Kut University College

Medical instrumentation techniques engineering

Computer Applications

M.Sc. Ghofran Saleem

What is MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-touse environment where problems and solutions are expressed in familiar mathematical notation.

MATLAB Desktop Tools

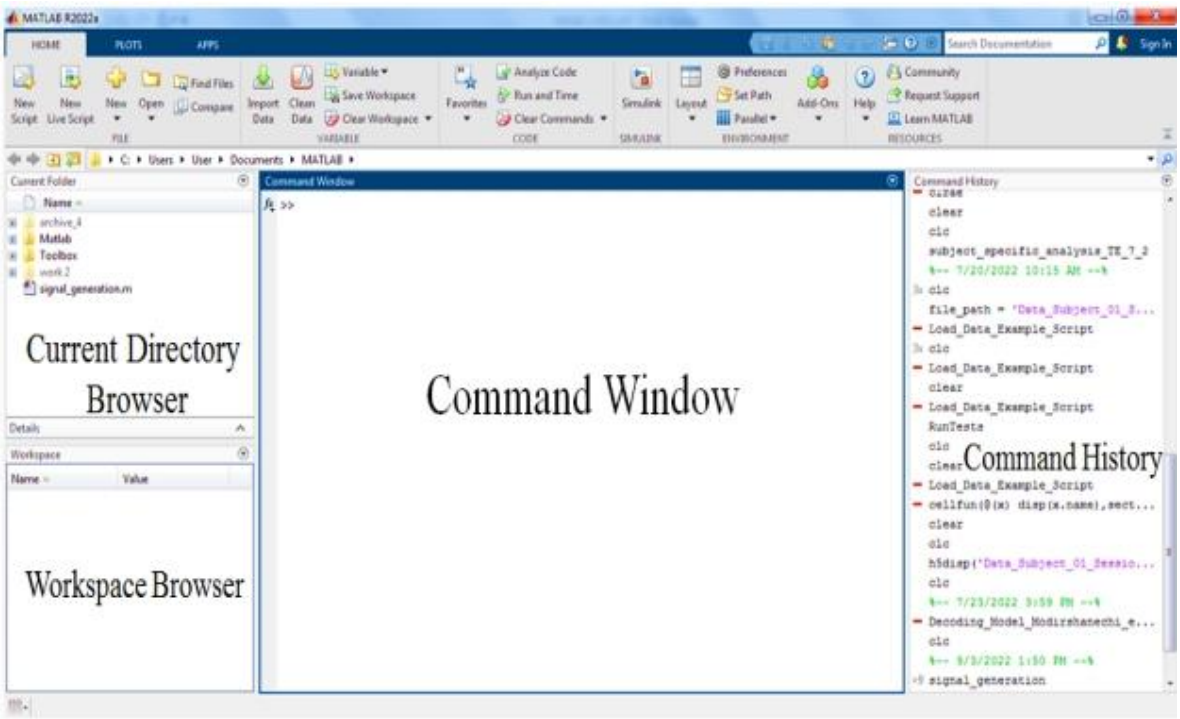
When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

