PROJECT 3

Math 441

Due: Thursday, Sept. 25

All "by hand" calculations require that you show your work. All MATLAB code and output must be turned in.

1. Do problem 1.7.10.
   (a) Calculate the determinants by hand, and show your work.
   (b) Do this part by hand. An operation of type 1 is defined on page 70 of your textbook.
   (c) Do this part by hand.
   (d) Call your answer **xstar**, and check it in MATLAB by entering **U\ y** and **A*xstar**.
2. Do problem 1.7.18. Feel free to use MATLAB to do the forward and backward substitutions: Enter **y=L\b** and **x=U\y**. Or do it all at once via **x=U\(L\b)**.
3. Do problem 1.7.34.
4. Do problem 1.7.35.
5. Do problem 1.7.45.
6. Consider the system of equations $Ax = b$ that we talked about in class, where $A = \begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, and ϵ is some small number.
   (a) Give the LU decomposition of A for any ϵ.
   (b) Show that the true solution to $Ax = b$, is $x_{true} = \begin{pmatrix} \frac{1}{\epsilon-1} \\ \frac{1}{1-\epsilon} \end{pmatrix}$.
   (c) Write a MATLAB program (and save it as some m-file) that finds the numerical solution to this system by implementing the following steps.
   - Set **epsilon = 10^(-1)** and **b=[1 0]'**.
   - From your answer to #6a, set L and U equal to the appropriate 2x2 matrices.
   - Use the LU decomposition to solve $Ax = b$ using forward-substitution and then backward-substitution. Execute **y=L\b** and **x=U\y**; or execute **x=U\(L\b)**.

   (d) Now make a new m-file in which you will modify your code from #6c to investigate the relative error, $\frac{||x_{true} - x_{approx}||}{||x_{true}||}$, as $\epsilon \to 0$, where x_{approx} is the solution that your code gives. "Wrap" your code with a **for**-loop like this:

```
for i=1:20
  epsilon = 10^(-i);
  xtrue = 1/(1-epsilon) * [-1 1];

  PUT YOUR CODE FROM PART (C) HERE TO GET xapprox

  relerr(i) = norm(xtrue - xapprox)/norm(xtrue);

end
```

(e) Plot the relative error as a function of i, the negative of the exponent of $\epsilon = 10^{-i}$ by entering **semilogy(1:20,relerr)**. Label the horizontal and vertical axes appropriately. Explain what you observe in the plot. Include this plot in your solutions.

(f) To what vector does x_{approx} "converge"? Why is this vector different than x_{true}?

(g) Solve $\begin{pmatrix} 10^{-20} & 1 \\ 1 & 1 \end{pmatrix} x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ for x in MATLAB using Gauss Elimination by entering **A\b**. Why does this command yield the correct solution, while x_{approx} in #6f is incorrect?

7. Do problem 1.8.4. Do the Gauss Elimination with pivoting by hand, but feel free to use MATLAB for the forward and backward solves.

8. Do problem 1.8.7.

9. (a) Write a MATLAB program which, given any $n \times n$ matrix A, calculates A^{-1}. You'll need a for-loop so that your program cycles through the n columns of I (call them e_i) and then solves $Ax_i = e_i$ for x_i using the LU decomposition (as well as the permutation matrix P which implements any row pivots). You can construct A^{-1} column by column by executing something like **Ainv(: , i)=** ...

   (b) Use your program to calculate the inverse of the 4×4 matrix from #1 (Problem 1.7.10 in your text). Call this inverse **Ainvapprox**.

   (c) Calculate the inverse of the 4×4 matrix from #1 (Problem 1.7.10 in your text) using MATLAB's **inv( )** command. Call this inverse **Ainvtrue**.

   (d) Now calculate the relative error:

      i. In MATLAB, enter: **abs((Ainvtrue - Ainvapprox)./Ainvtrue)**, which is the component by component relative error.

      ii. A scalar measure is found by entering **norm(Ainvtrue - Ainvapprox)/norm(Ainvtrue)**. We will talk about what the norm of a matrix is in Chapter 2.

   (e) Explain any **Inf**'s or **NaN**'s which you get.

   (f) Based on these two different measures, does your program accurately calculate A^{-1}? Explain.

10. Do problem 1.8.23. Use the fact that

$$1^2 + 2^2 + ... + (n-1)^2 + n^2 = \sum_{i=1}^{n} i^2 = \mathcal{O}\left(\frac{n^3}{3}\right).$$

Recall that we used this result when we computed the total FLOP count for Gaussian Elimination.