AMSC/CMSC 460 Homework 2: Due in class on Wednesday, Oct 4, 2016

1. Write Matlab functions that implement a matrix vector product

b=Ax
using each of the ways discussed in class. You can assume A is a n × n matrix

```
b = MatVecProd_a(A,x)%Using a pair of nested loops that access the matrix
entries in column major order

b = MatVecProd_b(A,x)%Using a pair of nested loops that access the matrix
entries in row major order

b = MatVecProd_c(A,x)%Using a single loop which access the matrix entries a
column vector at a time

b = MatVecProd_d(A,x)%Using a single loop which access the matrix entries a
row vector at a time

b=A*x; % Using Matlab's built in matrix vector product
```

Compare the time taken by these implementations against the size of the matrix n for each of these implementations by reporting the average over 10 runs. Use the Matlab functions `tic` and `toc` for timing.

Plot your results for each of the methods for n=128, 256, 512,1024, 2048, 4096. (You need not do the 4096 case, if your computer is too slow on the 2048 one). Make the axes log-log. Also use both lines and points for the plot. Finally you can include a legend, label the axes, and give a title. Explain your results.

2. Solve the following system of equations by performing first the LU=PA decomposition by hand (with pivoting), and then doing forward elimination and back substitution

$$\begin{bmatrix} 1 & -3 & 1 \\ 2 & -8 & 8 \\ -6 & 3 & -15 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ 9 \end{bmatrix}$$

Do these problems from your book. (http://www.mathworks.com/moler/lu.pdf)

3: Problem 2.7: Computing Determinants. Compare your results on a couple of test cases, in which you compute the determinant using your function, and using Matlab's built in function `det`.

4: Problem 2.8: Modify `lutx` to a new function `explutx` so that all Matlab trickery -- vectorized statements -- are removed and explicit statements are used. Compare the two programs to make sure that they produce the same results on the same random 10x10 matrix.

5. Problem 2.11: Using lutx to compute the inverse of a matrix. Again compare your code to the builtin Matlab function `inv` on a random 10x10 matrix.