Computer Science II

Environmental Engineering
Second Level
2024-2025
1st Course

COLLEGE OF ENGINEERING - كلية الهندسة
Tikrit University - جامعة تكريت

Lecturer Assist. Aalaa Ahmed

طريقك إلى النجاح
YOUR WAY TO SUCCESS

Lecture #3

Programming with MATLAB (Cont.)

COLLEGE OF ENGINEERING - كلية الهندسة
Tikrit University - جامعة تكريت

Lecturer Assist. Aalaa Ahmed

10. Basic plotting

MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands. You are highly encouraged to plot mathematical functions and results of analysis as often as possible. Trying to understand mathematical equations with graphics is an enjoyable and very efficient way of learning mathematics. Being able to plot mathematical functions and data freely is the most important step.

10. Basic plotting

Creating simple plots:

The basic MATLAB graphing procedure, for example in 2D, is to take a vector of x- coordinates, x = (x1, . . . , xN), and a vector of y-coordinates, y = (y1, . . . , yN), locate the points (xi, yi), with i = 1, 2, . . . , n and then join them by straight lines. You need to prepare x and y in an identical array form; namely, x and y are both row arrays or column arrays of the same length.

10. Basic plotting (Cont.)

Creating simple plots (Cont.):

The MATLAB command to plot a graph is ***plot(x,y)***. The vectors x = (1, 2, 3, 4, 5, 6) and y = (3, −1, 2, 4, 5, 1) produce the picture shown in Figure below:

```
>> x = [1  2  3  4  5  6 ];

>> y = [3  -1  2  4  5  1];

>> plot (x, y)
```

10. Basic plotting (Cont.)

Creating simple plots (Cont.):

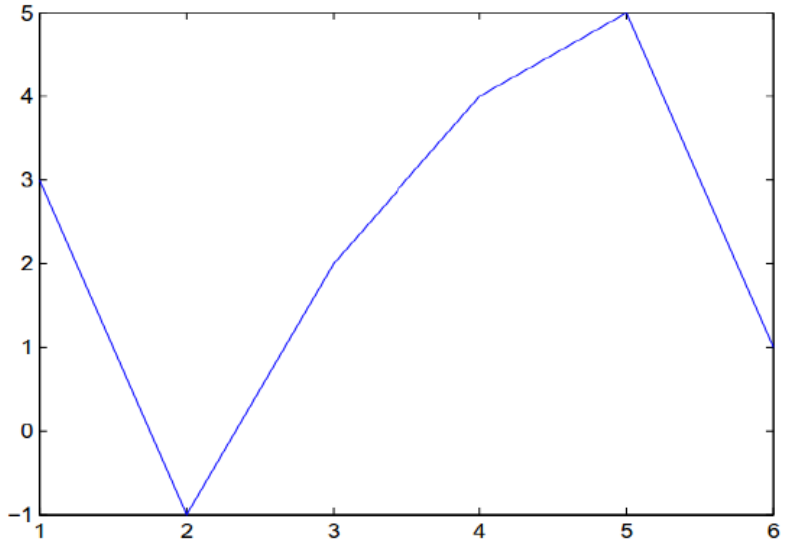
Note: The plot functions have different forms depending on the input arguments.

If y is a vector **plot(y)** produces a piecewise linear graph of the elements of y versus the index of the elements of y. If we specify two vectors, as mentioned above, ***plot(x,y)*** produces a graph of y versus x.

