



COLLEGE OF ENGINEERING AND TECHNOLOGIES
ALMUSTAQBAL UNIVERSITY

Digital Signal Processing (DSP)
CTE 306

Lecture 1

- Matlab Basics Review -

Dr. Zaidoon AL-Shammari

Lecturer / Researcher

zaidoon.waleed@mustaqbal-college.edu.iq

- ✚ Matlab ® is a software tool, originally developed as a “**Matrix Laboratory**” but now used in almost all areas of science and engineering.
- ✚ This software is used by scientists, engineers, teachers and students in both academic institutes and industries dealing with problem solving in **basic mathematics**, **graphics**, **differential equations**, **electric/electronic circuits**, **simulation**, **control systems & automation**, **signal processing**, **image processing**, **statistics**, **aerospace**, **telecom**, **bioinformatics** and **many other applications**.

Mathematical operations

Mathematical operations

Add

$$x + y$$

Subtract

$$x - y$$

Multiply

$$x * y$$

Divide

$$x / y$$

```
Command Window
>> 2+3
ans =
     5
>> 3.5-6.4
ans =
    -2.9000
>> 6*4.24
ans =
    25.4400
>> 5/2.1
ans =
     2.3810
fx >> |
```

Mathematical operations

Power

$$x^y \equiv x^{\wedge}y$$

- Square root

$$\sqrt{x} \equiv \text{sqrt}(x)$$

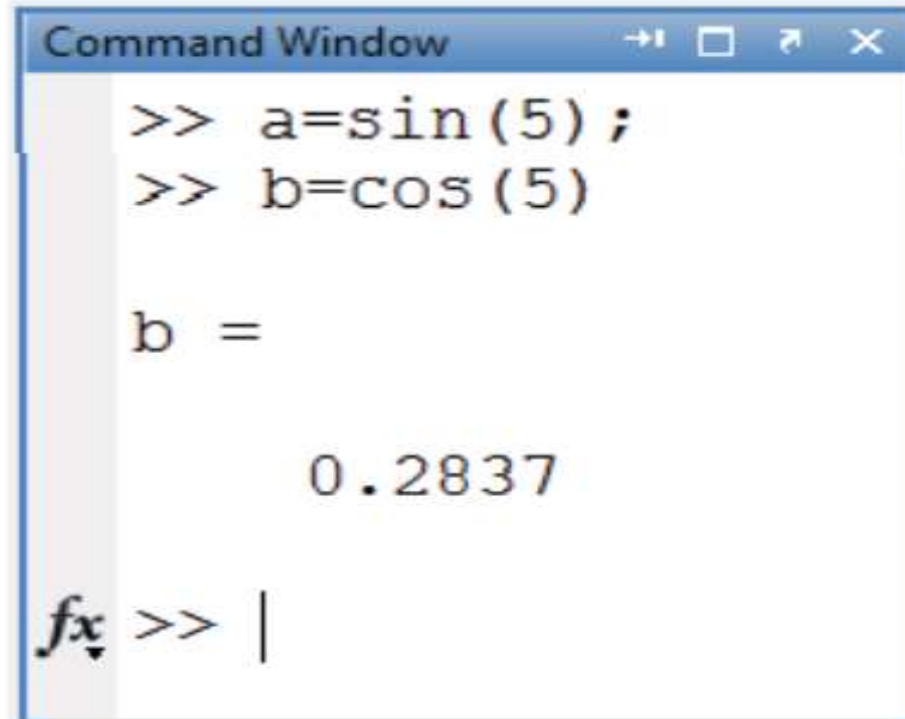
- Exponential

$$e^x \equiv \text{exp}(x)$$

```
Command Window
>> 2^3
ans =
     8
>> sqrt(9)
ans =
     3
>> exp(1)
ans =
    2.7183
fx >> |
```

