Signals & Systems Laboratory

CSE- 301L

Lab # 01

OBJECTIVES OF THE LAB

Matlab will be used extensively in all the succeeding labs. The goal of this first lab is to gain familiarity with Matlab and build some basic skills in the Matlab language. Some specific topics covered in this lab are:

- *Introduction to Matlab*
- *Matlab Environment*
- *Matlab Help*
- *Variable arithmetic*
- *Built in Mathematical Functions*
- *Input and display*
- *Timing functions*
- *Introduction to M-files*

1.1 WHAT IS MATLAB?

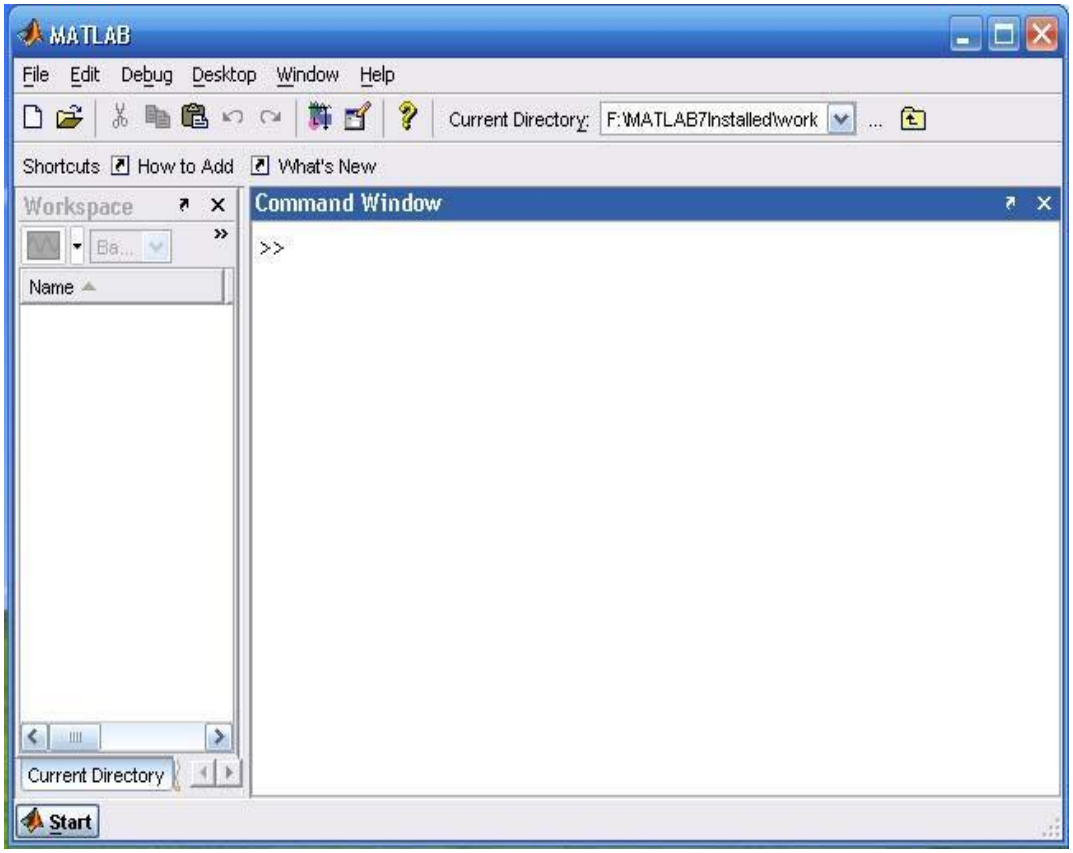
MATLAB is a commercial "**MAT**rix **LAB**oratory" package, by **MathWorks**, which operates as an interactive programming environment with graphical output. The MATLAB programming language is exceptionally straightforward since almost every data object is assumed to be an array. Hence, for some areas of engineering MATLAB is displacing popular programming languages, due to its interactive interface, reliable algorithmic foundation, fully extensible environment, and computational speed.

1.2 ENTERING AND RUNNING MATLAB

Double click on the MATLAB icon to launch and a command window will appear with the prompt:

>>

You are now in MATLAB. From this point on, individual MATLAB commands may be given at the program prompt. They will be processed when you hit the <ENTER> key. The following figure shows the screenshot of matlab.



1.3 LEAVING MATLAB

A MATLAB session may be terminated by simply typing

```
>> quit
```

or by typing

```
>> exit
```

at the MATLAB prompt.

1.4 MATLAB HELP

Online help is available from the MATLAB prompt, both generally (listing all available commands). >> help

```
[a long list of help topics follows]
```

and for specific commands:

```
>> help command_name
```

If you want to search for all the commands related to some particular functionality, use the keyword **lookfor** followed by a keyword that explains the functionality.

```
>> lookfor convolution
```

will return a number of commands that perform convolution related tasks.

1.5 VARIABLES

MATLAB has built-in variables like pi, eps, and ans. You can learn their values from the MATLAB interpreter.

```
>> eps
eps =
    2.2204e-16
>> pi
ans =
    3.1416
```

1.5.1 Variable Assignment

The equality sign is used to assign values to variables:

```
>> x = 3
x =
     3
>> y = x^2
y =
     9
```

Variables in MATLAB are case sensitive. Hence, the variables "x" and "y" are distinct from "X" and "Y" (at this point, the latter are in fact, undefined).

Output can be suppressed by appending a semicolon to the command lines.

```
>> x = 3;
>> y = x^2;
>> y
y =
     9
```

1.5.2 Active Variables

At any time you want to know the active variables you can use who:

```
>> who
Your variables
are: ans x y
```

1.5.3 Removing a Variable

To remove a variable, try this: `>> clear x`

To remove all the variables from workspace, use clear `>> clear`

1.5.4 Saving and Restoring Variables

To save the value of the variable "x" to a plain text file named "x.value"

use >> save x.value x -ascii

To save all variables in a file named mysession.mat, in reloadable format, use

>> save mysession

To restore the session, use

>> load mysession

1.6 VARIABLE ARITHMETIC

MATLAB uses some fairly standard notation. More than one command may be entered on a single line, if they are separated by commas.

```
>> 2+3;
>> 3*4, 4^2;
```

Powers are performed before division and multiplication, which are done before subtraction and addition. For example

```
>> 2+3*4^2;
```

generates ans = 50. That is:

```
2+3*4^2 ==> 2 + 3*4^2 <== exponent has the highest
precedence ==> 2 + 3*16 <== then multiplication operator
==> 2 + 48 <== then addition operator
==> 50
```

1.6.1 Double Precision Arithmetic

All arithmetic is done to double precision, which for 32-bit machines means to about 16 decimal digits of accuracy. Normally the results will be displayed in a shorter form.

```
>> a = sqrt(2)
a =
    1.4142
>> format long, b=sqrt(2)
b =
```

1.41421356237310

>> format short

1.6.2 Command-Line Editing

The arrow keys allow "command-line editing," which cuts down on the amount of typing required, and allows easy error correction. Press the "up" arrow, and add "/2." What will this produce?

>> 2+3*4^2/2

Parentheses may be used to group terms, or to make them more readable. For

example: >> (2 + 3*4^2)/2

generates ans = 25.

1.6.3 Built-In Mathematical Functions

MATLAB has a platter of built-in functions for mathematical and scientific computations. Here is a summary of relevant functions.

Function	Meaning	Example
sin	sine	sin(pi) = 0.0
cos	cosine	cos(pi) = 1.0
tan	tangent	tan(pi/4) = 1.0
asin	arcsine	asin(pi/2)= 1.0
acos	arccosine	acos(pi/2)= 0.0
atan	arctangent	atan(pi/4)= 1.0
exp	exponential	exp(1.0) = 2.7183
log	natural logarithm	log(2.7183) = 1.0
log10	logarithm base 10	log10(100.0) = 2.0

The arguments to trigonometric functions are given in radians.

Example: Let's verify that

sin(x)^2 + cos(x)^2 = 1.0

for arbitrary x. The MATLAB code is:

>> x = pi/3;

>> sin(x)^2 + cos(x)^2 - 1.0

ans =

0

1.7 TIMING COMMANDS

Timing functions may be required to determine the time taken by a command to execute or an operation to complete. Several commands are available to accomplish it:

1.7.1 Clock

CLOCK returns Current date and time as date vector. CLOCK returns a six element date vector containing the current time and date in decimal form:

CLOCK = [year month day hour minute seconds]

The first five elements are integers. The second's element is accurate to several digits beyond the decimal point. FIX(CLOCK) rounds to integer display format.

1.7.2 Etime

ETIME Elapsed time.
ETIME(T1,T0) returns the time in seconds that has elapsed between vectors T1 and T0. The two vectors must be six elements long, in the format returned by CLOCK:

T = [Year Month Day Hour Minute Second]

Time differences over many orders of magnitude are computed accurately. The result can be thousands of seconds if T1 and T0 differ in their first five components or small fractions of seconds if the first five components are equal.

```
t0 = clock;
operation
etime(clock,t0)
```

1.7.3 Tic Toc

TIC Start a stopwatch timer.

The sequence of commands

TIC, operation, TOC

prints the number of seconds required for the operation.

1.8 INPUT & DISPLAY

1.8.1 INPUT

INPUT prompts for user input.

```
R = INPUT('How many apples')
```

gives the user the prompt in the text string and then waits for input from the keyboard. The input can be any MATLAB expression, which is evaluated, using the variables in the current workspace, and the result returned in R. If the user presses the return key without entering anything, INPUT returns an empty matrix.

Example: Entering a single variable

```
>> x=input('Enter a variable: ')

Enter a variable: 4

x =

   4
```

Example: Entering a vector

A vector is entered by specifying [] and elements are inserted inside these brackets, separated by space.

```
>> x=input('Enter a vector: ')

Enter a vector: [3 4 1]

x =

   3   4   1
```

Example: A \n entered after the string results in starting a new line.

```
>> x=input('Enter a value\n')

Enter a value

5

x =

 5
```

1.8.2 DISP

DISP(X) displays the value of variable X, without printing the variable name. In all other ways it's the same as leaving the semicolon off an expression.

DISP('string') is another variation of the same function that is used to display a string on the command prompt.

