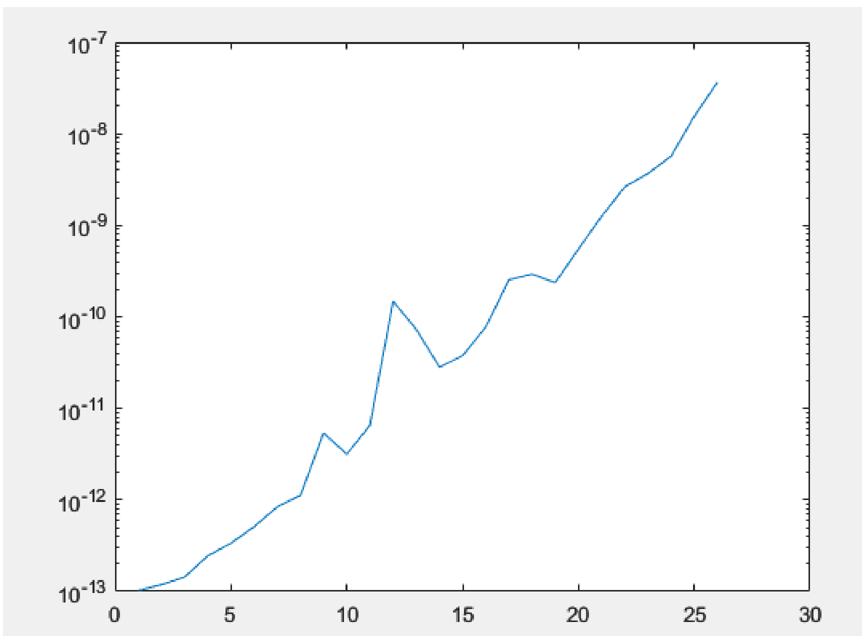
# NumComp - Fall 2022
# Project #8

## Due –
## Isaiah Thomas

Plots!

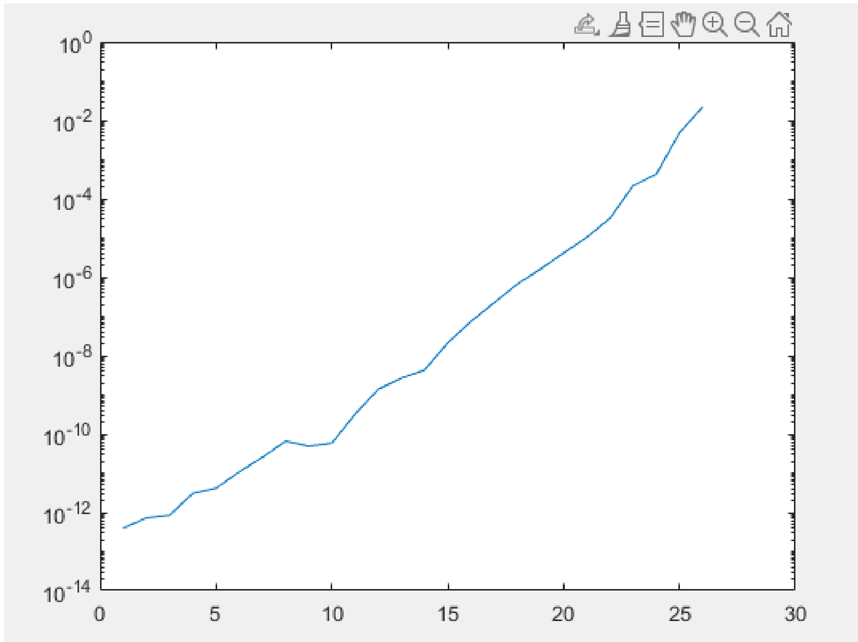
QR

value of k + 40 (x axis) vs average relative error (y axis)