Making Matlab Quiet

Any line in a script that ends with a semicolon (;) will execute without printing to the screen.



```
Command Window
>> a=sin(5);
>> b=cos(5)

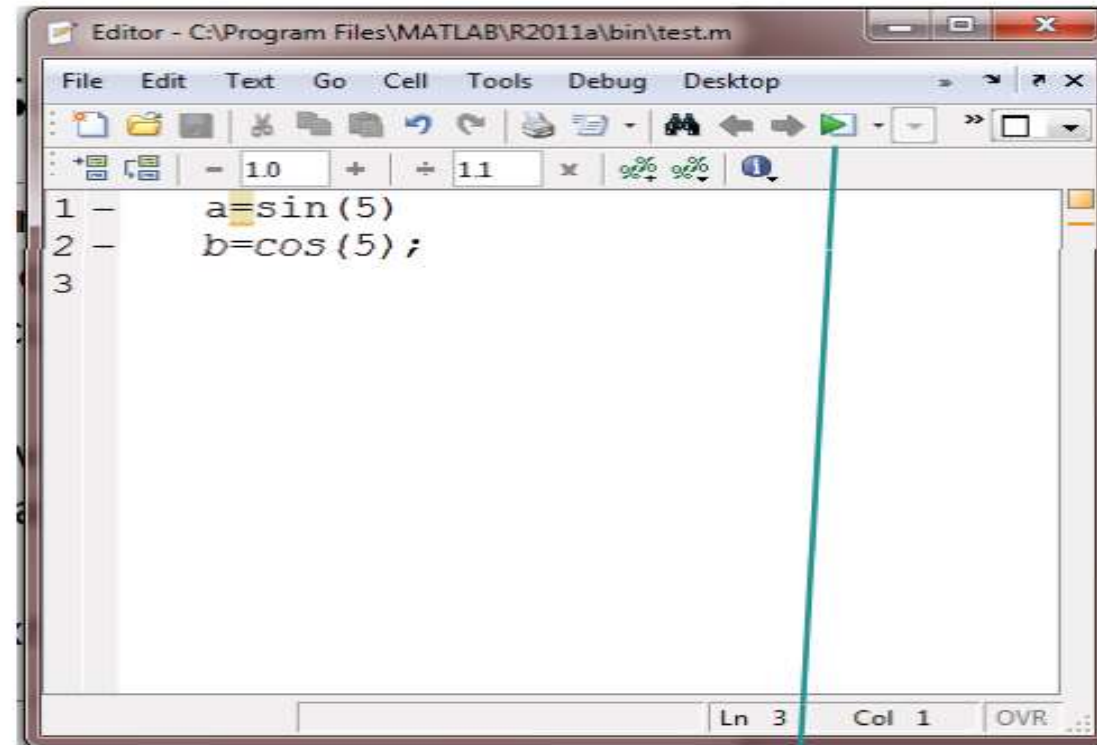
b =

    0.2837

fx >> |
```

Making Script Files

- Your work in Matlab can be stored in a script file to be executed over and over again.
- A script can be filled with a sequence of Matlab commands that will be executed from top to bottom
- These files have .m extensions



```
Editor - C:\Program Files\MATLAB\R2011a\bin\test.m
File Edit Text Go Cell Tools Debug Desktop
- 1.0 + ÷ 1.1 x
1 - a=sin(5)
2 - b=cos(5);
3
Ln 3 Col 1 OVR
```

execution

Add Comments to Programs

- Comments allow others to understand your code, and can refresh your memory when you return to it later.
- To add comments, we use the percent (%) symbol.

```
% Add up all the vector elements.  
y = sum(x)           % Use the sum function.
```

```
% This is a program that has a comment that is a little more than 75  
% columns wide.  
disp('Hello, world')
```

Clc

- Clear Command Window (clean screen)

