Kamal Giri
Scientific Computing
Homework4
Last Modified:
03/10/2023

Problem 2:

Main code:
Newton_method.m

```
%Newton Method Function implmentation


function [x, iter] = newton_Method(F, J, x0, tol, max_iter)
disp("    no of Iterations          ||f(x^k)||       " + ...
    "         x                   y                  z   ");


x = x0;
iter = 0;



while iter < max_iter
    f = F(x);
    j = J(x);
    delta = -j \ f;
    x = x + delta;
    iter = iter + 1;
    if(iter<6)
        print = [iter,norm(f) x'];
        disp(print);
    end

    % Check for convergence.
    if norm(f) < tol

        break;
```

```
    end
end

end
```

ScriptFile: script_Newton_Method.m

```
%Scipt for Newton Method

%Creating the functions and Jacobian Matrix of Partial Derivatives
F = @(x) [x(1) + x(2)^2 + x(3) - 4; x(1)^2 + x(2) + x(3) - 6; x(1) + x(2) + x(3)^2 - 4];
J = @(x) [1, 2*x(2), 1; 2*x(1), 1, 1; 1, 1, 2*x(3)];
%initial guess
x0 = [0; 0; 0];
%tolerance
tol = 1e-7;
%maximum iteration
max_iter = 100;

% Solve the system using Newton's method.
[x, iter] = newton_Method(F, J, x0, tol, max_iter);
```

Output:

```
>>scripforNewtonMethod
 no of Iterations          ||f(x^k)||              x                   y                    z
 1.000000000000000  8.246211251235321  1.000000000000000  3.000000000000000  3.000000000000000

 2.000000000000000 12.767145334803704  1.916666666666667  1.583333333333333  1.583333333333333

 3.000000000000000  2.960020059053654  1.953363154406892  1.092859509609013  1.092859509609012

 4.000000000000000  0.340212345145298  1.999012337992055  1.003016760258766  1.003016760258766

 5.000000000000000  0.011603781163044  1.999998475836334  1.000003534560095  1.000003534560095
```

Problem3:

Scriptfile:

```
%Data Set
clear all;
format long;

x  = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5];
y = [1.364, 1.637, 1.911, 2.633, 3.221, 3.977];
xnew = 0:0.01:0.5;



%plot(x, y, '*');

x2 = x.*x;

%Matrix
M = [sum(x2) sum(x) ; sum(x) length(x)];

ylog = log(y);
xy = x .* ylog;

V = [sum(xy); sum(ylog)];

coeff = M\V;
A = coeff(1)
B = coeff(2);

C = exp(B)
yfit = C * exp(A *xnew);

%Actually plotting the graph
plot(x, y, '*', xnew, yfit);
```

```
title("Exponential fit ");
xlabel("Values of X");
ylabel("Values of f(x)");

legend("Actual data point ");
```

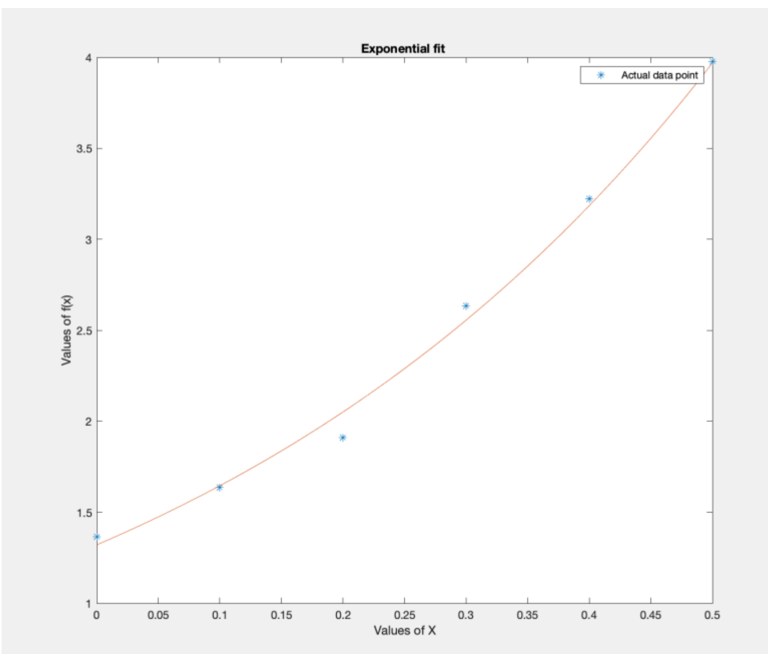
Output:

A = 2.200430847187897

B = 0.278101794895891

C = 1.320620622805869

Graph:



Problem 4:
Script File: quadraticfit.m

```
clear all;
format long;
```

```
x = 0:4;
y = [0.685, -1.375, -1.255, 0.781, 4.76];
xfit = 0:0.1:4;

%plot(x,y,'*');

x2 = x.*x;
x3 = x2.*x;
x4 = x2.*x2;
n = length(x);
xy = x.*y;
x2y= x2.*y;

%Defining matrix and vector

M  = [sum(x4) sum(x3) sum(x2); ...
      sum(x3) sum(x2)  sum(x);...
      sum(x2) sum(x)  n];

V = [sum(x2y);sum(xy); sum(y)];

coeff = M\V;

%Getting A, B and C

A = coeff(1)
B = coeff(2)
C = coeff(3)

yfit = A * xfit.^2 + B * xfit + C;
```

```
plot(x, y, '*', xfit, yfit);

title("Quadratic Fit ");
xlabel("Values of X");
ylabel("Values of F(x)");

legend("Actual data point ");
```

Output:

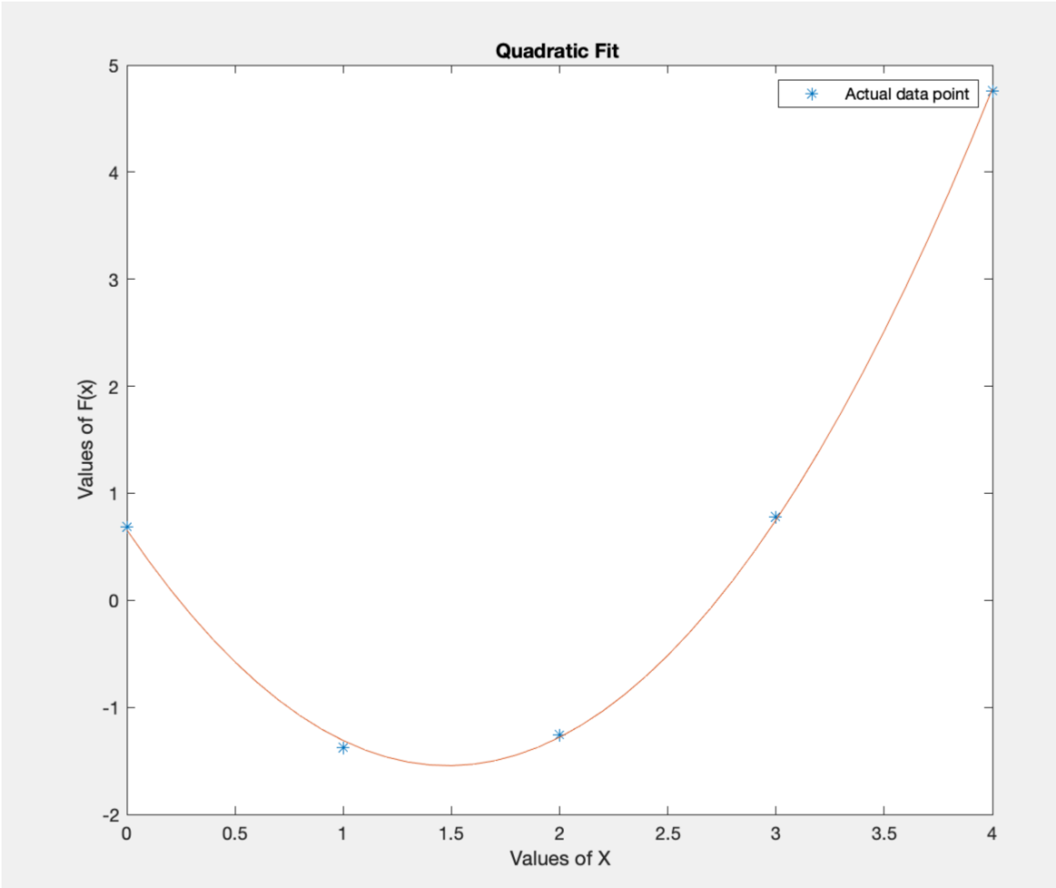
```
>> quadraticfit

A = 0.999571428571431

B = -2.967685714285726

C =  0.657142857142866
```

Plot:



Problem 3:
Here I am trying to implement the different/ improved version of the matlab code as provided in text book.

Scriptfile: exponentialfitv2.m

```
%Data Set
clear all;
format long;

x  = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5];
y = [1.364, 1.637, 1.911, 2.633, 3.221, 3.977];
xfit = 0:0.01:0.5;

%Trying the exponential fit as given in the text book which leads to
%modified system of equations
```

```
x2 = x.*x;
x2y = x2.*y;
xy = x.*y;
ylog = log(y);
xylogy = xy.*log(y);
ylogy = y.*log(y);

%Matrix of the above system

M = [sum(x2y) sum(xy); sum(xy) sum(y)];

%vector
V = [sum(xylogy); sum(ylogy)];

%Finding the coeff

coeff = M\V;

A = coeff(1)
B =  coeff(2)
C = exp(B)

yfit =  C * exp(A * xfit);

%Actually plotting the graph
plot(x, y, '*', xfit, yfit);
title("Exponential Fit Improved ");
xlabel("Values of X");
ylabel("Values of f(x)");

legend("Actual data point ");
```

Output:

>> exponentialfitv2

A = 2.218359428030302

B = 0.274179620308858

C = 1.315451062745675

Plot: