CS 322: Introduction to Scientific Computing
Spring 2003
Problem Set 2

Handed out: Wed., Feb. 12.

Due: Wed., Feb. 19 in lecture.

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

- You should hand in your on-time problem set at the beginning or end of lecture on the day it is due in the box at the front of the room. 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 on Feb. 20. 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.cs322. The professor reads this newsgroup and will post an answer.
- Write your names and section number (like this: "Section 2") at the top of the front page of your paper. 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. Download the following m-files from the textbook: CubicSpline.m, pwCEval.m, Locate.m. They are posted on the course home page.

   Use these m-files to answer the following question. Interpolate a not-a-knot cubic spline through the function $f(x) = \sqrt{1-x^2}$ at six evenly spaced breakpoints in $[-1, 1]$, i.e., at breakpoints $[-1; -.6; -.2; .2; .6; 1]$ (using CubicSpline). Then evaluate the difference between the true function and the interpolant at 100 evenly spaced x-coordinates in the same interval (using pwCEval to evaluate the spline), and print out the maximum value of the absolute difference.

   Now repeat all of this for the function $g(x) = \sqrt{1.2 - x^2}$ on the same interval.

   You should discover that the interpolant of $g(x)$ is much more accurate than the interpolant of $f(x)$. Write a few sentences or formulas to explain why there is such a big difference in accuracy. [Hint for the explanation: consider M_4 mentioned on p. 129 of the text.]

Hand in: a listing of your m-file, two plots, each one showing the true function and interpolant on the same axes (i.e., two plots each one analogous to the plot you made for Q3 of PS1), a printout of the two numerical differences in the two test cases, and the explanation requested in the preceding paragraph.

2. Suppose we want a piecewise polynomial interpolant to data points (x_1, y_1), $\ldots$, (x_n, y_n), $x_1 < x_2 < \cdots < x_n$, that has a continuous first derivative. Argue that piecewise quadratic is the lowest order that will work. How many end conditions are necessary to uniquely determine the piecewise quadratic interpolant? (Answer in a non-rigorous fashion by counting coefficients and constraints as in lecture.)

3. Download the m-file pwQSpline.m from the course web page, which constructs the piecewise quadratic spline described in the previous question. The comments explain how it works.

   (a) Write an m-file function pwQEval.m that evaluates a piecewise quadratic at m datapoints. In other words, pwQEval takes as input the spline interpolant output by pwQSpline and also an additional vector of x-coordinates, and evaluates the spline at those coordinates, returning the corresponding vector of y-coordinates. You can use the function pwCEval on the course web page for inspiration, but pwCEval is not vectorized whereas yours should be. You can use VLocate.m from the course webpage, which is a vectorized version of Locate.m. Your routine should be well documented with comments for full credit. [Hint: Assuming you use vector subscripts, this problem can be solved with a function that has just two statements in its body.]

   (b) Test out pwQSpline/pwQEval versus CubicSpline/pwCEval on the function $f(x) = \sin x$ over the interval $[0, \pi]$. Try them out with both exact first-derivative end conditions (i.e., the true value of the derivative, which is $f'(0) = 1$ for the quadratic case and $f'(0) = 1, f'(\pi) = -1$ for the cubic case) and an erroneous left-end condition of 0.7 (and -0.7 at the right end for the cubic case). Use ten evenly spaced points to compute the interpolant, and then 100 to plot it as in Q1. The quadratic interpolant in the case of incorrect end conditions has an undesirable property. Explain what is undesirable about the quadratic interpolant compared to the cubic spline. The issue is what happens away from the ends. [Note: This drawback of quadratic interpolation is the main reason why people rarely use it, but instead prefer to use cubics.]