FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Spring 2018

MATH2059 Numerical Methods

Homework #1

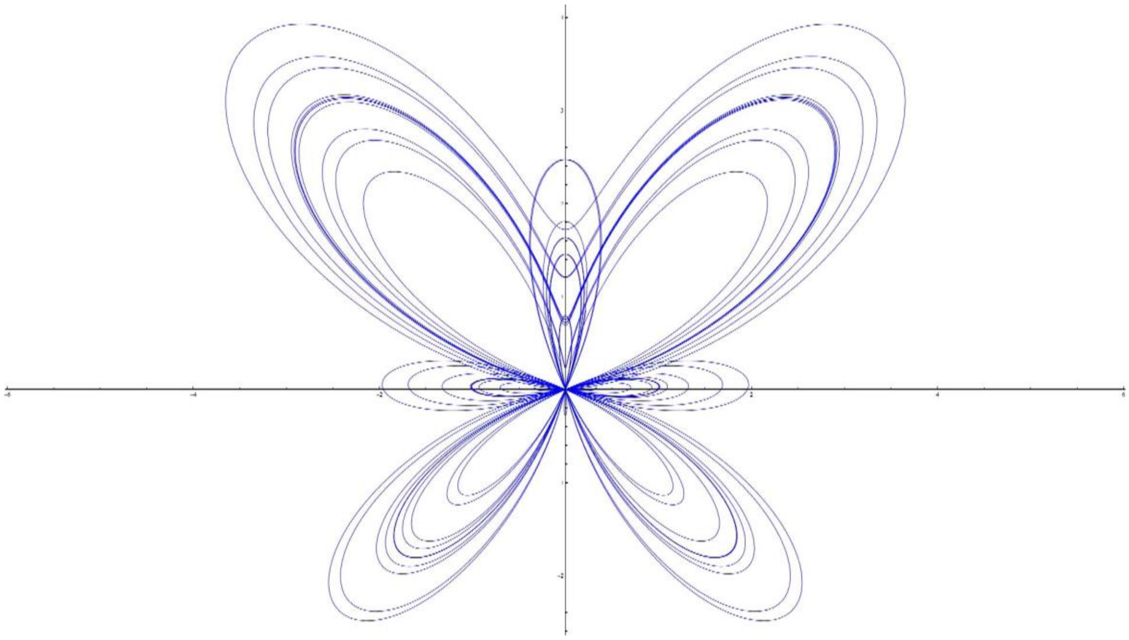
Student Name : Hilal

Student Surname : EKİNCİ

Student ID : 150114057

Exercise 1:

The butterfly curve is given by the parametric equations below. Generate values of x and y for values of t from 0 to 100 with delta t = 1/16. Construct 2D plots of (a) x and y versus t and (b) y versus x. Use subplot to stack these 2 plots vertically and make the plot in (b) square. Include titles and axis labels on both plots and a legend for (a). For (a), employ a dotted line for y in order to distinguish it from x.



(Butterfly Curve)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sin(t) \cdot \left(\left(e^{\cos(t)} - 2\cos(4t) - \left(\sin\left(\frac{t}{12}\right)^5 \right) \right) \right) \\ \cos(t) \cdot \left(e^{\cos(t)} - 2\cos(4t) - \left(\sin\left(\frac{t}{12}\right)^5 \right) \right) \end{bmatrix}$$

(Equations for Butterfly Curve)

Implementation Exercise 1 in MATLAB:

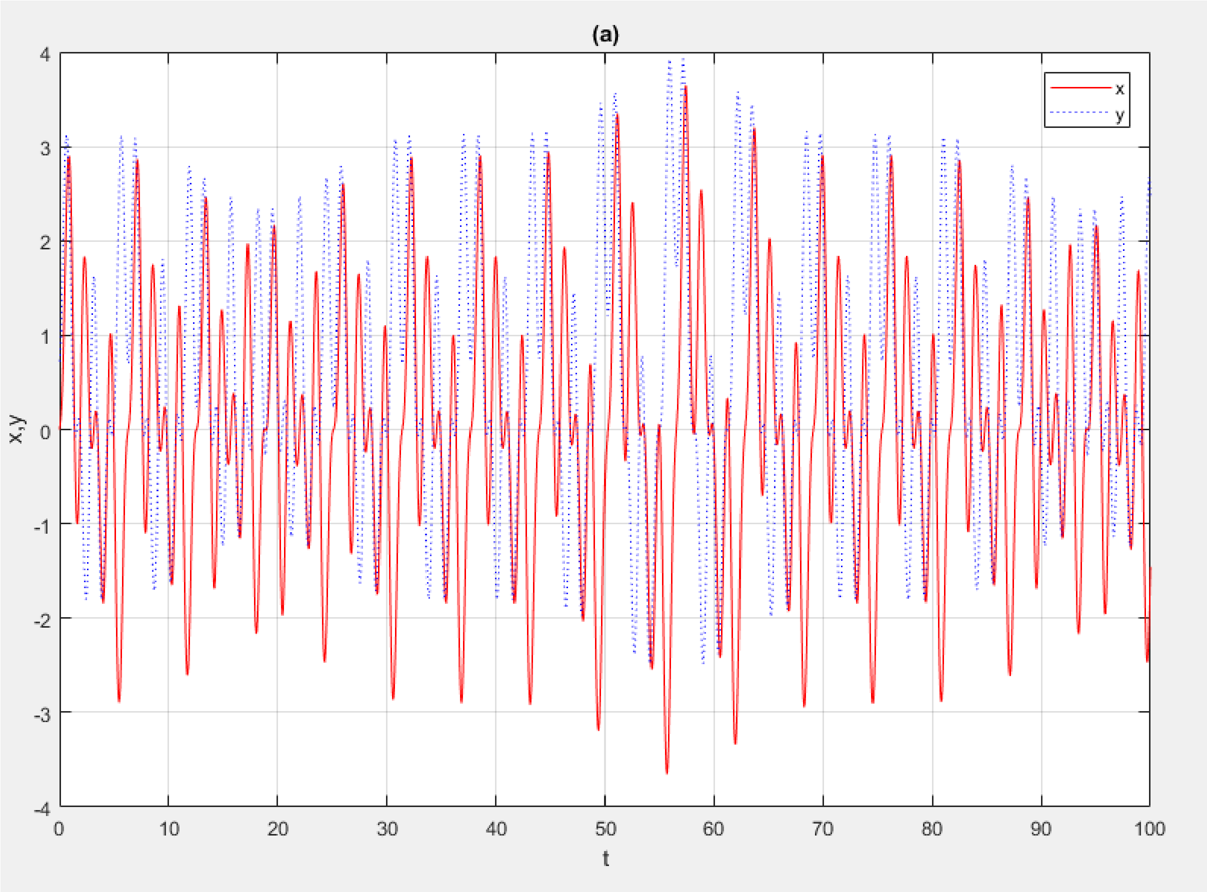
```
%Exercise 1
clc;
clear all;
t=0:1/16:100;
x=sin(t).*(exp(cos(t))-2*cos(4.*t)-sin(t./12).^5);
y=cos(t).*(exp(cos(t))-2*cos(4.*t)-sin(t./12).^5);
subplot(1,1,1);
%We need figures
figure(1)
plot(t,x,'color','r'); hold on;
plot(t,y,':b'); % for dotted line, used :
xlabel('t') % x-axis label
ylabel('x,y') % y-axis label
legend('x','y') % legend
title('(a)')
grid
figure(2)
plot(x,y)
title('(b)')
grid
```