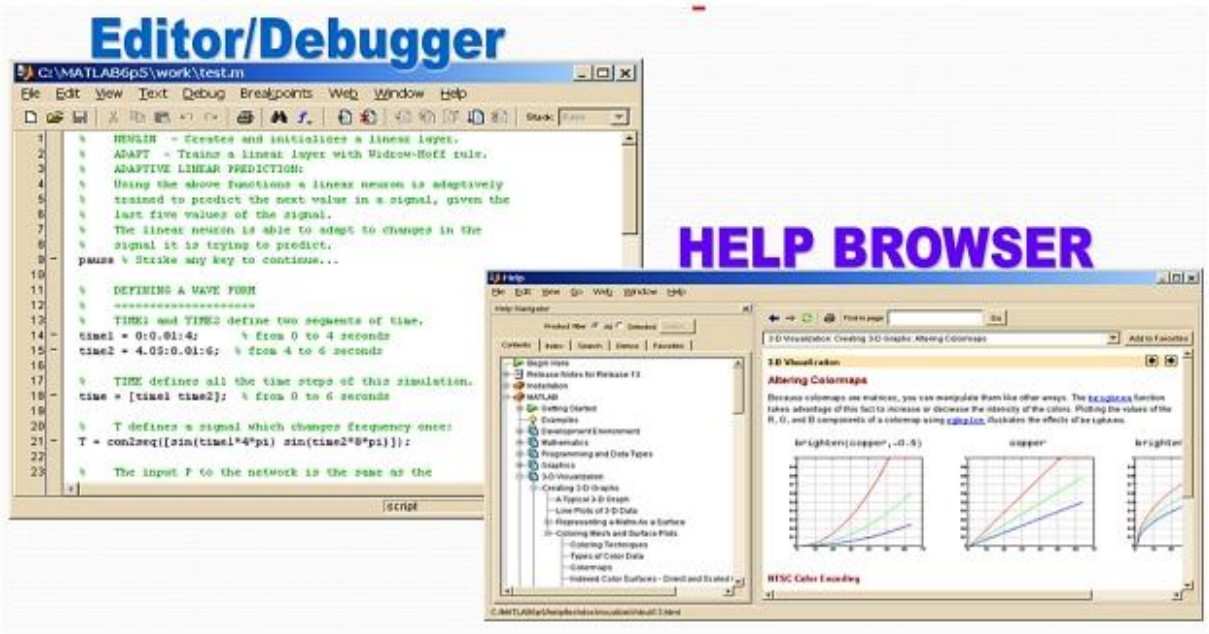
- Command Window
- Command History
- Current Directory Browser
- Workspace Browser
- Editor/Debugger
- Help Browser

To select what you want to see, go under “Desktop” tab.

Desktop Tools

- **Command Window:** Use the Command Window to enter variables and run functions and M-files.
- **Command History:** Statements you enter in the Command Window are logged in the Command History. In the Command History, you can view previously run statements, and copy and execute selected statements.
- **Current Directory Browser:** MATLAB file operations use the current directory reference point. Any file you want to run must be in the current directory or on the search path.
- **Workspace:** The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory.


Current Directory Browser
Command Window
Command History
Workspace Browser


Editor/Debugger
HELP BROWSER

Using MATLAB as a calculator

For example, let‟s suppose you want to calculate the expression, 1+2*3

You type it at the prompt command (>>) as follows,

```
>>1+2*3
ans=
 7
```

Note that the variable ans is created (or overwritten, if it is already existed). To avoid this, you may assign a value to a variable or output argument name. For example,

```
>> x=1+2*3
 x=
 7
```

Notes:

1-A **semicolon " ; "** at the end of a MATLAB statement suppresses printing of results .

For example:

```
>>x=3:
>>y=x+5
```

When you click execute bottom, MATLAB executes it immediately and the result returned is:

```
>>y=8
```

2-If a statement does not fit on one line, use " . . . ", followed by Enter to indicate that the statement continues on the next line .

For example:

>>S=sqrt(225)*30/…

(20*sqrt(100))

3-Adding comments

The percent symbol (%), is used for indicating a comment line .

For example:

>>x = 9 % assign the value 9 to x

the statement will appear in green color.

A First Program

1-Expressions

What is Expression in general?

The expressions consist of the various math functions like as arithmetic , trigonometric, logarithmic, exponential, constant term value, and so on . These functions have proper syntax.

Numbers, symbols, operators (arithmetic symbol such as : + , – , * , and /) grouped together and obtain the results.

For example : 7 + 3 is an expression.

Let's suppose you want to calculate the expression, 2 + 4 * 5. Type it in command window

```
>>2+4*5
 ans = 22
```

Example the MATLAB expressions

Example the MATLAB expressions

- The expressions are given by:

$1+\frac{2}{3}4-5$, in MATLAB `1 + 2/3*4-5`

(((1/2)/3)/4) , in MATLAB `1/2/3/4`

$(2-3*1)\frac{4}{5}$, in MATLAB $(2-3*(4-3))*4/5$

2- .Rules for Naming of Variables

The rules for naming variables in MATLAB can be summarized as

follows:

- Variable names in MATLAB must start with a letter and can be up to a - z characters long. The trailing characters can be numbers , letters or underscores (some other characters are also available but in this text we shall stick to these .(
- Variable names in MATLAB are case sensitive, so that (a) and (A) are two different objects.
- It is good programming practice to employ meaningful variable names .

