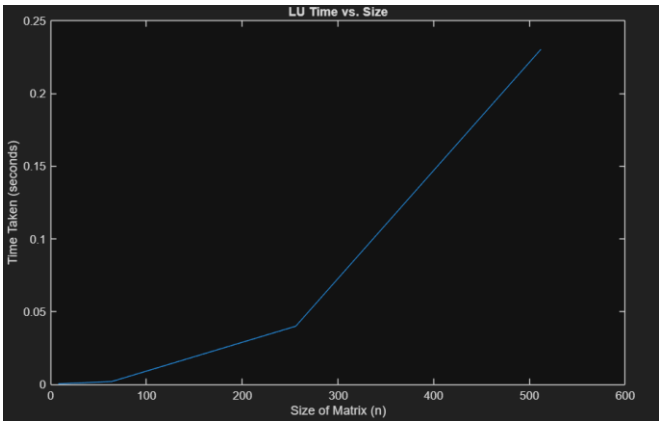
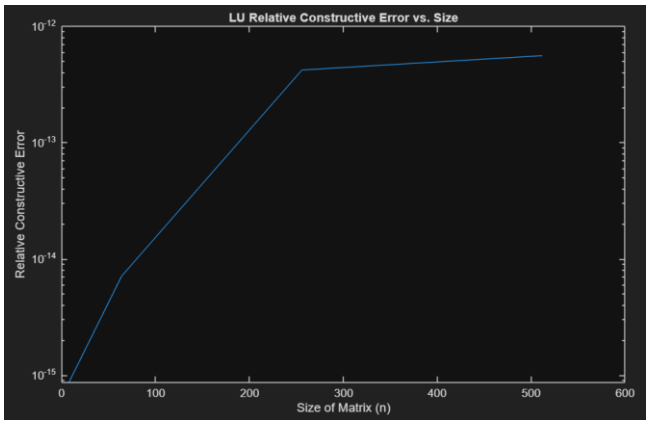
Matrix Decompositions Analysis Report

Naema Ahmed — KU ID : 100064664

1) LU Factorisation by Doolittle's Method





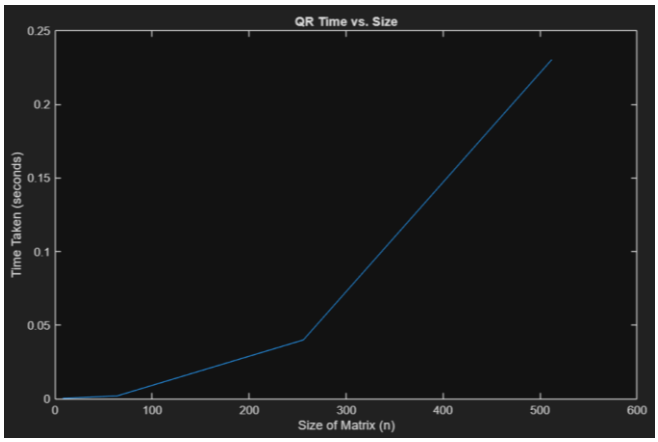
(i) Time vs. Size Plot

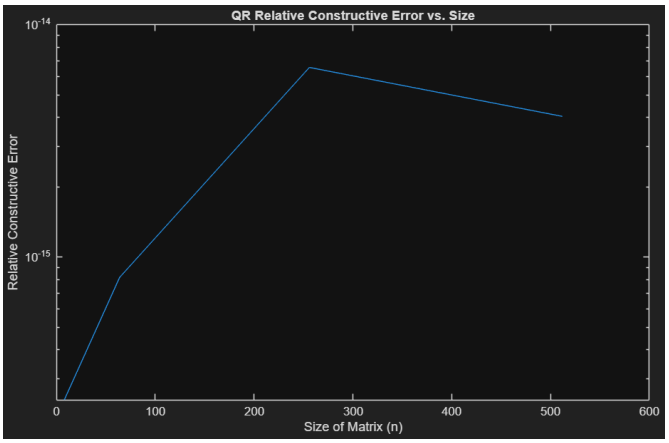
The time taken to compute the LU factorisation increases with the size of A approximately cubically. This increase is due to the larger number of computations necessary, and upon research, I see that this matches the time complexity of Doolittle (and Gaussian method) both being O(n^3).

