UNIVERSIDAD ANA G. MÉNDEZ | UAGM Recinto de Gurabo

Engineering Department

COMP 411- FINAL PROJECT

Frank F. Fabian - S01242357

Nombre del Profesor de .: YAHYA MASALMAH

Fecha de entrega: 05/10/2021

Introduction

As we progress to a professional level, we find that mathematics progressively increased in complexity. We can see that from a more realistic perspective the problems that are planted in real life the problems that are planted in everyday life, especially in engineering, which include the mathematical content as a starting plan to solve the problems raised.

As engineers we not only find a solution to the problems but also an efficient, theoretical and practically application that will undoubtedly be affected by means other than practical values that. we take into account to successfully conclude a situation by optimizing it thanks to numerical. methods obtaining an exact solution. and process the problem.

In the work that follows, the basic concepts to take into account and successfully start the course. of numerical methods are presented in a simple way, advancing to complex situations to use. computational means and developing small software for large solutions.

Problem 1: Write separate MATLAB functions for Newton's, the Secant method, the modified secant method. The functions should be written so that they can be called in. MATLAB by typing.

1. [X, NumIters]=Newton(@f, @df, x0, TOL, MaxIters)

2. [X, NumIters]=Secant(@f, x0, x1, TOL, MaxIters)

3. [[X, NumIters]=ModifiedSecant(@f, δ, x0,TOL, MaxIters)

1) <u>Newton method</u>

```
clc
f= @(x) x^3-x-1;
df= @(x) 3*x^2-1;
x0= 1.5;
TOL= 0.001;
MaxIters=50;


Newton(f,df,x0,TOL,MaxIters);
function [ X, NumIters] = Newton( f,df,x0,TOL,MaxIters)

  for i = 1: MaxIters
      f0=vpa(subs(f,x0)); %Calculating the value of
function at x0
      f0_der=vpa(subs(df,x0));
       X=x0-(f0/f0_der);
       NumIters=i-1;
       err = abs(X-x0);
       if err <TOL
           break;
       end
  end

fprintf('Root is : %.2f with accuracy: %.2f\n',X,err);
fprintf('Number of iterations:%d\n',NumIters);
end
```

2) **<u>secant method</u>**

```
% Matlab code of secant method
% f is input function,x0 is first point of guess
interval,
%x1 is second point of guess interval,TOL is given
tolerance
function [X,Numitr]= secant(f,x0,x1,ToL,Maxitr)
x(1)=x0;
x(2)=x1;
Numitr=0;
for i=3:Maxitr
   x(i) = x(i-1) - (f(x(i-1)))*((x(i-1) - x(i-
2))/(f(x(i-1)) - f(x(i-2))));
    Numitr=Numitr+1;
    if abs((x(i)-x(i-1))/x(i))*100<ToL
        X=x(i);
        Numitr;
        break
    end

end
```

```
>> [X,Numitr]=secant(@(x) cos(x)+2*sin(x)+x^2,0,-0.01,0.001,15)

X =

   -0.6593


Numitr =

     7
```

```
Root is : 1.35 with accuracy: 0.15
Number of iterations:49
>>
```

3) <u>ModifiedSecant</u>

```
% Matlab code of  Modified secant method
% f is input function,x0 is starting point
%TOL is given tolerance,delta is small perturbation
factor
function [X,Numitr]=
Modified_secant(f,x0,ToL,delta,Maxitr)
x(1)=x0;
Numitr=0;
for i=2:Maxitr
   x(i) = x(i-1) - (f(x(i-1))*delta*x(i-1))/(f(x(i-
1)+delta*x(i-1)) - f(x(i-1)));
    Numitr=Numitr+1;
    if abs((x(i)-x(i-1))/x(i))*100<ToL
        X=x(i);
        Numitr;
        break
    end

end
```

```
>> [X,Numitr]=Modified_secant(@(x) exp(-x)-x,1,0.01,0.01,100)

X =

    0.5671


Numitr =

     4
```

Problem 2: Figure 1 shows a circuit with a resistor, an inductor, and a capacitor in parallel. Kirchhoff's rules can be used to express the impedance of the system as

$$\frac{1}{Z} = \sqrt{\frac{1}{R^2} + (wC - \frac{1}{wL})^2}$$

Where Z =impedance (Ω) and w=the angular frequency. Find the w that result in animpedance of 75 Ω using the implemented methods in problem 1.

