COMPUTER SCIENCE 349A, FALL 2018
ASSIGNMENT #4 - 20 MARKS

DUE FRIDAY NOVEMBER 2, 2018 (11:30 p.m. PST)

This is a really large class and the logistics of grading assignments are challenging. Me and the markers require your help in making this process go smoothly. Please ensure that your assignments conform to the following requirements - any violation will result in getting a zero for the particular assignment.

- All assignments should be submitted electronically through the ConneX course website and shoud be **SINGLE PDF FILES**. No other formats will be accepted. Handwritten answers are ok but they will need to be scanned and merged into a single pdf file together with any code examples and associated plots.
- The assignment number, student name and student number should be clearly visible on the top of every page of your assignment submission.
- **PLEASE DO NOT COPY THE ASSIGNMENT DESCRIPTION IN YOUR SUBMISSION**
- The asnwers to the questions should be in the same order as in the assignment specification.
- Some of the questions of the assignments are recycled from previous years but typically with small changes in either the description or the numbers. Any submission that contains numbers from previous years in any questions will be immediately graded with zero.
- Any assignment related email questions should have a subject line of the form CSC349A Assignment X, where X is the number of the corresponding assignment.
- The total number of points for this assignment is 20.

1. Let

$$A = \begin{bmatrix} 0 & -2 & -2 & -4 \\ -1 & -1 & 1 & 0 \\ 2 & 4 & -2 & 0 \\ 1 & 1 & -1 & 0.5 \end{bmatrix}$$

   (a) **(4 points)** Use Gaussian elimination with partial pivoting to compute the fourth column vector of A^{-1} . Do this by hand computation, no programming. (Do not compute all of A^{-1}.) Explicitly interchange rows as required, and show all of the derived linear systems and the back substitution. Do not do any scaling.

   (b) **(2 points)** From the computations in (a), what is the determinant of A? (Show explicitly how you obtain this, including how the sign of the determinant is obtained.)

2. (a) **(2 points)** Let A be an $n \times n$ nonsingular lower triangular matrix with all of its nonzero entries on three diagonals of the matrix: the main diagonal, the first sub-diagonal and the second sub-diagonal; that is,

$$\begin{bmatrix} a_{1,1} & & & & & & \\ a_{2,1} & a_{2,2} & & & & & \\ a_{3,1} & a_{3,2} & a_{3,3} & & & & \\ & a_{4,2} & a_{4,3} & a_{4,4} & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & a_{n-1,n-3} & a_{n-1,n-2} & a_{n-1,n-1} & \\ & & & & a_{n,n-2} & a_{n,n-1} & a_{n,n} \end{bmatrix}$$

   Note that A is nonsingular if and only if all of the entries on its main diagonal are nonzero, that each row of A contains at most 3 nonzero entries, and that entries on the first sub-diagonal and the second sub-diagonal may be nonzero or zero. For example, if $n = 6$, then

$$A = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ -2 & 1.1 & & 0 & 0 & 0 \\ 2.3 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2.2 & 3 & -2.11 & 0 & 0 \\ 0 & 0 & 0 & 2 & 3.3 & 0 \\ 0 & 0 & 0 & 2.4 & 1.1 & -2.5 \end{bmatrix}$$

   is such a lower triangular matrix.

   Let $b = [b_1, b_2, \ldots, b_n]^T$ denote a (column) vector with n entries. An $n \times n$ system of linear equations $Ax = b$, where A is as described above, can be efficiently solved by forward substitution, which is similar to back-substitution but starts with the first equation. That is, the first equation can be used to solve for x_1, the second equation can be used to solve for x_2, the third equation for x_3, and so on.

Fill in the blanks in the following algorithm (use pseudocode, not MATLAB) so that it will solve such a system of linear equations.

$x_1 \leftarrow$ _____

$x_2 \leftarrow$ _____

for $i =$ _____

_____ $\leftarrow$ _____

end

DELIVERABLES: Your complete pseudocode, it can be hand written or typed.

(b) **(2 points)** Give a floating-point operation (flop) count for this forward substitution algorithm. That is, determine the total number of floating-point additions, subtractions, multiplications and divisions (as a function of n) that this algorithm will execute.

DELIVERABLES: All the steps in your count, including your reasoning.

(c) **(4 points)** Convert your pseudocode to MATLAB code and write a function that takes as input the nonsingular lower triangular matrix A, and column vector b and returns the results of "forward" substitution. The input matrix should be the full matrix including the zeros. Show how your function can be called and what the result is for the following matrices A and b:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 3 & 0 & 0 \\ 4 & 5 & 6 & 0 \\ 0 & 7 & 8 & 9 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 5 \\ 15 \\ 24 \end{bmatrix}$$

DELIVERABLES: A copy of the m-file function, a diary of the steps taken to solve for the given A and b and the solution x.

3. The following system of linear equations $Ax = b$

$$\begin{bmatrix} -0.2345 & 2.107 \\ 0.1234 & -1.115 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -2.345 \\ 1.001 \end{bmatrix}$$

has exact solution $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 345.404... \\ 37.329... \end{bmatrix}$. This problem is **ill-conditioned**; for example, if the (2,2)-entry of A is perturbed to be -1.111, then the exact solution of this perturbed linear system is

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 943.861... \\ 103.934... \end{bmatrix}$$

In general, it is very difficult to determine an accurate computed solution to an ill-conditioned linear system using floating-point arithmetic (even if a good algorithm such as Gaussian elimination with partial pivoting is used). This is illustrated by the following computation.

(a) **(4 points)**
Use base 10, precision $k = 4$, idealized rounding floating-point arithmetic and Gaussian elimination with partial pivoting to solve the above linear system. Specifically, using floating-point arithmetic, show the computed approximations for the multiplier, and the entries a_{22} and b_2 of the first derived system (that is, when A has been reduced to upper triangular form). (There is no need to compute a_{21} as it will become 0.) Then, using floating-point arithmetic and back-substitution to compute x_2 and x_1. Explain the results in context of the above.

DELIVERABLES: Your floating-point calculations and results plus the explanation.

(b) **(2 points)** Use the MATLAB built-in function *cond* to compute a condition number for the above coefficient matrix A. We wont be discussing the details of this condition number in this course, but there is some discussion of this on page 294 of the 7th edition of the textbook (page 290 of the 6th edition). Roughly speaking, if this condition number is between 1 and 10, then A is well conditioned, and if this condition number is greater than 1000, then A is ill-conditioned.

DELIVERABLES: Show your MATLAB input and output.