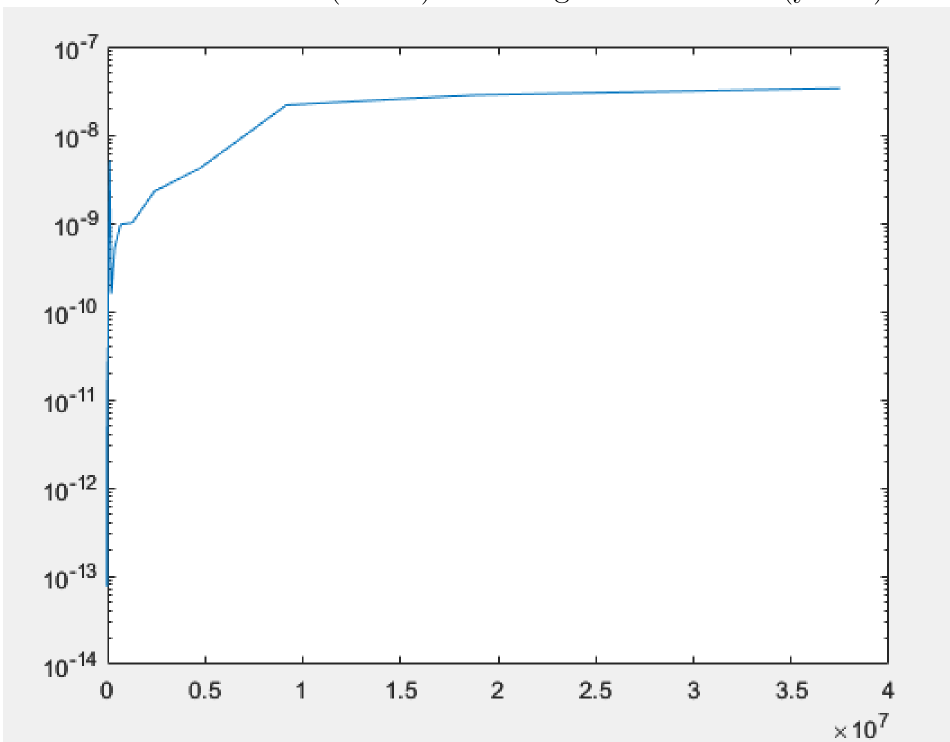
Normal Equations

value of k + 40 (x axis) vs average relative error (y axis)



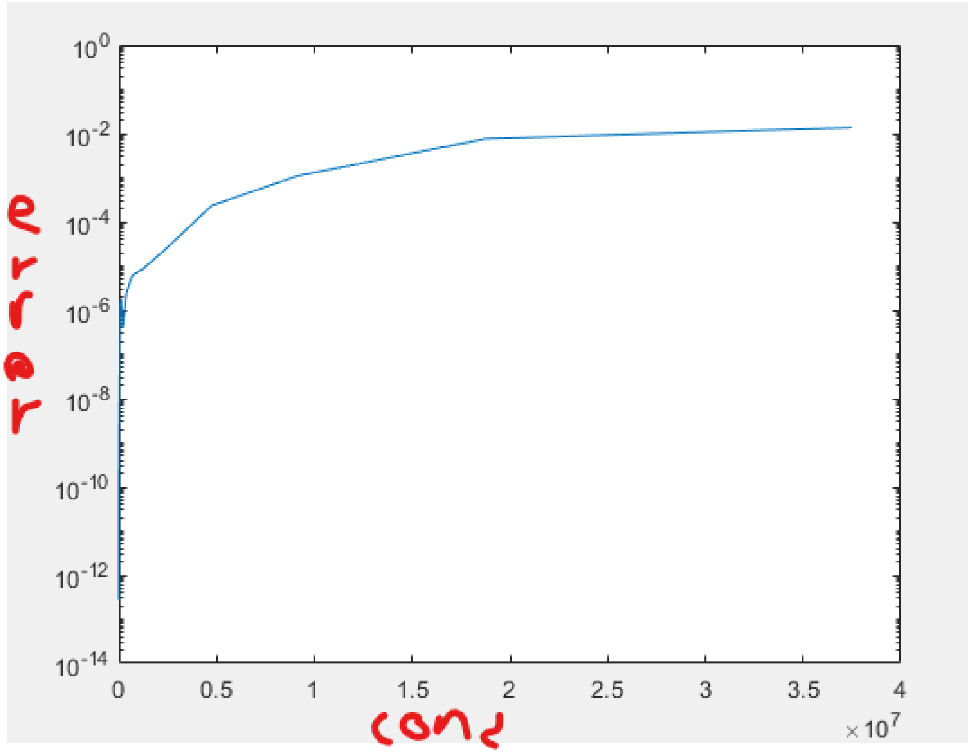
QR

condition number (x axis) vs average relative error (y axis)