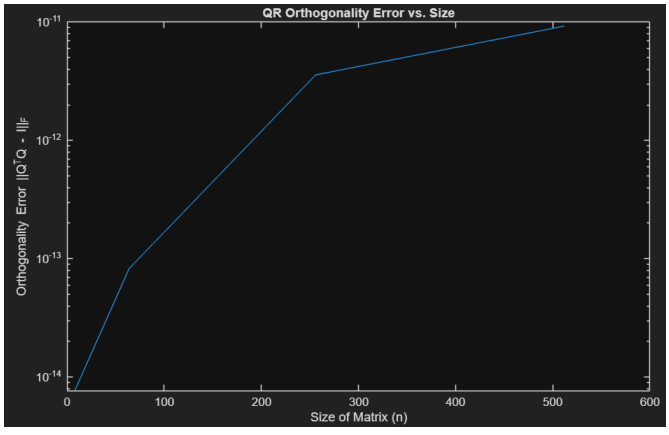
(ii) Relative Reconstruction Error vs. Size Plot

The shape of the graph does not hold significance, as it fluctuates wildly based on the randomised matrix A chosen. The insight is in the range of the relative Reconstruction error being consistently between 1e-15 and 1e-11, which is the expected error from the machine's rounding errors (since MATLAB doesn't perform symbolic calculations).

2) QR Factorisation by Classical Gram-Schmidt (CGS)







(i) Time vs. Size Plot

The time taken to compute the CGS QR decomposition of a matrix A increases as A grows in size because of the increased number of computations needed to obtain Q and R. This time increase is also approximately cubic, and I've found that the time complexity for QR by CGS is O(2n^3) for square matrices, and O(2mn^2) for rectangular ones.

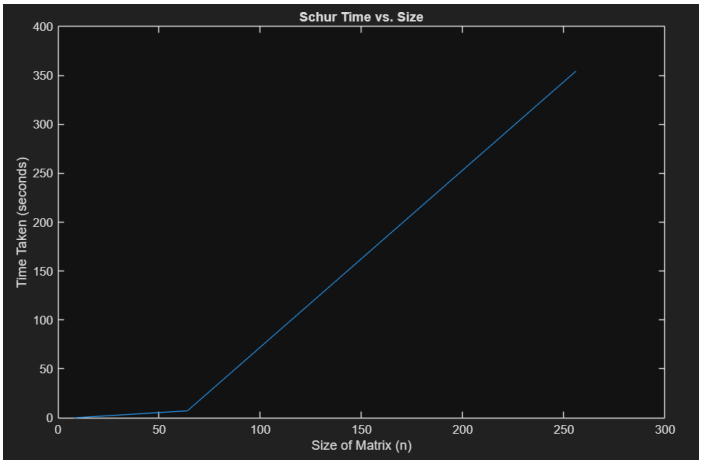
(ii) Relative Reconstruction Error vs. Size Plot

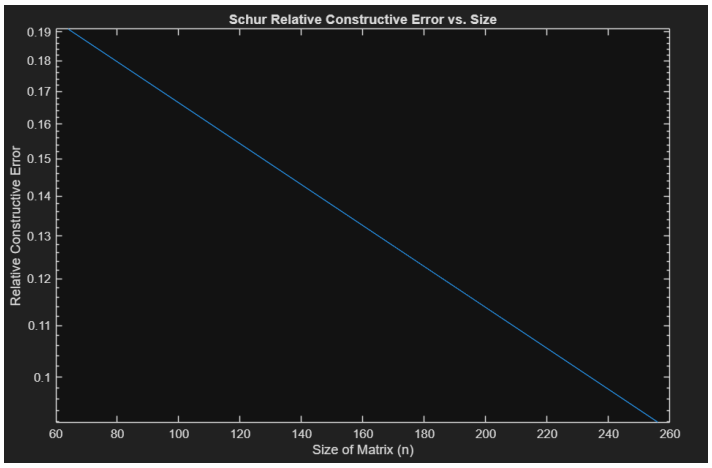
This plot fluctuates a lot based on the randomised A chosen, since in some cases A is an ill-conditioned matrix, and other times it is more well-conditioned. However, I notice that the range of the error generally remains between 1e-13 and 1e-16, which can be attributed to the machine precision error.