1. Newton x0=1

2. Secant method with x0=1, x1=100

3. Modified Secant method with x0=1

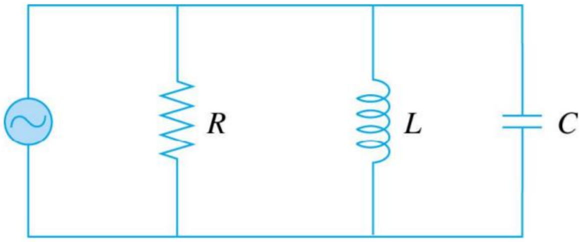


Figure 1. RLC Circuit.

Use the following parameters: R=225 Ω, C x F 6 0.6 10 − = and L = 0.5H . Determine how many iterations of each method are necessary to determine the answer to ε s = 0.1%. If any difficulties arise, please use the graphical method to explain them.

a) From newton's implimentation method

$$X_{n+1} = X_n - \frac{F(X_n)}{F'(X_n)} \quad \ldots\ldots (1)$$

Where, $F(X_n)$ is the given function for the n^{th} value of X

$F'(X_n)$ is the derivative of the given funtion

From the given expression

$$\left(\omega C - \frac{1}{\omega L}\right)^2 = \frac{1}{Z^2} - \frac{1}{R^2}$$

$$\omega C - \frac{1}{\omega L} = \sqrt{\left(\frac{1}{Z^2} - \frac{1}{R^2}\right)}$$

$$= \sqrt{\left(\frac{1}{(75)^2} - \frac{1}{(225)^2}\right)}$$

$$\omega C - \frac{1}{\omega L} == 0.0125$$

$$\boxed{\frac{\omega^2 CL - 1}{\omega L} = 0.0125} \quad \ldots\ldots\ldots (2)$$

From equation (2)

$$\omega^2 CL - 0.0125\,\omega L - 1 = 0$$

$$\omega^2 (0.6 \times 10^{-6})(0.5) - 0.0125\,\omega\,(0.5) - 1 = 0$$

$$\boxed{(3 \times 10^{-7})\,\omega^2 - 0.0625\,\omega - 1 = 0} \quad \ldots\ldots (3)$$

Implimenting Newton's method in equation (3)

$$F(w) = (3\times10^{-7})w^2 - 0.0625w - 1 = 0 \quad \ldots\ldots (4)$$

Take the derivative of equation (4)

$$F'(w) = 2(3\times10^{-7})w - 0.0625 = 0 \quad \ldots\ldots (5)$$

therefore.

$$w_1 = w_0 - \frac{F(w)}{F'(w)}$$

$$w_1 = 1 - \frac{(3\times10^{-7})w^2 - 0.0625w - 1}{2(3\times10^{-7})w - 0.0625} \quad \text{For Newton } (w_0 = 1)$$

After solving the above expresion we get

$$\boxed{w_1 = 2.08\times10^6 \text{ Hz}}$$

b) similarly.

Calculate for secant method with $w_0 = 1, w_1 = 100$

$$X_{n+1} = X_n - \frac{F(X_n)(X_{n-1} - X_n)}{F(X_{n-1}) - F(X_n)} \quad \ldots\ldots (6)$$

c) Similarly

Calculate for 'Modified secant method' with $\omega_0 = 1$

$$x_{n+1} = x_n - \frac{\delta x_n \, f(x_n)}{f(x_n + \delta x_n) - f(x_n)} \quad \text{..........} (7)$$

Where, δ is the small perturbation (0.01)

Hence,

The value of angula frequency (ω) is,

$$\boxed{\omega = 2.08 \times 10^6 \, Hz}$$

Problem 3: Given the following data points,

x	1.6	2	2.5	3.2	4	4.5
y	2	8	14	15	8	2

1. Construct the divided difference table for this table
2. Find the Newton's interpolating polynomial using the divided difference technique (1st , 2nd order) find f(2.8).
3. Repeat the previous step using Lagrange interpolation
4. Repeat 1 and 2 by running the given code for each technique in the handouts
5. Write a MATLAB code that plot the polynomials found in 2 and 3 on the same scale.

