Table of Contents


```
function submission4

%*Benjamin Palay: 1815594
%*Lab 4: Polynomial Interpolation
```

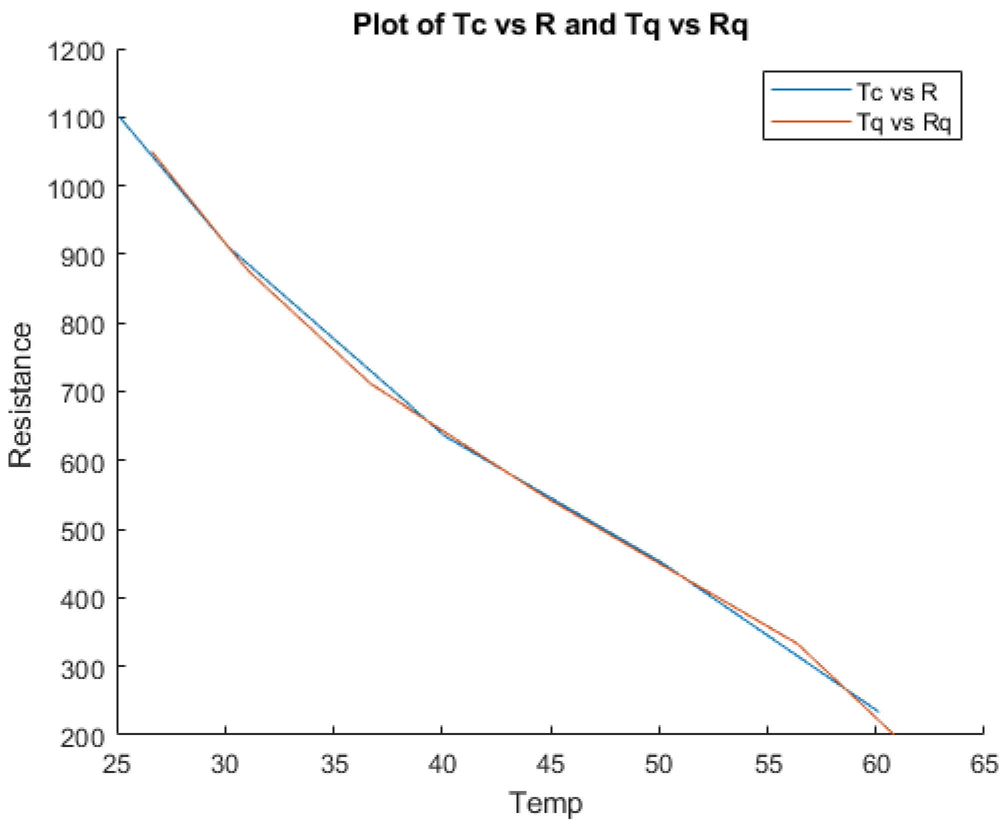
Question 1A)

```
format long
R = [1101.0; 911.3;636.0;451.1;233.5];
Tc = [25.113; 30.131;40.120;50.128;60.136];
[co,T] = NewtonInterp(R,Tc)


co =

  Columns 1 through 3

  25.113000000000000  -0.026452293094360   0.000021143571323

  Columns 4 through 5

  -0.000000027123585  -0.000000000131573


T =

  Columns 1 through 3

  25.113000000000000                   0                   0
  30.131000000000000  -0.026452293094360                   0
  40.119999999999997  -0.036284053759535   0.000021143571323
  50.128000000000000  -0.054126554894538   0.000038771188907
  60.136000000000003  -0.045992647058824  -0.000020208466673

  Columns 4 through 5

                   0                   0
                   0                   0
                   0                   0
  -0.000000027123585                   0
   0.000000087016311  -0.000000000131573
```

Question 1B)

```
Rq = [1050.1, 901.56, 875.11, 711.40, 545.27, 333.1, 200];
Tq = NewtonInterp2 (R,Tc,Rq)

figure(1); hold on
title('Plot of Tc vs R and Tq vs Rq')
xlabel('Temp ')
ylabel('Resistance ')
plot(Tc,R,'DisplayName', 'Tc vs R');
hold on
plot(Tq, Rq, 'DisplayName', 'Tq vs Rq'); hold off
legend('Tc vs R','Tq vs Rq')
```

```
Tq =

   26.6200   30.3852   31.0991   36.7051   44.8435   56.3732   60.8444
```

