1. (Written) Let $A \in \mathbb{R}^{n \times n}$ be real, symmetric, and positive definite. Consider solving $Ax = b$ using Gaussian elimination without pivoting(!). The purpose of this problem is to justify that the pivots will be nonzero.

(a) Show that all of the diagonal elements satisfy $a_{ii} > 0$. This shows that $a_{11}$ can be used as a pivot element.

(b) After elimination of $a_{1j}$ for $j = 2, \ldots, n$, let the resulting matrix $A^{(2)}$ be written as

$$A^{(2)} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n}^{(2)} \\ 0 & \cdots & 0 & a_{nn} \end{bmatrix}$$

Show that $\hat{A}^{(2)}$ is symmetric and positive definite.

This approach can be continued inductively to each stage of the elimination process, thus justifying the existence of nonzero pivots at every step. Hint: To prove $\hat{A}^{(2)}$ is positive definite, first prove the identity

$$\sum_{i,j=2}^{n} a_{ij}^{(2)} x_i x_j = \sum_{i,j=1}^{n} a_{ij} x_i x_j - a_{11} \left( x_1 + \sum_{j=2}^{n} a_{j1} x_j \right)^2$$

for any choice of $x_1, x_2, \ldots, x_n$. Then choose $x_1$ suitably.

2. An X-matrix Let $A$ be an $n \times n$ matrix ($n$ must be an EVEN integer), whose entries are all zeros, except for $a_{i,i} = i$, $a_{i,n-i+1} = i/2$, $i = 1, 2, \ldots, n$

If you “spy” this matrix in MATLAB, you will see that the structure of this matrix is an “X.” Let $\vec{b} \in \mathbb{R}^n$ be the vector with all ones, $b_i = 1$ for $i = 1, 2, \ldots, n$. We will consider several approaches to solving $Ax = \vec{b}$.

(a) Use the LUfactor(), LUsolve() routines you wrote earlier to solve this problem. Make some changes to your routines: declare a global integer variable flops (flops stands for FLoating-point OPerationS) in your program and each time your LU routine does a floating-point multiplication or division, add one to flops. Track the total number of floating point operations needed.

(b) Write a separate program to solve $Ax = \vec{b}$ using Jacobi iterations with successive-over-relaxation (SOR) for a given value of the relaxation parameter. Like (a), track the number of flops needed.

Since $A$ only has $2n$ nonzero entries, calculating the product $Ax$ should require only $O(n)$ flops. Write your code to be this efficient. Stop your criterion when $\|x_{k+1} - x_k\|_\infty < 10^{-12}$.

(c) Repeat (b) to write a code for Gauss-Seidel iterations. \footnote{Test (b) and (c) with $n = 6$ and $\omega = 1$; verify that your iteration code is correct; make sure it converges to the solution from (a).}
(d) For each of (b) and (c), put your code in a loop over a range of \( \omega \) values (with \( \omega \) incrementing by 0.001 or less). Starting from the same initial guess \( \vec{x}_0 \) (like \( \vec{x}_0 = 0 \), find the optimal value of \( \omega^* \) that leads to convergences with the smallest number of iterations.

(e) Save the results (\( n \) vs. \texttt{flops} ) to data files for the following cases:

- LU
- Jacobi (without SOR)
- Gauss-Seidel (without SOR)
- Jacobi-SOR with \( \omega^* \)
- Gauss-Seidel-SOR with \( \omega^* \)

Plot these as curves on a (one) log-log graph.

What is the smallest matrix (\( n \)-value) for which iterative solution becomes more computationally efficient than direct solution in this problem?

Extrapolate and make a table predicting the \texttt{flops} needed for \( n = 1000 \) for each method.

\footnote{Test to see if you believe this statement: \( \omega^* \) is independent of \( n \) for this problem.}