LAB # 1: To Identify the Continuous and Discrete Time Signals Using MATLAB

Objectives

After completing this lab, the student will be able to:

- ✓ Describe the basic operations and commands in MATLAB.
- ✓ Describe the steps involved in plotting the Continuous and Discrete time signals in MATLAB.

Pre Lab

Part I –Introduction to MATLAB

What is MATLAB?

MATLAB is a high-level technical computing language equipped with a user-friendly interface. Its name stems from the words ***MATrix*** and ***LABoratory*** as it is based on the use of matrices. MATLAB is an extremely powerful tool useful for scientists and engineers from various disciplines. For example, MATLAB can be used in a wide range of applications, such as telecommunications, signal and image processing, control, mathematics, financial modeling, bioengineering, aeronautics, and many more.

M-Files

In order to write many commands that are executed all together, the program must be written in a text editor. In this editor, one can type all the needed commands to form a program, save the program, and execute it any time he or she wants. The text files are called M-files due to their suffix *.m.

There are two categories of M-files: the Scripts and the Functions.

Scripts

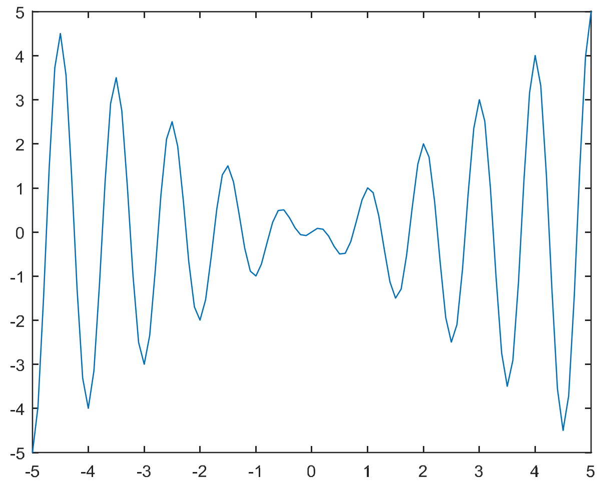
Scripts are the M-files with MATLAB commands. Their name must have a .m suffix. Scripts are suitable for solving the problems that require many commands. The advantage of the scripts is that they are implemented very easily.

Lab Tasks

Lab Task

Write a script file and execute.

```
% Program to  understand the use of script file
% f(t) = tcos(2πt), -5 ≤ t ≤ 5

t=-5:0.1:5;
f=t.*cos(2*pi*t);
plot(t,f)
```



Functions

Functions are also M-files, That is, are files with extension *.m* and must be saved in the *current Directory* of MATLAB. The difference between functions and scripts is that a function accepts one or more input arguments and returns one or more output arguments. To declare that an M-file is a function the first line of the m file must contain the syntax definition. More specifically, the first line of the M-file must be of the form *function [y1,y2,y3,…yn] = name(x1,x2,x3… xm).* The variable *y1,y2,…yn* are the outputs of the function while *x1,x2,…xm* are the input arguments. In case there is only one output, the square brackets are not necessary. The "*name*" specifies the name of the function. In order to execute a function, first the M-File is saved in *Current Directory*.

Lab Task

Write a function file and execute. Function should accepts as input two matrices and returns their sum and product.

```
% Program to  understand the use of a function file
% This function computes the sum and the product of 2 matrices

function [sm,pro]=oper(A,B)
sm=A+B;
pro=A*B;
end
```

```
% to test the function operation
% Enter the matric A
A= [ 2 3; 4 5]
% Enter Matrix B
B= [4 5; 5 6]
```

Useful Commands

Here we will learn and practice useful (when working with vectors and matrices) commands. As already discussed, the command sum returns the sum of the elements of a vector. The command cumsum returns a vector whose elements are the cumulative sum of the previous elements, the command prod is the product of the vector elements, while the command diff returns a vector in which each element is given by its subtraction with the previous element. The command max and min return the largest and smallest elements of the vector, respectively, as well as their index. The command sort sorts the vector elements in ascending (by default) or descending order. The command mean computes the mean value, while the command median returns the median value. All these commands are suitable also for matrices by slightly changing their syntax.