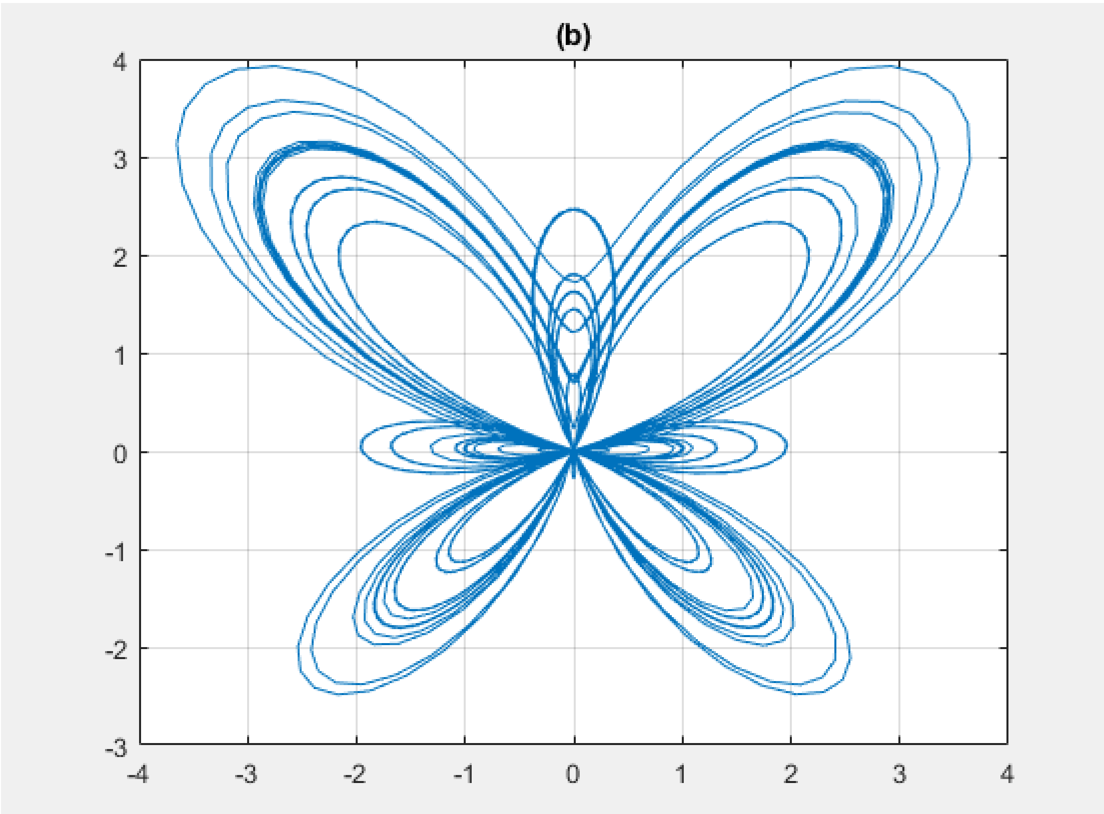
Firstly, I implemented t from 0 to 100 with delta t = 1/16 by writing t= 0: 1/16: 100 (startValue: incrementValue: endValue), and x and y equations according to t.

x=sin(t).*(exp(cos(t))-2*cos(4.*t)-sin(t./12).^5);
y=cos(t).*(exp(cos(t))-2*cos(4.*t)-sin(t./12).^5);

I used subplot function with parameters (1,1,1) to see the graphs big and in more detailed and I used figure functions to see graph a and b separately.

Output graphs for exercise 1:

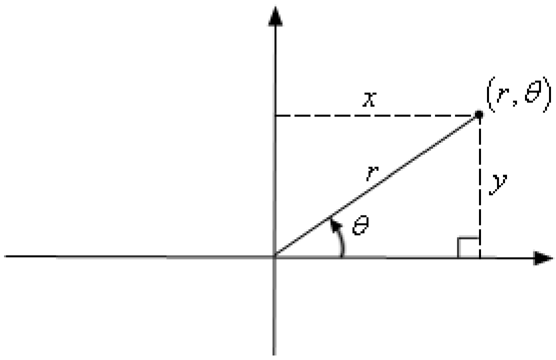




Exercise 2:

The butterfly curve given above can also be represented in polar coordinates as given below. Generate values of r for values of from 0 to with Use the MATLAB function polar to generate the polar plot of the butterfly curve with a dashed red line. Employ the MATLAB Help to understand how to generate the plot.

What is polar coordinate?



(Coordinates in this form are called **polar coordinates**.)

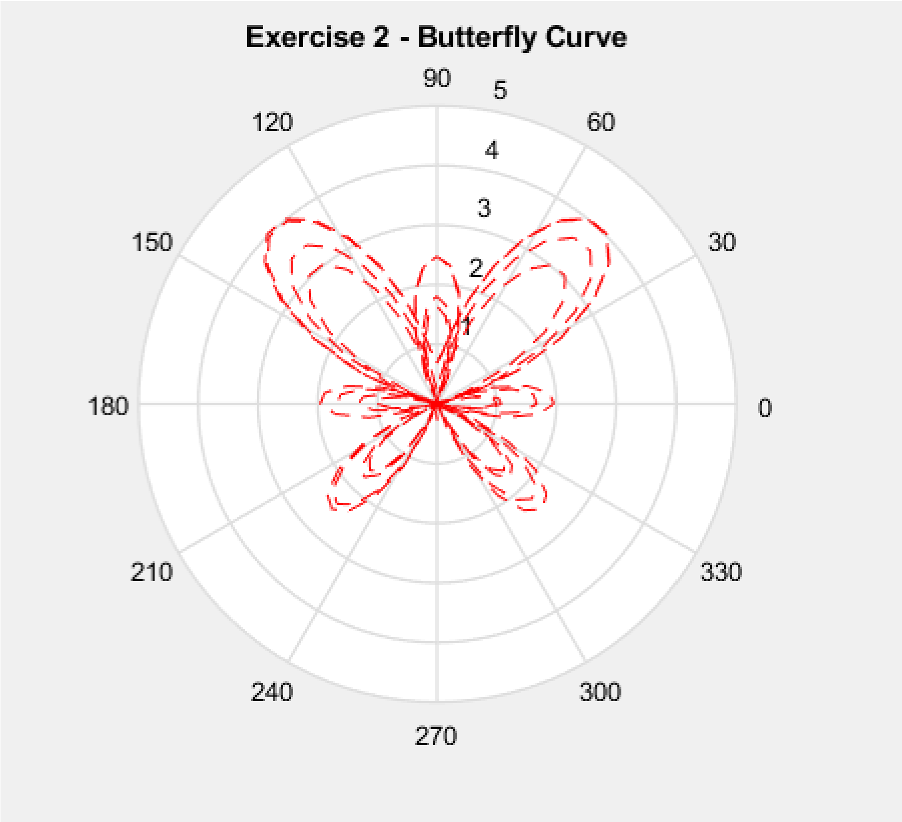
In mathematics, the polar coordinate system is a two-dimensional coordinate system in which each point on a plane is determined by a distance from a reference point and an angle from a reference direction.