1. here , $X = x_i$ $y = f(x_i)$

Divided difference table:

x_i	$f(x_i)$	1st order differences	2nd "	3rd "	4th "	5th "
1.6	2					
		$\frac{8-2}{2-1.6} = 10$				
2	8		$\frac{12-10}{2.5-1.6} = 2.22$			
		$\frac{14-8}{2.5-2} = 12$		$\frac{-8.81-2.22}{3.2-1.6} = 6.9$		
2.5	14		$\frac{1.43-12}{3.2-2} = -8.81$		$\frac{1.01-(-6.9)}{4-1.6} = 3.3$	
		$\frac{15-14}{3.2-2.5} = 1.43$		$\frac{-6.79-(-8.81)}{4-2} = \Rightarrow 1.01$		$\frac{0.6-3.3}{4.5-1.6} = -1$
3.2	15		$\frac{-8.75-1.43}{4-2.5} = \Rightarrow -6.79$		$\frac{2.15-1.01}{4.5-2} = 0.6$	
		$\frac{8-15}{4-3.2} = -8.75$		$\frac{-2.5-(-6.79)}{4.5-2.5} = 2.15$		
4	8		$\frac{-12-(-8.75)}{4.5-3.2} = -2.5$			
		$\frac{2-8}{4.5-4} = -12$				
4.5	2					

2) The Newton's interpolating polynomial using the divided difference technique (1st, 2nd order)

$$F(x) = F[x_0] + (x-x_0)\, F[x_0, x_1] + (x-x_0)(x-x_1)\, F(x_0, x_1, x_2)$$

$$f(2.8) = f[1.6] + (2.8-1.6)(2.22) + (2.8-1.6)(2.8-2)(-6.9)$$

$$f(2.8) = 2 + 2.664 - 6.624$$

$$F(2.8) = -1.96$$

3. Lagrange's interpolation formula:

$$F(x) = \frac{(x-x_1)(x-x_2)(x-x_3)(x-x_4)(x-x_5)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)(x_0-x_5)} F(x_0) + \ldots + \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)}{(x_5-x_0)(x_5-x_1)(x_5-x_2)(x_5-x_3)(x_5-x_4)} F(x_5)$$

$$F(2.8) = \frac{(2.8-2)(2.8-2.5)(2.8-3.2)(2.8-4)(2.8-4.5)}{(1.6-2)(1.6-2.5)(1.6-3.2)(1.6-4)(1.6-4.5)} \times 2$$

$$+ \frac{(2.8-1.6)(2.8-2.5)(2.8-3.2)(2.8-4)(2.8-4.5)}{(2-1.6)(2-2.5)(2.-3.2)(2-4)(2-4.5)} \times 8$$

