University Mohamed Boudiaf of M'Sila **Faculty of Technology**
Department of Electrical Engineering **2nd Year Licence in REE**
Module: Practical work in numerical methods **Teacher: BOUGUERRA abderrahmen**

PW 01: AN INTRODUCTION TO MATLAB

Abstract

MATLAB® is a software suite developed by The MathWorks, Inc. (http://www.mathworks.com) that is compatible with various systems, from personal computers to supercomputers, including parallel computing environments. This practical Work is to offer a comprehensive introduction to MATLAB, supplying adequate context for the numerical approaches under consideration.

MATLAB provides the user with:

- 1. Effortless manipulation of matrix configurations.
- 2. A multitude of robust built-in routines that are always expanding and evolving.
- 3. Robust two- and three-dimensional graphing capabilities.
- 4. A scripting system enabling users to create and customize the software according to their requirements.
- 5. Collections of functions, referred to as toolboxes, that can be integrated into the fundamental MATLAB functionalities. These are intended for particular applications, such as neural networks, optimization, digital signal processing, and higher-order spectrum analysis.

I. MATRICES IN MATLAB

In MATLAB the arithmetic operations of addition, subtraction, multiplication, and division can be performed in the usual way on scalar quantities, but they can also be used directly with matrices or arrays of data.

In command window or mfile

```
>> A = [1 2 3;4 5 6;7 8 9] % matrix A
>> Y=[1 2 3;4 5 6;7 8 9] ; % What is the difference between A and Y?
>> Y
>> y
>> B = [10 11 12;13 14 15;16 17 18] ; Look at the workspace
>> C = A+B
>> C = A-B % Does S retain its original value?
```

```
>> C
>> whos
>> C = A+B
>> C = A+B'
>> clear A
>> A
>> B = [ ];
>> C = zeros(4,4);
>> B = ones(5,3);
>> [p q] = size(A)
>> L = length(B)
>> X(1,3) = A(2,3)+B(1,2)
>> A = [2 3 4 5 6;-4 -5 -6 -7 -8;3 5 7 9 1; ...
4 6 8 10 12;-2 -3 -4 -5 -6]
>> v = [1 3 5];
>> b = A(v,2)
>> C = A(v,:)
>> D = zeros(3);
>> D(:,1) = A(v,2)
>> E = A(1:2,4:5)
>> A(4,5)
>> A(end-1,end)
>> >> P = [1 2 3 4;5 6 7 8; 9 10 11 12; 13 14 15 16]
>> reshape(P,8,2)
```

What is the difference between the two processes?

What is the difference between the two processes?

II. TRANSPOSING MATRICES

A fundamental operation that may be executed on a matrix is transposition, which swaps rows and columns. In MATLAB, transposition is represented by the symbol ':

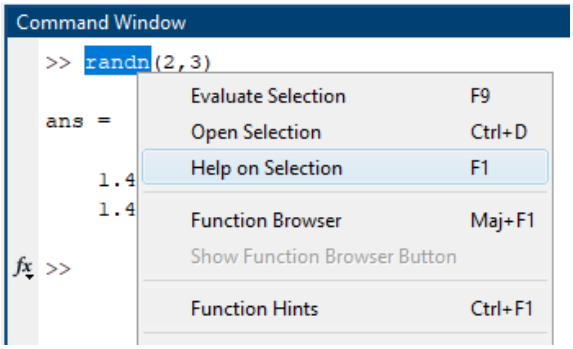
In command window or mfile

```
>> A = [1 2 3;4 5 6;7 8 9]
>> B = A'
>> A = [1+2i 3+5i;4+2i 3+4i] % A is complex
>> B = A' % MATLAB operator ' gives the complex conjugate transpose
```

```
>> C = A.' % To provide the transpose without conjugation we execute
```

III. SPECIAL MATRICES

```
>> rand(2,3)
>> randn(2,3)
```



function eye(m,n)generates a matrix of m rows and n columns with a diagonal of ones; thus:

```
>> A = eye(3,4)
```

IV. GENERATING MATRICES AND VECTORS WITH SPECIFIED ELEMENT VALUES

Here we confine ourselves to some relatively simple examples thus:

x = -8:1:8 (or x = -8:8) sets x to a vector having elements -8,-7, ...,8.

y = -2:.2:2 sets y to a vector having elements -2,-1.8,-1.6, ..., 1.8, 2.

z = [1:3 4:2:8 10:0.5:11] sets z to a vector having the elements

[1 2 3 4 6 8 10 10.5 11]

The MATLAB function linspace also generates a vector. However, in this function the user defines the beginning and end values of the vector and the number of elements in the vector. For example:

```
>> w = linspace(-2,2,5)
>> H = diag([2 3 4])
>> P = rand(3,4)
>> diag(P)
>> A1 = [1 2 5;3 4 6;3 4 5];
>> A2 = [1.2 3.5,8;0.6 0.9,50];
>> blkdiag(A1,A2,20)
```

V. ELEMENT-BY-ELEMENT OPERATIONS

```
>> A = [1 2;3 4]
>> B = [5 6;7 8]
```

```
>> A*B
>> A.*B
>> A.^B
```

Table 1.1 Selected MATLAB mathematical functions

Function	Function gives	Example
sqrt(x)	square root of x	y – sqrt(x+2.5);
abs(x)	if x is real, is the positive value of x	
	if x is complex, is the scalar measure of x	d – abs(x)*y;
real(x)	real part of x when x is complex	d – real(x)*y;
imag(x)	imaginary part of x when x is complex	d – imag(x)*y;
conj(x)	the complex conjugate of x	x – conj(y);
sin(x)	sine of x in radians	t – x+sin(x);
asin(x)	inverse sine of x returned in radians	t – x+sin(x);
sind(x)	sine of x in degrees	t – x+sind(x);
log(x)	log to base e of x	z – log(1+x);
log10(x)	log to base 10 of x	z – log10(1-2*x);
cosh(x)	hyperbolic cosine of x	u – cosh(pi*x);
exp(x)	exponential of x, i.e., e^x	p – .7*exp(x);
gamma(x)	gamma function of x	f – gamma(y);
bessel(n,x)	nth-order Bessel function of x	f – bessel(2,y);

```
>> >> x = [-4 3];
>> abs(x)
>> x = 3+4i;
>> abs(x)
>> imag(x)
x = linspace(0,pi,5)
>> sin(x)
```

VI. INPUT AND OUTPUT IN MATLAB

```
>> >> disp('This will display this test')
>> x = 2.678;
>> disp(['Value of iterate is ', num2str(x), ' at this stage'])

%%%%% mfile

% exemple (save & load)
x = [1 0 2 3 1]; y = [-1 0 4 1 1];
A = [x;x.^2; x.^3;x.^4;x.^5];
```

```
B = [x;2*x; 3*x; 4*x;5*x];
save test A B
clear all
whos
disp('nothing listed')
load test
C = A*B;
disp('Results of matrix multiplication')
disp(C)
```

VII. MATLAB GRAPHICS

```
%%%%% mfile
x = -4:0.05:4;
y = exp(-0.5*x).*sin(5*x);
figure(1), plot(x,y)
xlabel('x-axis'), ylabel('y-axis')
hold on
y = exp(-0.5*x).*cos(5*x);
plot(x,y), grid
gtext('Two tails...')
hold off
```

VIII. CREATE MATLAB FUNCTION

```
function [f1] = myfunction(x)
f1 = sin(0.1*pi*x); % Calculate function f1
subplot(2,2,1); stem(x,f1,'filled');
xlabel('x') ; % Label of X-axis
ylabel('f1[x]') % Label of Y-axis
end
```

(matlab window)

```
m=[-5:5]
myfunction(m)
```