Math 329 Unit 2 Fall 2014

All the problems this week involve the computer. You should zip the files and send them to me. The files should include: lusolve.m test_lusolve.m unit2.m output_unit2.txt. The unit2.m file should run the test_lusolve script and then proceed to answer questions 3-5.

1. LU-decomposition of A allows us to solve $Ax = P'LUx = b$ by solving three fast problems instead of one slow problem (the slow part of the problem is done while finding the LU–decomposition). This is helpful if you have many right hand sides b for which to solve $Ax = b$. You can find P, L and U once and then only do the fast parts for each b.

   Download the file **lusolve.m** from our webpage and adapt it to solve for x given L, U, P and b. It currently solves $P'Ux = b$ given U, P and b. It contains two internal functions that perform permutation and back–substitution as well as the main function which returns x. You will need to create a front substitution function (start with the ideas from the back–substitution routine provided) and adjust the main function to handle inputs L, U, P and b.

2. Write a test script (test_lusolve.m) which uses the lusolve.m function. Consider the matrix A from later in this assignment and let b be the first column of A. (Note that you know what the solution x should be.) The Matlab command [L,U,P] = lu(A); finds the matrices L, U, P for a given matrix A. Use it to compute the factorization and then call your lusolve function to find the solution x.

   Next create a 100x100 matrix A with random entries using rand(100,100). Similarly create a random right hand side b. Factor, solve and then check your solution by printing the norm of the residual.

3. Set up a linear system of equations which find the parabola through the points (2,3) (3,7) and (4,5). Set up column vectors for the x-values (xpts) and the y-values (ypts). You want coefficients a, b, and c so that $y = ax^2 + bx + c$ for each of these points. Three equations and three unknowns. The unknowns are a, b, c so the coefficients are `x^2`, `x` and `1` respectively. The matrix for the equations can be created as `A=[xpts.^2 xpts ones(3,1)];` (What is the right hand side?) Use Matlab's backslash operator to solve for the unknowns.

   Create a plot of the parabola using 120 points for x between 1 and 6. On the same figure plot the 3 points as circles. If `coeff` is the vector of coefficients, then code like the following should work:

```
x=linspace(1,6,120)';  % Transpose to make a column
y=coeff(1)*x.^2 + coeff(2)*x + coeff(3);
plot(xpts, ypts, 'o' )  % formatting string 'o' draws circles at each point
hold on     % future plot commands do not erase the figure
plot(x,y)   % no formatting string -> connect the points
hold off    % future plot commands erase the figure
title('Parabola through 3 points - Unit 2')
xlabel('x')
ylabel('y')
```

4. Find the sixth order polynomial through the points (0,1) (1,3) (2,2) (3,1) (4,3) (5,2) and (6,1). Plot both the polynomial (using 100 points from -1 to 7) and the given points (using circles) on the same graph. Do the plots agree?

   Plotting curves to check agreement is sometimes called using the "eyeball norm". What "is" the eyeball norm of the error for this example?

5. Consider the following matrices:

$$\mathbf{A} = \begin{bmatrix} 2 & 4 & -6 \\ 1 & 5 & 3 \\ 1 & 3 & 2 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 4424 & 978 & 224 & 20514 \\ 4424 & 978 & 224 & 54 \\ 224 & 978 & 1 & 54 \\ 54 & 14 & 4 & 224 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 20514 & 4424 & 978 & 224 \\ 4424 & 978 & 224 & 54 \\ 978 & 224 & 54 & 14 \\ 224 & 54 & 14 & 4 \end{bmatrix} \quad \mathbf{D} = \begin{bmatrix} 1 & 2 & 3 \\ 40 & 50 & 60 \\ 75 & 85 & 95 \end{bmatrix}$$

   For each of the matrices, do the following:

   a) Print the condition number for the matrix.

   b) Set up a right hand side b with corresponding solution being a column vector of ones. So the true solution x_{true} is all ones. The solve for the computed solution x using the backslash operator. Print the norm of the absolute error and the norm of the residuals.

   Your print statements can look similar to:

```
fprintf('For Matrix A:')
fprintf('a) Condition Number: %18.12e\n',condition)
fprintf('b) Absolute error norm: %18.12e\n',norm_abs_err);
fprintf('   Residual error norm: %18.12e\n',norm_resid));
```