Note that spaces are not important in MATLAB.

Example : Try to type the following assignment:

>>x = 2;

>> y = 4 ;

>> z = x*y

z=

8

Trigonometric Symbol	Operation / Function
sin(t)	Performs Sin operation on variable 't'.
cos(t)	Performs cosine operation on variable 't'.
tan(t)	Performs tangent operation on variable 't'.
asin(t)	Performs arc sin operation on variable 't' or Inverse of the sin function.
acos(t)	Performs arc cosine operation on variable 't' or Inverse of the cos function.
atan(t)	Performs arc tangent operation on variable 't' or Inverse of the tan function.

Exponential functions

Exponetial Symbol	Operation
exp(t)	Performs exponential operation on variable 't'.

Square functions

Symbol	Operation
^	Power or Square
sqrt(t)	Performs square root operation on variable 't'.

Logarithm functions

Symbol	Operation
log(t)	Performs a natural logarithmic operation on variable 't'.
log10(t)	Performs a common logarithmic operation on variable 't'.

Maximum & Minimum functions

Symbol	Operation
min(t)	Finds minimum value from array 't'.
max(t)	Finds maximum value from array 't'.

Writing Expressions with MATLAB

1- Consider the mathematical expression a(b + c) which you might read as “a times b plus c” ,the expression would appear to translate to the MATLAB command as a*b+c. Hopefully you can see that this actually is equal to ab+c. The correct MATLAB command for a(b + c) is a*(b+c).

2-The brackets have been used to force MATLAB to first evaluate the expression (b+c) and then to multiply the result by a. We should avoid falling into the trap of assuming that commands are performed from left to-right, for instance c+a*b is equal to c + ab not (c + a)b as if the addition was performed first.

3-Constants

Constant properties are used to define constant values that can be accessed by name. MATLAB comes with a set of predefined constant values. The following are some of the most common values:

Symbol / Constant	Associated Constant value
pi	The ‘π’ number = 3.14159...
i, j	The imaginary unit √-1
Inf	The infinity, ∞

4-Entering Matrices

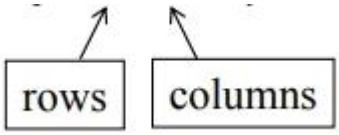
The definition of the Matrix is a two-dimensional array which consists of both the rows and columns. Matrices can be generated in several ways:

Vector : array with one dimension.

Matrix : array with more than one dimension

Size : array is specified by number of rows and number of columns .

For example : n x m array



In the MATLAB matrix, the rows and columns are created by using the commas (,) / line-spaces ( ) and semicolon (;) respectively.

Let's look at the general representation of matrix with the row and column.

Row Matrix is A = [x1, x2, x3, ... xn] or [x1 x2 x3 ..... xn]

This is (1 × n) vector i.e. a single row consisting the nth term elements.

Column Matrix is A = [x1; x2; x3; ..... xm]

This is (m × 1) vector i.e. a single row consisting the mth term elements.

Note: Row matrix and column matrix are nothing but the vectors.

- More explanation

$$a = \begin{bmatrix} 2 & 6 \\ 8 & 3 \\ 4 & 7 \end{bmatrix}$$ 3 x 2 matrix ⟶ 6 elements

$$b = [1 \quad 2 \quad 3 \quad 4]$$ 1 x 4 array ⟶ 4 elements, **row vector**

$$c = \begin{bmatrix} 1 \\ 4 \\ 6 \end{bmatrix}$$ 3 x 1 array ⟶ 3 elements, **column vector**

How to generate or create n ×m Matrix in MATLAB?