10. Basic plotting (Cont.)

Creating simple plots (Cont.):

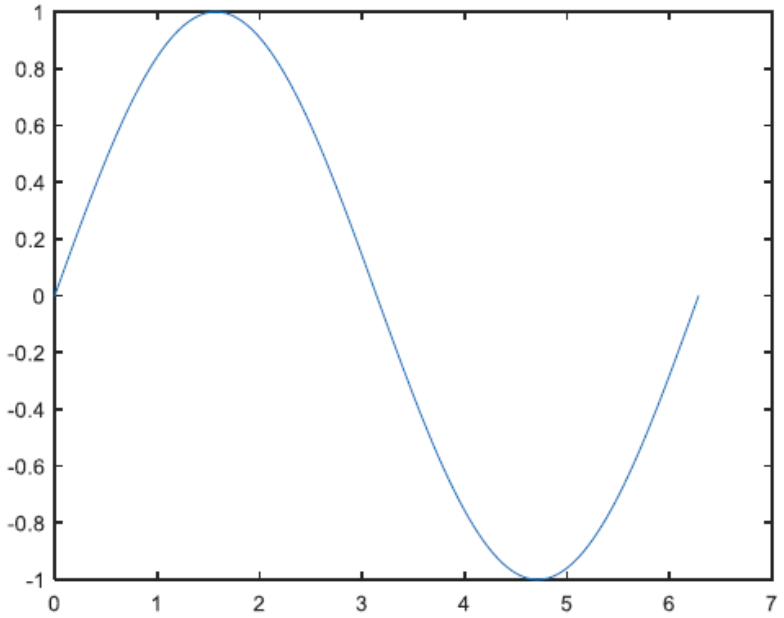
For example, to plot the function ***sin (x)*** on the interval [0, 2π], we first create a vector of x values ranging from 0 to 2π, then compute the ***sine*** of these values, and finally plot the result:



10. Basic plotting (Cont.)

Creating simple plots (Cont.):

```
>> x = 0: pi/100 :2*pi ;

>> y = sin (x) ;

>> plot (x, y)
```



10. Basic plotting (Cont.)

Creating simple plots (Cont.):

Notes:

- 0 : pi/100 : 2*pi yields a vector that:
  - starts at 0,
  - takes steps (or increments) of π/100,
  - stops when 2π is reached.
- If you omit the increment, MATLAB automatically increments by 1.

10. Basic plotting (Cont.)

Adding titles, axis labels, and annotations:

MATLAB enables you to add axis labels and titles. For example, using the graph from the previous example, add an x-axis and y-axis labels.

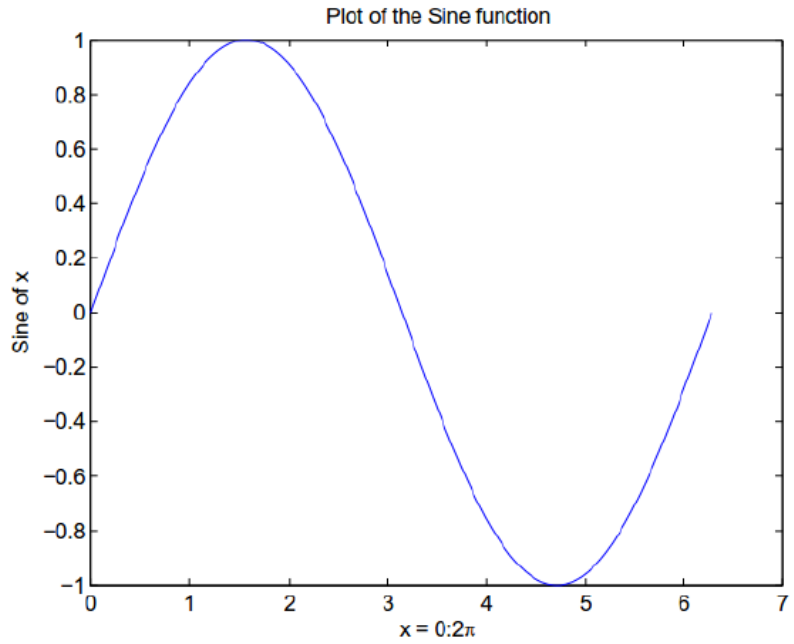
Now label the axes and add a title. The character \pi creates the symbol π. An example of 2D plot is shown in Figure below:

```
>> xlabel('x = 0:2\pi')

>> ylabel('Sine of x')

>> title('Plot of the Sine function')
```

10. Basic plotting (Cont.)

Adding titles, axis labels, and annotations (Cont.):

The color of a single curve is, by default, blue, but other colors are possible. The desired color is indicated by a third argument. For example, red is selected by plot(x,y,'r'). Note the single quotes, ' ', around r.



