PROBLEM SET

Assignment 4

Math 4330, Spring 2017

March 30, 2017

- Write all of your answers on separate sheets of paper. You can keep the question sheet.
- You **must** show enough work to justify your answers. Unless otherwise instructed, give exact answers, not approximations (e.g., $\sqrt{2}$, not 1.414).
- This problem et has 3 problems. There are **300 points total**.

Good luck!

Problem 1. Write a program to solve a linear system $Ax = b$ using Gaussian Elimination with scaled partial pivoting and backsubstitution. You can just type in the system matrices at the beginning of the file. Store the matrices as numpy arrays.

Recall that scaled partial pivoting works like this. Suppose the matrix we're work on is an $n \times n$ matrix W. (Leaving b out of the question.) Suppose we're working on column i of W, so we want to zero out the elements below position W_{ii}.

First, for each row j with $i \leq j \leq n$ find the maximum absolute value of the element in or to the right of column i, call it $d(j)$. In other words,

$$d(j) = \max\{|W_{jk}| \mid k = i \ldots n\}.$$

Then find j in the range $j = i \ldots, n$ that maximizes the quantity

$$\frac{|W_{jj}|}{d(j)}$$

Now switch rows i and j. Proceed to zero out the elements in the rows below the diagonal in column i.

Problem 2. In this problem, we'll do the LU decomposition. Again, you can type in the system at the top of the file.

Write a function that takes and $n \times n$ invertible matrix A and returns matrices P, L and U so that $A = PLU$ where P is a permutation matrix, L is lower trianglar with 1's on the diagonal and U is upper trianbular. Use scaled partial pivoting.

Write a function that takes a vector b and uses the decomposition PLU to solve the system $Ax = b$ using backward substitution and forward substituion.

Store the matrics as numpy arrays.

Problem 3. Work an LU decomposition problem by hand, say 3×3 or 4×4. You don't have to use any particular pivoting strategy. Store the multiplers from the row operation in the correspoind row of the column you're working on. There postions are known to be zero in U, so we can reuse the space.

After you finish the computation, compare L and the part of U below the diagonal. What happens?