Commands	***Results/Comments***
`a = [4  2  7  0  6]`	`a = 4  2  7  0  6` %Definition of vector *a*
`s = sum(a)`	`s = 19` %Sum the elements of *a*c
`c = cumsum(a)`	`= 4  6  13  13  19` %Cumulative sum. The result is obtained as [4,4+2,4+2+7, 4+2+7+0, 4+2+7+0+6]
`p =  prod(a)`	`p = 0` %Product of all elements.
`d = diff(a)`	`d = -2  5  -7  6` % Difference between two consecutive elements i.e., d(1)=a(2)-a(1) ,etc.

`[m,i] = max(a)`	`m = 7`
	`i = 4`
	%The largest value is assigned to the variable *m*, and its index is assigned to variable *i*.

`[m,i] = min(a)`	`m = 0`
	`i = 3`
	%The smallest value is assigned to the variable *m*, and its index is assigned to variable *i*.

`max(a)`	`ans = 7` %If no output variable is specified, only the largest value is returned.
`mean(a)`	`ans = 3.8000` %Mean value of the elements.
`median(a)`	`ans = 4` %Median value of the vector.
`sort(a)`	`ans = 0  2  4  6  7` %Sorting in ascending order.
`sort(a,'descend')`	`ans = 7  6  4  2  0` %Sorting in descending order.

Special Forms of Matrices

The command ones(M,N) creates a matrix of size MxN with ones. Typing ones(M) returns a square matrix with ones. Similarly, the command zeros(M,N) creates a matrix of size MxN with zeros. The command rand(M,N) returns an MxN matrix with random elements. The command eye(M,N) defines an MxN matrix with ones in the main diagonal zeros everywhere. The command magic(M) creates a square matrix with the property that the sum of each row or column is equal.

Commands	Results/Comments

`ones(2,3)`	`ans = 1     1     1`
	`      1     1     1`
	%Matrix of size 2x3 with ones.

`zeros(1,4)`	`ans = 0   0   0  0` %Matrix of size 1x4(or vector of length) with zeros.

`rand(3)`	`ans = 0.8600      0.9000      0.4600`
	`      0.6000      0.1000      -0.3000`
	`      0.4540      0.6023      0.2700`
	%Matrix of size 3x3 with random elements. If there is one input argument to the command, the obtained matrix is square.

`eye(4,2)`	`ans = 1    0`
	`      0    1`
	`      0    0`
	`      0    0`
	%Magic of size 4x2 with ones in the main diagonal and zeros elsewhere.

`eye(3)`	`Ans = 1    0     0`
	`      0    1     0`
	`      0    0     1`
	%The identity matrix 1 of size 3x3.

`A = magic(3)`	`A = 8    1    6`
	`    3    5    7`
	`    4    9    2`
	%Magic matrix

Symbolic Variables

In MATLAB, a variable type is the symbolic variable (or object). A symbolic variable is defined by the command sym and syms. The use of symbolic variables allows the computation of limits, integrals, derivativesetc.

Part II- Plotting Signals in MATLAB

MATLAB is a very reliable and power full tool for plotting. A graph is constructed as a set of points in two or three dimensions. These points are typically connected with a solid line. Commonly a graph of a function is required. However in MATLAB a plot is done using the vectors and matrices not functions.

Plotting in Two Dimensions

Suppose that we want to plot a function $y(x)$, where x is an independent variable. The procedure to plot $y(x)$ is as follows:

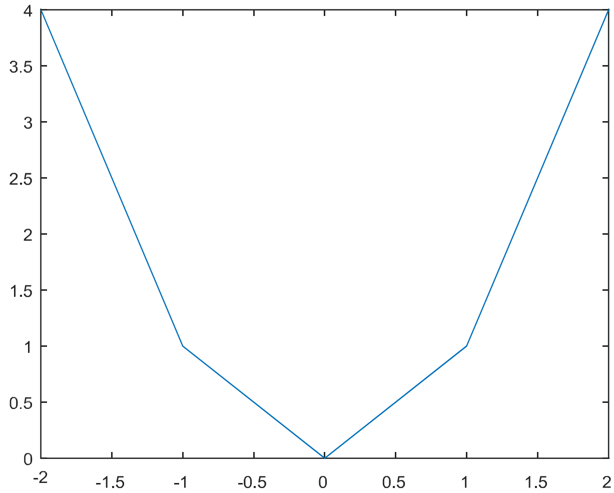
1. Vector x is created. Such as $a \le x \le b$, where a and b are scalars.
2. The function $y(x)$ will be plotted over the interval $[a, b]$.
3. Create the vector y , which is of the same length as x. The two vectors have an equal number of elements.
4. The value of each element of y is the value of $y(x)$ calculated for each value of$x$.
5. Finally, the function $y(x)$ is plotted by typing 'plot(x,y)'

Lab Task

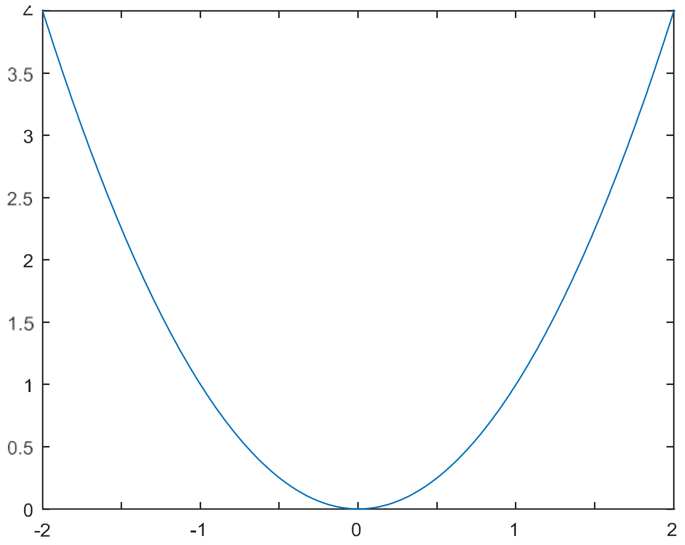
Plot the function $y(x) = x^2, -2 \leq x \leq 2$

```
x=-2:2 % independent variable ,length of the plot
length (x)
y=x.^2 % Function
length (y)
plot(x,y)
```

```
x = -2   -1   0   1   2

ans = 5


y= 4   1   0   1   4

ans = 5
```



Write down modified MATLAB code to plot exactly like below



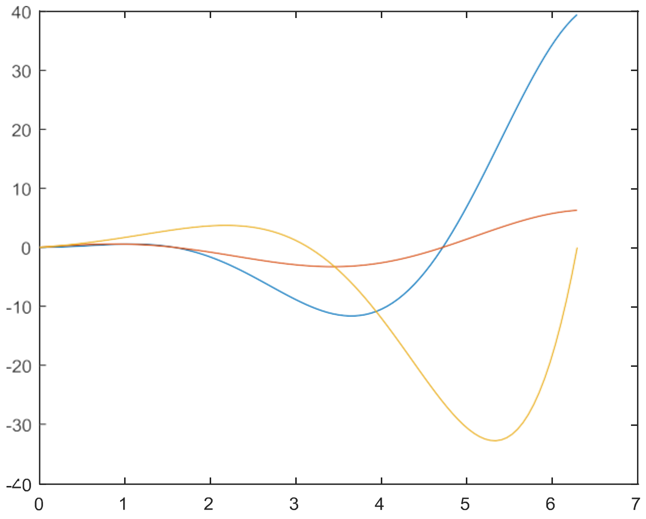
Plotting Several Function in One Figure

It is possible to plot more than one function in the name of figure by employing a different syntax of command Plot.

