# Spring 2022: Numerical Analysis
# Assignment 2 (due Feb. 27, 2022 at 11:59pm ET)

**Handing in code listings** When code listings are required, please add them when you upload your homework on Gradescope. We will try to look at them and point out where things could be improved–being able to write useful code is important for this class and in real life. In your programs, use meaningful variable names, try to write clean, concise and easy-to-read code and use comments for explanation.

1. **[Properties of LU factorization, 1+2pt]** We study basic properties of the LU-factorization.
   (a) Give an example of an invertible $3 \times 3$ matrix that does not have any zero entries, for which the LU decomposition without pivoting fails.
   (b) Show that the LU factorization of an invertible matrix $A \in \mathbb{R}^{n\times n}$ is unique. That is, if
   $$A = LU = L_1U_1$$
   with upper triangular matrices $U$, $U_1$ and unit lower triangular matrices $L$, $L_1$, then necessarily $L = L_1$ and $U = U_1$. You can use the results we discussed in class about products of lower/upper triangular matrices, and their inverses.

2. **[LU for transposed matrix, 2+2pt]** Let $n \geq 2$. Consider a matrix $A \in \mathbb{R}^{n\times n}$ for which every leading principal submatrix of order less than $n$ is non-singular.
   (a) Show that $A$ can be factored in the form $A = LDU$, where $L \in \mathbb{R}^{n\times n}$ is unit lower triangular, $D \in \mathbb{R}^{n\times n}$ is diagonal and $U \in \mathbb{R}^{n\times n}$ is *unit* upper triangular.
   (b) If the factorization $A = LU$ is known, where $L$ is unit lower triangular and $U$ is upper triangular, show how to find the LU-factors of the transpose $A^T$. Note that our requirement for an LU-factorization is that $L$ is *unit* lower triangular, and $U$ is upper triangular.

3. **[Cholesky factorization, 3+2pt]**
   (a) A matrix $A$ is called symmetric if $A = A^T$. If we compute $A = LDU$ for a symmetric matrix $A$, and find that each diagonal entry of $D > 0$, argue that $A$ can be factored as $A = RR^T$ and find $R$ in terms of $L, D, U$. Note that the factorization $A = RR^T$ can only be done for some types of symmetric matrices (specifically symmetric-positive-definite (spd) matrices), and is referred to as a Cholesky factorization.
   (b) The algorithm to find the Cholesky factorization of an $n \times n$ matrix $A$ is as follows:
   for $j = 1, \dots, n$
      i. $r_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} r_{jk}^2}$
      ii. for $i > j$: $r_{ij} = \frac{1}{r_{jj}} \left( a_{ij} - \sum_{k=1}^{j-1} r_{ik} r_{jk} \right)$

Use this to compute the Cholesky factorization of

$$A = \begin{bmatrix} 2 & 1 & 1/2 & 1/4 \\ 1 & 4 & 1 & 1/2 \\ 1/2 & 1 & 4 & 1 \\ 1/4 & 1/2 & 1 & 2 \end{bmatrix}.$$

To avoid computing the square root of a negative number (which happens for matrices that are not spd), check at each step that the quantity under the square root in the computation of $r_{jj}$ is positive. If it is not, display an error message[1] and stop the code. Please hand in your commented code.

4. **[Backward substituition implementation, 5pt]** Write a code for backward substitution to solve systems of the form $U\boldsymbol{x} = \boldsymbol{b}$, i.e., write a function `x = backward(A,b)`, which expects as inputs an upper triangular matrix $U \in \mathbb{R}^{n\times n}$, and a right hand side vector $\boldsymbol{b} \in \mathbb{R}^n$, which returns the solution vector $\boldsymbol{x} \in \mathbb{R}^n$. The function should find the size $n$ from the vector $\boldsymbol{b}$ and also check if the matrix and the vector sizes are compatible before it starts to solve the system. Please hand in your commented code. Apply your program for the computation of for $\boldsymbol{x} \in \mathbb{R}^4$, with

$$U = \begin{bmatrix} 1 & 2 & 6 & -1 \\ 0 & 3 & 1 & 0 \\ 0 & 0 & 4 & -1 \\ 0 & 0 & 0 & 2 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} -1 \\ -3 \\ -2 \\ 4 \end{bmatrix}.$$

5. **[Right hand sides with many zeros, 4pt]** For a given dimension $n$, fix some $k$ with $1 \le k \le n$. Now let $L \in \mathbb{R}^{n\times n}$ be a non-singular lower triangular matrix and let the vector $\boldsymbol{b} \in \mathbb{R}^n$ be such that $b_i = 0$ for $i = 1, 2, \ldots, k$.

   (a) Let the vector $\boldsymbol{y} \in \mathbb{R}^n$ be the solution of $L\boldsymbol{y} = \boldsymbol{b}$. Show, by partitioning $L$ into blocks, that $y_j = 0$ for $j = 1, 2, \ldots, k$.

   (b) Use this to give an alternative proof of Theorem 2.1 (iv), i.e., that the inverse of a non-singular lower triangular matrix is itself lower triangular.

6. **[Inverse matrix computation, 1+1+2pt]** Let us use the $LU$-decomposition to compute the inverse of a matrix[2].

   (a) Describe the algorithm (we discussed it in class) that uses the $LU$-decomposition of an $n \times n$ matrix $A$ for computing $A^{-1}$ by solving $n$ systems of equations (one for each unit vector).

   (b) Give the floating point operation count of this algorithm.

   (c) Improve the algorithm by taking advantage of the structure (i.e., the zero entries—see previous problem) of the right-hand side. What is the new algorithm's floating point operation count?

7. **[Stability of the Gaussian elimination, 2+1+2+1pt]**

   Consider the system

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{1}$$

[1] MATLAB has the command `error('message')` for doing that.

[2] This also illustrates that computing a matrix inverse is significantly more expensive than solving a linear system. That is why to solve a linear system, you should *never* use the inverse matrix!

where $A, L, E \in \mathbb{R}^{n\times n}$, $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$, with $A = L + E$. $L$ is unit lower triangular, where all the subdiagonal elements are $-1$, i.e., $l_{i,j} = -1$ for $i < j$, $l_{i,i} = 1$ for $i = 1, \ldots, n$, and $l_{i,j} = 0$ otherwise. Additionally, $E$ is a matrix such that $e_{i,n} = 1$ for $i = 1, \ldots, n-1$ and $e_{i,j} = 0$ otherwise. For example, when $n = 5$, we have

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix}.$$

(a) Prove that $A$ is invertible for any $n$, by induction.

(b) When $n = 5$ and $\boldsymbol{b} = (1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25})^T$, solve for $\boldsymbol{x}$ in (1) using Gaussian elimination, then backward substitution.

(c) Now consider $A$, $\boldsymbol{x}$, $\boldsymbol{b}$, where $n$ is not specified. Put (1) into the form $U\boldsymbol{x} = \boldsymbol{c}$, where $U$ is upper triangular using Gaussian elimination (you do not have write what $\boldsymbol{c}$ is since $\boldsymbol{b}$ is not given). What is $\max_{i,j} |u_{i,j}|$?

(d) For large $n$, e.g., $n = 2000$, what problems can you envision if you try to solve (1) using Gaussian elimination on a computer? Explain.