Clear all

- delete all stored variables in your workspace

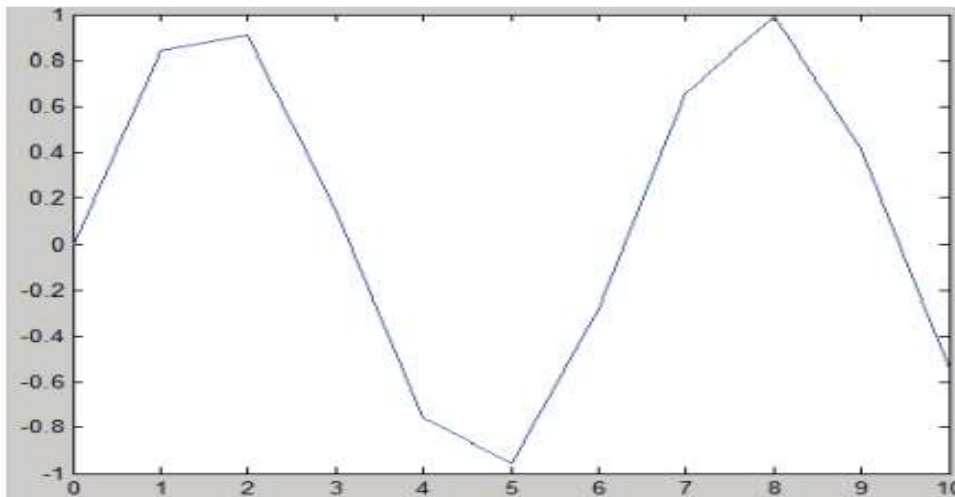
close all

- close all of figures that you have generated in your program

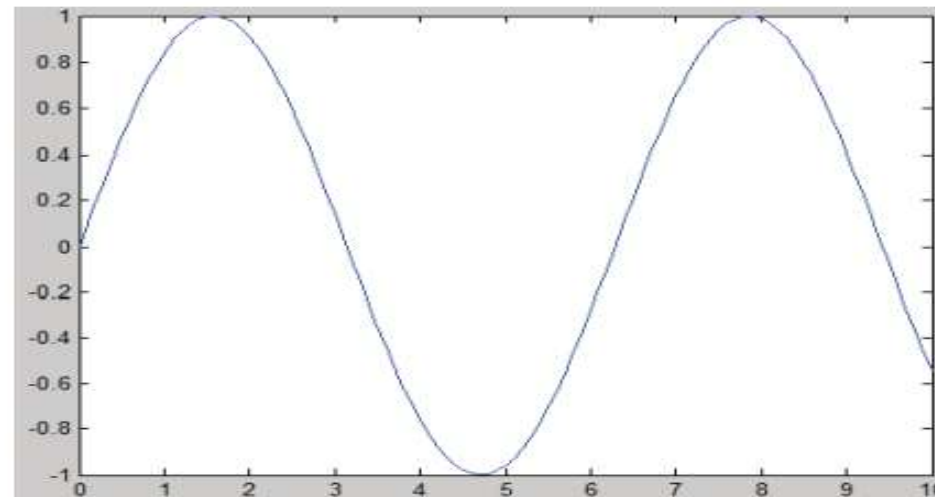
Two-dimensional Plot

plot(independent variable, dependent variable)

```
x=0:10;  
y=sin(x);  
plot(x,y)
```



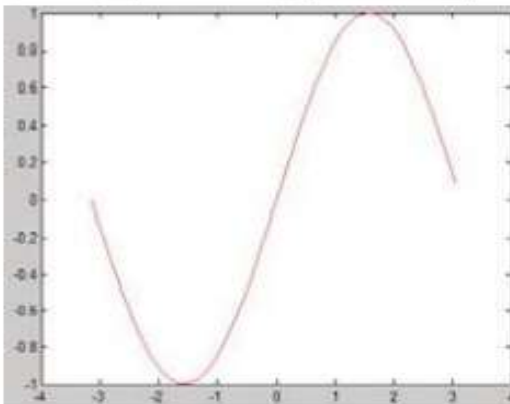
```
x=0:0.1:10;  
y=sin(x);  
plot(x,y)
```



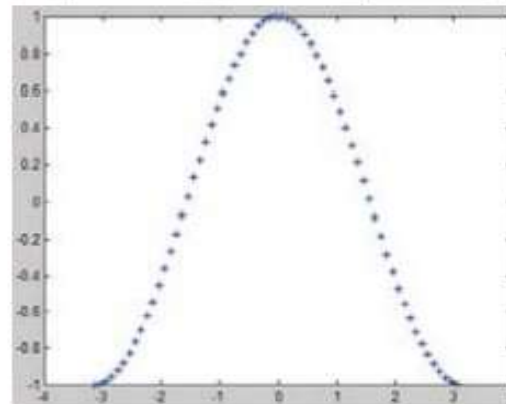
Two-dimensional Plot (properties)

Examples:

```
clc  
x=-pi:0.1:pi;  
y=sin(x);  
plot(x,y,'r')
```

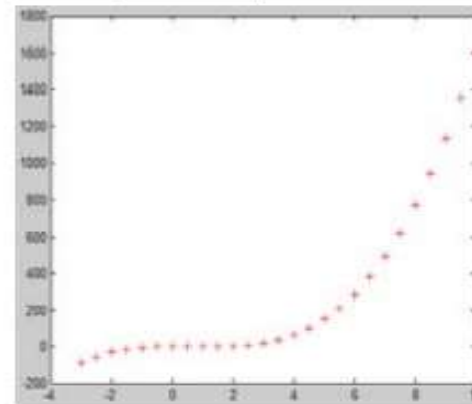


```
clc  
x=-pi:0.1:pi;  
y=cos(x);  
plot(x,y,'*')
```

