

Practical work 01: Presentation of Computing programming environment: MATLAB

1. Computing programming environment.....	2
2. MATLAB Programming Environment.....	2
3. Scilab Programming Environment.....	2
4. Alternatives numerically oriented programming language.....	3
5. Getting started with MATLAB and Scilab	3
6. Basic arithmetic	4
7. Priority of commands.....	5
8. Variables	6
9. Some reserved keywords and commands	7
10. Reusing previous commands	8
11. Clearing screen and variables and quitting MATLAB	9

Practical work 01: Presentation of Computing programming environment: MATLAB

In this practical work, we introduce one of the most used computing programming environment, MATLAB.

1. Computing programming environment

A "computing programming environment" can be defined as software and tools used to write, test, and debug computer programs. It offers an integrated set of tools and features that make the software development process more efficient and manageable. Several computing programming environments exist, and the choice of environment often depends on the programming language and the specific needs of the project.

2. MATLAB Programming Environment

MATLAB (an abbreviation of "MATrix LABoratory") is an interactive computing environment developed by **MathWorks**, which provides a technical computing environment designed to enable numerical computation and data visualization. MATLAB can be used to solve problems ranging from the very simple to the sophisticated and complex problems. MATLAB has proven its ability to help solve problems in many fields such as applied mathematics, physics, chemistry, engineering and any application dealing with complex numerical calculations.

In addition, MATLAB has optional toolbox allowing access to symbolic computing abilities and an additional package, Simulink, adds graphical multi domain simulation and for dynamic and embedded systems.

MATLAB was implemented by mathematician and computer programmer **Cleve Moler** in 1970 and was first released as a commercial product in 1984 at the Automatic Control Conference in Las Vegas.

3. Scilab Programming Environment

Scilab is a free, open-source and high-level, numerically oriented programming language. Scilab can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations and numerical optimization. The Scilab language provides an interpreted programming environment, with matrices as the main data type.

Scilab also includes a free open source package called **Xcos** for modeling and simulation of dynamical systems, including both continuous and discrete sub-systems. Xcos is equivalent

to Simulink package of MATLAB. As the syntax of Scilab is similar to MATLAB, it includes a translator for supporting the conversion of code from MATLAB to Scilab.

4. Alternatives numerically oriented programming language

MATLAB and Scilab have some alternative programming languages, which are showcased in what follows.

Commercials Alternatives

AMESim, GAUSS, IDL, Maple, Mathcad, Mathematica, OxMetrics, PyIMSL Studio, SAS/IML, Stata (Mata), Sysquake

Open alternatives

FreeMat, JMathLib, R, SageMath, SciPy

5. Getting started with MATLAB and Scilab

When MATLAB or Scilab software are started, a window opens in which the main part is the Command Window where the user can type their commands which are interpreted by the software to provide directly the solution.

The main parts of MATLAB and Scilab graphical users are indicated in Figure 1.

- **Current folder (File browser):** This section in the MATLAB desktop environment displays the current, the saved and the performed working directory;
- **Workspace:** This part refers to the environment where we create, store, and manage variables and data during MATLAB session. The Workspace provides a visual representation of the variables we have defined and their values;
- **Command window:** MATLAB's Command Window is where we can enter commands and see their results. It is the primary interface for interacting with MATLAB;
- **Command history:** It is a feature that allows us to view and access the history of commands we have entered during our current session.

