Lab Session 01

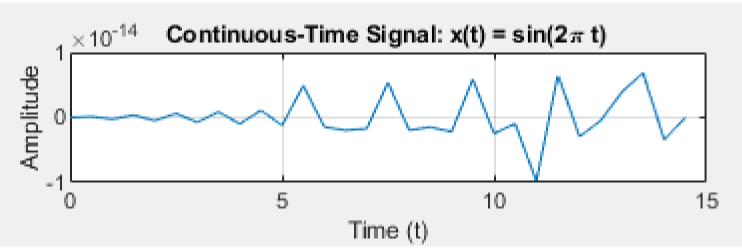
Explore characteristic of various signals using MATLAB

Signals

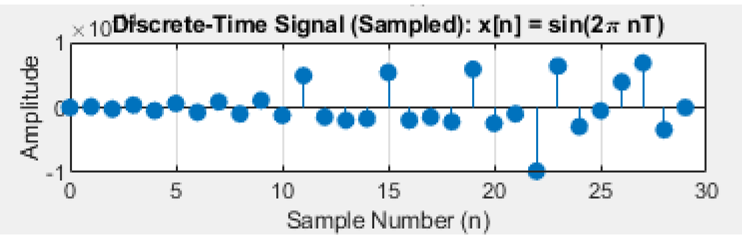
In the fields of communications, signal processing, and in electrical engineering more generally, a signal is any time-varying or spatial-varying quantity. This variable (quantity) changes in time.

- Speech or audio signal: Sound amplitude that varies in time
- Temperature readings at different hours of a day
- Stock price changes over days etc.

Signals can be classified as continues-time signal Fig. 1.1 (a) and discrete-time signal Fig. 1.1 (b). A continuous-time (CT) signal, usually denoted as x(t), is also called as an analog signal. The amplitude of CT signal continuously changes with time. A discrete-time (DT) signal, represented as x(n), is defined only at discrete instants of time. A DT signal is a time series signal that has been sampled from continuous time signal. A digital signal is a discrete-time signal that takes on only a discrete set of values.



(a)



(b)

Fig. 1.1 Signal types – (a) Continuous-time signal, (b) Discrete-time signal

MATLAB

MATrix LABoratory (MATLAB) is a powerful high-level programming language for scientific computations. It supports a rich suite of mathematical, statistical and engineering functions and its functionality is extended with interactive graphical capabilities for creating 2D as well as 3D plots. It provides comprehensive toolboxes and various sets of algorithms.

Default Desktop Environment

a) Command Window

The main window in which commands are keyed in after the command prompt '>>'. Results of the most printing commands are displayed in this window.

b) Command History Window

This window records all of the executed commands as well as the date and time when these commands were executed. This feature comes very handy when recalling previously executed commands. Previously entered commands can also be re-invoked using up arrow key.

c) Current Directory Window

This window keeps track of the files in the current directory.

d) Workspace

This window is used to organize the loaded variables and displays the information such as size and class of these variables.

Signal's Representation

A signal in MATLAB is represented by a vector:
Examples:

```
x = [2, 3, -5, -3, 1]
n = 2:3:17 %( here step size is 3)
n = 2:17   %( here default Step size 1 is used)
```

Plotting in MATLAB

While plotting in MATLAB one must be careful that a vector is plotted against a vector and lengths of vectors must match. Two functions **plot( )** and **stem( )** are used for plotting a CT and DT signals as shown in example below with plots in Fig. 1.2. and Fig. 1.3. respectively. The MATLAB commands **xlabel**, **ylabel**, and **title** are used to caption the x-axis, y-axis and plot itself. The command **figure** is used to display the output in a separate window.

Example:

MATLAB Commands to plot a CT signal:

t = [-2:0.002:2]
x = 10 * sin (pi * t)
figure, plot(t, x)
title('Example CT Sinusoid')
xlabel('time(sec)')
ylabel('Amplitude')

MATLAB Commands to plot a DT signal:

n = -6:3
x = [2, 3, -1, 5, 4, 2, 3, 4, 6, 1]
figure, stem(n, x)
title('Example DT Sinusoid')
xlabel('samples')
ylabel('Amplitude')