Normal Equations

condition number (x axis) vs average relative error (y axis)



1. Relation of error between QR and Normal Equations?

   First thing I noticed is the difference in range of possible average relative error. While QR has a range of about $(10^{-13} - 10^{-7})$ for our tested values of $k$, Normal Equations has a much larger range of $(10^{-12} - 10^{-2})$

   QR seems to be able of achieve higher level of accuracy with less columns, but quickly loses this as $k$ increases. From the first plot, it seems QR is less consistent when $k$ is within 50 and 60.

2. Relation of errors and condition number of A?

   Both methods relative error are heavily impacted by condition number. This makes sense as condition number is a ratio of possible error of of matrices $A$ and $b$. QR appears to be able to maintain a better level of accuracy than NE when condition numbers begin to increase.

3. Best method for ill-conditioned matrices?

   QR easily. For both plots, as condition gets larger, each method approaches a y-asymptote. QRs is about $10^{-7}$ and NEs is about $10^{-2}$. This makes sense as the condition number of NE is the square of what it's input matrix is.

Code!

```
function xout = thinqr(A,b)

    [Q,R] = qr(A,0);
    btld = Q' * b;
    xout = R \ btld;

end

function xout = normeq(A,b)

    R = A' * A;
    y = A' * b;
    xout = R \ y;

end
```

this function was normally just going to generate a matrix of the needed $A_k$ matrices, but it wound up pretty much doing everything else since I could lol

```
function [kdata,kcond, xtrue,xne,xqr,nekmeans, qrkmeans] = firstk(A)
    [am,an] = size(A);

    %creates the matricies of A up until the ith column
    %each cell of Aindexed is some A_k
    Aindexed = cell(1,101);
    Aindexed{1} = A(:,1);
    for i = 2:1:an
        can = A(:,i);
        Aindexed{i} = [Aindexed{i-1} can];
    end


    %gathers size condition and rank
    %size and rank are in a cell, condition in a matrix
    kdata = cell(26,2);
    kcond = zeros(1,26);
    for i = 1:1:26
        j = i + 39;
        %disp(j);
        kdata{i,1} = size(Aindexed{j});
        kdata{i,2} = rank(Aindexed{j});
        kcond(i) = cond(Aindexed{j});
    end



    bn = cell(1,100); %100 randomly generated bs of size R^m

    %cells containing all generated x values for each A and b
    xtrue = cell(100, 26 );
    xne = cell(100, 26 );
    xqr = cell(100, 26 );
    %ouput cell columns index A_k
    %rows are for each random b

    %these store the relative error of every x value for all As and bs
    errorne = cell(100, 26 );
    errorqr = cell(100, 26 );
```

```
%these hod the average error of all xs for all A and b
meannee = zeros(100,26);
meanqre = zeros(100,26);

%average error of all of xs for all 100 bs, indexed by A_k
nekmeans = zeros(26);
qrkmeans = zeros(26);

for i = 1:1:100
    bn{i} = rand(am,1);
    for j = 1:1:26
        kindex = j+39;
        disp(size(Aindexed{kindex}));
        disp(size(bn{j}));

        %current truth matrix x
        ctrue = Aindexed{kindex} \ bn{i};


        %my current solutions and their calculated relative error
        cne = normeq(Aindexed{kindex}, bn{i});
        cnere = abs((ctrue - cne) ./ ctrue);
        cqr = thinqr(Aindexed{kindex}, bn{i});
        cqrre = abs((ctrue - cqr) ./ ctrue);


        %storin them
        xtrue{i,j} = ctrue;
        xne{i,j} = cne;
        xqr{i,j} = cqr;
        errorne{i,j} = cnere;
        errorqr{i,j} = cqrre;

        meannee(i,j) = mean(cnere);
        meanqre(i,j) = mean(cqrre);


    end

end
nekmeans = mean(meannee,1);
```

```
    qrkmeans = mean(meanqre,1);
end
```