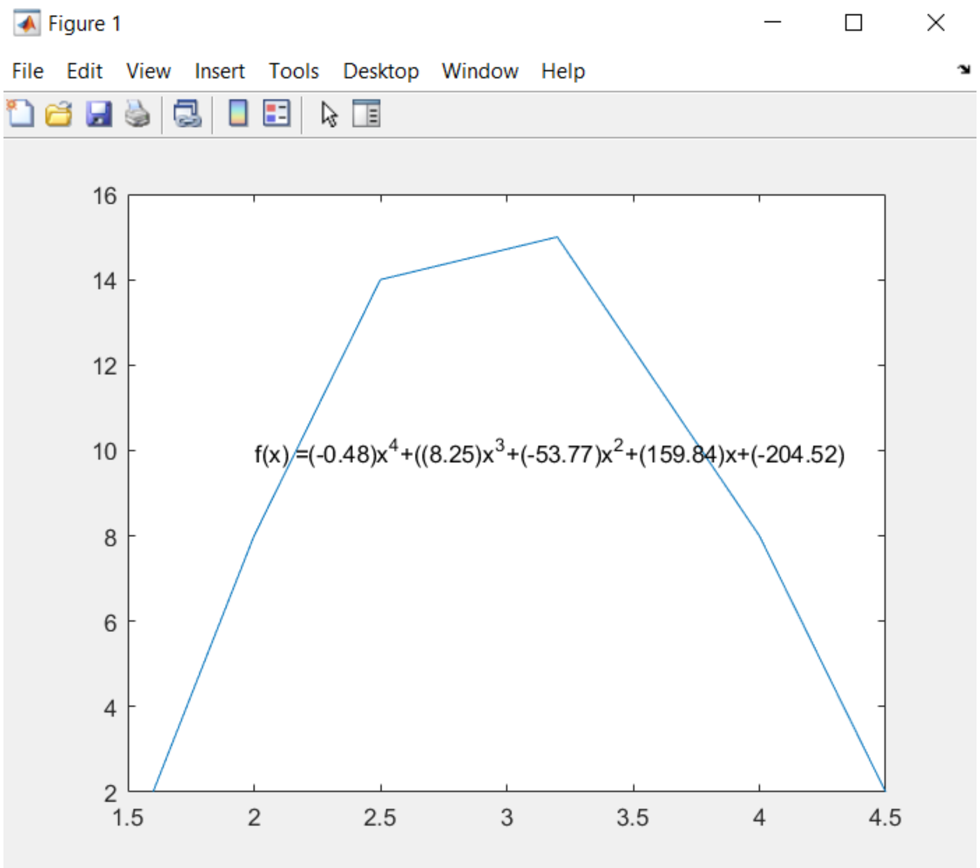
$$+ \frac{(2.8-1.6)(2.8-2)(2.8-3.2)(2.8-4)(2.8-4.5)}{(2.5-1.6)(2.6-2)(2.5-3.2)(2.5-4)(2.5-4.5)} \times 14$$

$$+ \frac{(2.8-1.6)(2.8-2)(2.8-4)(2.8-4.5)}{(3.2-16)(4.5-2)(4.5-2.5)(4.5-3.2)(4.5-4)} \times 2$$

$$\Rightarrow f(2.8) = 15.60$$

4) <u>(1) divided diffrence table</u>

```
function  [C,D] = newpoly(X,Y)
%___________________________________________
%Sample calls
% [C] = newpoly(X,Y)
% [C,D] = linewpoly(x,y)
%inputs
%  X  vector of abscissas
%  y  vector of ordinates
% Return
%  C  coefficient list for the newton polynomial
%  D  divide difference table

%____________________________________________
X=[1.6 2 2.5 3.2 4 4.5]
Y=[2 8 14 15 8 2]
plot(X, Y)
hold on
n = length(X);
D = zeros(n,n);
D(:,1) = Y';
for j=2:n,
    for k=j:n,
        %Value of divide diffrence table
        D(k,j) = (D(k,j-1)-D(k-1,j-1))/(X(k)-X(k-j+1))
    end
end

C= D(n,n);
for k=(n-1):-1:1
    C= conv(C,poly(X(k)));
    m= length(C);
    C(m) = C(m)+D(k,k);

end
%polynomial of divide diffrence table
text(2,10,sprintf('f(x)
=(%.2f)x^4+((%.2f)x^3+(%.2f)x^2+(%.2f)x+(%.2f)', C(1),
C(2), C(3), C(4), C(5)))
%interpolating value of newton polynomial
x=2.8;
```