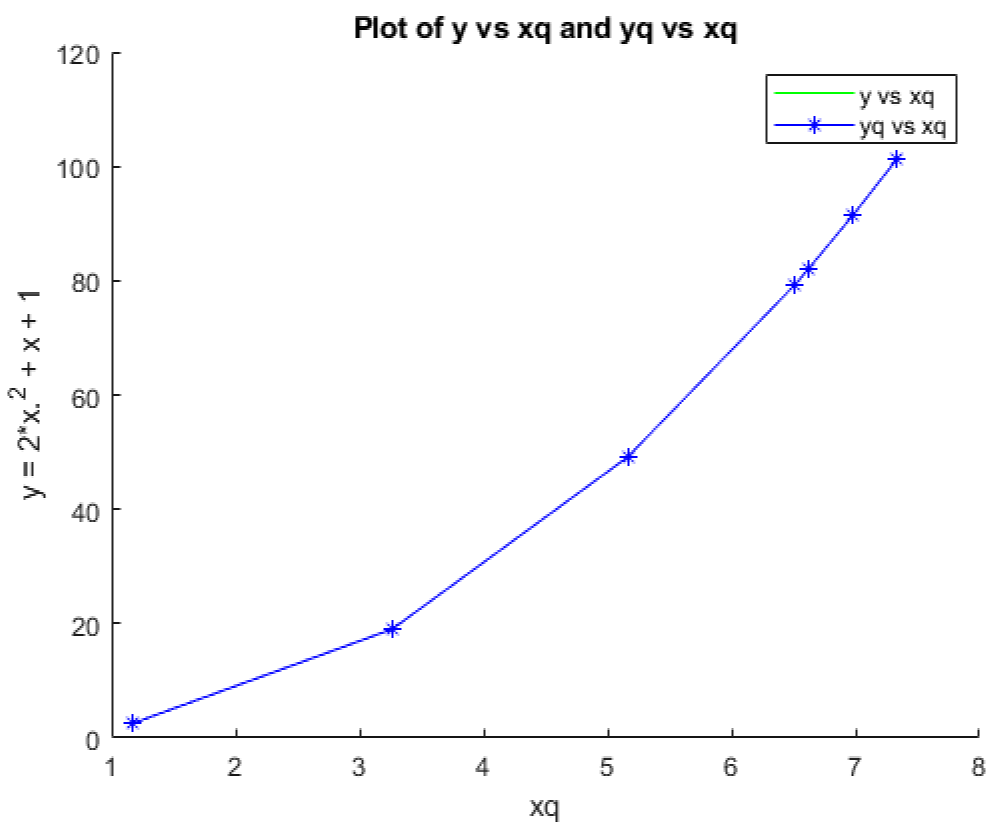


Question 2)

```
N = 14;
xdata = unifrnd(0,10,1,N);  %generate N random numbers between 0 and
 10
```

```
xdata = sort(xdata) %put in ascending order

y = 2*xdata.^2  - xdata + 1   %f(x) = 2x^2 + x + 1
xq = unifrnd(0,10,1,N/2);
xq = sort(xq)
y2 = 2*xq.^2 - xq + 1;
yq = NewtonInterp2 (xdata',y',xq)

figure(2.0); hold on
title('Plot of y vs xq and yq vs xq')
xlabel('xq')
ylabel('y = 2*x.^2 + x + 1')
plot(xq,y2, 'g');
hold on
plot( xq,yq, '-b*'); hold off
legend('y vs xq','yq vs xq')
```

```
xdata =

  Columns 1 through 7

    0.6928    0.7021    0.9240    1.0482    1.1403    1.3601    2.3787

  Columns 8 through 14

    2.4365    5.9863    6.1785    6.8148    7.8889    7.9625    8.5835


y =

  Columns 1 through 7

    1.2671    1.2839    1.7835    2.1493    2.4603    3.3395    9.9376

  Columns 8 through 14

   10.4364   66.6849   71.1694   87.0678  117.5810  119.8389  139.7704


xq =

    1.1757    3.2639    5.1627    6.5053    6.6178    6.9820    7.3374


yq =

    2.5887   19.0420   49.1444   79.1327   81.9718   91.5146  101.3381
```