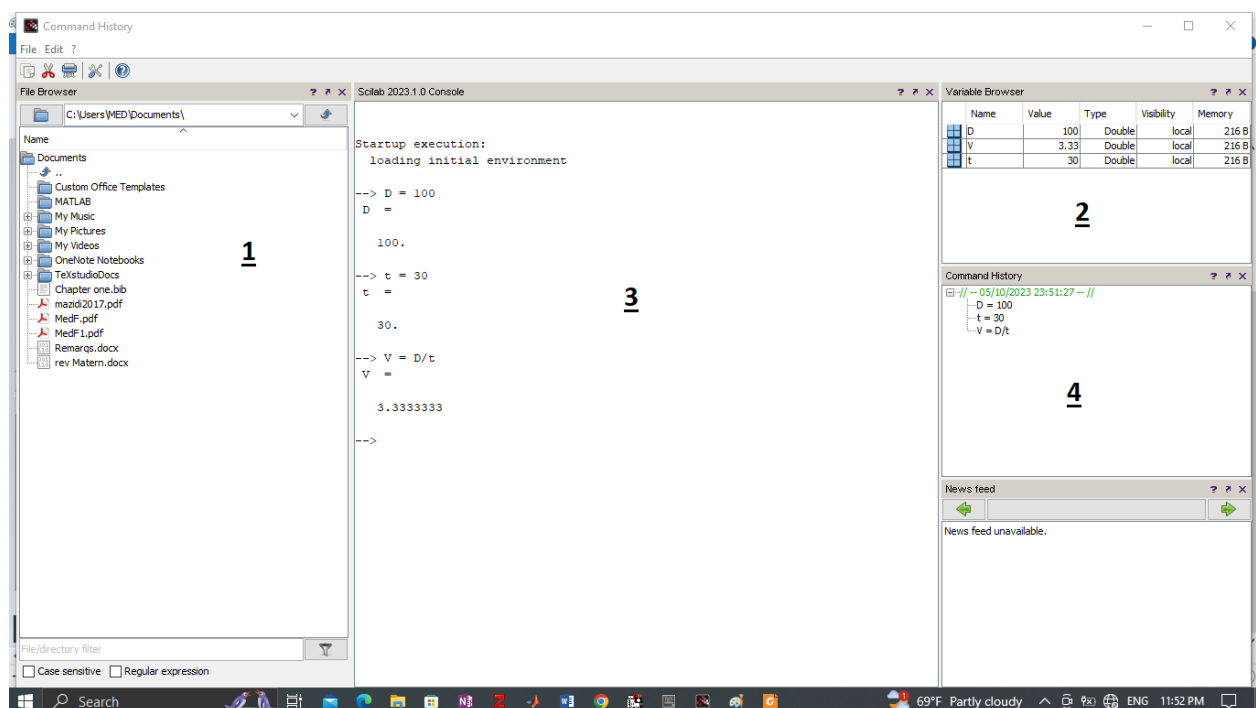
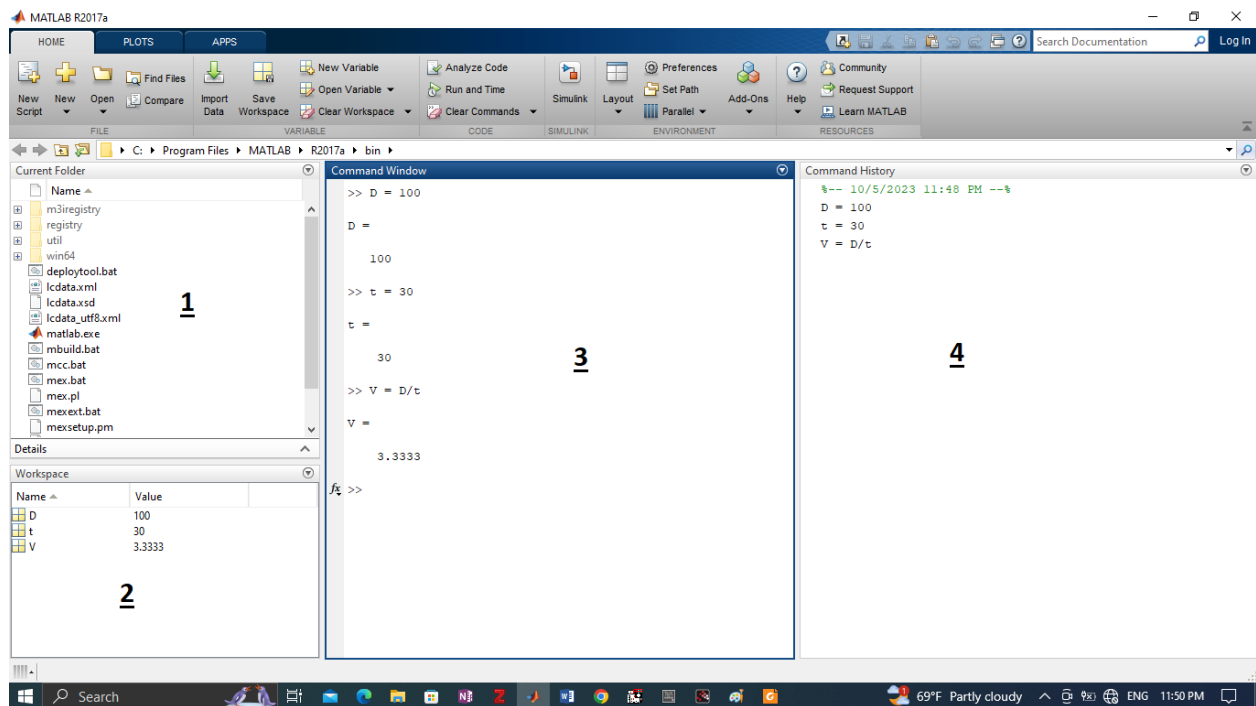