(2) **by laneange interpolation**

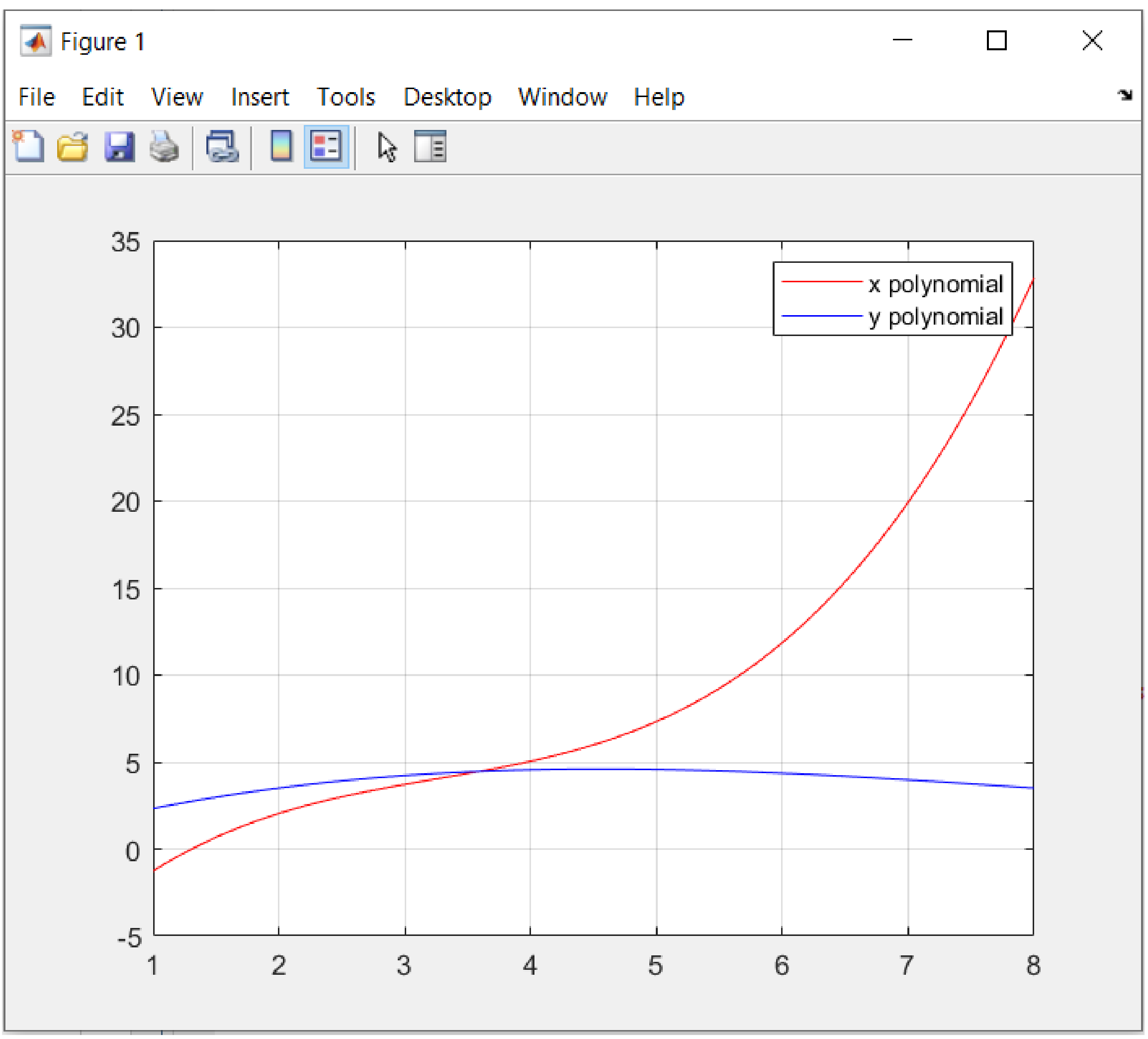
```
function s=LagrangeInter(x,y,x0)
%%LagrangeInter: find Lagrange interpolation polynomial
function
% x is the node X coordinate vector
% y is the node y coordinate of the point to be
calculated
x=[1.6 2 2.5 3.2 4 4.5]
y=[2 8 14 15 8 2]
x0=2.8
syms p;
n = length (x); %read x vector dimension
s=0;
for i=1.3
    Li=y(i);
    %contrutor basis function
    for j=1:i-1
        Li=Li*(p-x(j))/(x(i)-x(j));
```

```
      End
    for j=i+1:n
          Li=Li*(p-x(j))/(x(i)-x(j));
      end
      s=s+Li;
      simplify(s);
  end

  %If there are only two parameters, specify the number of
  input parameters
  % Provides a direct connection if there are three
  parameters. Receive the result of the% heading, t

  if (nargin==2)
      s=subs(s,'p','x');
         s = collect (s); %expansion polynomial
       %s = vpa(s, 4);
  else
      s = subs (s,'p',x0);
          %m =lenth (x0); %read t length
   %Interpolates each component od t separately
   %for  i i=1:m
   %temp(i)=subs(s,'p' ,x0(i));
   %end
   % s=temp;
                                                    end
```

```
>> LagrangeInter

x =

  Columns 1 through 5

    1.6000    2.0000    2.5000    3.2000    4.0000

  Column 6

    4.5000


y =

     2     8    14    15     8     2


x0 =

    2.8000
```

