Problem set 1

Due Friday, July 2 at 1:00 pm

Question 1 (25 points) How many flops are required to evaluate once the Lagrangian form of the polynomial of degree $n-1$ interpolating n points? Give the exact flop count, assuming that each subtraction and multiplication shown is performed (i.e. ignore the fact that you could save some flops by, for example, only computing $(x - x_1)$ once). Compare this to the total number of flops required to calculate the coefficients of the same polynomial in the Newton representation and evaluate it once using Horner's method (leading term only for this comparison). Which method requires the fewest flops? How many flops (leading term only) would each method require to compute the polynomial and evaluate it n times? How about n^2 times? For full credit you must show all your work. Assume that InterpN is used to calculate the coefficients in the Newton representation.

Question 2 (25 points) P2.1.1 in Van Loan. Hand in a print out of your matlab function and email a copy of the m-file to gunsri@cs.cornell.edu. Please send Gun only one email submission per assignment.

Question 3 (25 points) Consider the degree $n-1$ polynomial, $p_{n-1}(x)$, interpolating the function $f(x) = \cos(x)$ on the interval $[-\pi, \pi]$ at n equally spaced points. Give an upper bound on the error $|f(x) - p_{n-1}(x)|$ in this approximation on the interval $[-\pi, \pi]$. What happens to your upper bound as $n \to \infty$? If you want the error in your approximation to be less than 10^{-5}, how large must you choose n? For the value of n you have determined, plot $f(x) - p_{n-1}(x)$ in the interval $[-\pi, \pi]$. Use your function InterpV from question 2 to calculate the coefficients of the interpolating polynomial. Use any reasonable u, v that you wish and Horner's method of evaluation. What do you observe? Hand in your calculations, plot, and print out the matlab code used to generate the plot.

Question 4 (25 points) As in section 2.3, consider interpolating the function

$$f(x) = \frac{1}{1 + 25x^2} \tag{1}$$

at evenly spaced points in the interval $[-1, 1]$. Plot the function and it's interpolating polynomial on the same axis for $n = 5, 11, 15, 21$ points (so you will generate four plots). Are the errors behaving as they did in question 3? Now, instead of using evenly spaced points, use the points

$$x_k = -\cos\left(\frac{(2k-1)\pi}{2n}\right) \quad \text{for } k = 1, ..., n. \tag{2}$$

These are the Chebyshev points for the interval $[-1, 1]$, and are more dense near the ends of the interval than near the middle. Plot function 1 and the polynomial that interpolates it at the Chebyshev points for $n = 5, 11, 15, 21$ points, generating four more plots. What do you notice? In general, if you interpolate at the Chebyshev points a function f such that f and f' are continuous on $[-1, 1]$ then as you increase the number of points (and the degree of the interpolating polynomial increases) the sequence of polynomials generated converges uniformly to f on $[-1, 1]$. Hand in your plots and analysis together with a print out of the matlab code used to generate them.