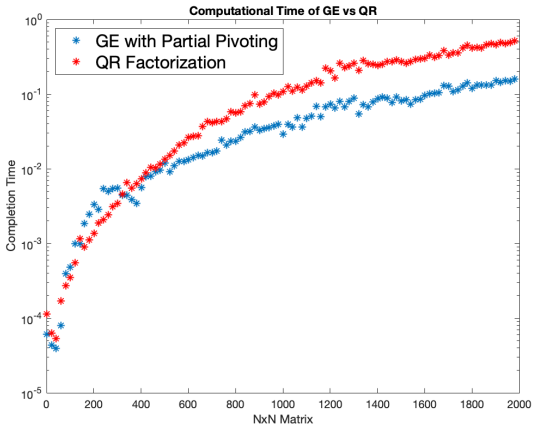
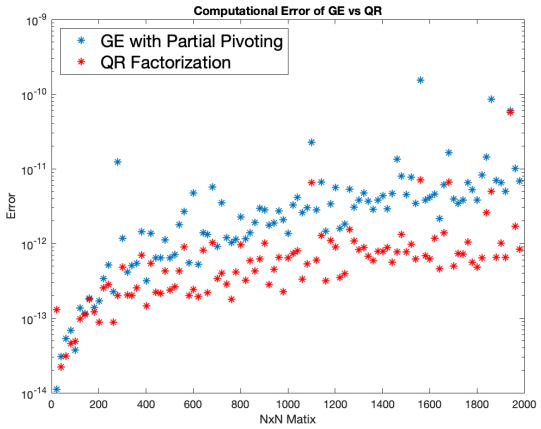
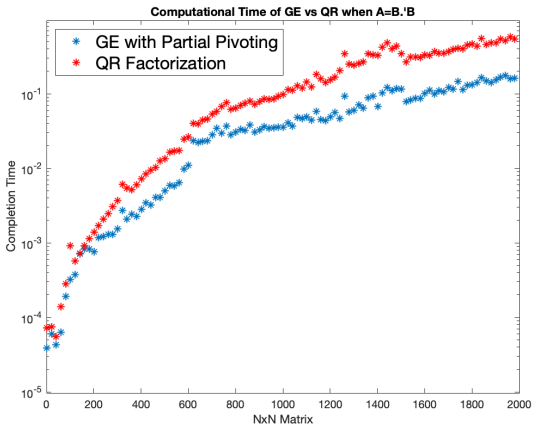
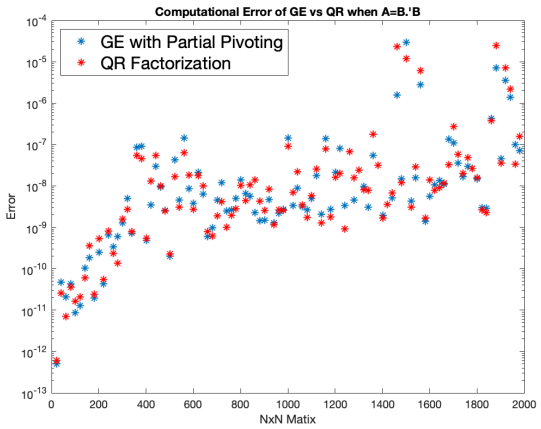
MACM316 ASSIGNMENT 2 – Ngawang Kyirong 301312227









Part 1: trials: 20, max matrix size: 2000**,** steps: 20

* The choices for the number of trials, steps, and matrix size are to ensure the calculations are more robust than just running the calculations once. The averages are taken to enforce outliers as well.

Efficiency: For the first experiment, the Gaussian Elimination with pivoting method tends to be more efficient than QR. While including the time it takes to factorize, QR will take much longer and especially as the matrix size increases. This can not only be seen through these experiments, but the flop count for QR is also larger than GE.

Robustness: For the first experiment, the error tends to be less for QR in comparison to GE. As the sizes of matrices increase, GE becomes unstable. QR is more robust as N starts to increase in size.

Part2: trials: 20, max matrix size: 2000**,** steps: 20

* Same explanation as above for part2 in terms of choosing values.

Efficiency: For the second experiment, the efficiency of GE vs QR remains similar. QR still takes longer than GE. We can again count the flops and see that QR has more steps than GE.

