CS 222: Introduction to Scientific Computing
Spring 2001
Problem Set 3

Handed out: Wed., Mar. 7.

Due: Wed., Mar. 14 in lecture.

The policies for this (and other problem sets) are as follows:

- You should hand in your on-time problem set in lecture in the box at the front of the room on the day it is due. Problem sets handed in elsewhere (TA's office, Upson 303, etc.) will be considered late. See the next bullet.
- Late papers may be handed in up to 24 hours late. For instance, this problem set may be handed in up to 11:00 am on Mar. 15. You can hand in a late paper in Upson 303. Late papers get an automatic deduction of 10%. The full late penalty is applied even if you turn in part of the solution on time.
- Problem sets may be done individually or in teams of two. Put your name or names on the front page. Re-read the academic integrity statement on the web for the policy concerning working in larger groups.
- Problem sets count for 20% of the final course grade. The lowest scoring problem set will be dropped.
- You need Matlab for some of the questions. Matlab is available in the following CIT labs: Upson and Carpenter.
- If you need clarification for a homework question, please either ask your question in section, lecture, or office hours or else post it to the newsgroup cornell.class.cs222. The professor reads this newsgroup and will post an answer.
- Write your section number (like this: "Section 2") at the top of the front page of your paper, and circle it. This is the section where your graded paper will be returned. As a reminder, Section 1 is Th 12:20, Section 2 is Th 3:35, Section 3 is F 2:30, Section 4 is F 3:35.

1. Sometimes it is necessary to integrate a function over an infinite interval, say $[0, \infty)$. One strategy is to apply a change of variables, say $y = x/(x+1)$, so that the interval is reduced to a finite length. Then a quadrature rule can be used on that finite interval. Implement this strategy in Matlab for approximating

$$\int_0^\infty \exp(-x^2)\, dx$$

(In other words, first, on paper, figure out a change of variables that transforms the infinite interval to a finite interval, and then analytically substitute the change of variables before beginning any computation). Once the interval is finite, use the `quad` function in matlab to evaluate the integral. Try various tolerances. How accurate is

this method? Note that the exact value of this integral is $\sqrt{\pi}/2$. Does the tolerance that is input to `quad` correspond to the actual accuracy (that is, the difference between the output of `quad` and the true solution)?

Hand in a listing of m-files that you wrote, and comparison of the tolerance to the actual accuracy.

Note that in Matlab 5.3, this quadrature won't work because of endpoint singularity. In this case, offset the endpoints by a very small amount, say `eps` when calling `quad`. This problem is fixed in `quad` in Matlab 6.0.

2. "Sparse Gaussian elimination" means carrying out the GE process on a matrix that is already mostly zeroed out. In sparse Gaussian elimination, many operations can be skipped since the result of the operation would be an update of zero anyway. Consider an $n \times n$ matrix A that is upper triangular except for the last row (i.e. $A(i,j) = 0$ whenever $j < i < n$). Write out a Matlab function (on paper) for Gaussian elimination with partial pivoting on a matrix with this special structure. Analyze the number of flops accurate to the leading term, which should come out to be quadratic. Your function should return `[L,U,piv]` as in lecture.

3. Cubic spline interpolation is generally a well-conditioned problem, except in the case that two of the knots are very close. In this case, it is ill-conditioned. Carry out two or three experiments using the cubic spline functions from last problem set to demonstrate these claims. To talk about conditioning, you must have a well-defined notion of what is the "input" and "output" are. The "input" for the cubic spline problem is the list of datapoints and the values for the end-conditions. The "output" for the cubic spline function is the interpolant. For measuring sensitivity of the output, you can treat the output as a long vector by evaluating the interpolant at a grid of closely space points and storing the interpolant values at those points in a vector. Hand in a description of your experiments, and what you discovered about sensitivity of the output with respect to the input.