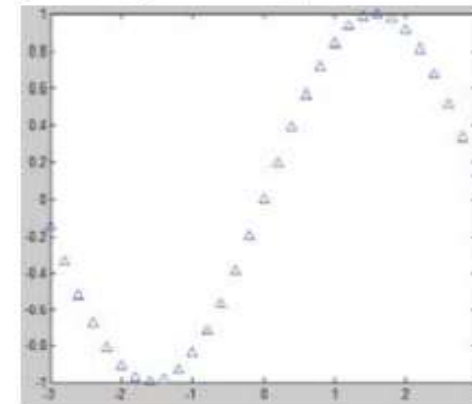


$$y = 2x^3 - 4x^2 + 2$$

```
clc  
x=-3:0.5:10;  
y=2*x.^3-4*x.^2+2;  
plot(x,y,'r*')
```



```
clc  
x=-3:0.2:3;  
y=sin(x);  
plot(x,y,'b^')
```



Two-dimensional Plot (properties)

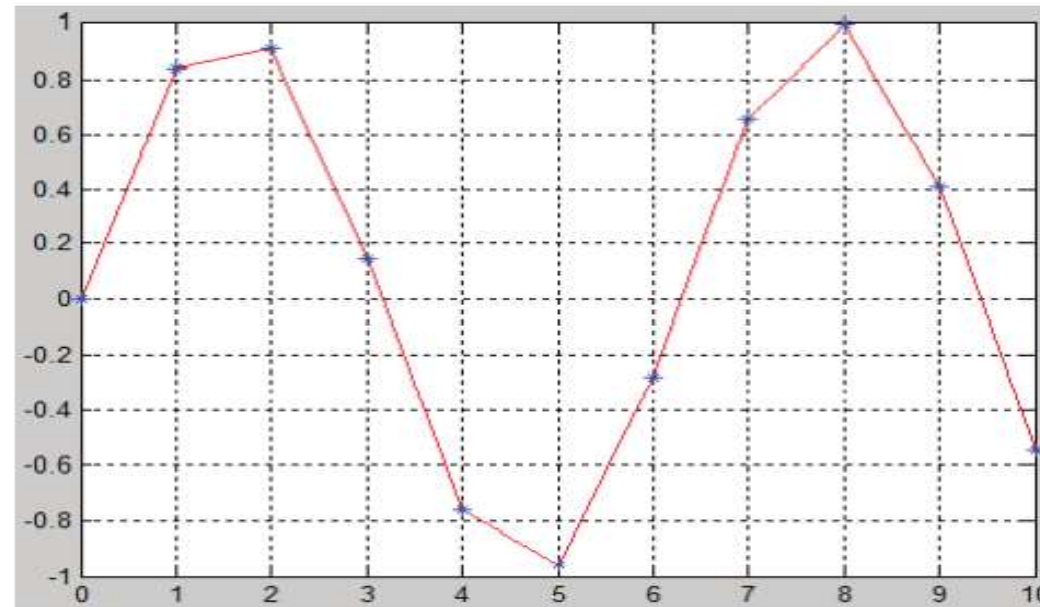
plot(independent variable, dependent variable, 'PropertyName')

1st char	meaning	2nd char	meaning	3rd char	meaning
y	yellow	.	point	-	solid
m	magenta	o	circle	:	dotted
c	cyan	x	x-mark	-.	dashdot
r	red	+	plus	--	dashed
g	green	*	star		
b	blue	s	square		
w	white	d	diamond		
k	black	v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		

Two-dimensional Plot (grid plotting)

