isec Engenharia
Department of Computer and Systems Engineering

Optimization and Decision Support Methodologies

Introduction to MATLAB

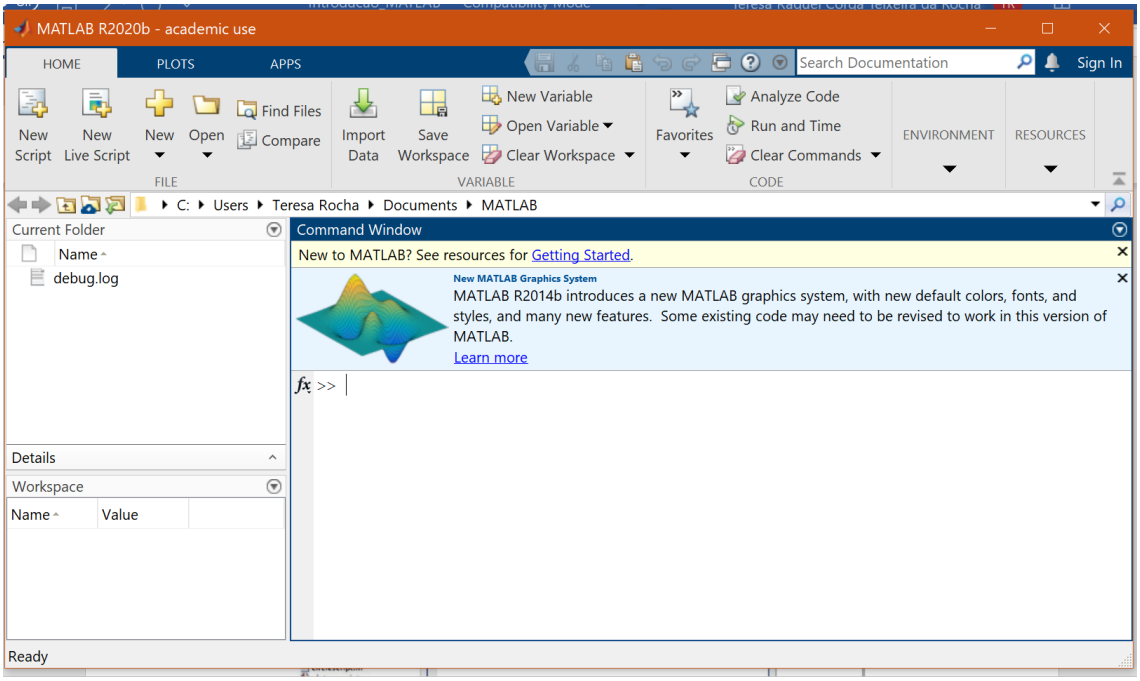
1. THE MATLAB

MATLAB, whose name derives from matrix laboratory, is an interactive system whose basic data element is an array that does not need dimensions.

It allows solving many computational problems of a technical nature, especially those involving vector and matrix formulations, in a fraction of the time it would take to write an equivalent program in a language such as, for example, C.

MATLAB consists of a package of functions fundamentally oriented to scientific calculation. It is presented in the form of a set of libraries (toolboxes), each comprising specific functions such as: Statistics and Machine Learning Toolbox, Optimization Toolbox, Deep Learning HDL Toolbox, …

When MATLAB is started, the desktop appears and the main window is the Command Window where you can see a prompt (>>). This window can be used interactively by entering any MATLAB command or expression after the prompt (see figure below).



It is also possible to write programs in MATLAB using files called M-Files (files with a .m extension), presented later in the document.