The reference point (analogous to the origin of a Cartesian coordinate system) is called the *pole*, and the ray from the pole in the reference direction is the *polar axis*. The distance from the pole is called the *radial coordinate* or *radius*, and the angle is called the *angular coordinate*, *polar angle*, or *azimuth.*

Implementation Exercise 2 in MATLAB:

```
 %Exercise 2
theta = 0:pi/32:8*pi; %theta from 0 to 8pi with delta theta = pi/32
r = exp(sin(theta))-2*cos(4.*theta)- sin((2.*theta - pi)/24).^5 ;
subplot(1,1,1);
figure
polar(theta,r,'--r') % to represent in polar coordinates
title('Exercise 2 - Butterfly Curve')

```

Output graph of Exercise 2:



Exercise 3:

The sine function can be evaluated by the following infinite series:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots$$

Create an M-file to implement this formula so that it computes and displays the values of sin *x* as each term in the series is added. In other words, compute and display in sequence the values for

$$\sin x = x$$

$$\sin x = x - \frac{x^3}{3!}$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!}$$

$$\vdots$$

up to the order term of your choosing. For each of the preceding, compute and display the percent relative error as

$$\%\text{error} = \frac{\text{true} - \text{series approximation}}{\text{true}} \times 100\%$$

As a test case, employ the program to compute sin(0.9) for up to and including eight terms—that is, up to the term *x*15/15!.