10. Basic plotting (Cont.)

Multiple data sets in one plot:

Multiple (*x, y*) pairs arguments create multiple graphs with a single call to plot.

For example, these statements plot three related functions of x: y1 = 2 cos(*x*), y2 = cos(*x*), and y3 = 0.5 * cos(*x*), in the interval $0 \le x \le 2\pi$.

```
x = 0:pi/100:2*pi;

y1 = 2*cos(x);

y2 = cos(x);

y3 = 0.5*cos(x);
```

10. Basic plotting (Cont.)

Multiple data sets in one plot (Cont.):

```
plot (x,y1,'--',x,y2,'-',x,y3,':')

xlabel ('0 \leq x \leq 2\pi')

ylabel ('Cosine functions')

legend ('2*cos(x)','cos(x)','0.5*cos(x)')

title ('Typical example of multiple plots')

axis ([0 2*pi -3 3])
```

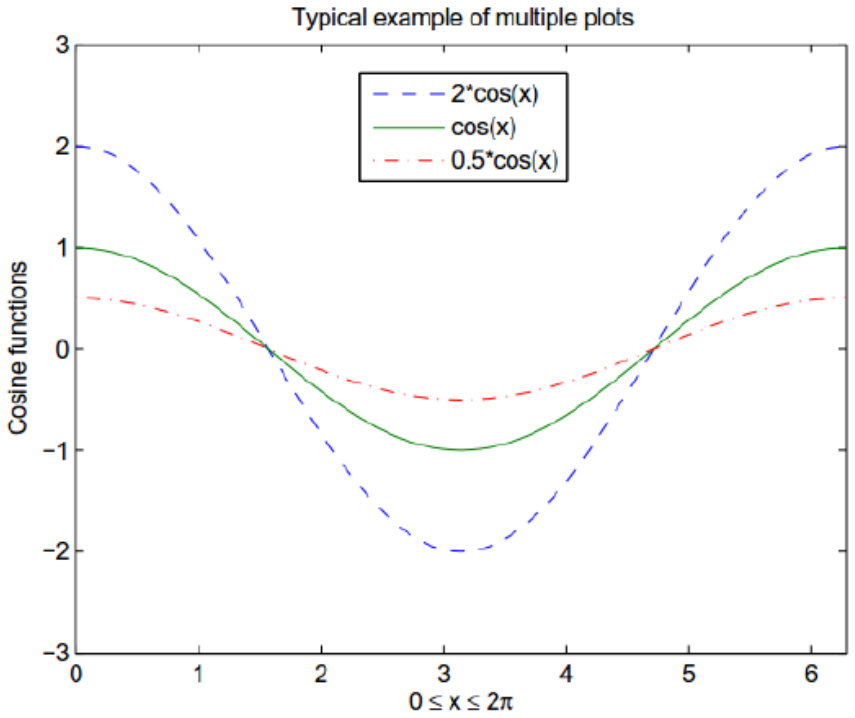
10. Basic plotting (Cont.)

Multiple data sets in one plot (Cont.):

The result of multiple data sets in one graph plot is shown in Figure below:

By default, MATLAB uses line style and color to distinguish the data sets plotted in the graph.

However, you can change the appearance of these graphic components or add annotations to the graph to help explain your data for presentation.



10. Basic plotting (Cont.)

Specifying line styles and colors:

It is possible to specify line styles, colors, and markers (e.g., circles, plus signs, . . .) using the plot command: plot(x,y,'style_color_marker')

where style_color_marker is a triplet of values from Table below. To find additional information, type help plot or doc plot.

Attributes for `plot`

Symbol	Color	Symbol	Line Style	Symbol	Marker
k	Black	–	Solid	+	Plus sign
r	Red	−−	Dashed	o	Circle
b	Blue	:	Dotted	*	Asterisk
g	Green	−.	Dash-dot	.	Point
c	Cyan	none	No line	×	Cross
m	Magenta			*s*	Square
y	Yellow			*d*	Diamond

10. Basic plotting (Cont.)

Specifying line styles and colors (Cont.):

Ex:- plot (x,y, 'r-.')

plots the points using red dash-dotted line.

plot (x,y, 'bx')

plots the points using blue crosses without joining them with lines.

plot (x,y, 'b:x')

plots the points using blue crosses and joins them with a blue dotted line.

10. Basic plotting (Cont.)

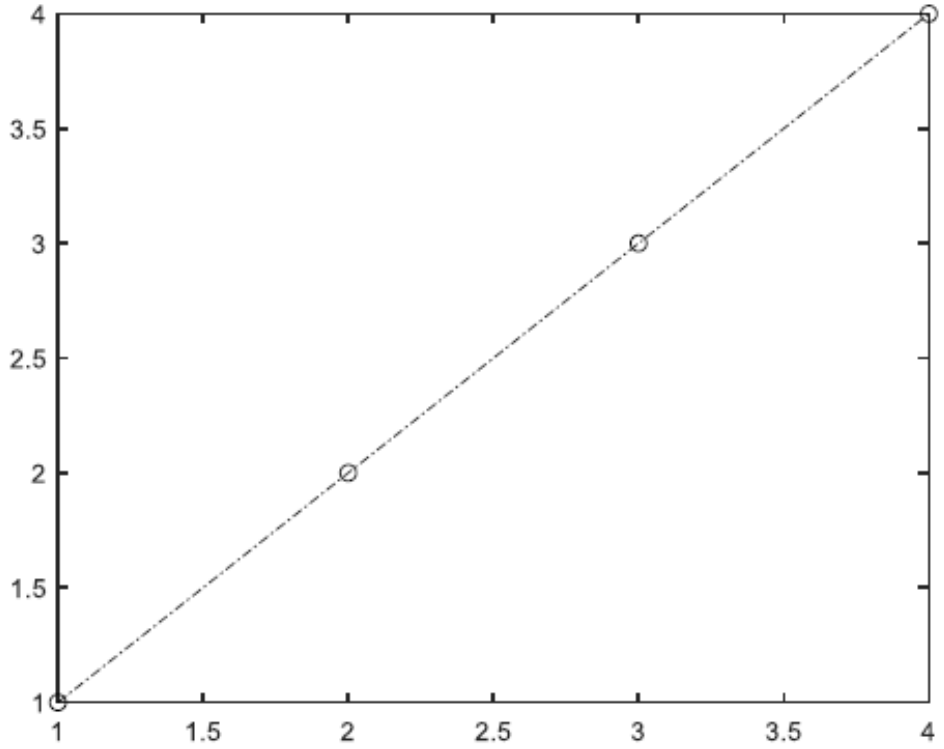
Specifying line styles and colors (Cont.):

Ex:-

w = [1 2 3 4]

v = [1 2 3 4]

plot (w,v,'k-.o')



10. Basic plotting (Cont.)

Specifying line styles and colors (Cont.):

There are several commands the appearance of the plot. These include:

Command	Description
axis ([xmin xmax ymin ymax])	define minimum and maximum values of axis
grid on	adds dashed grid lines at the tick marks
box on	add axes box, consisting of boundary lines and tick marks on top and right of plot
title ('text')	labels top of plot with text in quotes
xlabel ('text')	labels horizontal (x) axis with text in quotes
text (x,y,'text')	adds text in quotes to location (x,y) on the current axes.