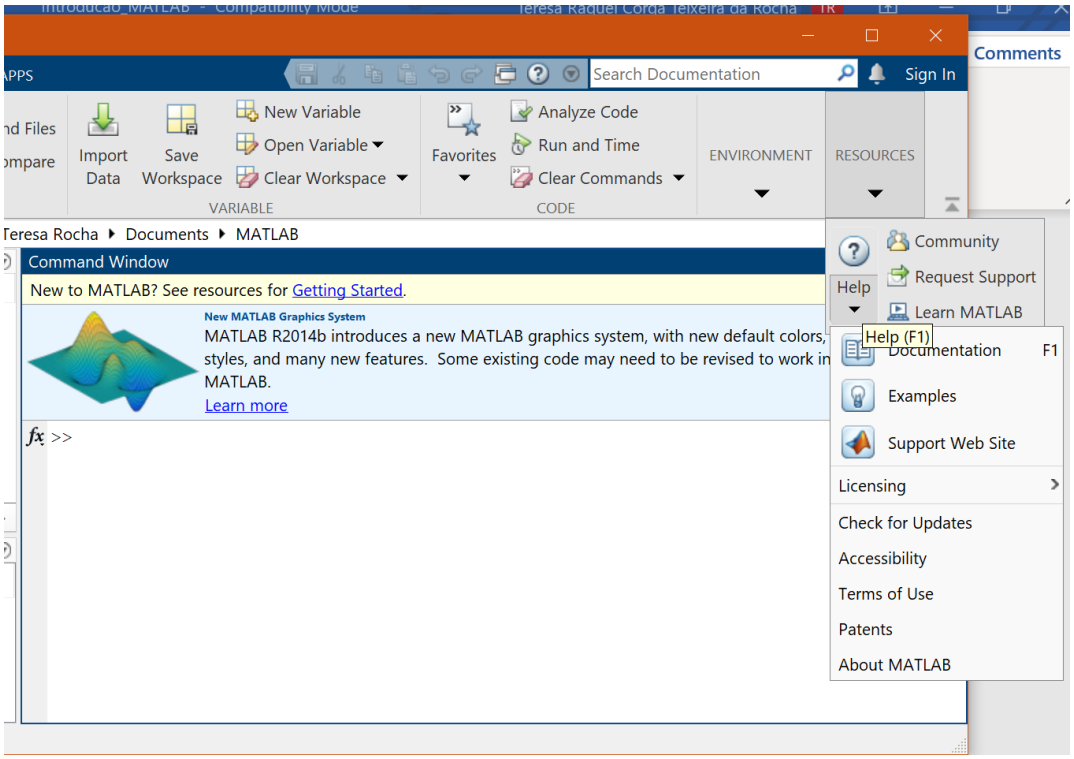
Some other windows of the desktop are:

Current Folder - provides a very quick way to view or change the current folder.

Workspace Browser – the MATLAB workspace consists of a set of variables created during a work session and stored in memory. To visualize the workspace and information about each variable, use this browser or, alternatively, the *who* and *whos* functions in

the command window. By double-clicking on one of the variables, you access the Array Editor, where you can view/change the values of that variable.

Help Browser – is a browser accessible from the Resources option of the main menu, which allows searching and viewing documentation related to MATLAB.



There are some help commands that can be entered in the command window and that might also be useful. Namely:

- ***demo*** – gives access to examples of some of MATLAB's features in the Help Browser
- ***help <name>*** - provides online help for the <name> command/function
- ***doc*** – equivalent to opening the Help Browser window

2. Matrix definition and matrix operations

As mentioned before, in MATLAB the basic element is the matrix.

2.1 Introduction / definition of matrices

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

2.1.1 Direct or explicit introduction

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

☞ **Note:** The ";" works as a row switching operator. Alternatively, it is possible to define a matrix by pressing the “ENTER” key whenever you want to make a new row in the matrix.

2.1.2 Definition based on the result of an operation

```
>> C=2*A
C =
 2 4 6
 8 10 12
```

2.2 Accessing array elements

2.2.1 Access only one element

A(i,j) – Allows access to the element located in row i, column j.

```
>> res = A (1.2)
res = 2
```

2.2.2 Access a set of elements

A(imin:imax , jmin:jmax)

```
>> res = A(1:2,2:3)
res =
 2 3
 5 6
```

If you simply use the ':' operator without specifying any threshold values, it is equivalent to selecting 'all rows' or 'all columns'

```
>> res = A(1,:)
res =
 1 2 3
```

2.3 Elementary operations

$$B = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix}$$

```
>> B=[0 1 2;3 4 5]
B =
 0 1 2
 3 4 5
```

- Adding matrices:

```
>> A+B
ans =
 1 3 5
 7 9 11
```

- Transposed of a matrix:

```
>> A'
ans =
 1 4
 2 5
 3 6
```

- Matrix multiplication:

```
>> A*B
??? Error using ==> *
Inner matrix dimensions must agree.
```

```
>> D=A'
D =
 1 4
 2 5
 3 6
>> B*D
ans =
 8 17
 26 62
```

- Element-by-element matrix multiplication (use the '.' operator before the '*' operator):

```
>> A.*B
ans =
 0 2 6
 12 20 30
```

- Inverse of a matrix:

```
>> C = [4 1 9; 4 5 6; 7 8 9]
C =
 4 1 9
 4 5 6
 7 8 9
>> inv(C)
ans =
 0.0909 -1.9091 1.1818
 -0.1818 0.8182 -0.3636
 0.0909 0.7576 -0.4848
```

- Main diagonal of a matrix:

```
>> diag(C)
ans =
 4
 5
 9
```

- Matrix of ones (*ones(m,n): m=number of rows; n=number of columns*):

```
>> ones(3,4)
ans =
 1 1 1 1
 1 1 1 1
 1 1 1 1
```

- Matrix of zeros (*zeros(m,n): m=number of rows; n=number of columns*):

```
>> zeros(2,5)
ans =
 0 0 0 0 0
```

```
0 0 0 0 0
```

- Identity Matrix (*eye(m) m=number of rows=number of columns*)

```
>> eye(3)
ans =
 1 0 0
 0 1 0
 0 0 1
```

2.4 Other functions

- Absolute value: abs(X)