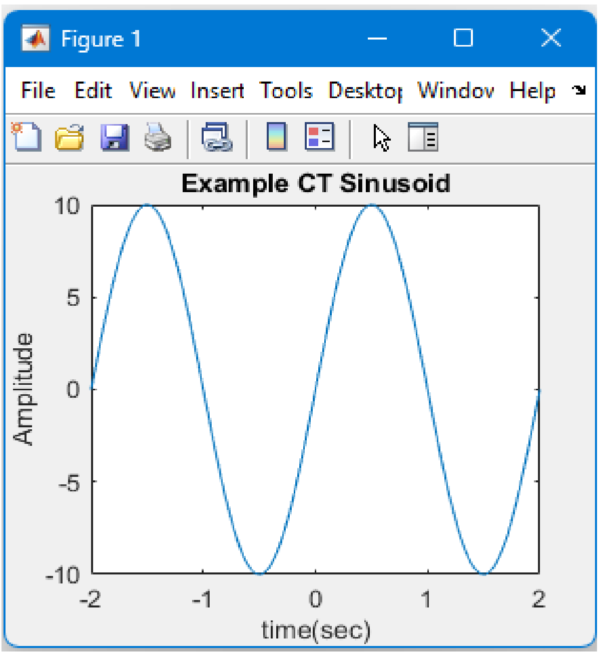


Fig. 1.2. Plotting a CT signal

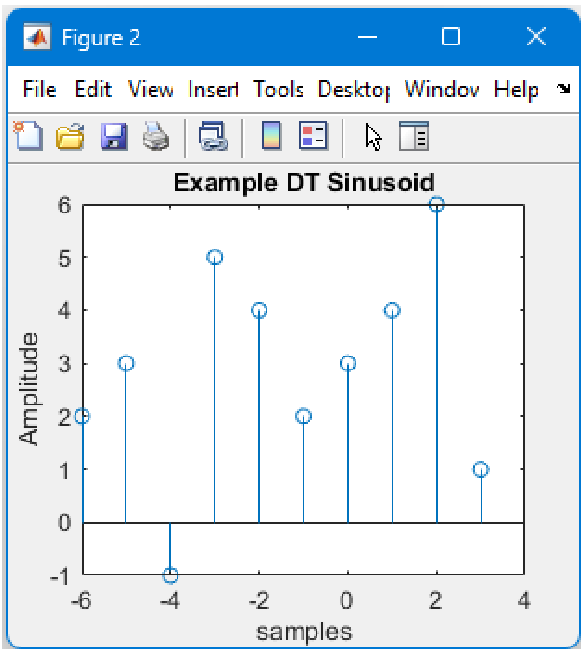


Fig. 1.3. Plotting a DT signal

Multiple Plots

For drawing multiple signals on the same graph, write first signal's x and y axis vectors followed by the next signal. In order to differentiate signals in a multi-plot by colors or line styles, character codes as listed in Table 1.1 are used. The **legend** command can be used to describe legend information of signals.

Table 1.1. Line and color styles in MATLAB

Code	Color name	Example	Code	Line Styles	example
y	yellow	yellow	.	point	………
m	magenta	magenta	o	circle	oooooo
c	cyan	cyan	x	x-mark	xxxxxx
r	red	red	+	plus	++++++
g	green	green	-	solid	———
b	blue	blue	*	start	******
w	white	white	:	dotted	:::::::::::
k	black	black	-.	dashdot	-.-.-.-.-.-
			--	dashed	---------

Example

t = [-2:0.002:2]
x = 3 * sin (pi * t)
y = 3 * cos (pi * t)
plot(t, y, 'r-', t, x, 'b--')
legend('Sine curve', 'Cosine curve')

