Introduction to Numerical Simulation (Fall 2003)
Problem Set #3 - due September 26th

Note: This problem set has only one problem, in order to give everyone a little time to catch up. It does not cover the difference between modified and unmodified QR. We will cover that issue in the next problem set.

1) In this problem, you will write your own factorization algorithm based on row orthogonalization. Such an approach makes it simpler to do numerical pivoting and is more easily compared to LU factorization.

a) Write a matlab program for solving $Mx = b$, where M is square, which is based on making the *rows* of M into a set of orthonormal vectors. Please remember to normalize the rows so that they correspond to vectors of unit length. You may find it helpful to look at our *qr.m* matlab program which is based on orthogonalizing columns.

b) Consider using your row orthogonalization algorithm to solve a tridiagonal matrix (a tridiagonal matrix is one whose only nonzero entries are $M_{i,i}, M_{i,i+1}$, and $M_{i,i-1}$). Compare the operation counts for sparse orthogonalization and sparse LU factorization of an $n \times n$ tridiagonal matrix.

c) If a row that is about to be normalized in your row orthogonalization algorithm corresponds to a vector with a very short length, then the normalization will increase the size of those matrix elements, contributing to matrix growth and worsening round-off errors. It might be better to exchange the unnormalized row with one of the other rows that has yet to be normalized, preferably the one with the largest associated length. Such a row exchange is analogous to the row exchange used in partial pivoting for LU factorization. Please modify your matlab row orthogonalization program to perform such a pivoting algorithm.

d) What will happen if you apply your row orthogonalization algorithm with pivoting to a problem in which M is singular? Please test your code on a singular example.