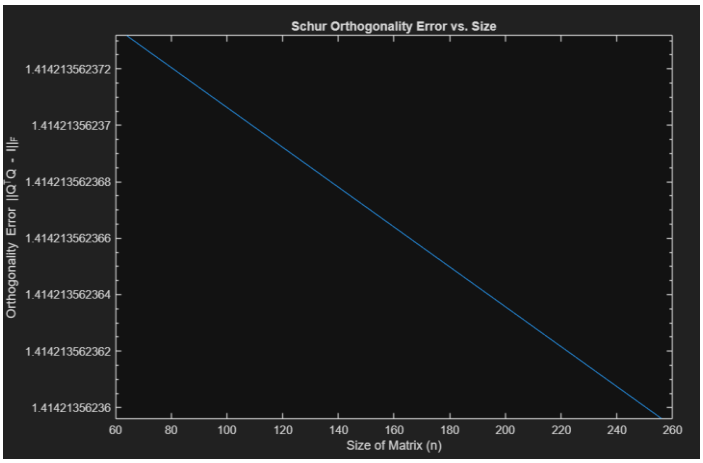
(iii) Orthogonality Error (‖Q^TQ − I‖_F) vs. Size Plot

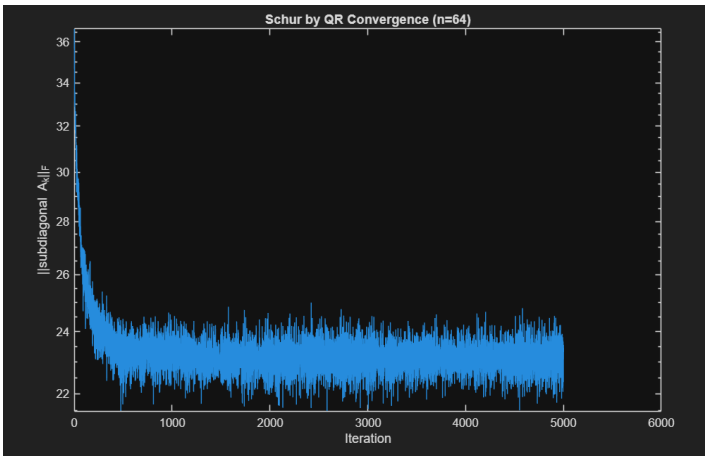
The orthogonality error should be 0 if the CGS was done by hand or symbolically. However, in MATLAB, rounding errors lead the calculation of the next vector of Q to not be perfectly orthogonal to the ones before it. This error adds up with each step, and that is exactly what the plot shows. The orthogonality error increases, ranging from 1e-14 to 1e-11 as the size of the matrix A grows. To reduce this error, adding a reorthogonalization step should be considered (Modified Gram-Schmidt).

3) Real Schur by QR Algorithm with Shifts









(i) Time vs. Size Plot

The time taken for computing the Schur factorisation of A is shown to increase quickly as the size of A grows. It seems like a cubic growth once again, which makes sense as this Schur function uses CGS QR. The time is significantly longer than the previous 2 functions, so much so that the n=512 case would not converge and computing 5000 iterations took too long (over 40 minutes). Hence, it is not included in the plot.

(ii) Relative Reconstruction Error vs. Size Plot

This plot looks concerning at first, since the range of errors is up to around 0.20, which is much larger than errors caused by machine precision. So, the decomposition QTQ^T is not that good of an approximation of A. This is very likely due to using CGS QR in the Schur function. The loss of orthogonality at each iteration, as well as hitting the maximum number of iterations before convergence are factors that led to this poor result.

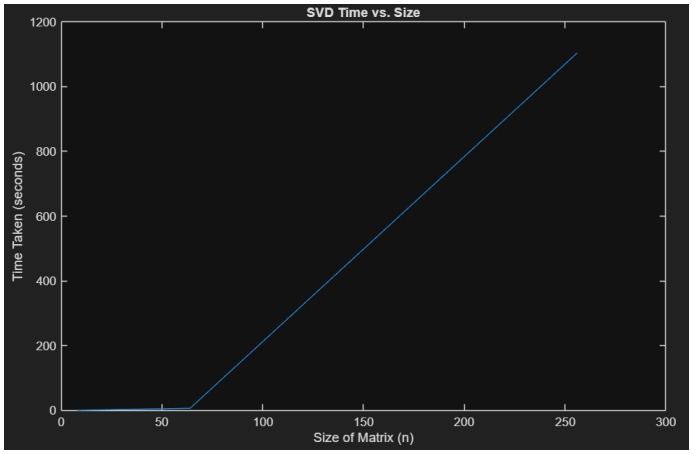
(iii) Orthogonality Error (‖Q^TQ − I‖_F) vs. Size Plot