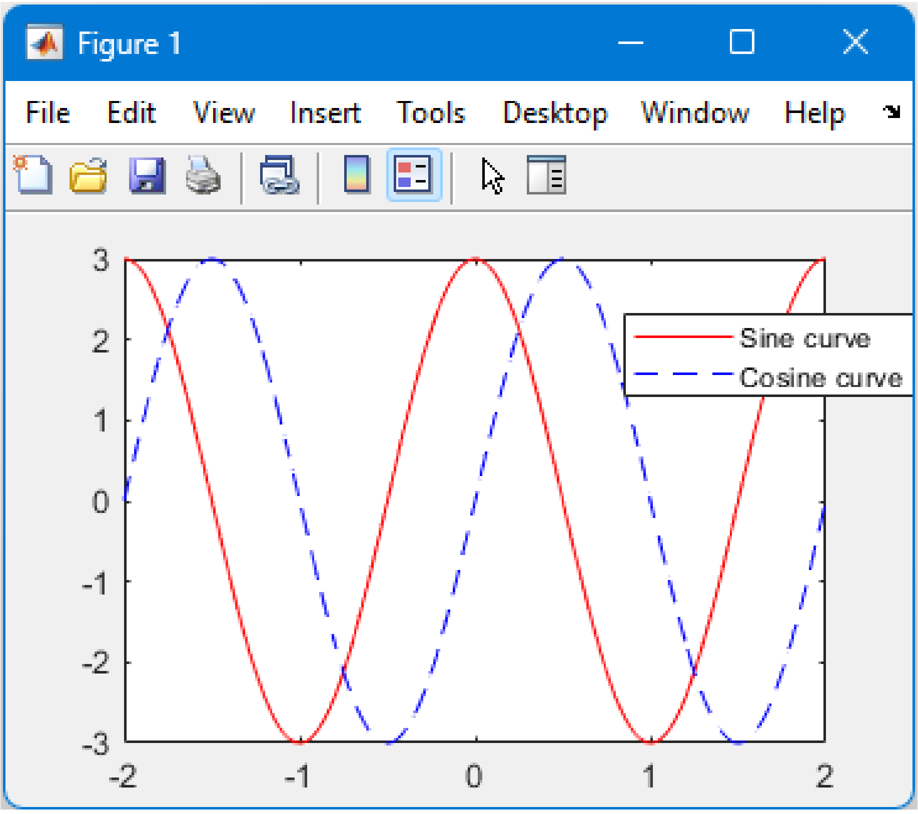


Fig. 1.4. Multiple plot example

Generating Subplots

A **subplot(m, n, p)** command is used to display multiple plots inside a grid of m rows and n columns inside a single window, where each plot is placed in position p inside the grid. If a single-digit m, n, p are needed then the command can be condensed as **subplot(mnp)**. The subplot command is followed by the plot command which necessary describes the plot to be placed at position p in the grid. The example to display multiple plots using subplot command is given below with the output as shown in Fig. 1.5.

Example

```
x=10*sin(-2*pi*t)
y=10*cos(-2*pi*t)
u=10*sin(-5*pi*t)
v=10*cos(-5*pi*t)
t = [-2:0.002:2]
subplot(2, 2, 1), plot(t, x, 'r')
xlabel('t'), ylabel('x')
subplot(222), plot(t, y, 'g')
xlabel('t'), ylabel('y')
subplot(2, 2, 4), plot(t, u, 'b')
xlabel('t'), ylabel('u')
subplot(223), plot(t, v, 'k')
xlabel('t'), ylabel('v')
```