- **Using:** *grid*
- **Note:** *grid* command must be come after plot command

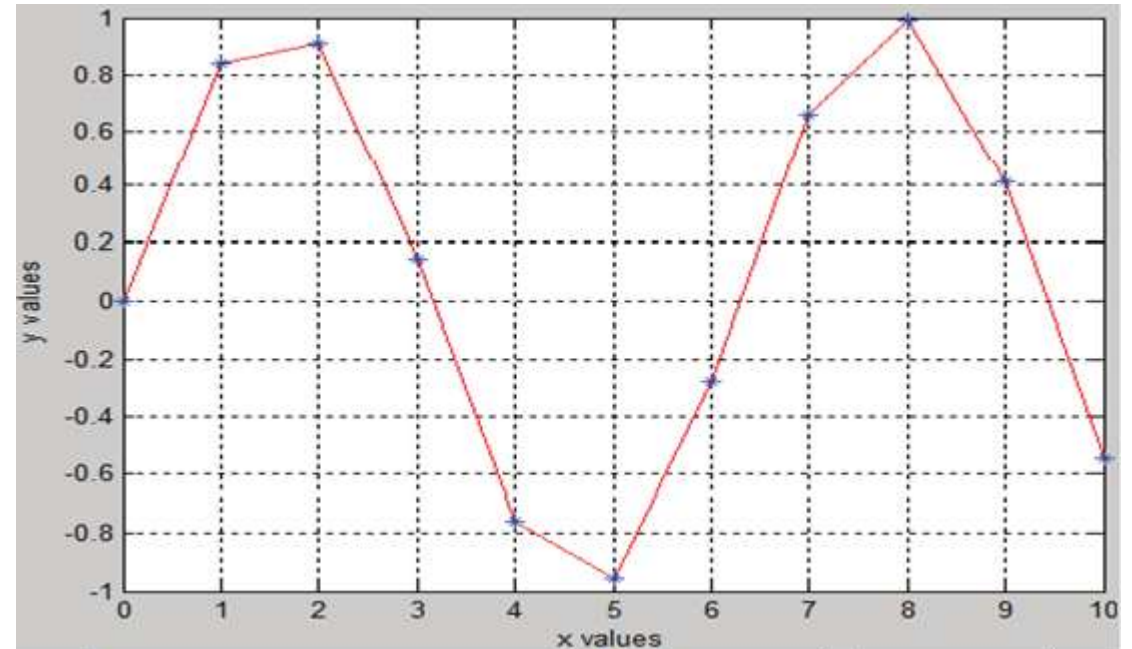
```
clc  
x=0:10;  
y=sin(x);  
plot(x,y,'r',x,y,'b*')  
grid
```



Two-dimensional Plot (axes label)

- **X-axis labeling:** `xlabel('name')`
- **Y-axis labeling:** `ylabel('name')`
- **Note:** labeling commands must be come after plot command

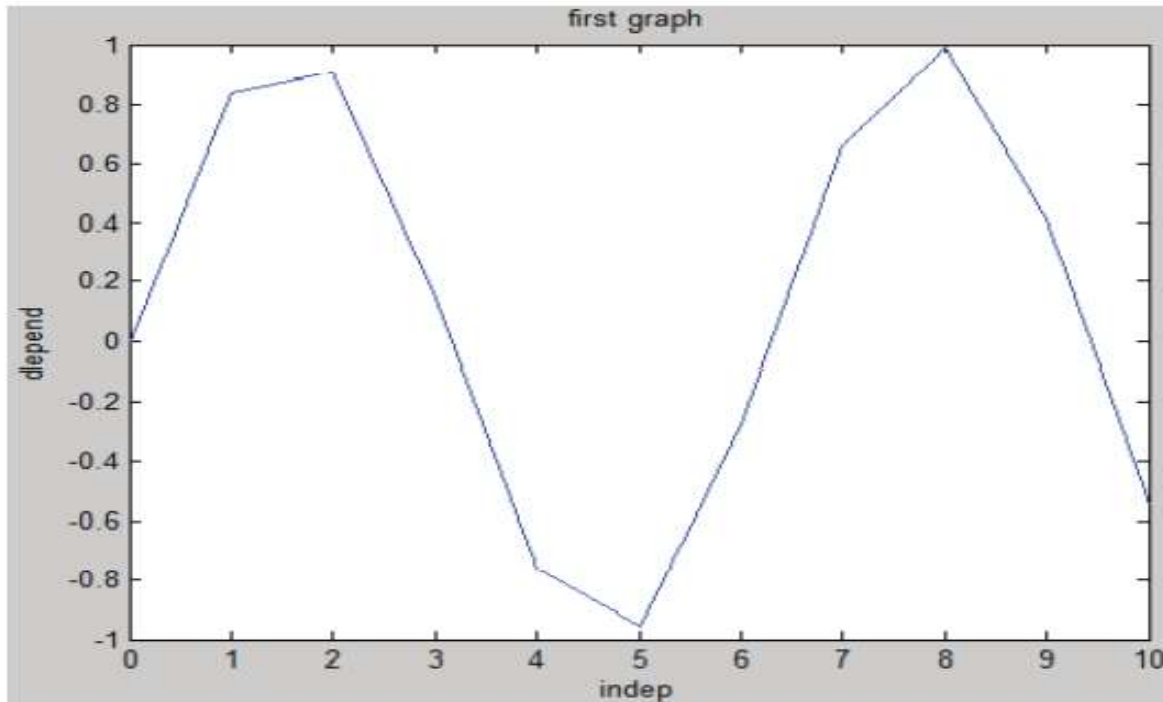
```
x=0:10;  
y=sin(x);  
plot(x,y,'r',x,y,'b*')  
xlabel('x values')  
ylabel('y values')
```