5)

```
% defining data points
x = [1.6 2 2.5 3.2 4 4.5];
y = [2 8 14 15 8 2];
range = 1:6;
% fitting the points
p1 = polyfit(x,range,3);
p2 = polyfit(y,range,3);
% setting range in plot
t = 1:.1:8;
% evaluting the polynomial
y1 = polyval(p1,t);
y2 = polyval(p2,t);
plot(t,y1,'r',t,y2,'b')
legend('x polynomial','y polynomial');
grid on
```



Problem 4: Generate eight equally-spaced points from the function

$$f(t) = sin\text{^}2(t)$$

From t=0 to 2π. Fit these data with

a. Seventh Order interpolating polynomial

b. A cubic spline.

4) Given function

$F(t) = \sin^2 t \quad , t = 0 \text{ to } 2\pi$

so, Length of interval $= \dfrac{2\pi - 0}{7}$

$= \dfrac{2\pi}{7}$

so,

t:	0	$\frac{2\pi}{7}$	$\frac{4\pi}{7}$	$6\pi/7$	$8\pi/7$	$10\pi/7$	$12\pi/7$	2π
$F(t)$:	0	.6113	.9505	.1883	.1883	.9505	.6113	0

a) seventh degree Polynomial is

$$F_7(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 + a_6 t^6 + a_7 t^7 \rightarrow ①$$

use $F(t) = 0$ at $t = 0 \Rightarrow a_0 = 0$

$$.6113 = a_1\left(\frac{2\pi}{7}\right) + a_2\left(\frac{2\pi}{7}\right)^2 + \ldots\ldots a_7\left(\frac{2\pi}{7}\right)^7 \rightarrow ②$$

$$.9505 = a_1\left(\frac{4\pi}{7}\right) + a_2\left(\frac{4\pi}{7}\right)^2 + \ldots\ldots + a_7\left(\frac{4\pi}{7}\right)^7 \rightarrow ③$$

⋮

and soon

we have a system of linear equation $Ax = B$ where A is of order 7×7 $\quad X = [a_1\ a_2\ a_3 \ldots\ldots a_7]^T$

By solving this system $Ax = B$, we get $a_1, a_2, a_3 \ldots\ldots a_7$ and Putting all $a_0, a_1, a_2, \ldots\ldots a_7$ in eq① we get required Polynomial

b) A Cubic Spline is $[X_0, X_n]$ is

Written by

$$S_3(X) = \sum_{K=-1}^{N+1} a_K B_K(k)$$

so for $[0, 2\pi]$ we have eight points

$$h = \frac{2\pi}{7}$$

$$\text{so } S_3(X) = \sum_{K=-1}^{8} a_K B_K(n)$$

$$B_0(n) = \begin{cases} 0 & X < 0 - \frac{4\pi}{7} \\ \frac{1}{6}\left(\frac{4\pi}{7}+n\right)^3 & -\frac{4\pi}{7} \le Y \le 0 - \frac{2\pi}{7} \\ \frac{2}{3}\left(\frac{2\pi}{7}\right)^3 - \frac{1}{2}n^2\left(\frac{4\pi}{7}+x\right) & \frac{2\pi}{7} \le n \le 0 \\ \frac{2}{3}\left(\frac{2\pi}{7}\right)^3 - \frac{1}{2}x^2\left(\frac{4\pi}{7}-x\right) & 0 \le n \le \frac{2\pi}{7} \\ \frac{1}{6}\left(\frac{4\pi}{7}-x\right)^3 & n \ge \frac{4\pi}{7} \\ 0 & \end{cases}$$

We get a linear system of the form $AX=B$ of order 8×8. solving we have $a_0, a_1, a_2 \ldots\ldots a_7$

Then find $S_3(n)$.