  Calculates the absolute value of the elements of X.

```
>>M = [-1 2 -3; 0 -84];
>> abs(M)
ans =
 1 2 3
 0 8 4
```

- Average value: mean(X)

  If X is a vector, returns the mean value of the elements of X;

  If X is a matrix, returns a row vector where each element is the mean value of each column of X.

```
>>V = [7 3 1];
>> mean(V)
ans =
 3.6667
>> mean(M)
ans =
 -0.5000 -3000 0.5000
% To get the matrix minimum:
>> mean(mean(M))
ans =
 -1
```

- Maximum / Minimum Value: max(X) / min(X)

  If X is a vector, returns the maximum / minimum value of the elements of X;

  If X is a matrix, returns a row vector where each element is the maximum/minimum value of each column of X.

- **...**

3. *Scripts* and functions

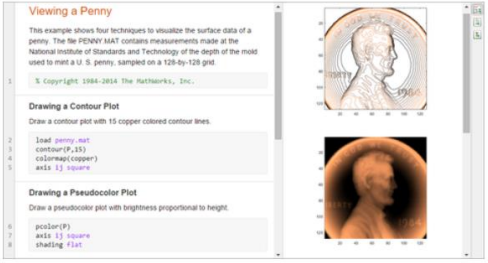
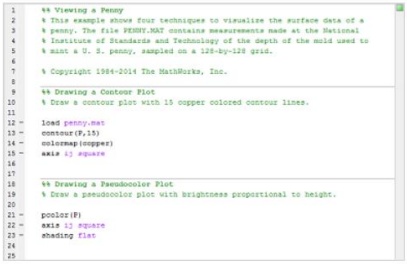
The files that contain code in the MATLAB language are files with the ".m" extension. You can create these files with a text editor and then use them like any other MATLAB function or command.

There are two types of files:

- ***Scripts*** – do not accept input arguments nor return any output arguments. They operate on data from the workspace.
- ***Functions*** – can accept either input or output arguments. Internal variables are local to the function.

In MATLAB 2021, there are Live Scripts and Live Functions that are interactive documents that combine MATLAB code with formatted text, equations and images in a single environment called Live Editor. They are especially suited to visually analyzing and exploring problems, creating interactive presentations, and more.

The following table illustrates the differences between the two types of scripts and functions.

	Live Scripts and Functions	**Plain Code Scripts and Functions**
File Format	Live Code file format. For more information, see Live Code File Format (.mlx)	Plain Text file format
File Extension	`.mlx`	`.m`
Output Display	With code in the Live Editor (live scripts only)	In Command Window
Text Formatting	Add and view formatted text in the Live Editor	Use publishing markup to add formatted text, publish to view
Visual Representation		

4. Logical Expressions

- Less than **<**
- Bigger then **>**
- Less than or equal to **<=**
- Greater than or equal to **>=**
- Equal to **==**
- Different than **~=**
- Logical AND **&&**
- Logical OR **||**
- Denial **~**

5. Flow control structures

- **IF:** *execute instructions conditionally*

  The general format is as follows:

```
if expression
 instructions
elseif expression
 instructions
else
 instructions
end
```

Example:

```
if i == j
        A(i,j) = 2;
elseif i < j
        A(i,j) = 1;
else
        A(i,j) = 0;
end
```

- **FOR:** *repeat instructions a certain number of times*

  The general format is as follows:

```
for variable = expression
        instruction
         ...
        Instruction
end
```

Example:

```
for i = 1:N
        for j = 1:N
                A(i,j) = i+j;
         end
end
```

- **WHILE:** *repeat instructions a certain number of times*

  The general format is as follows:

```
while expression
        instruction
         ...
        Instruction
end
```

Example:

```
i=1;
j=1;
while i<=N
        while j<=N
                A(i,j) = i+j;
                i=i+1;
        end
        j=j+1;
end
```

Exercise:

Develop a small program that counts the positive and negative numbers present in a matrix of the user's choice.

6. Functions

In MATLAB, the name that identifies the function is the name of the ".m" file that contains it. In this perspective, the function should always be given the same name as the ".m" file. If that file contains more than one function, only the first one can be called from the command window or by another ".m" file.

```
function [output_parameters] = function_name (input_parameters)
...
...
        Calculation of output parameters
...
```

Exercise:

Turn the previously developed program into a function. It must have as input parameter an array and, as output parameters, the total of positive, negative and null numbers of that array.

7. *.mat files

MATLAB allows you to store data from a specific session for later use through a *.mat file.

- Store variables — **save** *file_name variables*
- Load stored variables — **load** *file_name variables*
- Identify active variables — **who** or **whos**
- Clear workspace — **clear**

Exercise:

- Define the set of desired variables
- Check which variables make part of the workspace (active)
- Create “variables.mat” file
- Clear all defined variables
- Load the previously created file
- Check workspace

Loading a *.mat file is an alternative way of defining the workspace (the set of variables).