EE2021: Computer Tools for Electrical Engineering

Adopted from materials of David Houcqye, Northwestern U.

Lab 1

Fall 2018

1 Introduction

In this lab session, you will get started quickly in MATLAB and will be familiar with the basics of MATLAB.

2 A Minimum MATLAB Session

A minimum MATLAB session consists of four steps:

- Logging on
- Invoking MATLAB
- Doing a few simple calculations
- Quitting MATLAB

2.1 Starting MATLAB

Turn on your computer and log in to the operating system. Thereafter, you can execute MATLAB by double-clicking on the MATLAB shortcut icon on desktop. Once you start MATLAB, a new window which consists of the following windows appears

- The command window
- The command history window
- The workspace window
- The current folder window
- The editor window
- The help browser
- The start button

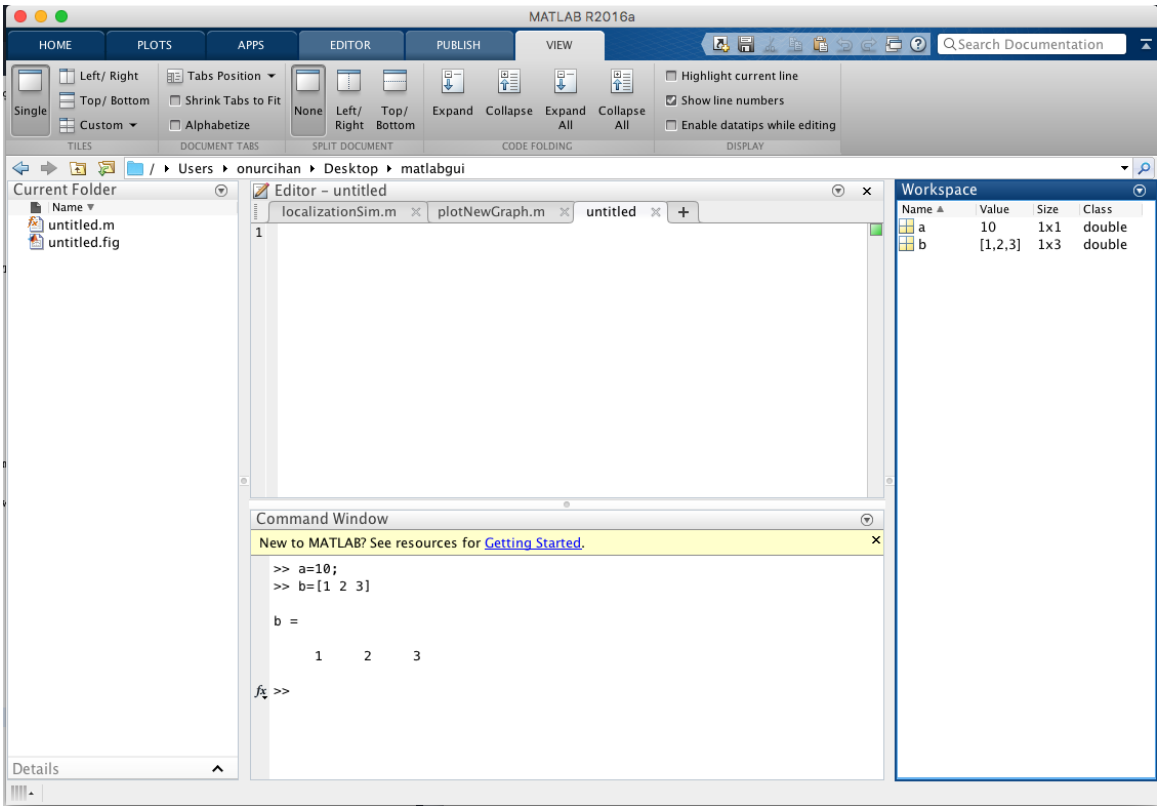

Figure 1: The graphical interface to the MATLAB application

When MATLAB is started for the first time, the screen looks like the one that shown in Figure 1. The illustration also shows a comfortable configuration of the MATLAB desktop. You can always customize the arrangement of tools and documents to suit your needs.