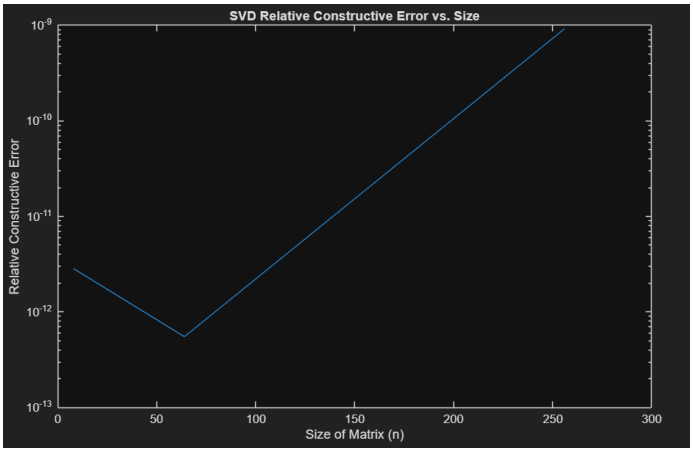
The orthogonality error shown in the plot is quite large, around 1.4142 (≈ √2). If Q was perfectly orthogonal (and normalised), then Q^TQ = I, leading to an error of 0. Given some rounding errors, this error would be bumped up to something around the machine precision (as in the QR's orthogonality error plot). However, the error for Schur is much higher, and this can be attributed to the fact that it is repeatedly undergoing CGS QR, each time losing some orthogonality, and carrying cumulative errors as it performs CGS QR on that un-orthogonal matrix. This shows just how unstable using CGS QR is, especially in functions that undergo many iterations using it. Adding a reorthogonalization step should be considered (Modified Gram-Schmidt) to reduce this significant error, or implement another method of QR (like Householder) to use for Schur.

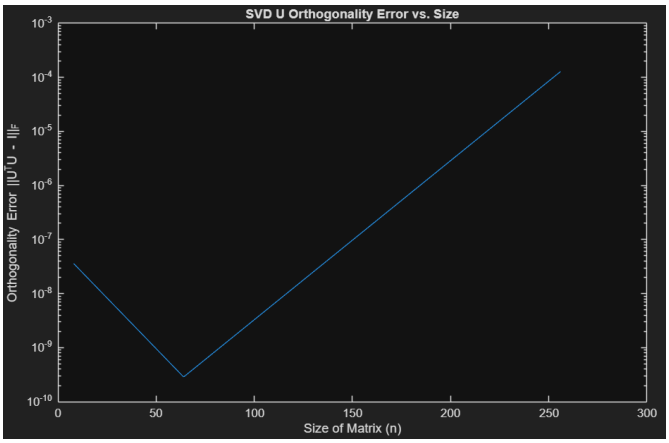
(iv) ‖ strictly subdiagonal (A_k) ‖_F vs. Iterations

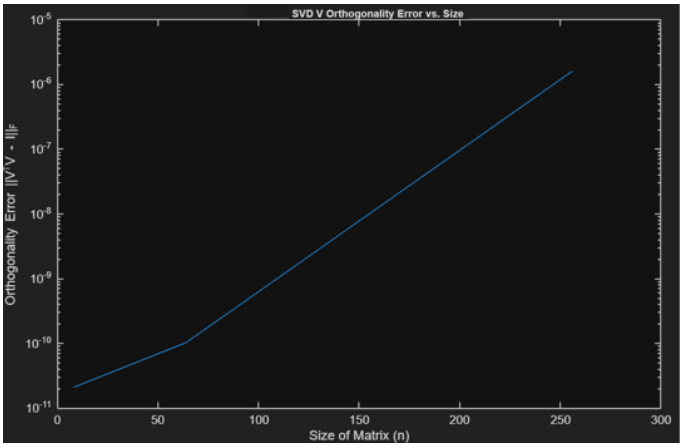
For this plot, I used a random square matrix A of size 64, with the default arguments provided. The norm of the subdiagonal elements rapidly decreases and then levels off till the maximum iteration is reached (meaning the minimum tolerance was not achieved before 5000th iteration). However, this norm is still significantly high, and that can be attributed to the rounding errors and loss of orthogonality from using CGS QR to implement this Schur function (CGS QR fails to make the subdiagonal elements of T go to 0). Using symmetric matrices, a modified GS, or a different QR method would decrease this norm a lot more and show a faster convergence.