Example:

```
>> disp('I am using MATLAB')
```

I am using MATLAB

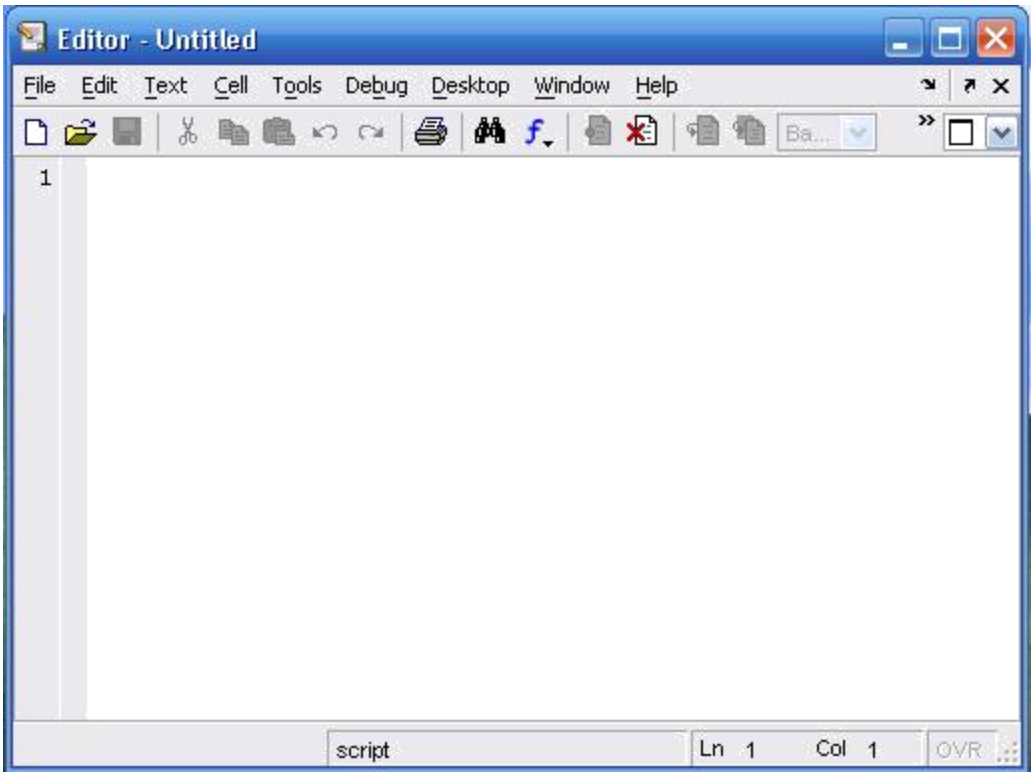
1.9 M-Files

Typing errors are time-consuming to fix if you are working in the command window because you need to retype all or part of the program. Even if you do not make any mistakes, all of your work may be lost if you inadvertently quit MATLAB. To preserve large sets of commands, you can store them in a special type of file called an ***M-file***. MATLAB supports two types of M-files: ***script*** and ***function M-files***. To hold a large collection of commands, we use a script M-file. The function M-file is discussed in coming lab. The script file has a `'.m'` extension and is referred to as an M-file (for example, `myfile.m` `myfuncion.m`, etc.). The commands in the script file can then be executed by typing the file name without its extension in the command window. Commands in a script utilize and modify the contents of the current workspace. It is possible to embed comments in a script file.

To make a script M-file, you need to open a file using the built-in MATLAB editor. There are two ways to accomplish it:

1. From file menu, click NEW
2. Type **edit** on command line

A new window appears like one shown in the figure below.



When you are finished with typing in this new window, click File->Save to save this file. The extension of this file be .m. In order to execute this program,

1. Write the name of file on command window (excluding the .m) or
2. Click Debug->Run

----------TASK 01----------

a. Matlab stores numeric data as double-precision floating point by default. To store data as an 8- bit integer, int8 (a conversion function) can be used. Type the sample code in matlab command window:

```
>> x = 26
>> whos
>> y = int8(x)
>> whos
```

What difference do you see? State your findings. (Also try uint16, uint32, uint64)

b. Take your name in command window e.g. name = 'Ali'. Convert it into 8-bit integer format using **int8** function.

----------TASK 02----------

Create an M-File to prove any five expressions from the following:

$\sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$ $\sin(\alpha-\beta) = \sin\alpha\cos\beta - \cos\alpha\sin\beta$ $\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$ $\cos(\alpha-\beta) = \cos\alpha\cos\beta + \sin\alpha\sin\beta$	$\tan(\alpha+\beta) = \frac{\tan\alpha+\tan\beta}{1-\tan\alpha\tan\beta}$ $\tan(\alpha-\beta) = \frac{\tan\alpha-\tan\beta}{1+\tan\alpha\tan\beta}$
$\sin\alpha+\sin\beta = 2\sin\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)$ $\sin\alpha-\sin\beta = 2\cos\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)$	$\cos\alpha+\cos\beta = 2\cos\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)$ $\cos\alpha-\cos\beta = -2\sin\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)$
$\sin\alpha\sin\beta = \frac{1}{2}[\cos(\alpha-\beta)-\cos(\alpha+\beta)]$ $\cos\alpha\cos\beta = \frac{1}{2}[\cos(\alpha-\beta)+\cos(\alpha+\beta)]$	$\sin\alpha\cos\beta = \frac{1}{2}[\sin(\alpha+\beta)+\sin(\alpha-\beta)]$ $\cos\alpha\sin\beta = \frac{1}{2}[\sin(\alpha+\beta)-\sin(\alpha-\beta)]$

Use ***etime*** or ***tic toc*** functions to evaluate time taken for solving each of the five chosen expressions.

---------- TASK 03 ----------

Write a CGPA Calculator program using M-File: Design a transcript for your second semester result i.e. take grade points and credit hours of each subject as input from user and store in variables. Take product of each subject grade points with its credit hours and divide by total credit hours in order to evaluate CGPA. Show the results in the form of well designed transcript using ***disp*** and ***input*** commands. Use the following table to display equivalent grades for each

grade point:

Grade	Grade Point
A	4.00
A-	3.67
B+	3.33
B	3.00
B-	2.67
C+	2.33
C	2.00
C-	1.67
D+	1.33
D	1.00
F	0

---------- TASK 04 ----------

Write a simple code to swap the values of two variables of double type using M-file. Create the logic in such a way that no third variable is used. Show the ***etime*** for this code.