In this step, we are interested in doing some simple calculations. It is assumed that you have sufficient understanding of your computer under which MATLAB is being run.

2.2 Using MATLAB as a calculator

As an example of a simple interactive calculation, just type the expression you want to evaluate. For instance, suppose that you want to calculate the expression $2 - 3 \times 4$. You type it a the prompt command (>>) as follows,

```
>> 2-3*4
ans =
   -10
```

You will have noticed that if you do not specify an output variable, MATLAB uses a default variable `ans`, short for `answer`, to store the results of the current calculation. 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= 2-3*4
x =
   -10
```

will result in `x` being given the value $2 - 3 \times 4 = -10$. This variable name can always be used to refer to the results of the previous computations. Therefore, computing $2x$ will result in

```
>> 2*x
ans =
   -20
```

Before we conclude the minimum session, Table 2 gives the partial list of arithmetic operators.

SYMBOL	OPERATION	EXAMPLE
$+$	Addition	$4 + 5$
$-$	Subtraction	$4 - 5$
$*$	Multiplication	$4 * 5$
$/$	Division	$4/5$

Table 1: Partial list of basic arithmetic operators

2.3 Quitting MATLAB

In order to end your MATLAB session, type quit in the command window or simply click X in the title-bar of MATLAB.

3 Getting Started

After learning the minimum MATLAB session, we will now learn to use some additional operations.

3.1 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 instance,

```
>> x = expression
```

where `expression` is a combination of numerical values, mathematical operators, variables and function calls. In other words, `expression` can involve:

- manual entry
- built-in functions
- user-defined functions

3.2 Overwriting variable

Once a variable has been created, it can be reassigned. In addition, if you do not wish to see the intermediate results, you can suppress the numerical output by putting a semicolon (`;`) at the end of the line. Then the sequence of commands looks like

```
>> t = 5;
>> t = t + 1;
t   =
    6
```

3.3 Error messages

If you enter an expression incorrectly, MATLAB will return an error message. For instance, in the following example, we left out the multiplication sign `*`, in the following expression

```
>> x = 10;
>> 5x
    |
Error: Unexpected MATLAB expression.

Did you mean:
>> 5*x
```

3.4 Making corrections

In order to make corrections, we can re-type the expressions. However, if the expression is lengthy, we can make more mistakes by typing a second time. A previously typed command can be recalled with the up-arrow key. When the command is displayed at the command prompt, it can be modified if needed and executed.

3.5 Controlling the hierarchy of operations or precedence

Let us consider the previous arithmetic operation, but now we will include parentheses. For instance,

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

and

```
>> (1+2)*3
ans =
    9
```

By adding parenthesis, these two expressions give different results: 9 and 7.

The order in which MATLAB performs arithmetic operations is exactly that is being taught in high school algebra courses. *Exponentiations* are done first, followed by *multiplications* and *divisions*, and finally by *additions* and *subtractions*. However, the standard order of precedence of arithmetic operations can be changed by inserting *parentheses*. For instance, the result of $1 + 2 \times 3$ is quite different than the similar expression with parentheses $(1+2) \times 3$. The results are 7 and 9 respectively.

Parentheses can always be used to overrule *priority*, and their use is recommended in some complex expressions to avoid ambiguity.

Thus, in order to make the evaluation of expressions unambiguous, MATLAB has established a series of rules. The order in which the arithmetic operations are evaluated is given in Table **??**.

PRECEDENCE	MATHEMATICAL OPERATIONS
First	The contents of all parentheses are evaluated first, starting from the innermost parentheses and working outward
Second	All exponentials are evaluated, starting from the left to the right
Third	All multiplications and divisions are evaluated, starting from the left to the right
Fourth	All additions and subtractions are evaluated, starting from the left to the right

Table 2: Hierarchy of arithmetic operators

MATLAB arithmetic operators obey the same *precedence* rules as those in most computer programs. For operators of *equal* precedence, evaluation is from the *left* to the *right*. Now, consider another example

$$\frac{1}{2+3^2}+\frac{4}{5}\times\frac{6}{7}$$

In MATLAB, it becomes

```
>> 1 /(2+3^2)+4/5*6/7
ans =
    0.7766
```

or, if parentheses are missing,

```
>> 1 /2+3^2+4/5*6/7
ans =
    10.1857
```

Therefore, we want to emphasize the importance of precedence rule in order to avoid ambiguity.

3.6 Controlling the appearance of floating point number

MATLAB by default displays only 4 decimals in the result of the calculations, for instance -163.6667, as shown in above examples. However, MATLAB does numerical calculations in *double* precision, which is 15 digits. The command `format` controls how the results of computations are displayed. Here are some examples of the different formats together with the resulting outputs.

