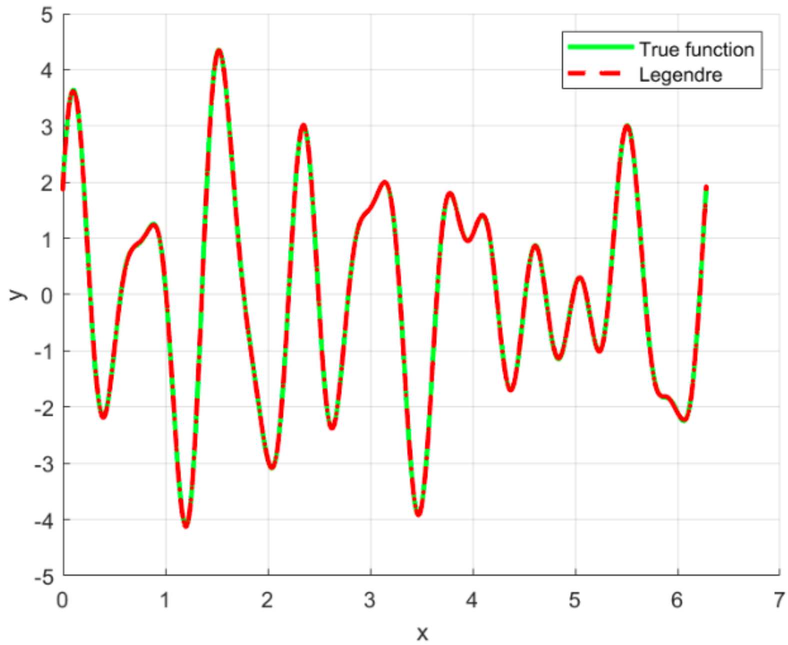
ECSE343 Group Project 1 Report

Part a

*The data for part a is obtained after implementing scaling.

By experimenting with different design choices (number of data points, location of data points, basis functions, and degree), we found a model that can almost mimic the real function as shown in the following graph (the result is after scaling):



The final design choices we take are as followed:

Number of data points: 100

Location of data points: roots of Legendre polynomials

Basis function: Legendre polynomials

Degree:50

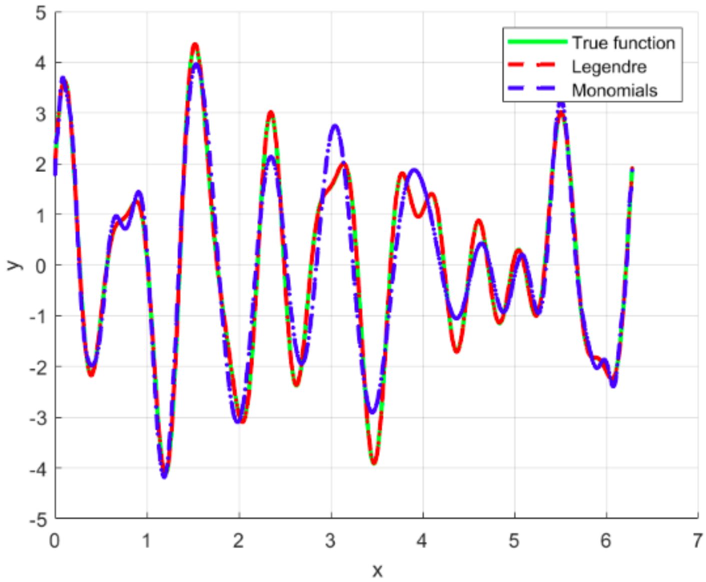
These design choices will impact the condition number for the matrix that we need to solve. For example, when experimenting with which basis function to choose, we found that the condition number for Legendre polynomials is significantly less than the one for monomials, demonstrated by the figures followed.

```
Mp = PolynomialMatrix(x,deg)
cond(Mp)
M = LegendrePolynomialMatrix(x,deg)
cond(M)
```

```
Mp = 50x51

    1.0000   -0.9989    0.9977   -0.9966    0.9955
    1.0000   -0.9940    0.9881   -0.9822    0.9763
    1.0000   -0.9854    0.9709   -0.9567    0.9427
    1.0000   -0.9729    0.9465   -0.9208    0.8958
    1.0000   -0.9566    0.9151   -0.8754    0.8374
    1.0000   -0.9367    0.8773   -0.8218    0.7697
    1.0000   -0.9131    0.8337   -0.7612    0.6951
    1.0000   -0.8860    0.7849   -0.6954    0.6161
    1.0000   -0.8554    0.7318   -0.6260    0.5355
    1.0000   -0.8216    0.6750   -0.5546    0.4556
       ⋮

ans = 4.0774e+17
```

```
M = 50x51

    1.0000   -0.9989    0.9966   -0.9932    0.9887
    1.0000   -0.9940    0.9821   -0.9645    0.9411
    1.0000   -0.9854    0.9564   -0.9137    0.8583
    1.0000   -0.9729    0.9197   -0.8427    0.7449
    1.0000   -0.9566    0.8727   -0.7536    0.6071
    1.0000   -0.9367    0.8160   -0.6494    0.4525
    1.0000   -0.9131    0.7506   -0.5335    0.2895
    1.0000   -0.8860    0.6774   -0.4096    0.1270
    1.0000   -0.8554    0.5976   -0.2818   -0.0264
    1.0000   -0.8216    0.5125   -0.1540   -0.1629
       ⋮

ans = 13.1456
```