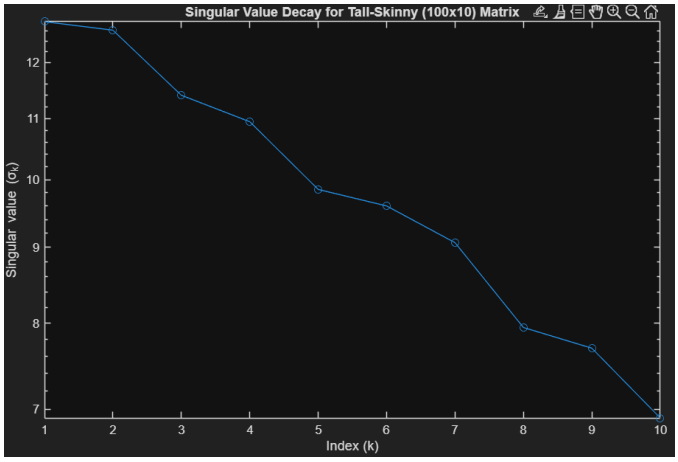
4) SVD by Eigen-Decomposition of A^TA

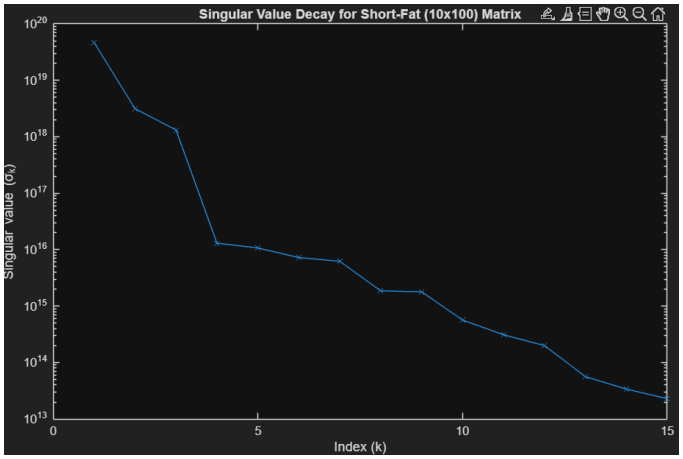


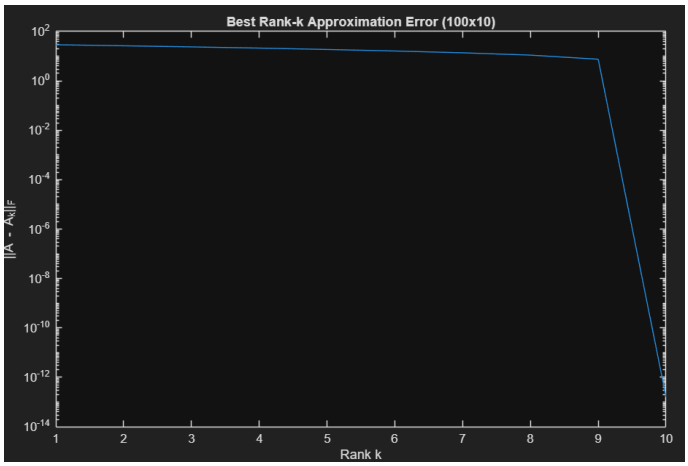


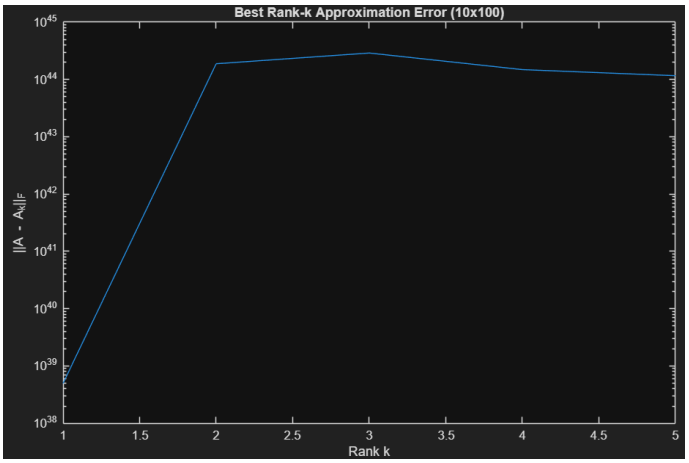












(i) Time vs. Size Plot

The observations for this plot are the same as with the previous functions. My SVD function shows a rapid increase in computation time as the size of the matrix grows, and the shape of the graph seems to be approximately cubic as well. I found that the time complexity for SVD on a square matrix is generally O(n^3), so this result is expected. However, the range of the time itself is extremely large, and the function becomes incredibly slow even by n=256, making the inclusion of n=512 in this plot not possible. This stems from how my SVD function was constructed using the slow CGS QR based Schur.

