Homework 3

Linear Systems and LU Factorization

1. (adapted from I.3 5) Show possible number of solutions for each linear systems $A_i\mathbf{x} = \mathbf{b}$, whose left hand side A_i is given as following:

$$A_1 = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \qquad A_2 = \begin{pmatrix} 1 & 2 & 6 \\ 2 & 4 & 7 \\ 3 & 6 & 5 \end{pmatrix} \qquad A_3 = \begin{pmatrix} 1 & 3 \\ 2 & 4 \\ 5 & 7 \end{pmatrix} \qquad A_4 = \begin{pmatrix} 1 & 2 & 4 \\ 3 & 6 & 7 \end{pmatrix}$$

2. For each of the following statements, indicate whether the statement is true or false.

   T/F If a triangular matrix has a zero on its main diagonal, then it is necessarily singular.

   T/F The product of two upper triangular matrices is upper triangular.

   T/F Once the LU factorization of an $n \times n$ matrix has been computed to solve a linear system, then subsequent linear systems with the same matrix but different right-hand-side vectors can be solved in $O(n^2)$ time without refactoring the matrix.

3. (adapted from I.4 3) What lower triangular matrix E puts A into upper triangular form $EA = U$? Multiply by $E^{-1} = L$ to factor A into LU:

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 6 & 4 & 2 \\ 0 & 3 & 5 \end{pmatrix}$$

4. (adapted from I.4 4) LU factorization is sometimes derived using elimination matrices such as E_1 and E_2 below, which consist of the identity matrix plus the negatives of sub-diagonal elements of one column of L. These matrices have the nice property that their inverses are trivial to compute, e.g., $E_1^{-1} = \begin{pmatrix} 1 & & \\ a & 1 & \\ b & 0 & 1 \end{pmatrix}$, and $E_2^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & c & 1 \end{pmatrix}$

   This problem shows how the elimination matrix inverses multiply to give L. You see this best when $A = L$ is already lower triangular with 1's on the diagonal. Then $U = I$:

   Multiply $A = \begin{pmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ b & c & 1 \end{pmatrix}$ by $E_1 = \begin{pmatrix} 1 & & \\ -a & 1 & \\ -b & 0 & 1 \end{pmatrix}$ and then $E_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -c & 1 \end{pmatrix}$.

   (a) Multiply $E_2 E_1$ to find the single matrix E that produces $EA = I$.

   (b) Multiply $E_1^{-1} E_2^{-1}$ to find the matrix $A = L$.

   The multipliers a, b, c are mixed up in $E = L^{-1}$ but they are perfect in L.

5. (adapted from I.4 8) Tridiagonal matrices have zero entries except on the main diagonal and the two adjacent diagonals. Factor these into $A = LU$. Symmetry further produces $A = LDL^T$, where D is a diagonal matrix.

$$A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 2 \\ 0 & 2 & 2 \end{pmatrix} \qquad A_2 = \begin{pmatrix} a & a & 0 \\ a & a+b & b \\ 0 & b & b+c \end{pmatrix}$$

6. (adapted from I.4 11) In some data science applications, the first pivot is the *largest number* $|a_{ij}|$ in A. Then row i becomes the first pivot row $\mathbf{u}_1^*$. Column j is the first pivot column. Divide that column by a_{ij} so ℓ_1 has 1 in row i.

Then remove that $\ell_1 \mathbf{u}_1^*$ from A.
This example finds $a_{22} = 4$ as the first pivot ($i = j = 2$). Dividing by 4 gives ℓ_1:

$$\begin{pmatrix} 1 & 2 \\ 3 & \mathbf{4} \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1 \end{pmatrix} \begin{pmatrix} 3 & \mathbf{4} \end{pmatrix} + \begin{pmatrix} -1/2 & 0 \\ 0 & 0 \end{pmatrix} = \ell_1 \mathbf{u}_1^* + \ell_2 \mathbf{u}_2^* = \begin{pmatrix} 1/2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 4 \\ -1/2 & 0 \end{pmatrix}$$

For this A, both L and U involve permutations. P_1 exchanges the rows to give L. P_2 exchanges the columns to give an upper triangular U. Then $P_1 A P_2 = LU$.
Permuted in advance $P_1 A P_2 = \begin{pmatrix} 1 & 0 \\ 1/2 & 1 \end{pmatrix} \begin{pmatrix} 4 & 3 \\ 0 & -1/2 \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$.
Question for $A = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$: Apply complete pivoting to produce $P_1 A P_2 = LU$.

7. (Matlab/Octave Programming. Note: If you prefer, you may translate the code below to Python and use Python and numpy instead.) Consider the Matlab function

```
function [L,U] = lu_cs210(A)
n = size(A,1);
L = zeros(size(A));
U = zeros(size(A));
A2 = A;
for k = 1:n
    if A2(k,k) == 0
        'Encountered 0 pivot. Stopping'
        return
    end
    for i = 1:n
        L(i,k) = A2(i,k)/A2(k,k);
        U(k,i) = A2(k,i);
    end
    for i = 1:n
        for j = 1:n
            A2(i,j) = A2(i,j) - L(i,k)*U(k,j);
        end
    end
end
```

(a) Try this code on the matrix `A = [ 10^-20 1; 1 1 ]`. What are L and U (You may need to access the elements as e.g., U(1,1) to see the values more accurately)?

(b) Try Matlab's `lu` function on the same matrix. What result do you get?

(c) Modify the code above to implement partial (row) pivoting. Try it on the matrix A above. What factors L and U do you get? What permutation? Try your code on `A2 = [ 1 2 3; 3 2 1; 1 -1 0]`. What do you get for L, U, and P? Compare with Matlab's `[L,U,P] = lu(A2)`. Include your code with your submission.