Scientific Computing Assignment

Erdem
Şamlıoğlu
2448843

Question 2

Part A

My code to print the errors and run times. Also plot the relative errors vs matrix sizes:

```
matrixsize = 300;

for n = 2:matrixsize
    A = hilb(n);

    %LU without pivoting
    tic;
    [L, U] = my_lu(A);
    time_np = toc;
    error_np = norm(A - L*U, 2) / norm(A, 2);

    %LU with partial pivoting
    tic;
    [L_pp, U_pp, ~] = my_lu_pp(A);
    time_pp = toc;
    error_pp = norm(A - L_pp*U_pp, 2) / norm(A, 2);

    %LU with complete pivoting
    tic;
    [L_cp, U_cp, ~, ~] = my_lu_cp(A);
    time_cp = toc;
    error_cp = norm(A - L_cp*U_cp, 2) / norm(A, 2);

    %Values
    fprintf('n = %d: Runtime (No Pivot= %.5f s, Partial Pivot= %.5f s, Complete Pivot= %.5f
    fprintf('Error (No Pivot= %.5e, Partial Pivot= %.5e, Complete Pivot= %.5e)\n', error_np,
end
```

```
% Plotting results
figure;
semilogy(2:matrixsize, errors(1, :), 'r'); % LU without pivoting
hold on;
semilogy(2:matrixsize, errors(2, :), 'g'); % LU with partial pivoting
semilogy(2:matrixsize, errors(3, :), 'b'); % LU with complete pivoting
xlabel('Matrix Size (n)');
ylabel('Relative Error');
legend('LU without pivoting', 'LU with partial pivoting', 'LU with complete pivoting');
title('Relative Error / Matrix Size');
```

1. Run times: I have printed the run times for the 3 algorithms to compare them.

   No pivoting: It is the fastest compared to the other 2 algorithms. Although it is the fastest, there are few exceptions to this. For example I have observed that for matrix size n=287, and 288, partial pivoting is faster. Since adding partial pivoting to the computation should have increased computational complexity, this was an unexpected result and my research in the internet showed me that this could be possibly due to particular size of a matrix might fit better in the cache, leading to faster computations for specific size.

   ```
   n = 287: No Pivoting = 0.20658 s, Partial Pivoting = 0.19868 s
   Complete Pivoting = 0.28463 s
   ```

   ```
   n = 288: No Pivoting = 0.23836 s, Partial Pivoting = 0.21684 s,
   Complete Pivoting = 0.27913 s
   ```

   Partial pivoting: It is faster compared to complete pivoting and slower compared to no pivoting. Although it is slower than the no pivoting, the run time values are really close and even sometimes better than no pivoting, which I explained in "No pivoting" part.

   Complete pivoting: It is the slowest algorithm among others, without any exception. This is due to the additional computation for finding the maximum element in the matrix.

2. Relative Error Analysis:

   No pivoting has the lowest errors for smaller matrices as it could be seen from Figure 1, but as the size of the matrix increases, the error also increases significantly compared to other algorithms. Although both partial

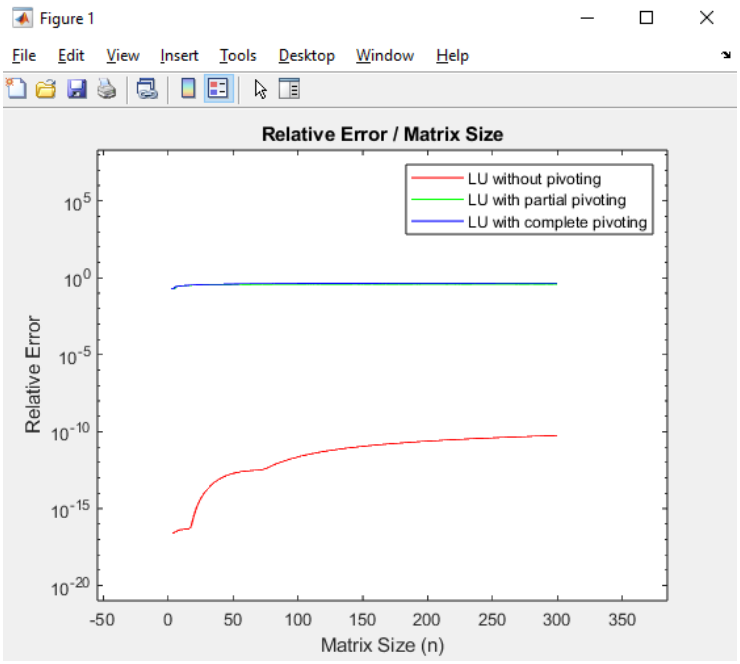

Figure 1: The relative error vs matrix size graph

pivoting and complete pivoting have higher errors for the given matrix size, the errors stabilize and do not increase as much as no pivoting as the size of the matrix increases. This situation shows that pivoting methods are better for numerical stability, especially for large, Hilbert matrices (ill conditioned).

3. Conclusion:
   For smaller matrices, No pivoting method offers both good speed and accuracy, but as the matrix size increase, the accuracy decreases significantly which makes it bad for very large matrices.
   For larger matrices, complete pivoting becomes the most stable but it is slower compared to partial pivoting. Partial pivoting is in the middle, which is faster than complete pivoting but not more stable than it.

   As a conclusion, if the matrix size is small one could choose no pivoting since it provides both good speed and accuracy. For a large matrix, if the speed is more important one should choose partial pivoting, if the accuracy is more important, then one should choose complete pivoting.

Part B

In theory, all of the algorithms should be able to factorize any non singular square matrix. In practice, having very small pivot elements can cause numer-

ical instability, especially in no pivoting. In no pivoting, when there is a zero or very close to zero pivot element, this could result in division by zero or insignificant numerical stability. In partial pivoting, this situation is handled by swapping rows to place the largest absolute value element in the pivot position. In complete pivoting, this situation is handled even better by swapping both rows and columns to place the largest available element in the pivot position.

As a conclusion, in practice, having a very small pivot element can cause instability, especially in no pivoting. With the pivoting methods, this situation is handled mostly. Therefore, for the ill conditioned matrices, pivoting methods should be preferred.