University of Biskra

Faculty of sciences and technology

Department of industrial chemistry

2nd year process engineering

Dr Khaled ATHMANI

khaled.athmani@univ-biskra.dz

Informatics 03

2024/2025

Contents

Contents

1 Introduction

This course, designed for second-year process engineering students, focuses on building practical MATLAB skills through a series of hands-on sessions. The course is structured to introduce students to key programming concepts in MATLAB, starting from the basics and gradually progressing to more advanced topics that are crucial for solving engineering problems.

Course Structure:

- **PW01 - Introduction to MATLAB:** This session will familiarize students with the MATLAB environment, basic commands, and the structure of a MATLAB script.
- **PW02 - Input and Output Data:** Students will learn how to handle data input and output, which is essential for reading, processing, and exporting information within MATLAB.
- **PW03 - Vectors and Matrices:** This session covers the creation and manipulation of vectors and matrices, fundamental structures in MATLAB used in engineering calculations.
- **PW04 - Control Flow in MATLAB:** Students will be introduced to loops, conditionals, and other control flow mechanisms, enabling them to write more efficient and dynamic programs.
- **PW05 - Basics of Plotting in MATLAB**: This session focuses on visualizing data using MATLAB's powerful plotting functions, helping students create meaningful graphs for data analysis.
- **PW06 - MATLAB Functions:** The final session teaches how to create and use functions, allowing students to structure their code efficiently for complex problem-solving.

The aim of this course is to equip students with the foundational MATLAB programming skills necessary for solving process engineering problems. By the end of the course, students will be able to:

1. Navigate the MATLAB environment and utilize its basic functionalities.
2. Manipulate data through vectors, matrices, and control flow structures.
3. Implement plotting techniques to visualize and analyze engineering data.
4. Develop MATLAB functions to solve specific engineering challenges.

This course will empower students with the tools to effectively use MATLAB in their future academic and professional endeavors.

2 PW01- Introduction to Matlab

The most common technical computing environment is Matlab. It is a modern programming language and has extensive uses in engineering computing. The name MATLAB is derived from the term MATrix LABoratory.

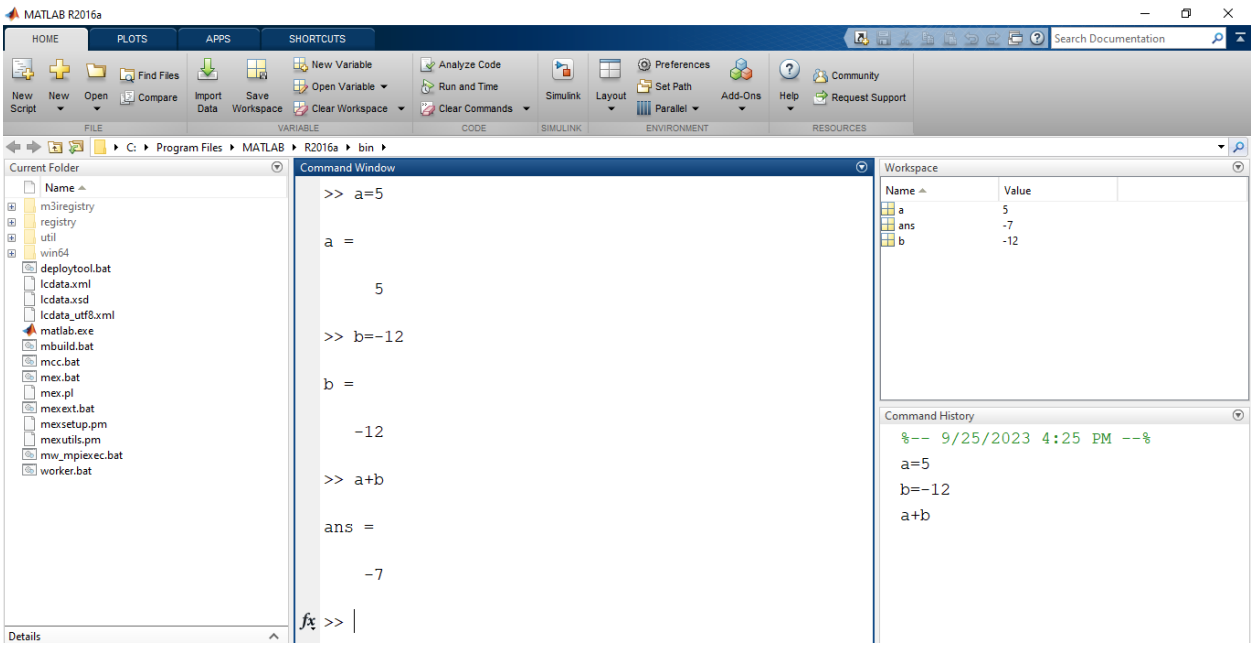
Through this tutorial and through a learning-by-doing approach, students will attain the following objectives:

- Become familiar with the Matlab interface.
- Do a few simple calculations.

2.1 Start MATLAB on Windows Platforms

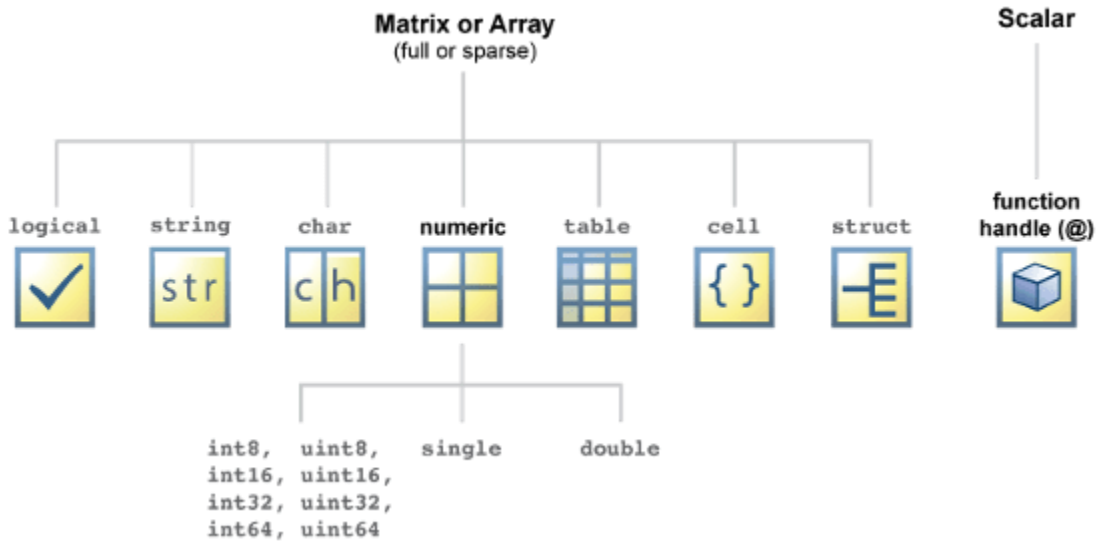
By double-clicking on the MATLAB shortcut icon, you will open MATLAB. The principal tools found on the Matlab desktop are:

- ✓ The Command Window
- ✓ The Command History
- ✓ The Workspace
- ✓ The Current Folder
- ✓ The Help Browser



2.2 Fundamental MATLAB Classes

Matlab provides many different data types, or classes. There are 17 fundamental classes in MATLAB:



Whos lists the current variables and their sizes and types.

```
>> x=single(15)

>> whos x

  Name      Size            Bytes  Class     Attributes

  x         1x1                 4  single

>> x=15

>> whos x

  Name      Size            Bytes  Class     Attributes

  x         1x1                 8  double
```

2.3 Creating MATLAB variables

MATLAB variables are created with an assignment statement. The syntax of variable assignment is:

variable name = a value (or an expression)

For example,

```
>> x = expression

ans =

    expression
```

Matlab displays intermediate results of the variable x, you can suppress the display of intermediate results by adding a semicolon (;) at the end of the x expression .

2.4 Arithmetic Operators

Symbol	Role
+	Addition
-	Subtraction
.*	Element-wise multiplication
*	Matrix multiplication
./	Element-wise right division
/	Matrix right division
.^	Element-wise power
^	Matrix power
.'	Transpose

2.5 Logical Operators

Symbol	Role	More Information
&	Find logical AND	and
|	Find logical OR	or

2.6 Relational Operators

Symbol	Role
==	Equal to
~=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

2.7 Mathematical functions

Matlab library includes a list of mathematical functions, the common functions are presented in this table:

cos (x)	Cosine	log (x)	Natural logarithm
sin (x)	Sine	log10(x)	Common logarithm
tan (x)	Tangent	abs (x)	Absolute value
acos (x)	Arccosine	sign (x)	Signum function
asin (x)	Arcsine	max(x)	Maximum value
atan (x)	Arctangent	min(x)	Minimum value
exp (x)	Exponential	rem (x)	Remainder after division
sqrt (x)	Square root		

2.7.1 Appliation:

Using MATLAB, find the value of y in the following problem:

01) $x = 1, y = exp(x)$

02) $a = 9, b = -2, x = 2.5, y = a + b\,x$

03) $a = 1, b = 6.33\,, x = 19, y = (a + x)\,(b\,sin(x))$

04) $x = \sqrt{b}\,, b = 3.14 * ln(a), a = 3.14, y = a + b + x$

05) $y = \frac{(a+b)^2}{x},\ a = 2.3, b = \frac{a}{9}, x = a + b$

2.8 M-File Scripts

A MATLAB script is a plain text file with a .m extension that contains a sequence of MATLAB commands.

You can create a blank script by Ctrl+N.

Exemple 02:

Consider the following problem:

$$y = \frac{(a+b)^2}{x}, \qquad a = 2.3, b = \frac{a}{9}, x = a + b$$

Find the solution y?

Write all the required operations on a script file and run it.

References

- Houcque, David. *Introduction to MATLAB for Engineering Students*. Northwestern University, 2005.
- MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025).

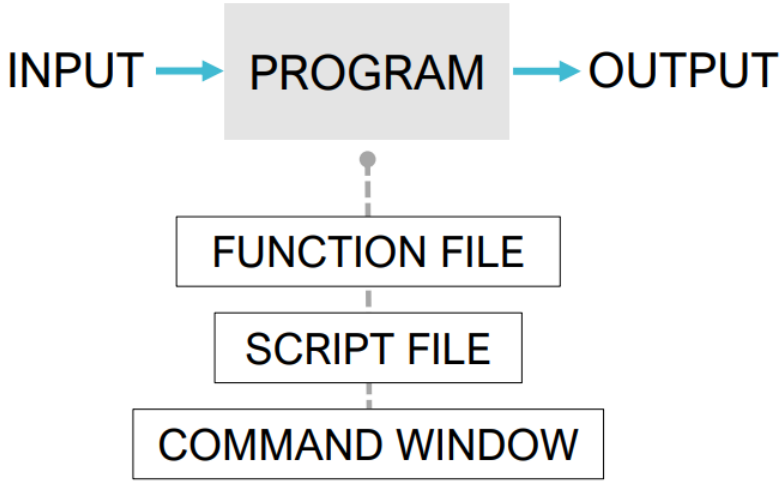
3 PW 02- Input and output data

Handling input and output data is a fundamental aspect of programming. Input refers to the data that a program receives from external sources, such as user input, files, or network connections, while output is the data that a program generates and sends to external destinations, such as displays, files, or other systems.

Through this tutorial and through a learning-by-doing approach, students will attain the following objectives:

- Prompt the user for input to an M-file program.
- Display output using the disp function.
- Create formatted output using fprintf and sprintf.
- Import and export files using load and save functions.

In MATLAB programming, input data can be provided to a program through various means, and



the output is generated after the code is executed. MATLAB allows you to work with both scripts and functions, and you can interact with the command window for quick testing and exploration.

3.1 Input:

Request user input.

Syntax:

x = **input**(prompt)

txt = input(prompt,'s') , prompt='message'

Description

- x = input(prompt) displays the text in prompt and waits for the user to input a value and press the Enter key. The user can enter expressions, like pi/4, and can use variables in the workspace.
- txt = input(prompt,"s") returns the entered text, without evaluating the input as an expression.
- If the user presses the Enter key without entering anything, then input returns an empty matrix.
- When running a script or function, function input() can be used to request a user input data at the command window.

Exemple:

01) Request a numeric input, and then multiply it by 5.

02) Request a simple text (your name).

Solution

01)

```
clear
clc
x=input('enter the value of x')

y=5*x
```

02)

```
clear
clc
x=input(' enter your name', 's')
your name
```

3.2 Output: displaying output

3.2.1 Disp

Display value of variable

Syntax

disp(x)

Description

disp(x) displays the value of variable x without printing the variable name. Another way to display a variable is to type its name, which displays a leading 'x=' before the value.

Example

01) Create a variable with numbers (your birth year) and display it using disp().

02) Display "Hello world".

Solution

01)

```
clear
clc
x=2003;
dip(x)
```

02)

```
clear
clc
x='Hello world'
disp(x)
```

3.2.2 fprintf

Write data to text file

Syntax

fprintf(formatSpec,A1,...,An)

Description

fprintf(formatSpec,A1,...,An) formats data and displays the results on the screen.

A1,...,An : elements of arrays.

Formatting Operator

A formatting operator starts with a percent sign, %, and ends with a conversion character.

Field Width

Minimum number of characters to print.

Precision

Number of digits to the right of the decimal point.

Conversion Character

This table shows conversion characters to format numeric and character data as text

Value Type	Conversion
Integer, signed	%d or %i
Integer, unsigned	%u
Floating-point number	%f Fixed-point notation (Use a precision operator to specify the number of digits after the decimal point.) %e Exponential notation, such as 3.141593e+00 (Use a precision operator to specify the number of digits after the decimal point.)
Characters or strings	%c: Single character. %s: Character vector or string array.

%4.2f in the formatSpec input specifies that the first value in each line of output is a floating-point number with a field width of four digits, including two digits after the decimal point.

Exemple:

1) Print the value of x, where x=2023.
2) Print the value of x with two digits after the decimal point, where x=11,8827.
3) Print 'Hello world'.

Solution

1)

```
clear; clc
x=2023;    fprintf('x = %d \n', x)
```

2)

```
clear; clc;
x=11,8827;
fprintf(' x = %5.2f' \n,x)
```

3)

```
clear; clc;
x='Hello world'
fprinff('%s \n',x)
```

or

```
clear; clc;
fprinff('Hello world \n')
```

\n is a control character that starts a new line.

3.2.3 sprintf

Format data into string or character vector.

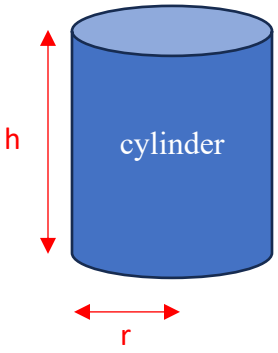
The sprintf function is similar to fprintf, but fprintf prints to a file or to the Command Window.

3.3 Exercise:

Write a code that prompts the user to specify the radius and height of the cylinder. It then uses the formula for the volume of a cylinder to calculate the result and displays it with five digits after the decimal point.

The volume of a cylinder can be calculated using the following formula:

$$V = \pi r^2 h$$



Where:

"r" is the radius of the base of the cylinder.

"h" is the height of the cylinder.

3.4 Import & export dat

3.4.1 Export data

Matlab allows saving the data from the workspace to an external file.

save

Save variables from workspace to file

Syntax

save(filename**)**

Description

Save(filename) saves all variables from the current workspace in a binary MATLAB® file (MAT-file) named filename.

Exemple

You can save the results of your last code (cylinder volume) in an external file as:

write at the end of your code:

save(‘pw 02’)

3.4.2 Import data

Matlab allows getting the data from an external file to the workspace.

Load

Load variables from file into workspace

Syntax

load(filename)

Description

load(filename) loads data from filename into the MATLAB® workspace.

Exemple

You can load the data of ‘pw 02.mat’ as:

Write on the command widow:

```
clear; clc;
load(‘pw 02.mat’)
```

3.5 References

- MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025).
- Sheikh, Usman Ullah, Mokji, Musa Mohd, et al. *Input and Output Data*. UTM.

4 PW 03- Vectors and matrices

Vectors and matrices are fundamental mathematical concepts that play a crucial role in engineering for various reasons: representation of data, linear Algebra (solving engineering problems), numerical simulations and optimization.

Through this tutorial and through a learning-by-doing approach, students will attain the following objectives:

- Learn to create vectors and matrices in MATLAB.
- Learn to perform essential mathematical operations involving vectors and matrices, including addition, subtraction, scalar multiplication, and other fundamental manipulations.

4.1 Vectors

4.2 Row vector

4.2.1 Explicit list

You can create a vector by enclosing the elements in square brackets:

x=[1 2 3 4 5 6 7 8 9 10]

or

x=[1, 2, 3 ,4 ,5 ,6 ,7 ,8 ,9 1,0]

Commas are optional.

4.2.2 colon:

Description

The colon is one of the most useful operators in MATLAB. It can create vectors.

Syntax

$x = x_0 : x_f$ (Create Unit-Spaced Vector)

$x = x_0 : \Delta x : x_f$ (Create Vector with Specified Increment)

Examples:

01) Create a unit-spaced vector of numbers between 1 and 8. The colon operator uses a default increment of +1.
02) Create a vector whose elements increment by 0.1.

Solutions

01)

```
clear; clc
x=1:8
```

02) Let's consider the vector x, its first and last values are: $x_0 = 2$ and $x_f = 3.2$.

```
clear; clc
x=2:0.1:3.2
```

4.2.3 linspace

Generate linearly spaced vector

Syntax

$x = linspace(x_0, x_f, n)$

Description

$x = linspace(x_0, x_f, n)$ generates n points. The spacing between the points is $\frac{x_f - x_0}{n-1}$.

Exemple

Use linspace to generate this vector: x=2:0.1:3.2.

This vector is used in the previous exemple.

Solution

```
clear; clc

x=linspace(2,3.2,13)
```

4.3 Column vectors

Explicit list

x=[1; -5; 9; -2; 7]

Transposing a row vector

x=[1 -5 9 -2 7]' note that : x'=transpose(x)

4.4 Matrices

4.4.1 Explicit list

To create a matrix that has multiple rows, separate the rows with semicolons.

A=[1 -2 -5; 4 2 3; 1 0 0]

4.4.2 Special matrices

Another way to create a matrix is to use a function, such as **ones** and **zeros**.

A=**zeros**(2,2)

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

B= **ones**(3.2)

$$B = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

4.4.3 Mathematical Operations

We have the folwing matrices:

$$A = \begin{bmatrix} 1 & 2 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 7 & 2 \\ -1 & 5 \end{bmatrix}$$

4.4.4 Addition and subtraction:

A+B

ans =

8 4

-1 5

A-B

ans =

-6 0

1 -5

4.4.5 Multiplication

We can multiply A by B only if the number of columns of A is equal to the number of rows of B.

Exemple

A*B

ans =

5 12

0 0

4.4.6 Matrix inverse

C = inv(B) computes the inverse of square matrix B.

B^{-1} is equivalent to inv(B).

Exemple

$$B = \begin{bmatrix} 7 & 2 \\ -1 & 5 \end{bmatrix}$$

inv(B)

ans =

0.1351 -0.0541

0.0270 0.1892

4.4.7 Transpose vector or matrix

C = A.' or C = transpose(A) : returns the nonconjugate transpose of A.

Exemple

transpose(a)

ans =

1 0

2 0

4.4.8 length and size of array

length(x) : returns the length of the largest array dimension in x.

size(x) : returns a row vector whose elements are the lengths of the corresponding dimensions of x.

Exemple:

$$B = \begin{bmatrix} 7 & 2 \\ -1 & 5 \end{bmatrix}$$

length(b)

ans =

2

size(b)

ans =

2 2

4.4.9 Matrix determinant

d = det(A) returns the determinant of square matrix A.

Exemple

```
det(B)

ans =

    37
```

4.5 References

MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025).

5 PW 04- Control flow in MATLAB

Control flow in MATLAB is essential for directing the execution of your code. It allows you to make decisions, repeat actions, and handle errors. MATLAB provides several control flow structures, including: the if statement, the for loop, the while loop, and the switch statement.

Through this tutorial and through a learning-by-doing approach, students will attain the following objectives:

- Learn how to use if-else statements for decision-making in MATLAB code.
- Master the usage of loops, including 'for' and 'while' loops, for efficient repetition and iteration in MATLAB.

5.1 Conditional Statements

To determine which block of code to execute at run time, use if or switch conditional statements.

5.1.1 if

Execute statements if condition is true, MATLAB supports the variants of "if" construct:

if expression
  statements
end
	if expression
	  statements
	else
	  statements
	end
		if expression
		  statements
		elseif expression
		  statements
		else
		  statements
		end

5.1.2 Relational and logical operators

Relational and logical operators in MATLAB are used to perform comparisons and logical operations on variables and expressions. These operators are essential for controlling the flow of your code and making decisions. The commonly used relational and logical operators in MATLAB are presented in the following tables:

Relational Operators

Symbol	Role
==	Equal to
~=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

Logical Operators

Symbol	Role	More Information
&	Find logical AND	and
|	Find logical OR	or

5.1.3 Exemple

Write a MATLAB script that classifies the type of flow, including the transitional case, based on a known Reynolds number. Your script should:

Define a variable Re and assign a known Reynolds number value to it.

Use conditional statements to classify the type of flow based on the value of `Re:

- If Re is less than 2,000, display "Laminar Flow."
- If Re is between 2,000 and 4,000 (transitional range), display "Transitional Flow."
- If Re is equal to or greater than 4,000, display "Turbulent Flow."

Test your script with different known Reynolds numbers to classify the type of flow.

Solution

```
clear; clc
Re=100;
if Re<1000
 disp('Laminar Flow');
elseif Re>=4000
 disp('Turbulent Flow');
else
 disp('Transitional Flow');
end
```

5.1.4 Switch

Execute one of several groups of statements, its syntax is:

switch switch_expression

  case case_expression

    statements

  case case_expression

statements

...

otherwise

statements

end

5.2 Loops:

MATLAB provides for and while loops for repeating a block of code.

5.2.1 for

for loop to repeat specified number of times, its syntax is:

for index = values

statements

end

Exemple:

write a MATLAB program to find the factorial of 10 using for loop.

Solution

```
clear
clc

f=1;

for j=1:10

   f=f*j;

end
fprintf('The factorial of 10 is %d ',f)
```

5.2.2 while

while loop to repeat when condition is true, its syntax is:

while expression

statements

end

Exemple

Write a MATLAB program to find the factorial of 10 using while loop.

Solution:

```
clear
clc

j=1;
f=1;
while j<=10

    f=f*j;
    j=j+1;

end
fprintf('The factorial of 10 is %d ',f)
```

Exercise:

Use if and for loop to generate the matrix A

$$A = \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & 0 \\ -5 & 0 & 1 \end{bmatrix}$$

5.3 References

- Computer Engineering Department, Faculty of Engineering, Al-Mustansiriyah University. *Control Flow in MATLAB*.
- MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025)..

6 PW 05- Basics of plotting in Matlab

Plotting is of paramount importance in engineering due to its role in data visualization, analysis, and communication. Engineers use plots to understand complex data, make informed decisions, troubleshoot issues, validate models, optimize designs, and ensure safety and quality.

Through this tutorial and through a learning-by-doing approach, students will attain the following objectives:

- Learn to create clear and visually appealing plots in MATLAB.
- Learn how to configure and enhance plots using functions for labeling, including xlabel and ylabel, and adding legends.

6.1 plot

2-D line plot

6.2 Syntax

plot(x,y)

plot(x,y,LineSpec)

plot(X1,Y1,LineSpec1,...,Xn,Yn,LineSpecn)

plot(x,y) creates a 2-D line plot of the data in y versus the corresponding values in x.

To plot a set of coordinates connected by line segments, specify x and y as vectors of the same length.

To plot multiple sets of coordinates on the same set of axes, specify at least one of x or y as a matrix.

plot(X,Y,LineSpec) creates the plot using the specified line style, marker, and color.

6.2.1 Specify Line Style, Color, and Marker

Line style, marker, and color, specified as a string scalar or character vector containing symbols:

Line Style	Description
"-"	Solid line
"--"	Dashed line
":"	Dotted line

Marker	Description
"o"	Circle
"+"	Plus sign
"*"	Asterisk

Color Name	Short Name
"black"	"k"
"red"	"r"
"blue"	"b"

To learn more about line style, color, and marker settings, visit this site:

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

6.2.2 Add Title and Axis Labels

You can add a title and axis labels by using the title, xlabel, and ylabel functions:

title('plot title')

xlabel('x-axis')

ylabel('y-axis')

6.2.3 Add legend

To add legend to axes use the function legend. Legend function creates a legend with descriptive labels for each plotted data series.

legend(label1,...,labelN)

6.3 Exercise 01:

01) Open MATLAB and create a new script file.
02) Define time and velocity vectors for two cars with different accelerations. Use the following data:

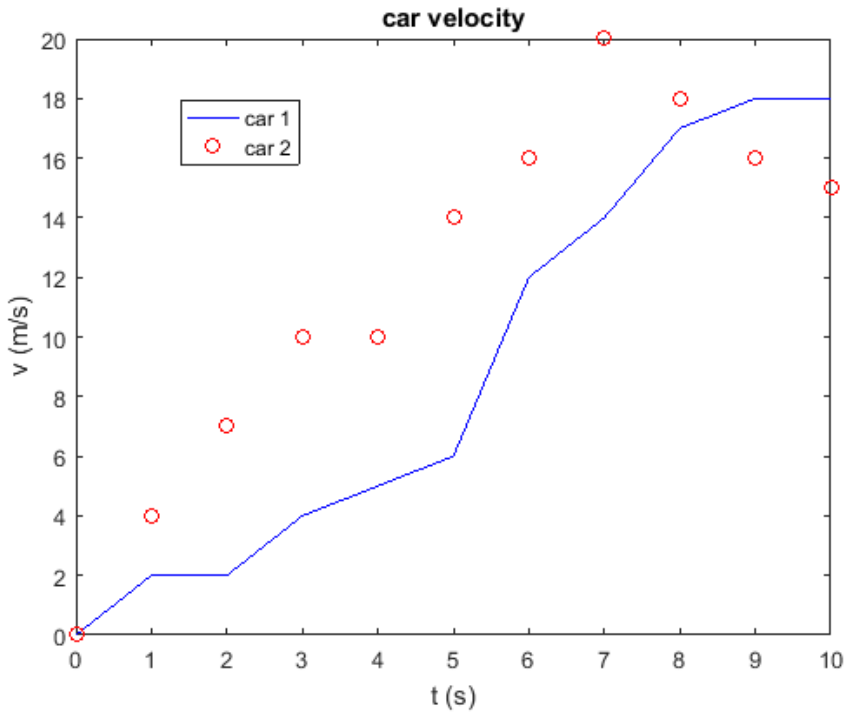
    t(s) = [0 1 2 3 4 5 6 7 8 9 10]

    v1(m/s) = [0 2 2 4 5 6 12 14 17 18 18]

    v2(m/s) = [0 4 7 10 10 14 16 20 18 16 15]

03) Create a single plot that includes both cars data using the plot function.
04) Add a title to this graph.
05) Add axis labels to the graph using the xlabel and the ylabel functions.
06) Add a legend to distinguish between Car 1 and Car 2.
07) Specify that "v1" should be plotted as a blue line.
08) Specify that "v2" should be plotted as a red circle.

At the end You will get this graph:



6.4 Exercise 02

01) Open MATLAB and create a new script file.

02) Plot the function f=x*exp(-x) over the range 0.1 to 2 with an increment of 0.1.

03) Add a title, axis labels, and a legend to the graph.

04) Specify that the curve should be plotted as a dashed black line.

6.5 References

- MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025).

2nd year process engineering **University of Biskra**

Informatics 3 **2024/2025**

7 PW 06- Matlab functions

In MATLAB, you can create functions to encapsulate a specific piece of code for reuse. Functions in MATLAB are defined in separate files with a .m extension.

Through this tutorial and through a learning-by-doing approach, students will attain the following objective:

- Gain proficiency in creating and using MATLAB functions.

7.1 function

Declare function name, inputs, and output

7.1.1 Syntax

function [y1,...,yN] = myfun(x1,...,xM)

7.1.2 Description

function [y1,...,yN] = myfun(x1,...,xM) declares a function named myfun that accepts inputs x1,...,xM and returns outputs y1,...,yN.

In a function file which contains only function definitions. The name of the file must match the name of the first function in the file.

7.2 Exercise:

Write a function with input parameters x and n that evaluates the nth order Taylor approximation of exp(x).

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x2}{2!} + \frac{x3}{3!} + \ldots$$

Test it for x=3 and n=5, 10 and 15.

Where exp(3)= 20.0855.

Write a script that calls the function for various values of n and plots the error in the approximation.

7.3 Solution

Create a new script and write the following:

```
function [y]=Taylorexp(n,x)

y=1;

for j=1:n

    y=y+x^j/factorial(j) ;
end
```

Save this script with the name **Taylorexp**.

In the command window type:

y= **Taylorexp(x,n)**

7.4 References

- MathWorks. *MATLAB Official Website*. Available at: https://www.mathworks.com/?s_tid=gn_logo (Accessed: 17 March 2025)

2nd year processes engineering **Name: ……**

Informatics 3 **Group: ……..…**

8 Homework (3 points)

8.1 Exercise 01 (1pts):

Write a MATLAB code that includes a for loop to calculate and print the values of s_j with three digits after the decimal point at each iteration, where:

$$j = [3\ 4\ 5\ 6\ 7], \quad S_0 = 10.123456, \quad S_j = S_0 + j$$

Solution:

……
……
……
……
……

8.2 Exercise 02 (1pts)

Write a Matlab program to compute the solution of a quadratic equation. check whether a = 0, to prevent a division by zero.

Solution:

……
……
……
……
……
……
……

……

…….

8.3 Exercise 03 (1 pts):

The probability density function of the random variable $\boldsymbol{T}$ described by the normal distribution is given as

$$f(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(t-\mu)^2}{2\sigma^2}}$$

$\mu = 0$; $\sigma = 1$;

1 Write a program to compute and display the values of $\boldsymbol{T}$ and f(t) over the range $(-3,3)$ with step 0.01 .

2 Plot a graph of $f(t)$, consider the following:

- write the text 'The probability density function of the normal distribution ' as a title on top of the graph.
- label the x-axis as $\boldsymbol{t}$
- label the y-axis as $\boldsymbol{f(t)}$.

Solution:

……

……

……

……

……

……

……

……

……

8.4 Exercise 02 solution

Method 1:

```
>> y = zeros(20,1);
>> y(1) = 1;
>> for n = 2:20
>> for m = 1:n
>> temp = temp + m^3;
>> end
>> y(n) = y(n-1)*temp;
>> temp = 0
>> end
```

Method 2 (avoids inner loop):

```
>> y = zeros(20,1);
>> y(1) = 1;
>> for n = 2:20
>> temp = 1:n;
>> y(n) = y(n-1)*sum(temp.^3);
>> end
```

Method 3 (avoids both the loops):

```
>> X = tril(ones(20)*diag(1:20));
>> x = sum(X.^3,2);
>> Y = tril(ones(20)*diag(x))+ …
```

triu(ones(20)) – eye(20);

>> y = prod(Y,2)

8.5 Exercise 03 solution

m = 0; % mean of probability distribution

s = 1; % standard deviation

t = -3 : 0.01 : 3; % random variable

T f = 1/(sqrt(2*pi)*s) * exp(-0.5*(t-m).^2/s^2); plot(t, f),

title('The probability density function of the normal distribution');

xlabel('t'),

ylabel('f(t)')

grid disp([t' f']) %display a table

Exam

Duration: 1.20 Name: ……

Course: Informatics 3 Group: ……..

9 Test 01 (10 points)

Q1(1.25 pts): Type the function y into MATLAB to calculate its value and print it with three digits after the decimal point, where:

أكتب الدالة y في الماتلاب من أجل حساب قيمتها و عرضها بحيث تضهر بثلاث ارقام بعد الفاصلة، حيث

$a = 1, b = 6.33\,, x = 19, y = (a^3 + \sqrt{x})\ (b\ sin(x))$

.....

.....

Q2 (1 pts): Using Matlab, create the vector A by three different techniques, where:

باستحدام الماتلاب قم بأنشاء الشعاع A بثلاث طرائق مختلفة، حيث:

A=[7 8 9 10 11]

01) …… 02) …… 03) ……

Q3 (0.5 pts): Which commands are used to clear the workspace and command window in MATLAB?:

ماهي الأوامر التي تستخدم لمحو workspace و command window

01) …… 02)……..

Q4 (0.5 pts): Which functions help you to save and load files:

ماهي الدوال التي تساعدك على حفض وقراءة الملفات

To save: …… ……. To load: …….

Q5 (1 pts): Executing in the command window the following code returns.

تنفيذ البرنامج التالي في command window ، يعطي :

C=[1 2 2; 3 5 6]

C=…….
……..
…….

length(C)

ans= ……..

size(C)

ans= …….

Q6 (0.75 pts) What functions in MATLAB are used to calculate the matrix inverse, matrix transpose, and matrix determinant.

ماهي الدوال التي تستخدم لحساب معكوس مصفوفة، مقلوب مصفوفة و محدد مصفوفة.

Inverse: ……..……. Transpose: …… Determinant: …….

Q7 (2.5 pts): Write a program that inputs any two numbers (which may be equal) and displays the larger one with a suitable message, if they are equal, displays a message to that effect:

أكتب برنامج الذي يطلب ادخال متغيرين(ممكن ان يكونا متساويين) ثم يحدد اكبر واحد منهما مع اضهار رسالة مناسبة ، اذا كانا متساويين يضهر رسالة ايضا لهته الحالة.

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Q8 (2.5 pts) Write a Matlab program to compute the value of the following function

أكتب برنامج ماتلاب لحساب قيمة الدالة التالية:

$y(n) = 1^3(1^3 + 2^3)(1^3 + 2^3 + 3^3) \dots (1^3 + 2^3 + \dots + n^3)$ for n = 1 to 20

.....

.....

.....

.....

.....

.....

.....

9.1 Test 01 solution (10 points)

Q1(1.25 pts): Type the function y into MATLAB to calculate its value and print it with three digits after the decimal point, where:

أكتب الدالة y في الماتلاب من أجل حساب قيمتها و عرضها بحيث تضهر بثلاث ارقام بعد الفاصلة، حيث

$a = 1, b = 6.33, x = 19, y = (a^3 + \sqrt{x})\ (b\ sin(x))$

a=1; b=6.33; x=19; y=(a^3 + sqrt(x)) *b* sin(x) ; 0.75pts

fprintf('y = %4.3f',y) 0.5pts

Q2 (1 pts): Using Matlab, create the vector A by three different techniques, where:

باستحدام الماتلاب قم بأنشاء الشعاع A بثلاث طرائق مختلفة، حيث:

A=[7 8 9 10 11]

02) **A=[7 8 9 10 11]** 0.25pts 02) **A=7:11** 0.25pts 03) **linspace(7,11,5)** 0.5pts

Q3 (0.5 pts): Which commands are used to clear the workspace and command window in MATLAB?:

ماهي الأوامر التي تستخدم لمحو workspace و command window

01) **clear** 0.25pts 02) **clc** 0.25pts

Q4 (0.5 pts): Which functions help you to save and load files:

ماهي الدوال التي تساعدك على حفض وقراءة الملفات

To save: **save('filename')** 0.25pts To load: **load('filename')** 0.25pts

Q5 (1 pts): Executing in the command window the following code returns.

تنفيذ البرنامج التالي في command window ، يعطي :

C=[1 2 2; 3 5 6]

C=

1 2 2

3 5 6 0.5pts

length(C)

ans= **3** 0.25pts

size(C)

ans= **2 3** 0.25pts

Q6 (0.75 pts) What functions in MATLAB are used to calculate the matrix inverse, matrix transpose, and matrix determinant.

ماهي الدوال التي تستخدم لحساب معكوس مصفوفة، مقلوب مصفوفة و محدد مصفوفة.

Inverse: **inv** 0.25pts Transpose: **transpose** 0.25pts Determinant: **det** 0.25pts

Q7 (2.5 pts): Write a program that inputs any two numbers (which may be equal) and displays the larger one with a suitable message, if they are equal, displays a message to that effect:

أكتب برنامج الذي يطلب ادخال متغيرين(ممكن ان يكونا متساويين) ثم يحدد اكبر واحد منهما مع اضهار رسالة مناسبة ، اذا كانا متساويين يضهر رسالة ايضا لهته الحالة.

```
clear; clc      0.25pts
x = input('Enter the first number (x): '); 0.25pts
y = input('Enter the second number (x) '); 0.25pts
if (x>y)   1pts
 disp('larger is: x'); 0.25pts
elseif (y>x)
 disp('larger is: y'); 0.25pts
else
 disp('Equal numbers'); 0.25pts
end
```

Q8 (2.5 pts) Write a Matlab program to compute the value of the following function

أكتب برنامج ماتلاب لحساب قيمة الدالة التالية:

$y(n) = 1^3(1^3 + 2^3)(1^3 + 2^3 + 3^3) \ldots (1^3 + 2^3 + \ldots + n^3)$ for n = 1 to 20

```
Clear; clc 0.25pts

y=1;   0.25pts
for j=2:20   0.25pts
    f=1;   0.25pts
    for k=2:j    0.5pts

      f=f+k^3 ;  0.25pts

    end

    y=y*f;   0.5pts
end
disp(y)   0.25pts
```