Lab Task

Plot the $y(x) = x^2\cos(x)\,, g(x) = x\cos(x), and\ f(x) = 2^x\sin(x)\,, \quad 0 \leq x \leq 2\pi$ in the same figure.

```
x = linspace(0,2*pi,100) % linspace could be used to create a vector.
x=0:pi/50:2*pi     % this is same value as above. Both method are correct
y = (x.^2).*cos(x);
g = x.*cos(x);
f = (2.^x).*sin(x);
plot(x,y,x,g,x,f)
```



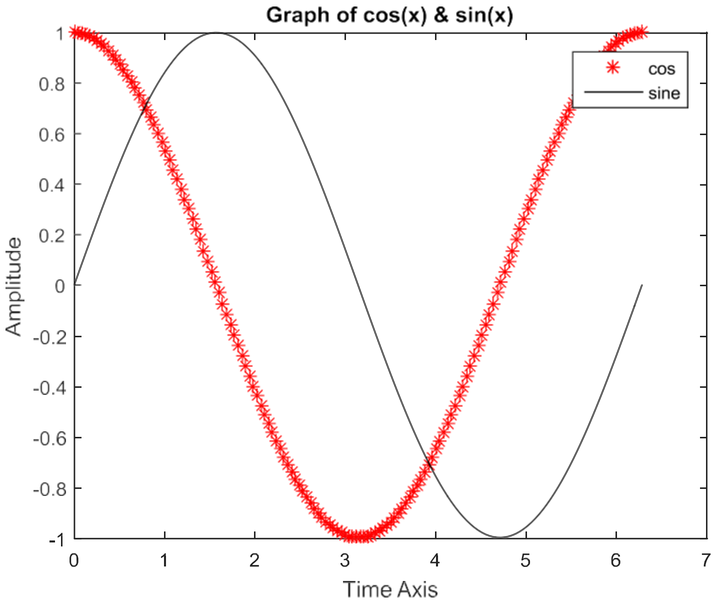
In the previous examples, the functions were plotted with predefined colors and line type (solid). It is possible to create a graph using colors, symbols used to draw the points, and type of line that connects the points of your choice. This is achieved by applying one more input argument to the command plot. The new argument is a series of special character given in single quotes. The available special characters are presented in the table below.

Symbols	*Color*	*Symbol*	*Point Type*	*Symbol*	*Line Type*
B	Blue	.	Point	-	Solid
G	Green	o	Circle	:	Dotted
R	Red	x	x-mark	-.	Dashdot
C	Cyan	+	Plus	_	Dashed
M	Magenta	*	Star		
Y	Yellow	s	Square		
K	Black	d	Diamond		
W	White	<,>	Triangle		
		P	Pentagram		
		h	Hexagram		

Formatting a Figure

The command grid on adds lines to the graph, while the command grid off removes the grid lines. Simply typing grid is switch between the two modes. Text besides the *x-axis* and *y-axis* can be added using the commands xlabel and ylabel, respectively. A graph title is inserted by the Command title.

```
% Program to  understand formatting a plot: axis labeling, title and legend %

x = linspace(0,2*pi,150);
plot(x,cos(x),'r*',x,sin(x),'k')
xlabel('Time Axis')
ylabel('Amplitude')
title('Graph of cos(x)  & sin(x)')
legend('cos','sine')
```



Plotting in Different Figures

Up to this point, all plots were made in single figure named *Figure 1*. By typing at the command prompt figure, a new figure with name *Figure 2* appears without closing the old figures. The command subplot(m,n,p) or subplot(mnp) splits the figure window into $m \times n$small subfigures and makes the *pth* subfigure active.

Lab Task

Plot the functions that were plotted in last lab task in the following two ways.