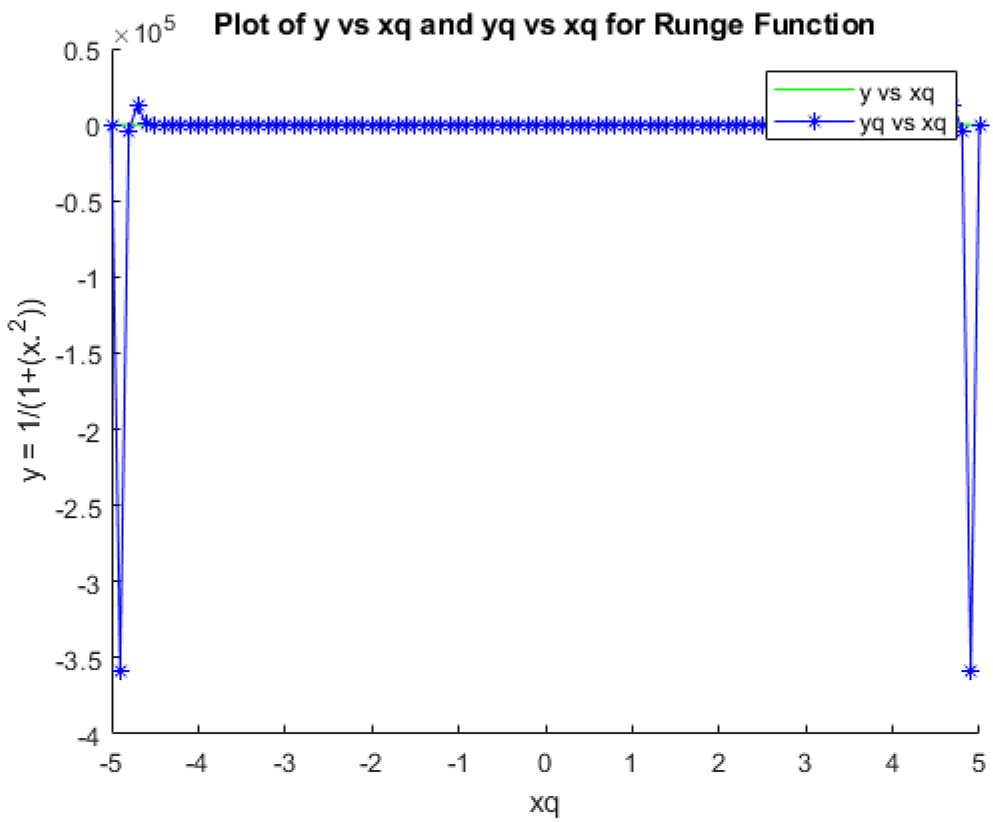
Question 3A)

```
n = [5 ,10 ,20 ,50];
xq = linspace(-5,5,101);

for i = 1:101
    y(i)= 1/(1+(xq(i).^2));
end

 for k = 1:4
xdata = linspace(-5,5, n(k));

for i = 1:n(k)
    ydata(i) = 1/(1+(xdata(i).^2));
end

yq = NewtonInterp2 (xdata',ydata',xq);

for i=1:101
md(i) = abs(y(i) - yq(i));
end
maxd(k) = max(md);
 end
 figure(3.0); hold on
title('Plot of y vs xq and yq vs xq for Runge Function')
```

```
xlabel('xq')
ylabel('y = 1/(1+(x.^2))')
plot(xq,y,'g');
hold on
plot( xq,yq, '-b*'); hold off
legend('y vs xq','yq vs xq')
 f(:,1) = n;
f(:,2) = maxd;
fprintf("   Maximum Difference Table:\n\n");
fTable = array2table(double(f),'VariableNames',...
    {'n','Max Difference'});
disp(fTable);
```

Maximum Difference Table:

n	*Max Difference*
5	*0.43813*
10	*0.2984*
20	*8.1665*
50	*3.5881e+05*



Question 3B)

According to the Weierstrauss Appoximation Theorem, if a function is continuous and on a closed interval, there exists a polynomial that is similar to the function. However, this is not true for every function As shown in 3a, certain functions have large oscillations when approximated using this method. For higher-degree functions, the interpolating polynomial diverges for large n. This is known as Runge's Phenomenon.

```
end
```

Published with MATLAB® R2020a