Robustness: For the second experiment, error is quite similar when comparing GE and QR. This can be further explored through the use of condition numbers when dealing with A = transpose(B) * B. As a condition number of a matrix is closer to 1, the robustness will be better than those with a condition number greater than 1. Furthermore, because of this, we see that this greatly both algorithms as the error is larger than it is for both algorithms in comparison to the first experiment. Since the matrix has a higher condition number, we are much more prone to errors in both algorithms when solving a system.

```
nmax = 2000;
trials = 20;
steps = 20;

times1 = []; % Vector of times for GE
times2 = []; % Vector of times for QR
times3 = []; % Vector of times for GE w/ A=transpose(B)*B
times4 = [];% Vector of times for QR w/ A=transpose(B)*B
errors1 = []; % Vector of error for GE
errors2 = []; % Vector of error for QRs
errors3 = []; % Vector of error for GE w/ A=transpose(B)*B
errors4 = []; % Vector of error for QR w/ A=transpose(B)*B

for N = 1:steps:nmax

    x = ones(N,1);
    times1_raw = zeros(trials,1); % raw Vector of times for GE
    times2_raw = zeros(trials,1); % raw Vector of times for QR
    times3_raw = zeros(trials,1); % raw Vector of times for GE w/ A=transpose(B)*B
    times4_raw = zeros(trials,1); % raw Vector of times for QR w/ A=transpose(B)*B
    errors1_raw = zeros(trials,1); % raw Vector of error for GE
    errors2_raw = zeros(trials,1); % raw Vector of error for QR
    errors3_raw = zeros(trials,1); % raw Vector of error for GE w/ A=transpose(B)*B
    errors4_raw = zeros(trials,1); % raw Vector of error for QR w/ A=transpose(B)*B
    for i = 1:trials

        A = randn(N,N);
        b = A*x;
        % ----GE----
        tic;
        xhat = A\b;
        times1_raw(i) = toc;
        errors1_raw(i) = max(abs(xhat - x));
        % ----QR----
        tic;
        [Q,R] = qr(A);
        xhat2 = R\(transpose(Q)*b);
        times2_raw(i) = toc;
        errors2_raw(i) = max(abs(xhat2 - x));
        % ------PART 2--------
        b2 = randn(N,N);
        a2 = transpose(b2)*b2;
        b3 = a2*x;
        % ----GE w/ A=transpose(B)*B----
        tic;
        xhat3 = a2\b3;
        times3_raw(i) = toc;
        errors3_raw(i) = max(abs(xhat3 - x));
        % ----QR w/ A=transpose(B)*B----
        tic;
        [Q,R] = qr(a2);
        xhat4 = R\(transpose(Q)*b3);
        times4_raw(i) = toc;
        errors4_raw(i) = max(abs(xhat4 - x));
    end

    times1 = [times1 mean(times1_raw)];
    errors1 = [errors1 mean(errors1_raw)];
    times2 = [times2 mean(times2_raw)];
    errors2 = [errors2 mean(errors2_raw)];
    times3 = [times3 mean(times3_raw)];
    errors3 = [errors3 mean(errors3_raw)];
    times4 = [times4 mean(times4_raw)];
    errors4 = [errors4 mean(errors4_raw)];
end
figure(1)
plt = semilogy(1:steps:nmax, times1, '*', 1:steps:nmax, times2, 'r*');
xlabel('NxN Matrix');
ylabel('Completion Time');
title('Computational Time of GE vs QR')
legend({'GE with Partial Pivoting','QR Factorization'},'fontsize', 16, 'Location','northwest')
figure(2)
plt2 = semilogy(1:steps:nmax, errors1, '*', 1:steps:nmax, errors2, 'r*');
xlabel('NxN Matix');
ylabel('Error');
title('Computational Error of GE vs QR')
legend({'GE with Partial Pivoting','QR Factorization'},'fontsize',16, 'Location','northwest')
% ------PART 2--------
figure(3)
plt = semilogy(1:steps:nmax, times3, '*', 1:steps:nmax, times4, 'r*');
xlabel('NxN Matrix');
ylabel('Completion Time');
title("Computational Time of GE vs QR when A=B.'B");
legend({'GE with Partial Pivoting','QR Factorization'},'fontsize', 16, 'Location','northwest')
figure(4)
plt2 = semilogy(1:steps:nmax, errors3, '*', 1:steps:nmax, errors4, 'r*');
xlabel('NxN Matix');
ylabel('Error');
title("Computational Error of GE vs QR when A=B.'B");
legend({'GE with Partial Pivoting','QR Factorization'},'fontsize',16, 'Location','northwest');
```