Two-dimensional Plot (axes title)

- **Title of axes:** `title('name')`
- **Note:** *title command must be come after plot command*

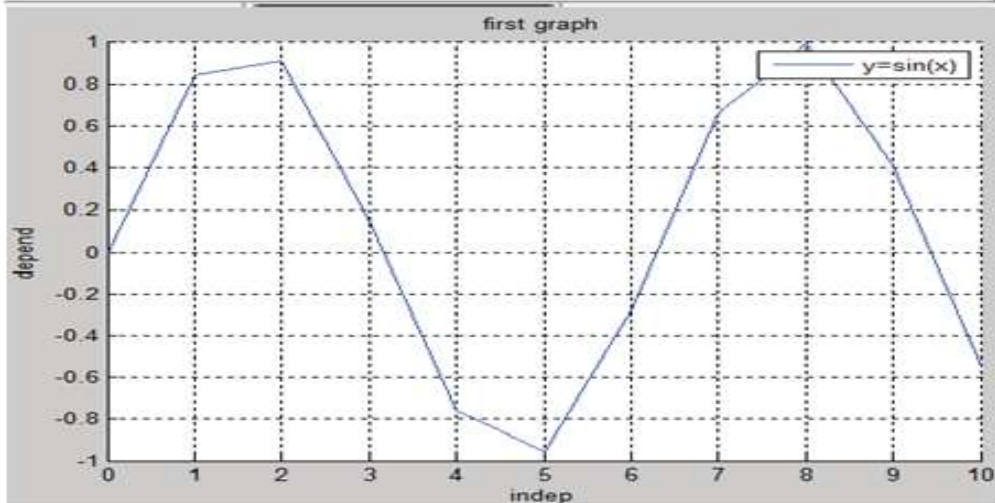
```
clc  
x=0:10;  
y=sin(x);  
plot(x,y,'b')  
title('first graph')  
xlabel('indep')  
ylabel('depend')
```



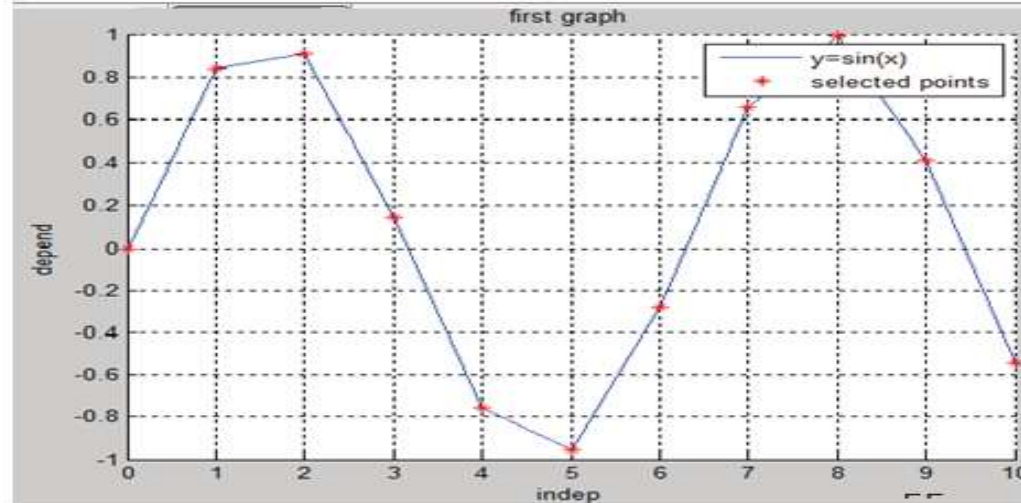
Two-dimensional Plot (legend)

➤ **Using: legend('name')**

```
x=0:10;  
y=sin(x);  
plot(x,y,'b')  
title('first graph')  
xlabel('indep')  
ylabel('depend')  
legend('y=sin(x)')  
grid
```