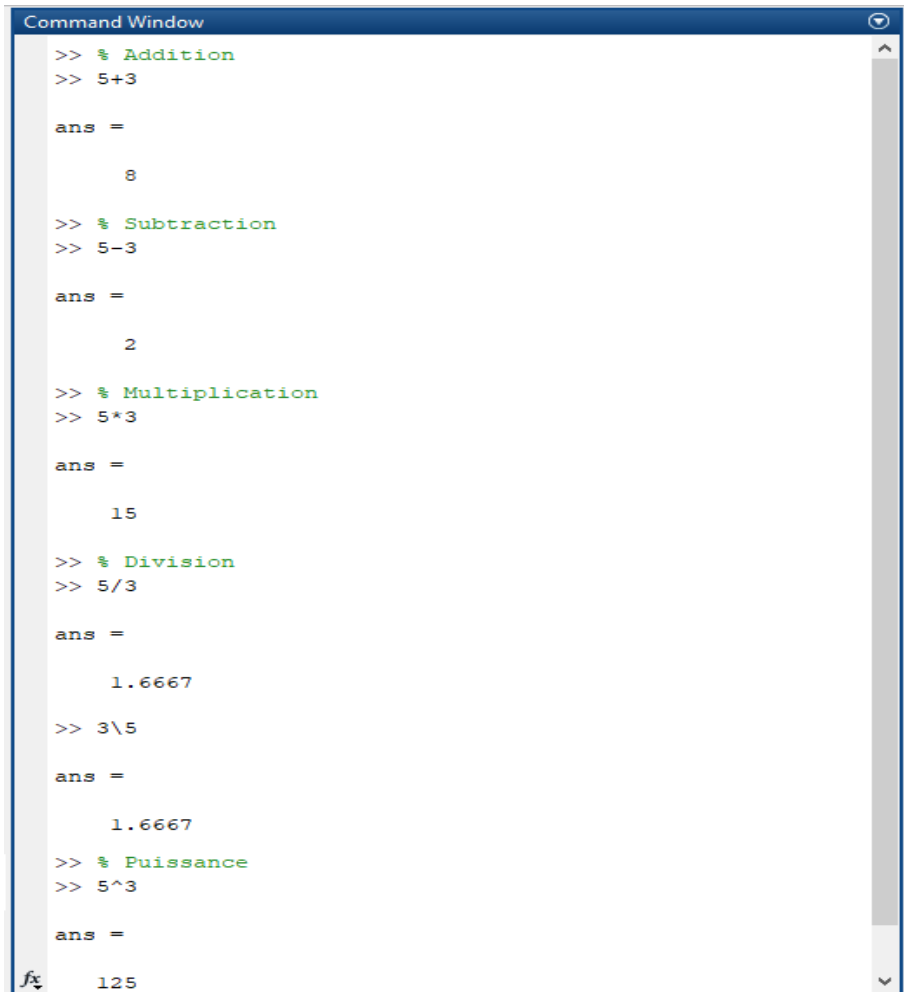