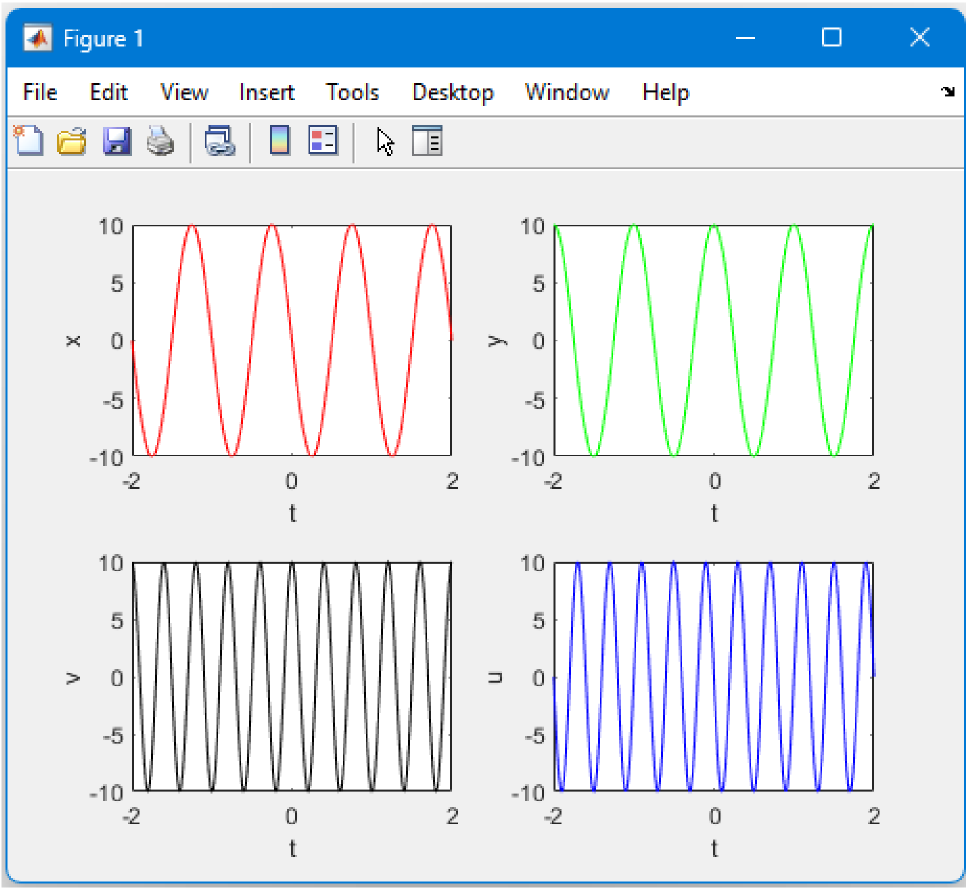


Fig. 1.5. Multiple plots using subplot command

Zero & One Vectors

To generate zero or one vectors, use following statements:
zeros(1, 5)
Output: [0 0 0 0 0]
ones(2, 10)
Output:
[1 1 1 1 1 1 1 1 1 1]
[1 1 1 1 1 1 1 1 1 1]

Some Common Types of Signals

For unit step:
x = (ones(size(n))).*(n>=0)
Note: *size(n)* returns dimension and number of elements in the array

For ramp:
x = n.*(n >= 0)

For exponential:
x = (ones(size(n))).*(n >= 0) %unit step
y = ((0.5).^n).*x
stem(n,y)

For Rectangle:
```
t = -1:0.001:1;
y = rectpuls(t);
plot (t,y);
```

For Triangle:
```
t = -1:0.001:1;
y = tripuls(t);
plot (t,y);
```

For Sawtooth:
```
fs = 10000;
t = 0:1/fs:1.5;
x = sawtooth(2*pi*50*t);
plot(t,x), axis([0 0.2 -1 1]);
```

For Square wave:
```
t = 0:20;
y = square(t);
plot(t,y)
```

For Sinc function:
```
t = -5:0.1:5;
y = sinc(t);
plot(t,y)
```

Exercise

1. Write MATLAB code to plot function x = sin(nπt). Generate 8 subplots using for loop. Use step size of 0.05.

2. Write a sequence of MATLAB commands in the space below to plot the curves ***y1 = cosx*** and ***y2 = x*** for $0 \le x \le 2$ on the same figure. Then zoom in to determine the point of intersection of the two curves (and, hence, the root of x = cosx) to two significant figures. Your plot must be properly labeled.

3. Draw graphs of the functions for x = 0:0.1:10 and label your graph properly.
i. y = sin(x)/x

ii. $u = (1/(x-1)^2) + x$

iii. $v = (x^2+1) / (x^2-4)$

iv. $z = ((10-x)^{1/3}-1) / (4 - x^2)^{1/2}$

4. Write MATLAB commands to plot following elementary DT signals. Properly label your graphs.
a) Unit Step

b) Unit Ramp

c) Real Exponential: $x(n) = 2(0.25)^n$, $0 < n < 10$

5. Generate multiple plots with the following data:
Suppose A = 1, f = 1Hz, t = 0:0.01:1: y1(t) = cos(2πt); y2(t) = cos(2πt+π/2); y3(t) = cos(2πt-π/2); y4(t)=sin(2πt) where A is the amplitude of signal. Use colors & line styles to distinguish the plots.

Instructions:

1. Solve the exercise questions on exercise sheet. You may use multiple exercise sheets with proper page numbering at the bottom.
2. Hand-write or print the question first before your answers.
3. Hand-write the MATLAB code. Figures / plots need to be printed and properly pasted but avoid micro sizing the figures.
4. Avoid placing lot of content on one page. However, to avoid the space wastage on the page, you may divide the page in double columns.
5. Keep the page left margin intact, since you need to punch holes and add the page(s) in punch folder.