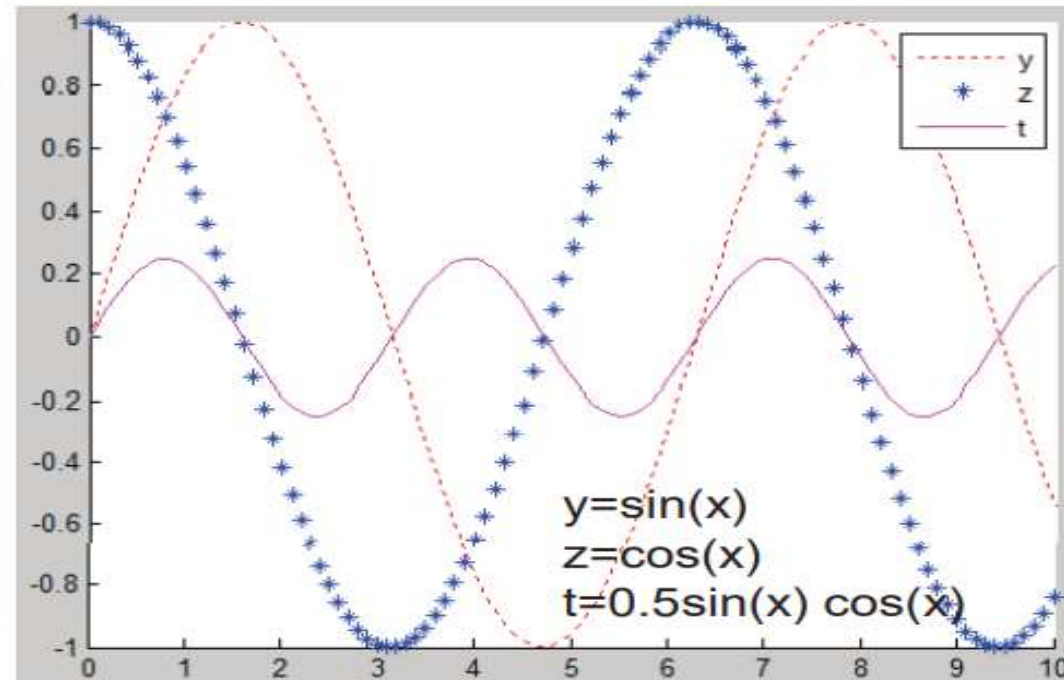
```
x=0:10;  
y=sin(x);  
plot(x,y,'b',x,y,'r*')  
title('first graph')  
xlabel('indep')  
ylabel('depend')  
legend('y=sin(x)', 'selected points')  
grid
```



Two-dimensional Plot (plotting several curves in the same window)

- **Using: hold on** (before plot commands)
- **Using: hold off** (after plot command)

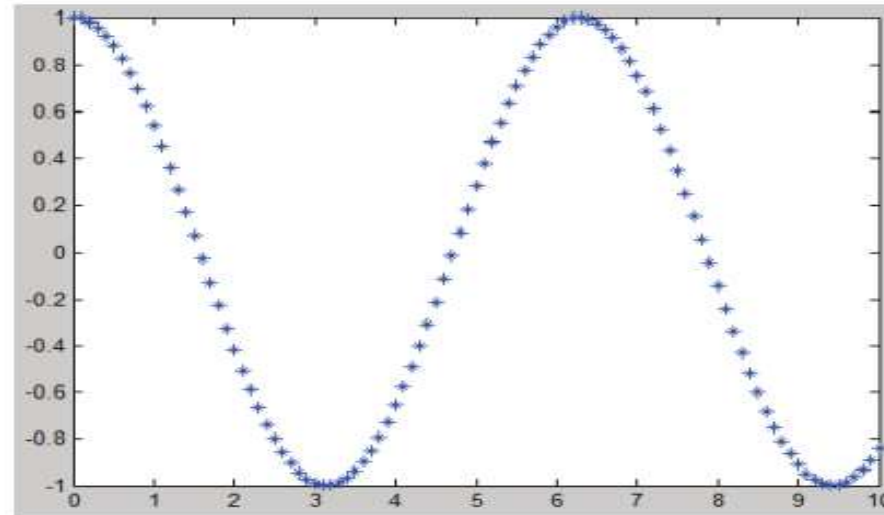
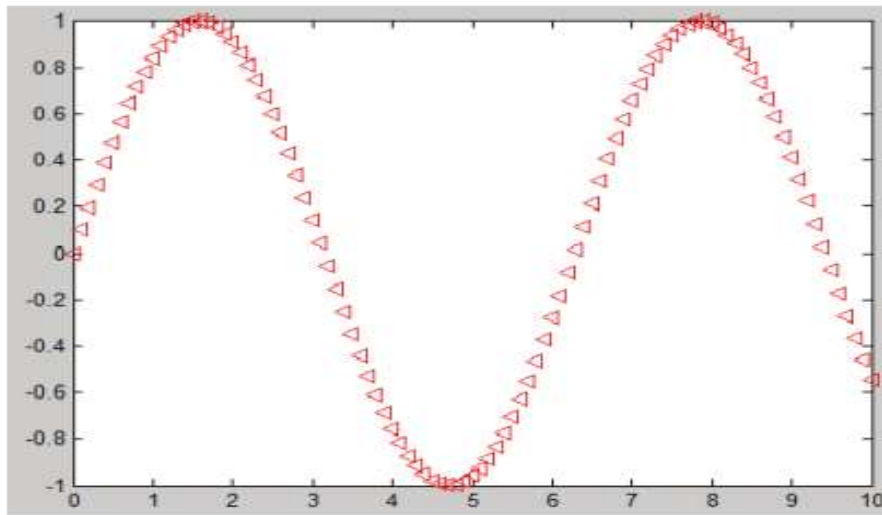
```
clc  
x=0:0.1:10;  
y=sin(x);  
z=cos(x);  
t=0.5*sin(x).*cos(x);  
hold on  
plot(x,y,'r:');  
plot(x,z,'b*');  
plot(x,t,'m');  
legend('y','z','t')  
hold off
```



