who design compilers and languages.

13. **The pitfalls of interpolation at equally spaced points.**

    A phenomenon to be discussed in class is the fact (discovered by Runge) that interpolation of smooth functions by polynomials fitted at equally spaced points can produce a terrible result. A related project would investigate the reasons for this phenomenon (and possibly the related "Gibbs phenomenon"), and examine interpolation techniques based on using Chebyshev points instead.

The Wikipedia article on Runge's phenomenon is a good start, and there is an excellent discussion in Nick Trefethen's book *Spectral Methods in Matlab* (2000), published by SIAM.

14. **Safeguarded line searches in optimization.**

    In class, we discussed the high points of safeguarded methods for one-dimensional zero-finding (Newton, secant, etc.). The analogous problem in optimization is one-dimensional minimization, which is closely associated with line search methods in higher-dimensional optimization. Broadly speaking, the line search problem is: given a nonlinear function $f(x)$ of one variable and a point x_0, find a step δ such that $f(x_0 + \delta)$ is "sufficiently less" than $f(x_0)$. The seminal paper "Line search algorithms with guaranteed sufficient decrease", by Jorge Moré and David Thuente, *ACM Transactions on Mathematical Software* 20, 286–307 (1994), discusses some of the many complications in devising and implementing a reliable and efficient line search. A project could (for example) explain the difficulties and their significance, and also involve illustrative numerical experimentation.

15. **Iterative methods for linear systems.**

    We will have very little time in class to consider the huge array of *iterative* methods for solving linear systems, which are typically used when the matrix A is too large to be stored. These are discussed in Chapter 7 of Ascher and Greif, which provides additional references. A good survey (as of 1997) is the book *Iterative Methods for Solving Linear Systems*, by Anne Greenbaum, published by SIAM.

    A project on this topic could describe the motivation and details of several of the most popular methods, including numerical issues, experimentation, and analysis of the results.

16. **Calculation of special functions.**

    Special functions, such as sine, log, and exponential, are taken for granted by most users of scientific software, but (unknown to those users) they rely on years of research about how to compute special functions efficiently, with guaranteed accuracy. The 2007 book *Numerical Methods for Special Functions*, by Amparo Gil, Javier Segura, and Nico Temme, published by SIAM and available in the Courant Library, discusses a wide array of special functions and the associated clever, subtle numerics. A project on evaluation of special functions could investigate the techniques used in calculating a selection of special functions, discussing the tradeoffs in accuracy, computation time, and dependence on specific hardware features.