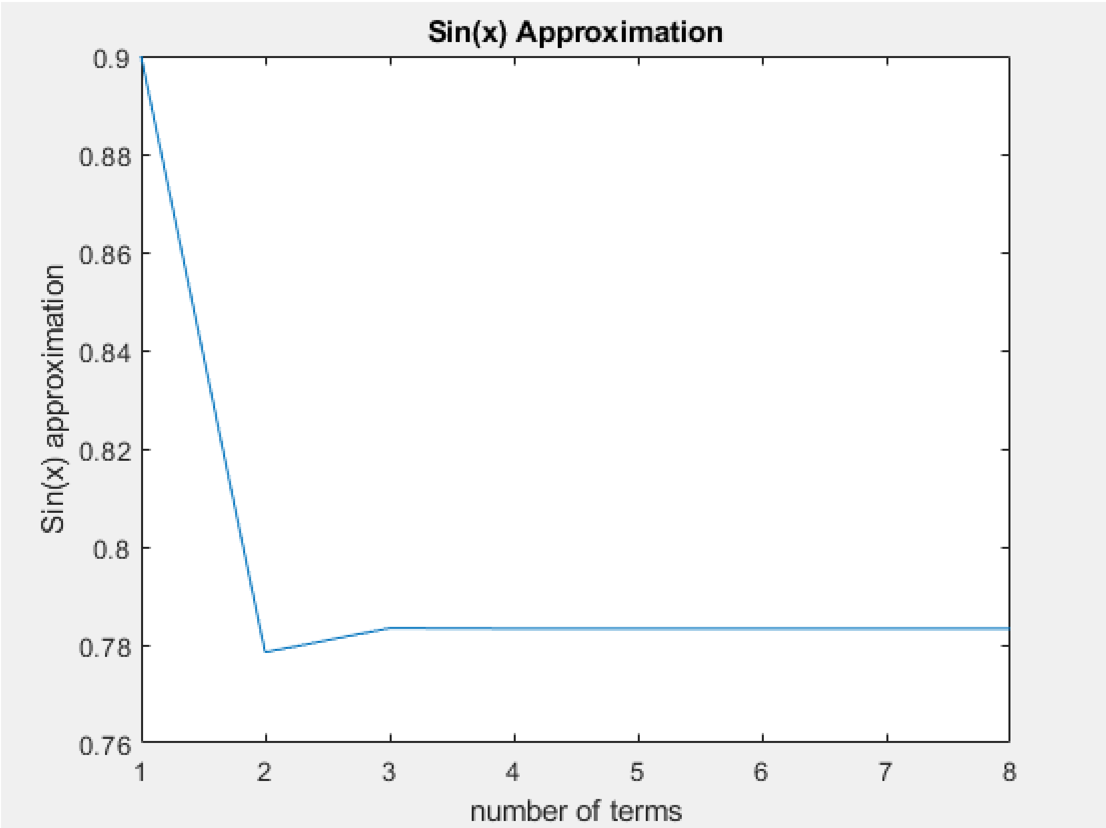
Implementation Exercise 3 in MATLAB:

```
x = 0.9; %Test Value for sinx approximation
n = 15; %Up to 15th term
FindPercentErrorSinx(x,n) %Calling function

function FindPercentErrorSinx(x,n)%Function that takes 2 parameters
                                  % x if or degree, n is for term up to
terms=1:2:n;%For sinx approximation terms will be 1,3,5,7,9,..,n
facts=factorial(terms); %Find factorial of term
x_power=x.^terms;        %Find power of terms
signs=(-1).^((terms-1)/2); %We need terms +,-,+,-,+,.. in order
quotes=x_power ./facts;
series=signs.*quotes;
sinxApproximation=cumsum(series);
display(sinxApproximation);
figure(1)
plot (sinxApproximation);
title('Sin(x) Approximation');
xlabel('number of terms'); ylabel('Sin(x) approximation');
true=sin(x); %Get true value of sin(x) and calculate error
percent_realative_error=abs((true-sinxApproximation)/true*100);
display(percent_realative_error)
figure(2)
plot(1:1:8,(abs(percent_realative_error)));
title('Percent Relative Error for sin(x)');
xlabel('number of terms'); ylabel('percent relatidve error');
end
```

Output of sinxApproximation term by term:

```
sinxApproximation =

    0.9000    0.7785    0.7834    0.7833    0.7833    0.7833    0.7833    0.7833
```

To understand question and to be sure whether percent relative error (that we found) is correct or not, I wanted to find and plot sin(x) approximation. If we calculate sin(x) where x=0.9 as a test value, we find sin(0.9) equals nearly 0.78333. After plotting sin(x) approximation, we can easily see that sin(0.9) approximates 0.78333 when look at the figure below.



Output of percent relative error term by term:

```
percent_realative_error =

   14.8946    0.6162    0.0120    0.0001    0.0000    0.0000    0.0000    0.0000
```

To see whether solutions of percent relative error is correct or not, calculating percent relative error for 1st term will be helpful to be sure.