If we also plot the basis functions for monomials and compare it to the Legendre and true function, we can see that Legendre fits the true function much better than monomials do. The graph will be as followed:



For the design choice of degree, the conduction number will increase if we increase the degree since the size of the matrix to solve increases. To illustrate, we show the following figures. The "ans" is the condition number.

```
deg = 50
Mp = 50x51
     1.0000   -0.9989    0.9977   -0.9966    0.9955 ·
     1.0000   -0.9940    0.9881   -0.9822    0.9763
     1.0000   -0.9854    0.9709   -0.9567    0.9427
     1.0000   -0.9729    0.9465   -0.9208    0.8958
     1.0000   -0.9566    0.9151   -0.8754    0.8374
     1.0000   -0.9367    0.8773   -0.8218    0.7697
     1.0000   -0.9131    0.8337   -0.7612    0.6951
     1.0000   -0.8860    0.7849   -0.6954    0.6161
     1.0000   -0.8554    0.7318   -0.6260    0.5355
     1.0000   -0.8216    0.6750   -0.5546    0.4556
        ⋮

ans = 4.0774e+17
M = 50x51
     1.0000   -0.9989    0.9966   -0.9932    0.9887 ·
     1.0000   -0.9940    0.9821   -0.9645    0.9411
     1.0000   -0.9854    0.9564   -0.9137    0.8583
     1.0000   -0.9729    0.9197   -0.8427    0.7449
     1.0000   -0.9566    0.8727   -0.7536    0.6071
     1.0000   -0.9367    0.8160   -0.6494    0.4525
     1.0000   -0.9131    0.7506   -0.5335    0.2895
     1.0000   -0.8860    0.6774   -0.4096    0.1270
     1.0000   -0.8554    0.5976   -0.2818   -0.0264
     1.0000   -0.8216    0.5125   -0.1540   -0.1629
        ⋮

ans = 13.1456
```

```
deg = 45

Mp = 50x46
     1.0000   -0.9989    0.9977   -0.9966    0.9955 ...
     1.0000   -0.9940    0.9881   -0.9822    0.9763
     1.0000   -0.9854    0.9709   -0.9567    0.9427
     1.0000   -0.9729    0.9465   -0.9208    0.8958
     1.0000   -0.9566    0.9151   -0.8754    0.8374
     1.0000   -0.9367    0.8773   -0.8218    0.7697
     1.0000   -0.9131    0.8337   -0.7612    0.6951
     1.0000   -0.8860    0.7849   -0.6954    0.6161
     1.0000   -0.8554    0.7318   -0.6260    0.5355
     1.0000   -0.8216    0.6750   -0.5546    0.4556
        ⋮

ans = 3.6422e+16
M = 50x46
     1.0000   -0.9989    0.9966   -0.9932    0.9887 ...
     1.0000   -0.9940    0.9821   -0.9645    0.9411
     1.0000   -0.9854    0.9564   -0.9137    0.8583
     1.0000   -0.9729    0.9197   -0.8427    0.7449
     1.0000   -0.9566    0.8727   -0.7536    0.6071
     1.0000   -0.9367    0.8160   -0.6494    0.4525
     1.0000   -0.9131    0.7506   -0.5335    0.2895
     1.0000   -0.8860    0.6774   -0.4096    0.1270
     1.0000   -0.8554    0.5976   -0.2818   -0.0264
     1.0000   -0.8216    0.5125   -0.1540   -0.1629
        ⋮

ans = 12.1322
```

The number of data points does not have a very significant change on the condition number: if we change the number of data points to 50, 100, or 200 for example, the changes in condition number are the following.

```
M = 50x51
    1.0000   -0.9989    0.9966   -0.9932    0.9887 ...
    1.0000   -0.9940    0.9821   -0.9645    0.9411
    1.0000   -0.9854    0.9564   -0.9137    0.8583
    1.0000   -0.9729    0.9197   -0.8427    0.7449
    1.0000   -0.9566    0.8727   -0.7536    0.6071
    1.0000   -0.9367    0.8160   -0.6494    0.4525
    1.0000   -0.9131    0.7506   -0.5335    0.2895
    1.0000   -0.8860    0.6774   -0.4096    0.1270
    1.0000   -0.8554    0.5976   -0.2818   -0.0264
    1.0000   -0.8216    0.5125   -0.1540   -0.1629
      ⋮

ans = 13.1456
```

```
M = 100x51
    1.0000   -0.9997    0.9991   -0.9983    0.9971 ..
    1.0000   -0.9985    0.9955   -0.9910    0.9850
    1.0000   -0.9963    0.9889   -0.9779    0.9633
    1.0000   -0.9931    0.9794   -0.9591    0.9323
    1.0000   -0.9890    0.9671   -0.9348    0.8926
    1.0000   -0.9839    0.9520   -0.9052    0.8446
    1.0000   -0.9778    0.9342   -0.8705    0.7890
    1.0000   -0.9708    0.9136   -0.8311    0.7266
    1.0000   -0.9628    0.8905   -0.7871    0.6584
    1.0000   -0.9539    0.8649   -0.7391    0.5851
      ⋮

ans = 12.9981
```

```
M = 200x51
    1.0000   -0.9999    0.9998   -0.9996    0.9993 ...
    1.0000   -0.9996    0.9989   -0.9977    0.9962
    1.0000   -0.9991    0.9972   -0.9944    0.9907
    1.0000   -0.9983    0.9948   -0.9896    0.9828
    1.0000   -0.9972    0.9917   -0.9834    0.9725
    1.0000   -0.9959    0.9878   -0.9758    0.9598
    1.0000   -0.9944    0.9833   -0.9667    0.9448
    1.0000   -0.9926    0.9780   -0.9562    0.9275
    1.0000   -0.9906    0.9720   -0.9443    0.9081
    1.0000   -0.9884    0.9653   -0.9311    0.8865
      ⋮

ans = 13.1087
```

We also consider how choosing the location of data points impacts the condition number. If we use linear spaced data, we will get the following condition number:

```
M = 50x51
    1.0000   -1.0000    1.0000   -1.0000    1.0000 ...
    1.0000   -0.9592    0.8800   -0.7674    0.6281
    1.0000   -0.9184    0.7651   -0.5588    0.3243
    1.0000   -0.8776    0.6551   -0.3732    0.0817
    1.0000   -0.8367    0.5502   -0.2094   -0.1060
    1.0000   -0.7959    0.4502   -0.0666   -0.2449
    1.0000   -0.7551    0.3553    0.0563   -0.3408
    1.0000   -0.7143    0.2653    0.1603   -0.3994
    1.0000   -0.6735    0.1803    0.2466   -0.4258
    1.0000   -0.6327    0.1004    0.3159   -0.4251
      ⋮

ans = 2.1196e+12
```