1. In two different figures
2. In a figure but separately using subplot.

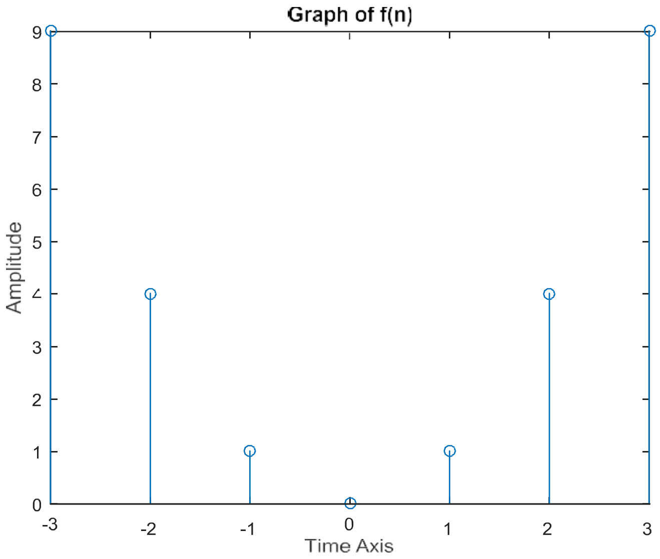
Plotting the Continuous Time and Discrete Time Functions

A discrete time function is a function of the form $f[n], n \in z, where\ z\ denotes\ the\ set\ of\ integers.$ In this case the appropriate command for plotting the function is ‘stem(n,f)’

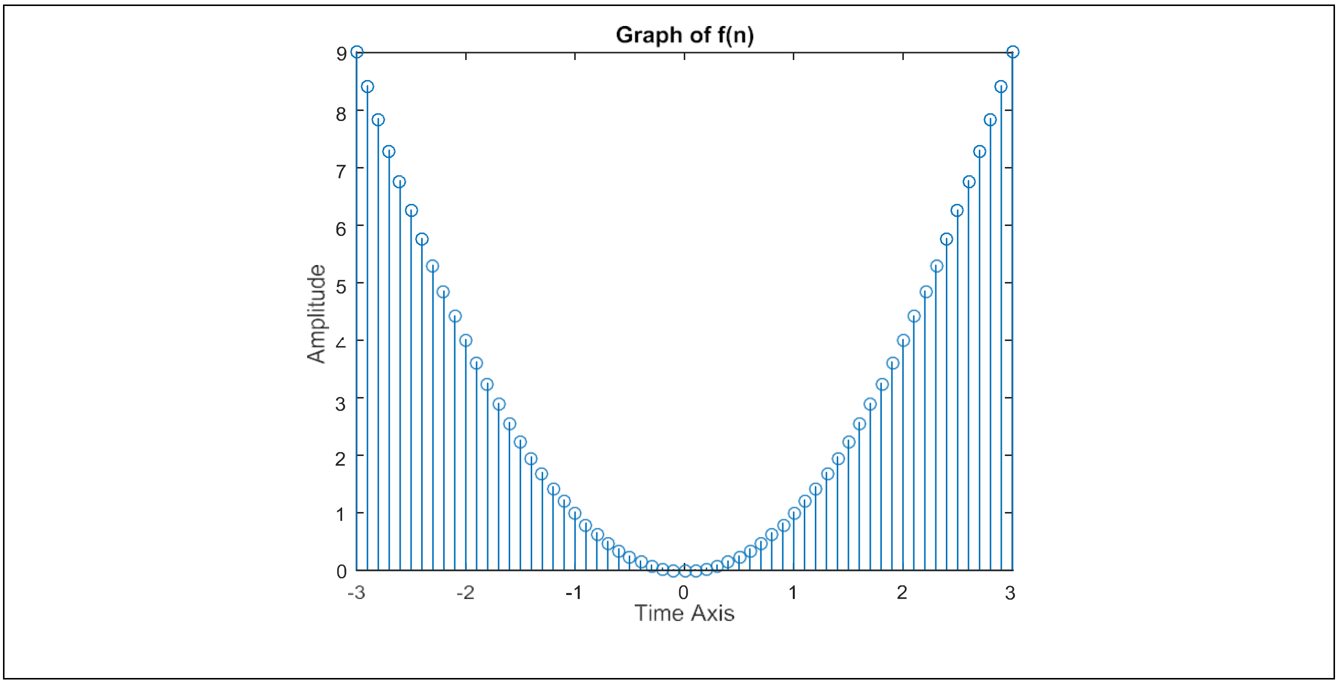
Lab Task

Plot the discrete function$f[n] = n^2, where - 2 \leq n \leq 2.$

```
n = -3:3
f= n.^2
stem(n,f)
xlabel('Time Axis')
ylabel('Amplitude')
title('Graph of f(n)')
```



Modify the above lab task to create figure like below.