Two-dimensional Plot (plotting several curves in separated windows)

➤ **Using: figure**

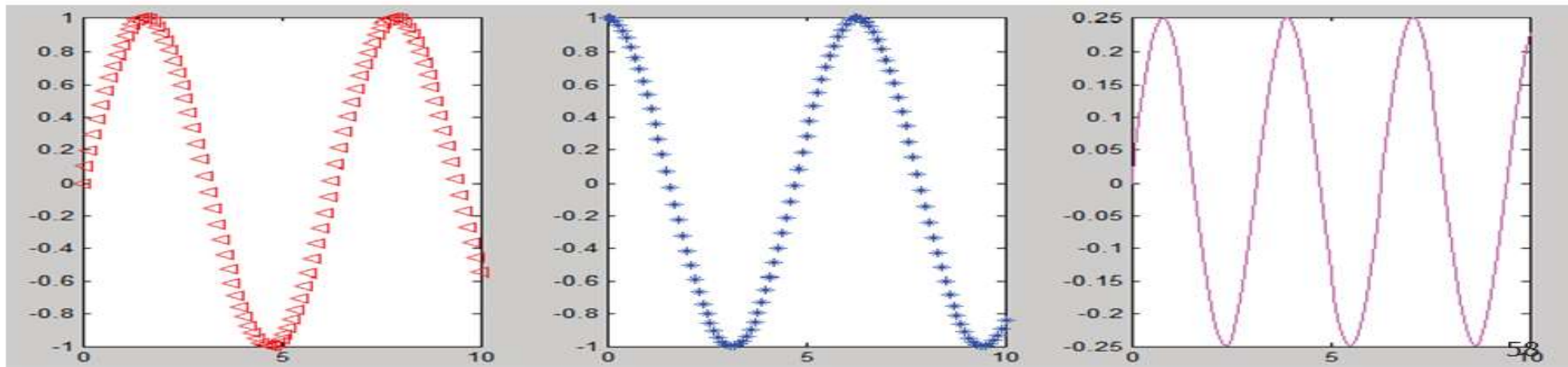
```
clc  
x=0:0.1:10;  
y=sin(x);  
z=cos(x);  
plot(x,y,'r<');  
figure  
plot(x,z,'b*');
```



Two-dimensional Plot (plotting several curves in three stacked subplots)

- **Using: subplot** (before plot command)
- **Command: subplot(m,n,p)**
- It divides the current figure into an **m-by-n** grid and creates an axes in the grid position specified by **p**.

```
clc  
x=0:0.1:10;  
y=sin(x);  
z=cos(x);  
t=0.5*sin(x).*cos(x);  
subplot(1,3,1)  
plot(x,y,'r<');  
subplot(1,3,2)  
plot(x,z,'b*');  
subplot(1,3,3)  
plot(x,t,'m');
```



Two-dimensional Plot (example 2) (plotting several curves in three stacked subplots)

```
clc  
x=0:0.1:10;  
y=sin(x);  
z=cos(x);  
t=0.5*sin(x).*cos(x);  
subplot(2,2,[1 2])  
plot(x,y,'r<');  
subplot(2,2,3)  
plot(x,z,'b*');  
subplot(2,2,4)  
plot(x,t,'m');
```