(ii) Relative Reconstruction Error vs. Size Plot

The relative Reconstruction error is between 1e-13 and 1e-9, some of this error can be attributed to machine precision, and the upper bound being a bit high is likely from the larger, more ill-conditioned matrices. The error range is a lot better than in Schur, likely because SVD uses Schur on AtA which is symmetric and does not require shifting.

(iii) Orthogonality Errors (|| U^TU – I ||_F and || V^TV −I ||_F) vs. Size Plots

The plots show that both U and Vt are near orthogonal up to machine precision since their respective errors, || U^TU – I ||_F and || V^TV −I ||_F, are 1e-10 to 1e-4 and 1e-11 1e-6, respectively. This is a lot better than what was seen with the Schur orthogonality error, and that can be attributed to the fact that SVD computes the Schur decomposition of a symmetric matrix, AtA (so its Schur decomposition has a diagonal T of eigenvalues, and its eigen vectors are orthogonal by default).

(iv) Singular Value Decay (Tall-Skinny & Short-Fat)

These plots show a key difference between the two types of matrices. In the tall-skinny case (100x10), AtA will have dimensions 10x10. Performing Schur via CGS QR on a 10x10 matrix doesn't involve many iterations, so the CGS QR errors from loss of orthogonality do not accumulate to be large. Hence, we can see the singular values are not too large, and decrease across k. On the other hand, the short-fat (10x100) has an AtA matrix with dimensions 100x100. This is very large to my numerically unstable model, and since it leads to many iterations using CGS QR, it accumulates a large and significant error. Since we obtain V and Σ from the Schur by CGS QR of AtA, the singular values are affected (very large values, slower convergence) and that is what can be seen in these plots. Once again, a more stable version of QR could fix this issue (MGS or Householder).

(v) Reconstruction Error of Best Rank-k Approximation (|| A - Ak ||_F) vs. k

The two plots show the same difference discussed in (iv). The tall-skinny matrix (100x10) has a rank-k approximation error that decreases across k, showing that the largest k is the best choice as it captures the most information from the matrix (since singular values are sorted and reflect importance). Since it has AtA of dimensions 10x10, it didn't require too many iterations of CGS QR and didn't get the chance to accumulate many errors due to loss of orthogonality. However, in the short-fat case (10x100), the error grows extremely large because AtA has dimensions 100x100, and that is a lot for the numerically unstable CGS QR based Schur to handle. The errors add up over the iterations, leading to such an inflated error plot. Once again, switching the CGS QR out for a more stable QR (MGS, Householder) would fix this issue.

(vi) Discussing Computational Costs/Behaviour Differences (e.g., forming ATA cost and conditioning)

Generally, the computational time taken by my SVD function shows an approximately cubic increase as the matrix size grows. However, there are differences in behaviour between a tall-skinny and short-fat matrix. Because SVD uses the Schur factorisation of AtA, which is smaller for a tall-skinny matrix and very large for a short-fat matrix, the behaviour of the function and the resulting errors that arise wildly differ. As aforementioned, short-fat matrices end up undergoing many iterations using CGS QR and accumulates significant rounding errors in the process, resulting in both a higher computational cost and poorer reconstruction accuracy due to both a larger size and being ill-conditioned.

Notes / Further Discussion:

- *The type of matrix (Gaussian, Hilbert, Vandermonde) is not specified for plot creation in the homework instructions, hence, I arbitrarily used Gaussian matrices for all plots shown.*
- *In LU, I understand that zero/near-zero pivots can cause errors or make the LU fail, so I included a line in my code to identify such pivots.*
- *My Schur function does not have deflation to accelerate convergence by removing eigenvalues from the matrix and being left with smaller matrices to work with. Instead, I implemented a shift (A(n,n)). I recognise that adding deflation to my function could make my function much faster and more stable, but for the scope of this project, I decided not to incorporate it here.*