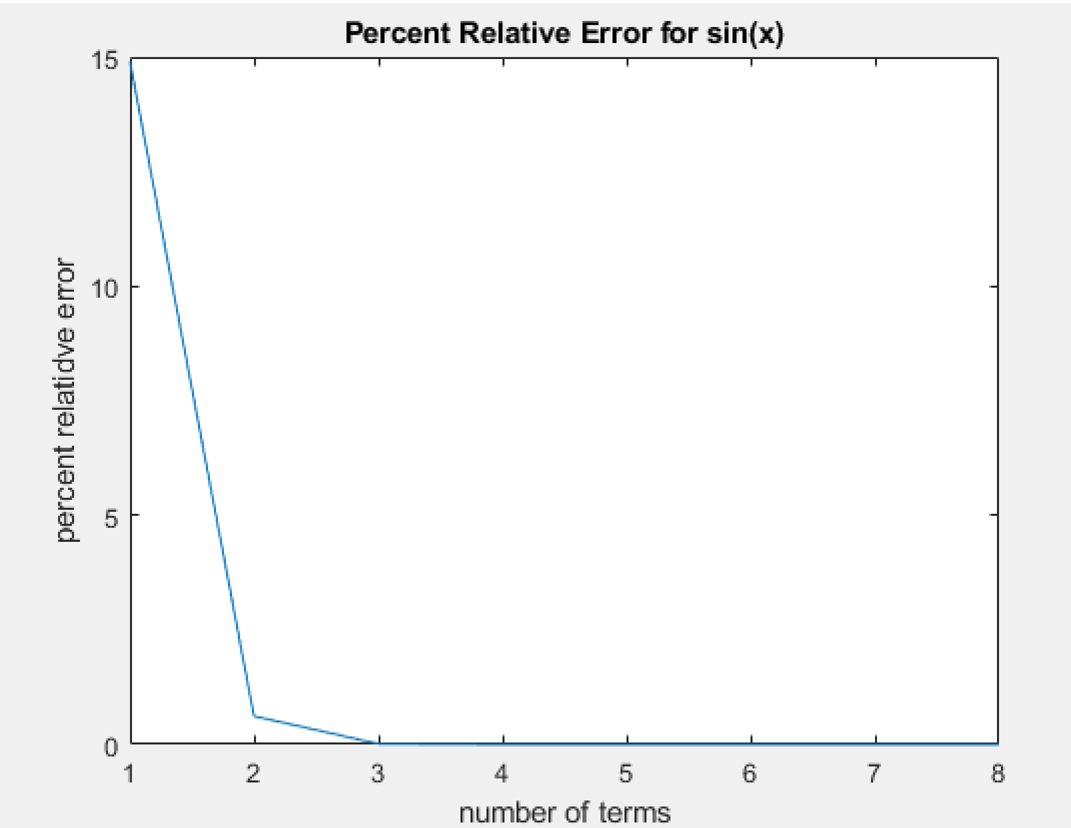
Percent relative error = ((true value – series approximation)/(true value))*100%

Percent relative error = (sin(0.9) – 1st term in series)/sin(0.9) * 100 %

1st term in sin(x) series is = x which is 0.9.

Percent relative error = (0.78333 – 0.9) / 0.78333 * 100 %

Percent relative error = 14.8941059 %



As expected, percent relative error approximates 0(zero).

Exercise 4:

Develop a function M-file that returns the difference between the passed function's maximum and minimum value given a range of the independent variable. In addition, have the function generate a plot of the function for the range. Test it for the following cases:

(a) $f(t) = 8e^{-0.25t}\sin(t-2)$ from $t = 0$ to 6π.
(b) $f(x) = e^{4x}\sin(1/x)$ from $x = 0.01$ to 2.
(c) The built-in humps function from $x = 0$ to 2.

Implementation Exercise 4 in MATLAB:

```
[difference] = FindDifference(func,x);%Calling function
function [difference]=FindDifference(f,t)%Parameter func is for function,
   plot(t, f);                                %Parameter t is for range
   ylabel('function');
   xlabel('range');
   grid on;
   difference = abs(max(f)- min(f)); %Find min and max values and
                                %Calculate difference using abs function
   sprintf('The min value = %f\nThe max value = %f', min(f), max(f))
   sprintf('\nThe difference value = %f ',difference)
end
```

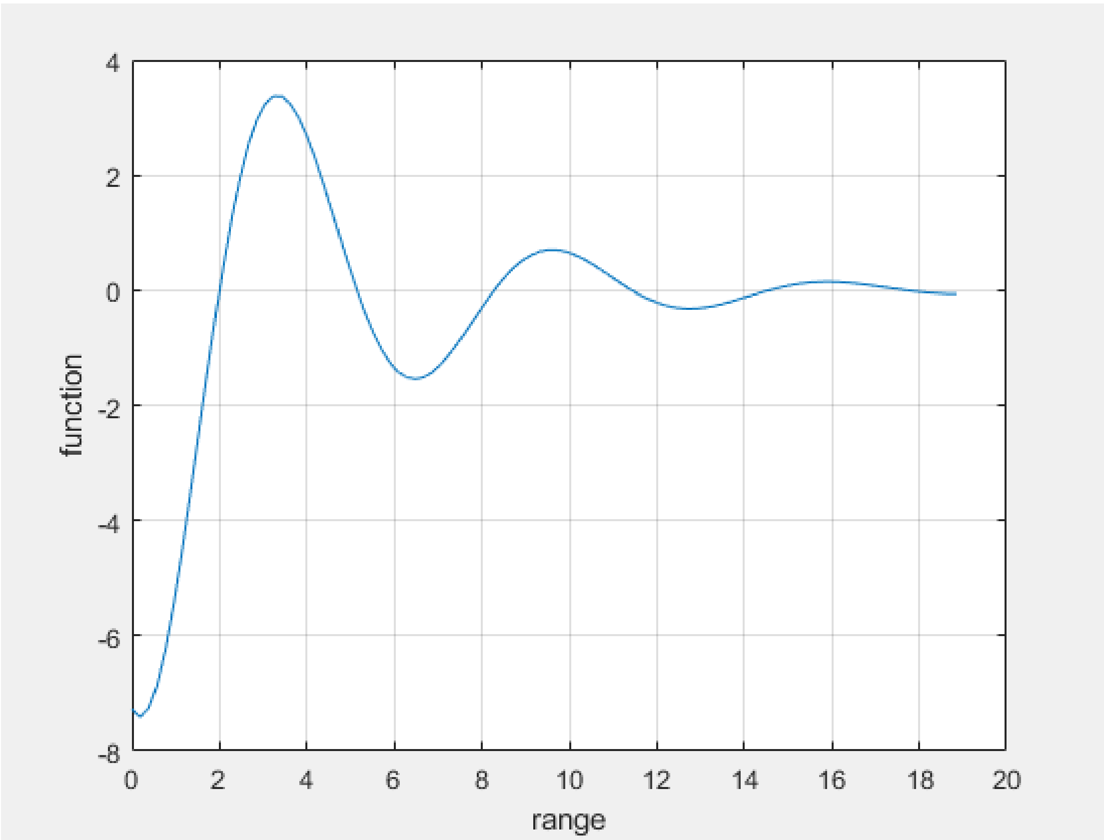
Function above takes two parameters which func and t, func refers to function that we pass, t refers to range of the function that we pass. Function finds minimum and maximum values of passes function and calculates difference and prints.