```
>> format short
>> x = -491/3
x =
-163.6667
```

If we want to see all 15 digits, we use the command `format long`

```
>> format long
>> x = -491/3
x =
-1.636666666666667e+02
```

In order to return to the standard format, enter `format short`, or simply `format`.

Note 1. *Up to now, we have let MATLAB repeat everything that we enter at the prompt (>>). Sometimes this is not quite useful, in particular when the output is lengthy. In order to prevent MATLAB from echoing what we type, simply enter a semicolon (;) at the end of the command. For instance,*

```
>> x = -491/3;
```

and then ask about the value of ***x*** *by typing,*

```
>> x
x   =
    -163.6667
```

3.7 Managing the workspace

The contents of the workspace persist between the executions of separate commands. Therefore, it is possible for the results of one problem to have an effect on the next one. To avoid this possibility, it is a good idea to issue a clear command at the start of each new independent calculation.

```
>> clear
```

The command `clear` or `clear all` removes all variables from the workspace. This frees up system memory. In order to display a list of the variables currently in the memory, type

```
>> who
```

while, `whos` will give more details which include size, space allocation, and class of the variables.

3.8 Keeping track of your work session

It is possible to keep track of everything done during a MATLAB session with the diary command.

```
>> diary
```

or give a name to a created file

```
>> diary FileName
```

where `FileName` could be any arbitrary name you choose.

The function `diary` is useful if you want to save a complete MATLAB session. They save all input and output as they appear in the MATLAB window. When you want to stop the recording, enter `diary off`. If you want to start recording again, enter `diary on`. The file that is created is a simple text file. It can be opened by an editor or a word processing program and exited to remove extraneous material, or to add your comments. You can use the function type to view the diary file or you can edit in a text editor or print. This command is useful, for instance, in the process of preparing a homework or lab submission.

3.9 Entering multiple statements per line

It is possible to enter multiple statements per line. Use commas (`,`) or semicolons (`;`) to enter more than one statement at once. Commas (`,`) allow multiple statements per line without suppressing output.

```
>> a = 7; b=cos(a), c =cosh(a)
b  =
   0.6570
c  =
   548.3170
```

3.10 Miscellaneous commands

Here are few additional useful commands:

- To clear the command window, type `clc`
- To abort a MATLAB computation, type `ctrl-c`
- To continue a line, type `...`

3.11 Getting help

To view the online documentation, select MATLAB Help from Help menu or MATLAB Help directly in the Command Window. The preferred method is to use the *Help Browser*. The Help Browser can be started by selecting the ? icon from the desktop toolbar. On the other hand, information about any command is available by typing

```
>> help Command
```

Another way to get help is to use the `lookfor` command. The `lookfor` command differs from the help command. The `help` command searches for an exact function name match, while the `lookfor` command searches the quick summary information in each function for a match. For example, suppose that we were looking for a function to take *the inverse of a matrix*. Since MATLAB does not have a function named `inverse`, the command help `inverse` will produce nothing. On the other hand, the command `lookfor inverse` will produce detailed information, which includes the function of interest, `inv`.

```
>> lookfor inverse
```

Note 2. *At this particular time of our study, it is important to emphasize one main point. Because MATLAB is a huge program; it is impossible to cover* all the details *of each function one by one. However, we will give you information how to get help. Here are some examples:*

- *Use* `help` *to request info on a specific function*

```
>> help sqrt
```

- *Use* `doc` *function to request documentation of a specific function*

```
>> doc plot
```

- *Use* `lookfor` *to find functions by keywords. The general form is*

```
>> lookfor FunctionName
```

Exercises

Complete the Exercies 1.1-1.8 in your textbook.