Solution: The 3×3 matrix must have 3 rows and 3 columns. These rows and columns are created with the help of space and semicolon.

For example: Matrix A is

```
>> A = [2 3 3; 1 2 8; 7 9 3]
A =
 2 3 3
 1 2 8
 7 9 3
```

3 x 3 matrix ⟶ 9 elements

Useful Matrix Generators

MATLAB provides four easy ways to generate certain simple matrices . These are:

zeros	a matrix filled with zeros
ones	a matrix filled with ones
rand	a matrix with uniformly distributed random elements
randn	a matrix with normally distributed random elements
eye	identity matrix

To generate matrices in MATLAB should be you give the functions the number of rows and columns .

For examples

```
>>a = zeros(2,3)
a =
0 0 0
0 0 0
>> b = ones(2,2)/2
b =
 0.5000 0.5000
0.5000 0.5000
>> u = rand(1,5)
u =
0.9218  0.7382  0.1763  0.4057  0.9355
>> n = randn(5,5)

n =
-0.4326  1.1909  -0.1867   0.1139  0.2944
-1.6656   1.1892   0.7258  1.0668   -1.3362
0.1253   -0.0376   -0.5883  0.0593  0.7143
0.2877  0.3273   2.1832     -0.0956   1.6236
-1.1465  0.1746   -0.1364   -0.8323   -0.6918
>>eye(3)
and=
```

1 0 0

0 1 0

0 0 1

Subscripting

Individual elements in a matrix are denoted by a row index and a column index.To pick out the third element of the vector u type:

```
>>u(3)
ans =
0.1763
>>u([1 2 3])
ans =
 0.9218  0.7382  0.1763
>>u(1:3)
ans =
 0.9218   0.7382   0.1763
>>a=[1 2 3;4 5 6]
a =
 1 2 3
 4 5 6
>>a(2,3)
ans = 6
>>a(1:2;3)
ans = 3
 6
```

Colon Operator

The symbol colon (:) is used as an operator in MATLAB programming and is a commonly used operator. This operator is used to construct a vector that is defined by a simple expression, iterate over or subscribe to an array, or access a set of elements of an existing vector.

```
>>a(2,☺
ans =
 2 5 6
>> a(:,3)
ans =
3
6
9
>>a(4)
ans=
 2
```

The colon symbol can be used as a single index to a matrix .

```
>>a(:)
ans =1
      4
      7
      2
      5
      8
```

3

6

9

End as a subscript

To access the last element of a matrix along a given dimension, use end as a subscript .

For example:

```
>>q = 4 : 10
q=
 4 5 6 7 8 9 10
>>q(end)
ans=
10
>> q(end-4:end)
ans=
 6 7 8 9 10
```

End as a subscript

```
>>q = [ 7 8 9 10; 6 1 2 20; 5 4 3 30]

q=

 7 8 9 10

6 1 2 20

5 4 3 30

>>q(end , end)

ans=

30

>>q(2,end-1:end)

ans=

   2 20

>> q(end-2:end,end-1:end)

ans=

 9 10
 2 20

 3 30

>> q(end-1,: )

ans=

 6  1  2  20
```

Transpose

Transpose method is a mathematical operation where:

1-The elements of the row are converted into the elements of the column. (or)

2-The elements of the column are converted into the elements of the row.

The transpose of the Matrix is represented by **an apostrophe** or **single quote** (‘) or **power of** T like as [ ]' 0r [ ]T

Note: The representation of the transpose of the Matrix is the same as the representation of the transpose of the vector.

How to determine the transpose of the given below matrix D?

For example:

```
>>D = [2 0 1; 9 6 8; 4 1 1]

D =

2 0 1

9 6 8
4 1 1
```

By using the transpose method,

```
>>D '

D=

 2 9 4

 0 6 1

1 8 1
```

Deleting Rows or Columns

To get rid of a row or column set it equal to the empty matrix.[]

```
>> a = [1 2 3;4 5 6;7 8 9]

a=

 1 2 3

4 5 6

7 8 9

>>a(:,2)=[]

a=

 1 3

4 6

7 9
```