Output and Plot of Test Value 1: f(t)

```

%Test Value 1
t=linspace(0, 6*pi);
func = 8 * exp(-0.25*t).*sin(t - 2);

```

Command Window

```
ans =

    'The min value = -7.411647
     The max value = 3.364889'


ans =

    '
     The difference value = 10.776537 '

fx >>
```

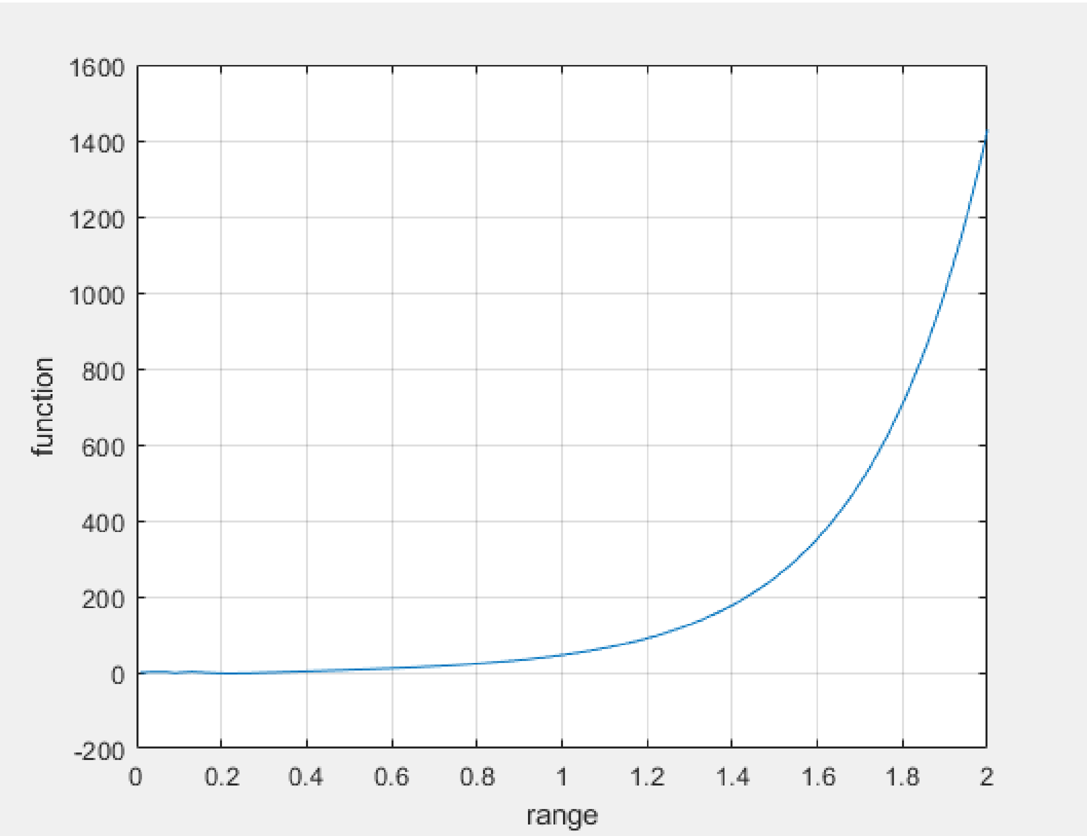


Output and Plot of Test Value 2: f(x)

```
%Test Value 2
x=linspace(0.01, 2);
func = exp(4*x).*sin(1./x);
```