If we change the location of data points to be Chebyshev nodes, we will get the following condition number:

```
M = 51x51
    1.0000   -1.0000    1.0000   -1.0000    1.0000 ...
    1.0000   -0.9980    0.9941   -0.9882    0.9804
    1.0000   -0.9921    0.9764   -0.9532    0.9225
    1.0000   -0.9823    0.9473   -0.8961    0.8298
    1.0000   -0.9686    0.9072   -0.8188    0.7075
    1.0000   -0.9511    0.8568   -0.7240    0.5624
    1.0000   -0.9298    0.7967   -0.6148    0.4028
    1.0000   -0.9048    0.7281   -0.4947    0.2373
    1.0000   -0.8763    0.6519   -0.3679    0.0752
    1.0000   -0.8443    0.5693   -0.2383   -0.0749
      ⋮

ans = 14.5154
```

If we change the location of data points to roots of Legendre polynomials, we will get the following condition number:

```
M = 50x51
     1.0000   -0.9989    0.9966   -0.9932    0.9887 ...
     1.0000   -0.9940    0.9821   -0.9645    0.9411
     1.0000   -0.9854    0.9564   -0.9137    0.8583
     1.0000   -0.9729    0.9197   -0.8427    0.7449
     1.0000   -0.9566    0.8727   -0.7536    0.6071
     1.0000   -0.9367    0.8160   -0.6494    0.4525
     1.0000   -0.9131    0.7506   -0.5335    0.2895
     1.0000   -0.8860    0.6774   -0.4096    0.1270
     1.0000   -0.8554    0.5976   -0.2818   -0.0264
     1.0000   -0.8216    0.5125   -0.1540   -0.1629
        ⋮

ans = 13.1456
```

If we change the location of data points to random samples, we will get the following condition number:

```
M = 50x51
     1.0000   -0.9111    0.7451   -0.5241    0.2767 ...
     1.0000   -0.8728    0.6427   -0.3531    0.0573
     1.0000   -0.8124    0.4899   -0.1217   -0.1944
     1.0000   -0.7884    0.4324   -0.0425   -0.2656
     1.0000   -0.7806    0.4140   -0.0182   -0.2856
     1.0000   -0.7616    0.3700    0.0381   -0.3283
     1.0000   -0.7223    0.2825    0.1415   -0.3907
     1.0000   -0.7000    0.2350    0.1925   -0.4121
     1.0000   -0.6159    0.0691    0.3397   -0.4180
     1.0000   -0.5626   -0.0251    0.3987   -0.3737
        ⋮

ans = 3.8569e+16
```

So, we can see Chebyshev and Legendre are both good options to be chosen as the locations of datapoints. Legendre is slightly better than Chebyshev, so we choose Legendre.