Problem 5: Use nonlinear regression to estimate α and β of the following model

$$y = \alpha_4 x e^{\beta_4 x}$$

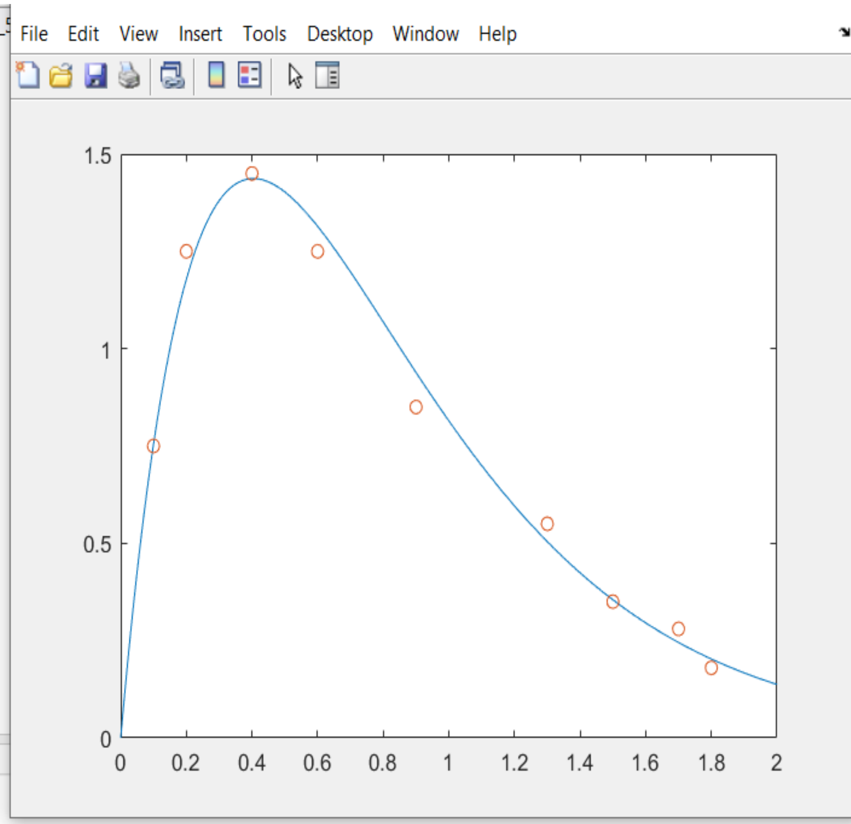
Based on the following data. Develop a plot of your fitting model along with the data .

X	0.1	0.2	0.4	0.6	0.9	1.3	1.5	1.7	1.8
Y	0.75	1.25	1.45	1.25	0.85	0.55	0.35	0.28	0.18

```
clc
clear all
close all
format long

x=[0.1,0.2,0.4,0.6,0.9,1.3,1.5,1.7,1.8];
y=[0.75,1.25,1.45,1.25,0.85,0.55,.35,.28,0.18];

A=[ones(numel(x),1) x(:)];
b= [log(y(:)./x(:))];
C=A\b;

    beta = C(2)
    alpha = exp(C(1))
    f=@(x) alpha*x.*exp(beta*x);

    xx=0:0.01:2;
    yy=f(xx);

    plot(xx,yy,x,y,'o');
```

newton.m | secant.m | secantModified.m | 5j.m | seve_problema_5

```
clc
clear all
close all
format long

x=[0.1,0.2,0.4,0.6,0.9,1.3,1.5,1.7,1.8];
y=[0.75,1.25,1.45,1.25,0.85,0.55,.35,.28,0.18];

A=[ones(numel(x),1) x(:)];
b= [log(y(:)./x(:))];
C=A\b;

beta = C(2)
alpha = exp(C(1))
f=@(x) alpha*x.*exp(beta*x);

xx=0:0.01:2;
yy=f(xx);

plot(xx,yy,x,y,'o');
```



Command Window

```
  -2.473308765704636


alpha =

   9.661785859642913
```

Conclusion

Numerical solutions allow the computational analysis process to be very accurate. In this technique, the correlation is represented by the correlation between the actual values and the approximate number assigned between them. Here a large number and an associated number of errors are considered. From this knowledge, you can improve and implement regular software and improve the error count you use. Numerical methods have become an important part of our calculations due to the success of our results and the research we use to generate problems accordingly.