Command Window

```
ans =

    'The min value = -2.335524
     The max value = 1429.147388'


ans =

    '
     The difference value = 1431.482913 '

fx >>
```

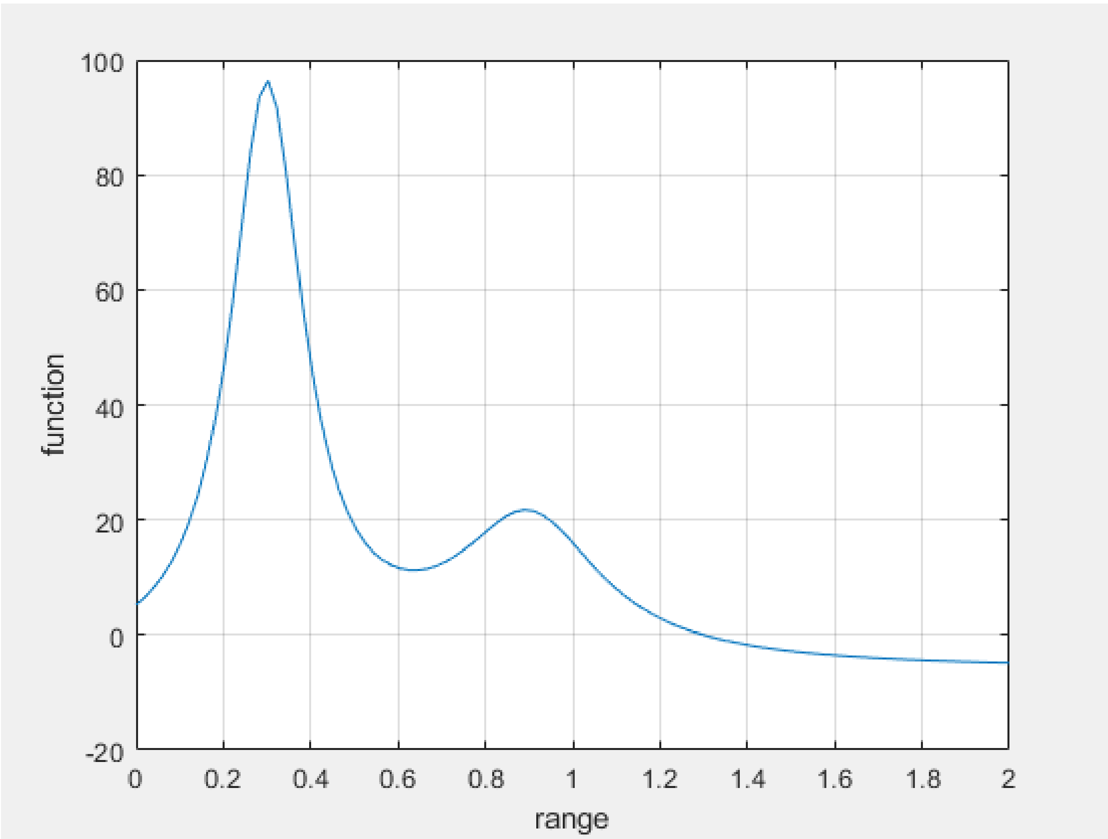


Output and Plot of Test Value 3: humps(x)

```
%Test Value 3
x=linspace(0, 2);
func =humps(x);
```

Command Window

```
ans =

    'The min value = -4.855172
     The max value = 96.431134'


ans =

    '
     The difference value = 101.286307 '

>>
```



REFERENCES

http://mayankagr.in/images/matlab_tutorial.pdf

http://www.wikizero.com/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvUG9sYXJfY29vcmRpbmF0ZV9zeXN0ZW0

http://tutorial.math.lamar.edu/Classes/CalcII/PolarCoordinates.aspx

https://www.mathworks.com/help/matlab/ref/subplot.html

https://www.mathworks.com/help/matlab/ref/plot.html?searchHighlight=plot&s_tid=doc_srchtitle

https://www.mathworks.com/help/matlab/ref/polar.html?searchHighlight=polar&s_tid=doc_srchtitle