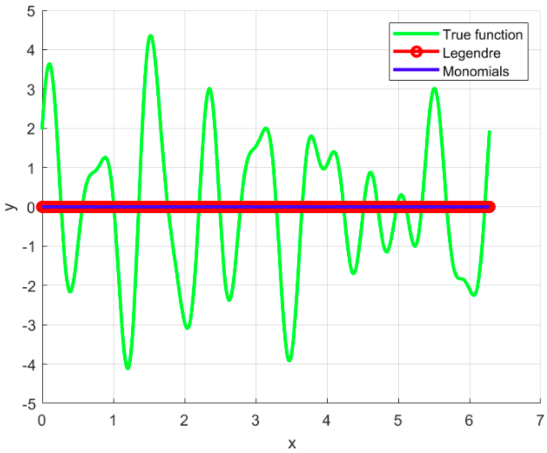
Part B

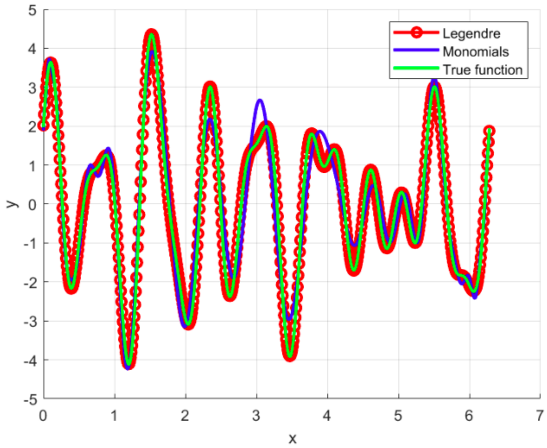
We rewrite two scaled versions for generating polynomial matrices, based on the equation given in the manual:

```
function M = ScaledPolynomialMatrix(x,n)
% write your code here
 % n is the degree of the polynomila
 % x is the vector of input data
 powers=0:n;
     for I = 1:length(powers)
         M(:,I)=(2/(max(x)-min(x))*(x-(min(x)+max(x))/2)).^powers(I);
     end
end
```

```
function M = ScaledLegendrePolynomialMatrix(x,n)
% write your code here
 % n is the degree of the polynomila
 % x is the vector of input data
 powers=0:n;
     for I = 1:length(powers)
         M(:,I)=legendreP(powers(I),(2/(max(x)-min(x))*(x-(min(x)+max(x))/2)));
     end
end
```

Using the scaled versions to replace the polynomial matrix generation methods given in the manual, we get the condition number for monomials(Mp) and Legendre(M). Below, we listed our regression graph and condition number of regression matrix after scaling on the right-hand side, and those before scaling on the left-hand side. We can see that the condition numbers are decreased dramatically by scaling the polynomial matrices.






```
Mp = 100x51
10^39 x
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000 ...
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
      :

ans = 2.7321e+47
M = 100x51
10^53 x
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000 ...
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
      :

ans = 5.3083e+60
```

```
Mp = 100x51
    1.0000   -1.0000    1.0000   -1.0000    1.0000   -1.0000 ...
    1.0000   -0.9988    0.9976   -0.9963    0.9951   -0.9939
    1.0000   -0.9966    0.9932   -0.9898    0.9864   -0.9830
    1.0000   -0.9934    0.9869   -0.9804    0.9739   -0.9675
    1.0000   -0.9893    0.9787   -0.9681    0.9578   -0.9475
    1.0000   -0.9842    0.9686   -0.9532    0.9381   -0.9233
    1.0000   -0.9781    0.9567   -0.9357    0.9152   -0.8951
    1.0000   -0.9711    0.9430   -0.9157    0.8892   -0.8635
    1.0000   -0.9631    0.9275   -0.8933    0.8603   -0.8286
    1.0000   -0.9542    0.9104   -0.8687    0.8289   -0.7909
      :

ans = 8.3580e+16
M = 100x51
    1.0000   -1.0000    1.0000   -1.0000    1.0000   -1.0000 ...
    1.0000   -0.9988    0.9963   -0.9927    0.9878   -0.9817
    1.0000   -0.9966    0.9898   -0.9796    0.9661   -0.9493
    1.0000   -0.9934    0.9803   -0.9608    0.9351   -0.9034
    1.0000   -0.9893    0.9680   -0.9365    0.8952   -0.8450
    1.0000   -0.9842    0.9529   -0.9068    0.8472   -0.7753
    1.0000   -0.9781    0.9350   -0.8721    0.7915   -0.6958
    1.0000   -0.9711    0.9144   -0.8326    0.7291   -0.6082
    1.0000   -0.9631    0.8913   -0.7886    0.6607   -0.5144
    1.0000   -0.9542    0.8657   -0.7406    0.5873   -0.4163
      :

ans = 13.1490
```