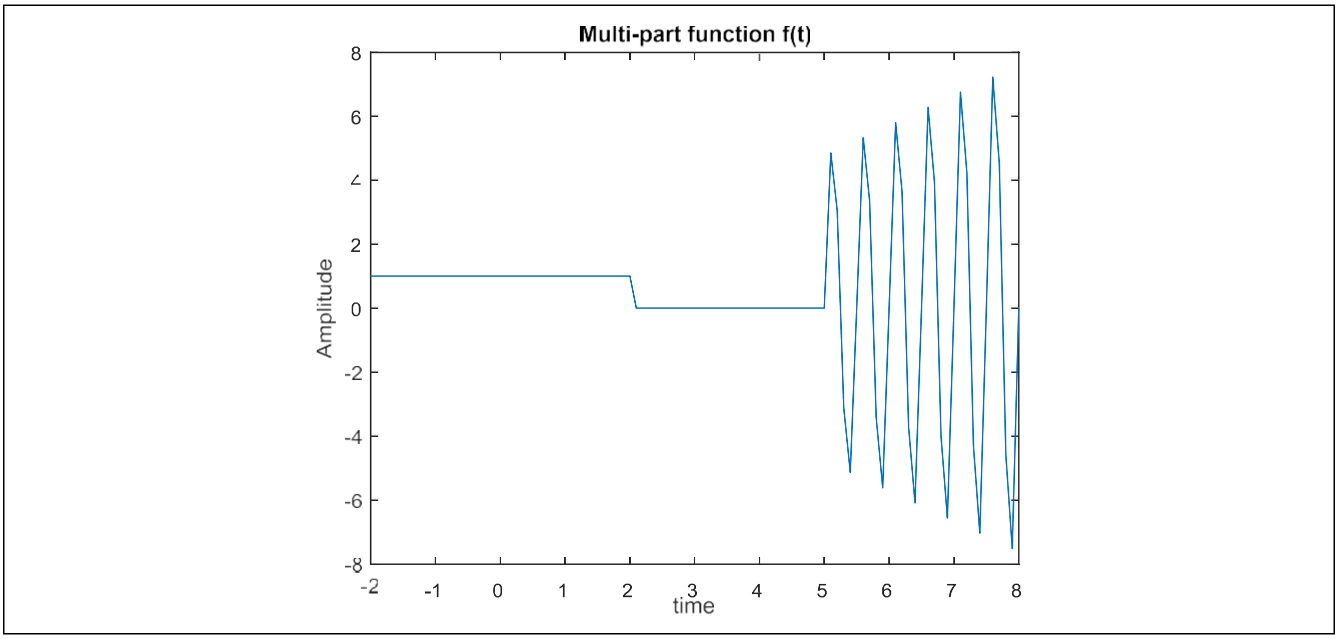
Now we will discuss the way of defining and plotting functions with more than one part.

Lab Task

Plot the following function

$$f(t) = \begin{cases} 1, & -2 \leq t \leq 2 \\ 0 & 2 < t < 5 \\ t\sin(4\pi t) & 5 \leq t \leq 8 \end{cases}$$

```
% Program to  understand piecewise functions plotting

t1=-2:.1:2;
t2=2.1:.1:4.9;
t3=5:.1:8;
f1=ones(size(t1));
f2=zeros(size(t2));
f3=t3.*sin(4*pi*t3);
t=[t1 t2 t3];
f=[f1 f2 f3];
plot(t,f)
title('Multi-part function f(t)')
xlabel( 'time')
ylabel( 'Amplitude')
```



Multi-part function f(t)
Amplitude
time
8
6
4
2
0
-2
-4
-6
-8
-2
-1
0
1
2
3
4
5
6
7
8