Figure.1 User interface Up: MATLAB desktop Down: Scilab desktop. 1. Current folder (File browser) 2. Work space 3. Command window 4. Command history.

6. Basic arithmetic

Matlab uses the standard arithmetic operators, addition, subtraction, multiplication, division and power as indicated in Figure.2.



```
Command Window
>> % Addition
>> 5+3

ans =

     8

>> % Subtraction
>> 5-3

ans =

     2

>> % Multiplication
>> 5*3

ans =

    15

>> % Division
>> 5/3

ans =

    1.6667

>> 3\5

ans =

    1.6667

>> % Puissance
>> 5^3

ans =

    125
```

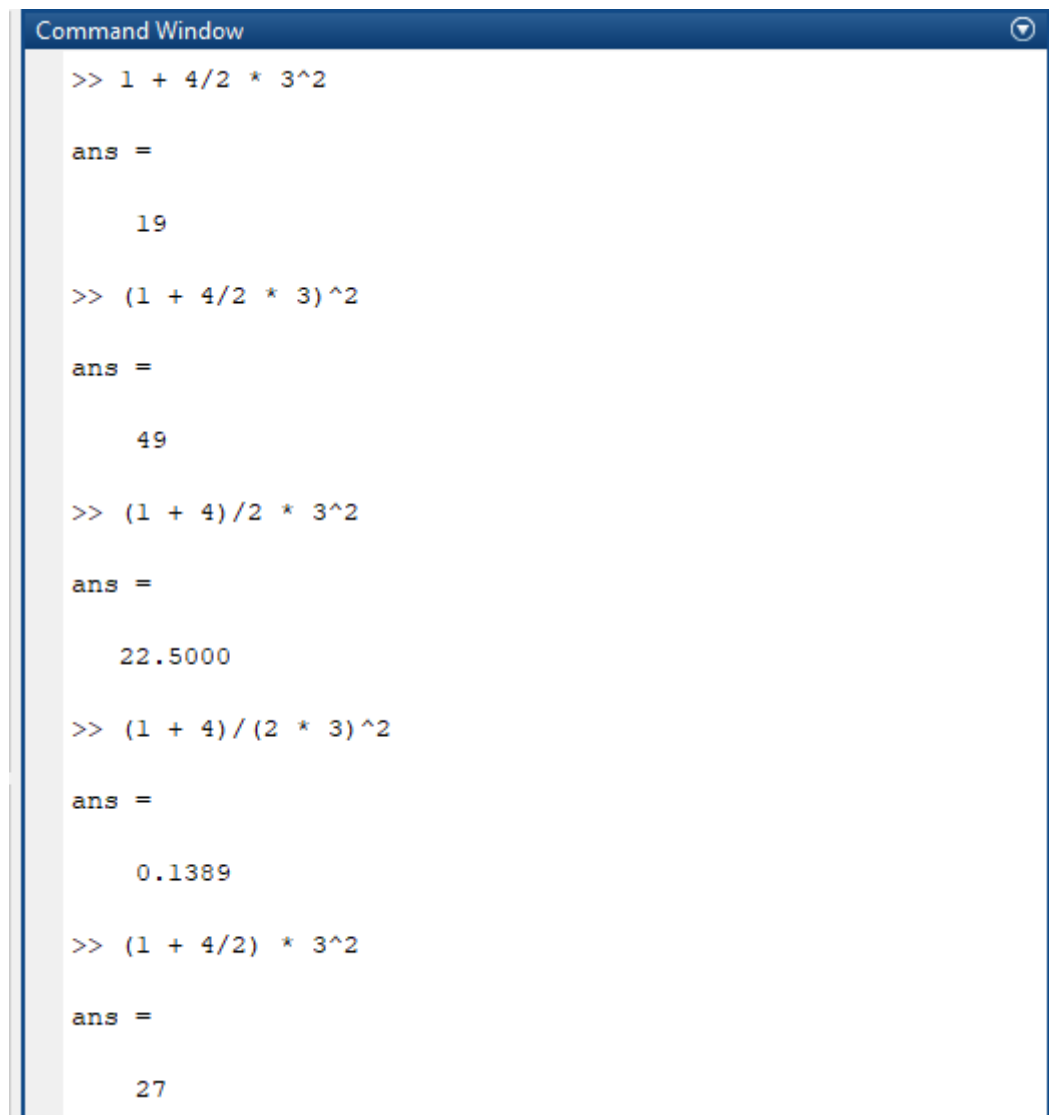
Figure.2 Arithmetic operators.

7. Priority of commands

The arithmetic operators have the following priority:

1. Brackets first;
2. Power next;
3. Multiplication and division next. Left to right in case of competition;
4. Addition and subtraction next. Left to right in case of competition.

Figure.3 shows an example of such priority.



```
Command Window

>> 1 + 4/2 * 3^2

ans =

    19

>> (1 + 4/2 * 3)^2

ans =

    49

>> (1 + 4)/2 * 3^2

ans =

   22.5000

>> (1 + 4)/(2 * 3)^2

ans =

    0.1389

>> (1 + 4/2) * 3^2

ans =

    27
```

Figure.3 Arithmetic operators priority.

8. Variables

Variables in MATLAB are defined by an identifier name, which must respect the following rules:

1. The variable name must begin with a letter and can contain letters, digits and underscore character (_);
2. MATLAB is case sensitive. So, variable named `Var_1` is different to variable named `var_1`;
3. MATLAB has reserved keywords, which cannot be used as identifier name (`sqrt`, `exp`, ...)

Figure.4 shows some operations using variables.

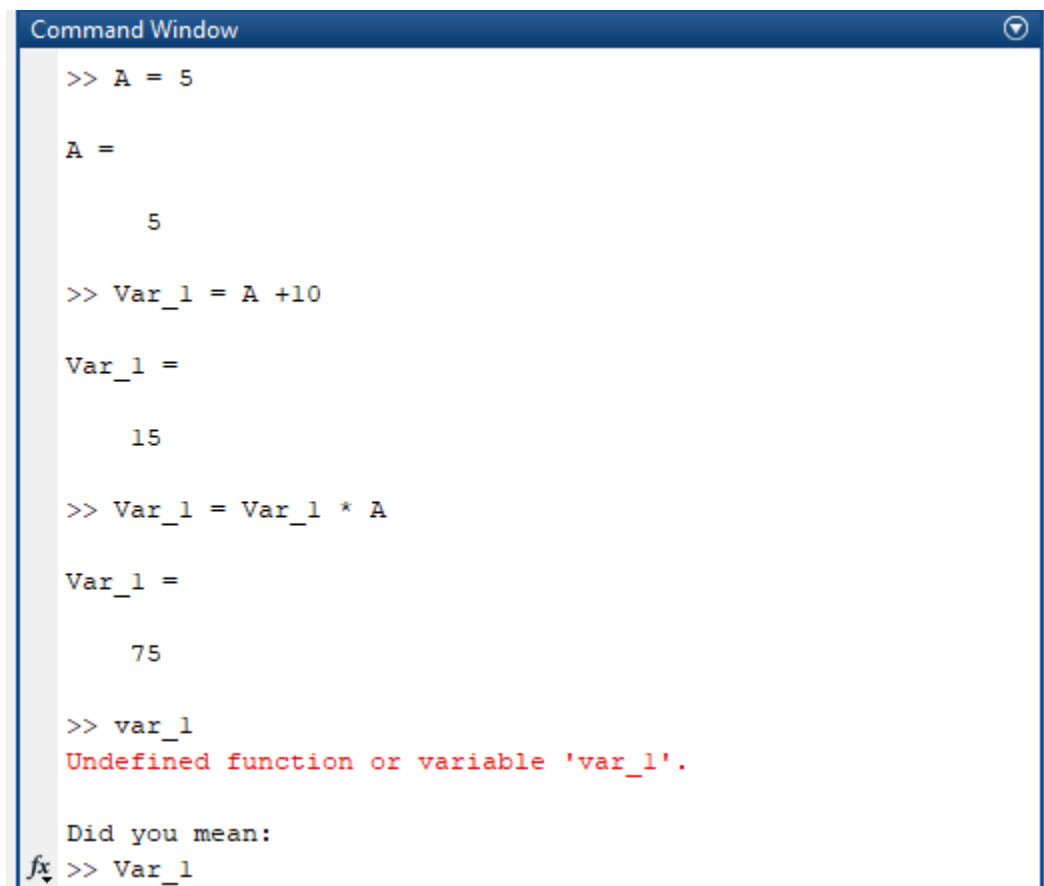
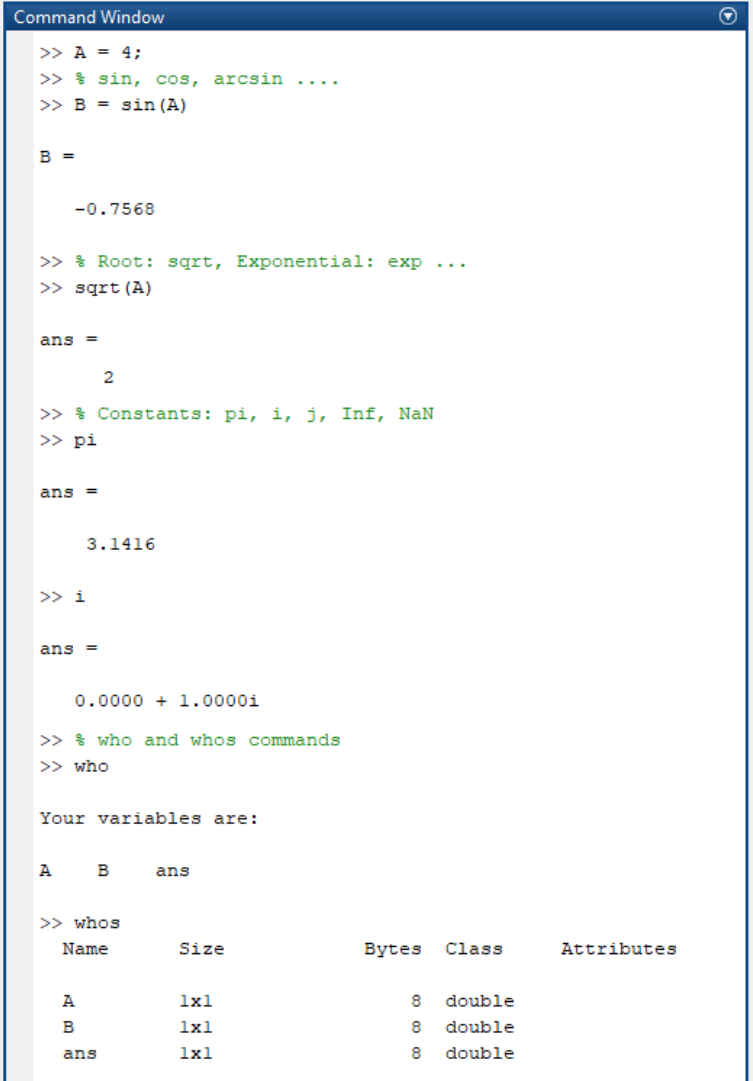
A screenshot of the MATLAB Command Window. The window has a dark blue title bar with the text 'Command Window' and a small circular icon on the right. The main area is white with black text. The commands and their outputs are as follows:
1. Command: >> A = 5
Output: A =
5
2. Command: >> Var_1 = A +10
Output: Var_1 =
15
3. Command: >> Var_1 = Var_1 * A
Output: Var_1 =
75
4. Command: >> var_1
Output: Undefined function or variable 'var_1'.
5. Prompt: Did you mean:
6. Command: fx >> Var_1 (where 'fx' is a cursor icon)
The text is monospaced and the output is indented relative to the command line.

Figure.4 Some operations using variables.

9. Some reserved keywords and commands

MATLAB has a list of reserved words and commands. Some of such keywords are indicated in Figure.5.



```

Command Window

>> A = 4;
>> % sin, cos, arcsin ....
>> B = sin(A)

B =

    -0.7568

>> % Root: sqrt, Exponential: exp ...
>> sqrt(A)

ans =

     2

>> % Constants: pi, i, j, Inf, NaN
>> pi

ans =

    3.1416

>> i

ans =

    0.0000 + 1.0000i

>> % who and whos commands
>> who

Your variables are:

A      B      ans

>> whos

  Name      Size      Bytes  Class  Attributes

  A         1x1         8  double

  B         1x1         8  double

  ans        1x1         8  double

```

Figure.5 Some MATLAB reserved keywords.

10. Reusing previous commands

To rerun previous commands do one of the following:

- Press the up arrow key (↑) until the command you want appears at the prompt, and then press **Enter**
- Double-click an entry or entries in the Command History window, or select an entry and press **Enter**.

To extend the selection to include multiple commands, press **Shift+↑**.

11. Clearing screen and variables and quitting MATLAB

`clc` Clear command window.

`clear` removes all variables from the workspace.

`clear VARIABLES` does the same thing.

`clear GLOBAL` removes all global variables.

`clear ALL` removes all variables

`clear VAR1 VAR2 ...` clears the specified variables.