Labwork Set # 2 – Math 371 – Fall 2009 Due date: 9/TBD/2009

0. [1 point each part] Number the lines of code in the lutx program on pages 60-61 in the Moler text, excluding comments and spaces and including the function header. Please answer the following questions precisely. You may want to use MATLAB to experiment with an input matrix A so that you can see more easily what is happening.

   (a) What does line (5) ([r,m] = ...) do? What does it return?

   (b) Why is line (6) necessary?

   (c) What exactly does the **vector** p represent?

   (d) What does line (13) do? Why??

   (e) If you ask MATLAB to print the final matrix A that results at the end of the lutx code, what is contained in this matrix? Is it the reduced form of A? Something else?

1. [3 points] Modify the lutx function so that it uses explicit for loops instead of MATLAB vector notation. For example, one section of your modified program will read

   for i = k+1:n A(i,k) = A(i,k)/A(k,k); end

   Use the built in MATLAB functions tic and toc to compare the execution time of your modified lutx program with the original lutx program by finding the sizes of the matrices for which each of the programs takes about 10 seconds on your computer.

2. [4 points for function] Write a MATLAB function lutx_matrix that will solve by LU decomposition the equation $Ax = b$, where x and b are $n \times r$ **matrices** and A is an $n \times n$ matrix. You should use lutx to decompose A (and it should be called only once!). Once you have this written, use lutx_matrix to:

   (a) [1 point] Check the solution you found in problem 2 of the written homework.

   (b) [2 points] Solve the systems where

      I. A = [1 2 3; 4 5 6; 7 8 9.001], b = [1 2 3; 1.1 2 3]'

      II. A = [1 2 3; 3 7 4; 8 3 6], b = [1 2 3; 1.1 2 3]'

   (c) [2 points] What do you notice about the results of part (b)? What is it about the two different matrices A that lead to such different results?

3. [3 points] Use the code you have written in problem 2 to write a function X = myinv(A) that computes the inverse of a square matrix A. (You should not use the MATLAB backslash operator or inv function) Test your function by comparing the inverses it computes with thte inverses obtained from the built-in inv(A) on a few test matrices.