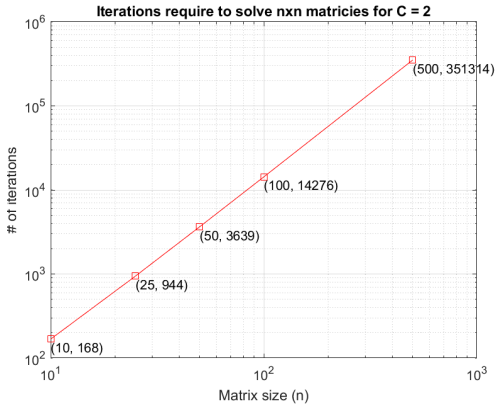
MACM 316 Computing Assignment 4:
Analysis of Jacobi Iteration and Performance

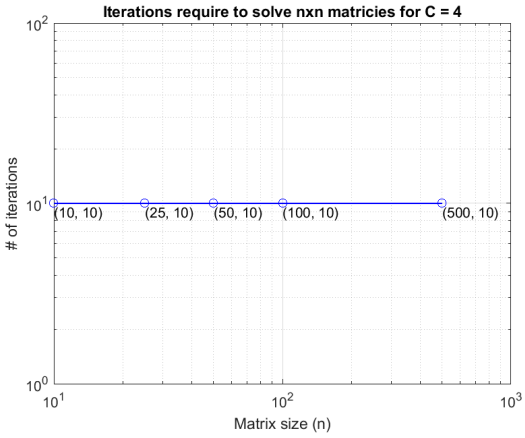
The Jacobi Iterative method is a well known algorithm for solving linear systems. The implementation is as follows:

$$\vec{x}_{n+1} = D^{-1}(\vec{b}_n - (L + U)\,\vec{x})$$

This study analyzes how the number of iterations depends on the size of the matrix for two classes of sparse, symmetric, and positive definite matrices. More specifically, $c = 2$ and $c = 4$. These 'C' values are scaling factors that affect how diagonally dominant a matrix is, affecting the performance. I.E, higher the scaling factor, the more diagonally dominant the matrix. The scaling factor is crucial in the performance analysis as it affects diagonal dominance and spectral radius, both of which are influential in Jacobi convergence. In both cases, our initial $\vec{x}$ is the zero vector, $\vec{0}$. To validate solutions, this study uses a tolerance of 10^{-3}. The study uses various n values, n = [10, 25, 50, 100, 500], for n by n matrices for both cases. The exact solution is solved by using MATLAB's backslash (\) operator. This value is used to compute the error and compare the Jacobi iteration result to the set tolerance.



Part A, $C = 2$) Once we have extracted D^{-1}, L, $and\ U$, we run the Jacobi iterative implementation in MATLAB until the exact solution within the tolerance is achieved. The graphical representation can be found in fig1, the red line graph. While all test values of n converge, what should be noticed is the polynomial increase in time and iterations required to achieve the exact solution for $n \geq 500$.



Investigating why the method is inefficient to converge at larger matrix sizes, we find that as n increases, the spectral radius ρ, gets closer to 1. The spectral radius is found by taking the max absolute eigenvalue of the matrix A_n

The spectral radius condition that guarantees Jacobi convergence is:

$$\rho < 1$$

If $\rho \geq 1$, the Jacobi iterative method will not converge. This follows the results we found above, that as n become larger, the spectral radius increases towards 1 meaning the algorithm will take significantly longer to converge. For example, $\rho = 0.99998$ for $n = 500$.

While we can model the Jacobi iteration for this scaling factor, $C = 2$, using log-log regression, the exponential jump cost to converge for the largest n value of the test cases leads to invalid data. In this case, we can take a more empirical approach. We expect a theoretical growth relation of $O(n^2)$, therefore:

$$\text{For valid iterations of n, r = mean } \frac{\#\ of\ iterations}{n^2}$$

Said plainly, we take the iterations required for convergence for size n, and divide by n^2. We repeat this for all n in the set, n = [10, 25, 50, 100, 500], and take the mean. This yields mean[1.6800, 1.5104, 1.4556, 1.4276, 1.4053] where the mean is 1.4958. The resultant is polynomial, close (≈ 2) to quadratic growth; therefore, Jacobi with c = 2 has a growth of $O(n^{1.4958})$.

Part B, C = 4) Fig2, The blue graph represents the growth of this Jacobi process, with scalar factor = 4. Following the steps in part A, we find the spectral radius, ρ, to be trending towards $\rho \approx 0.5$ and remains there. As indicated in the blue graph, even in the larger matrix sizes, the Jacobi continues to converge, consistently only requiring 10 iterations.

Using the combination of log-log regression and the power-law relation, we can find its big O relation. The power-law states that the slope of the log-log regression line is representative of the function's growth rate. Applied here:

$$log(\#\ of\ iterations) = R \cdot log(n) + constant$$

Where the slope, R, is representative of Jacobi's growth rate in big O. Through this process, we obtain the slope, $R = 6.9246E^{-18}$. A number so close to 0 indicates constant (low) growth. This is validated by the blue graphs results, being a constant and flat line. Therefore, we can conclude that the growth of Jacobi iterations, with C = 4, is represented by $O(1)$.