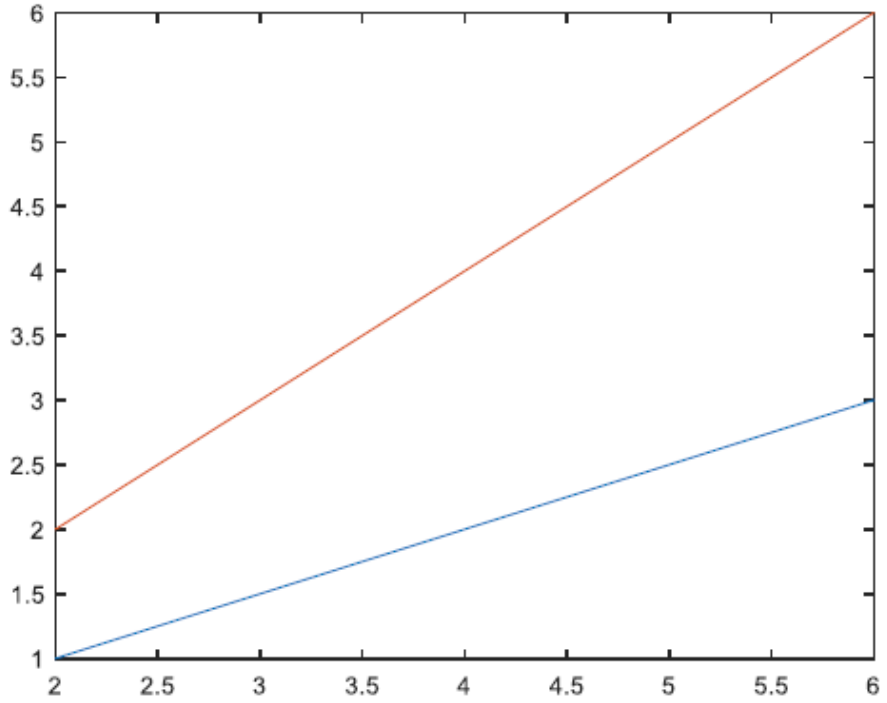
10. Basic plotting (Cont.)

Adding plots (Cont.):

When you issue a plot command MATLAB clear the axes and produces a new plot. To add to an existing plot, type hold on as example below:

```
x = [1 2 3]

y = [1 2 3]

plot (2*x, y)

hold

plot (2*x, 2*y)
```

To switch off the hold behavior, type hold off.



11. The Colon Operator:

To generate a vector of equally – spaced element MATLAB provides the colon operator. The syntax x:y means roughly generate the ordered set of numbers from x to y with increment 1 between them.

The syntax x:d:y means roughly generate the ordered set of numbers from x to y with increment d between them.

Ex:

1: 5 → 1 2 3 4 5

Ex:

0: 2: 10 → 0 2 4 6 8 10

11. The Colon Operator (Cont.):

Often we must deal with matrices or vectors that are too large to enter one element at a time. For example, suppose we want to enter a vector x consisting of points (0, 0.1, 0.2, 0.3, · · · , 5). We can use the command:

```
>>  x = 0: 0.1: 5 ;
```

The row vector has 51 elements.

12. Linear spacing:

On the other hand, there is a command to generate linearly spaced vectors: linspace. It is similar to the colon operator (:), but gives direct control over the number of points. For example, y= linspace (a,b)

To generate a vector of evenly spaces points between two end points, you can use function: linspace (start, stop, n points)

Ex: >> x = linspace (0 , 1, 10)

X= columns 1 through 6

0 0.1111 0.2222 0.3333 0.4444 0.5556

Columns 7 through 10

0.6667 0.7778 0.8889 1.0000

Generates a row vector y of n points linearly spaced between and including a and b. This is useful when we want to divide an interval into a number of subintervals of the same length.