# 1 LAB 1 (INTRODUCTION TO NUMERICAL ANALYSIS WITH MATLAB)

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.

MATLAB offers lots of additional Toolboxes for different areas such as Control Design, Image Processing, Digital Signal Processing etc. MATLAB is a tool for technical computing, computation and visualization in an integrated environment, e.g.,

- Math and computation
- Algorithm development
- Data acquisition
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is developed by The MathWorks. MATLAB is a short-term for MATrix LABoratory. MATLAB is in use world-wide by researchers and universities.

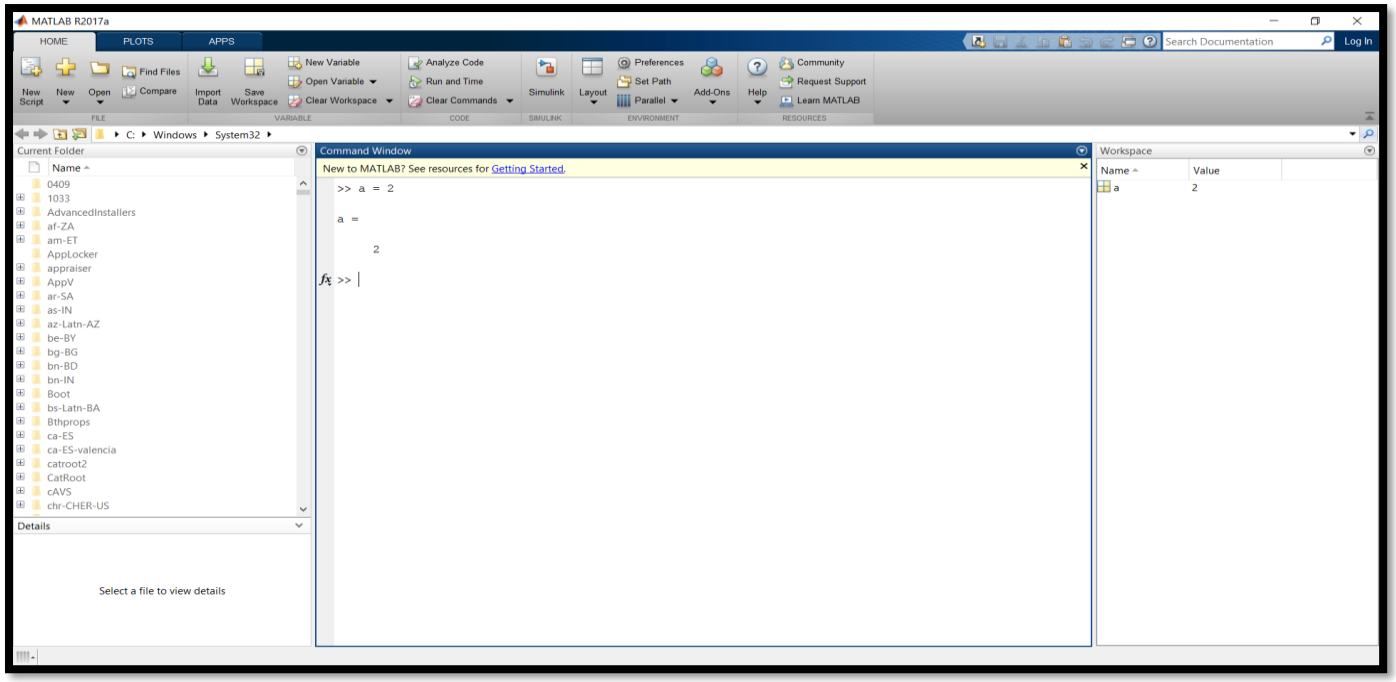


*Figure 1: MATLAB application (running on Windows 10)*

## 1.1 HOW TO GET STARTED?

1. Desktop machines in lab are already setup with this environment. Start MATLAB from its icon which is either placed on desktop or is available in search as shown in figure 2.
   (Note for installation on personal PCs, take setup from GA)

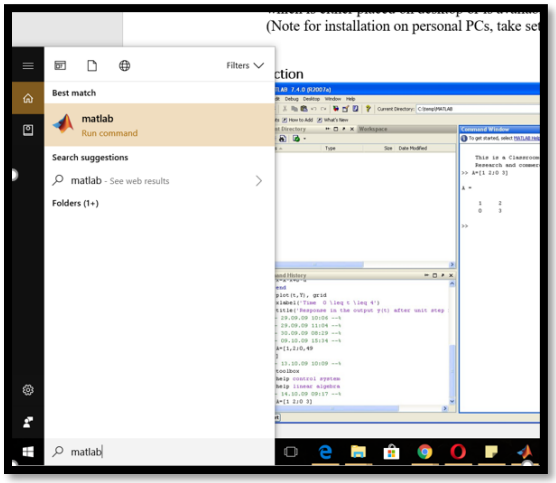


*Figure 2: How to start MATLAB*

2. Figure 1 shows MATLAB application. It contains several windows for our ease.
    a. Command windows is the main window. Use it to enter variables and to run functions and M-files scripts (more about m-file later).
    b. Command history (maintains history of previously entered commands)
    c. Workspace shows stored variables
    d. Current folder shows current directory where you will run your commands
3. **HELP:** In command window type *help*. List of all available APIs will be shown. You can click on entry to see list of available functions in MATLAB. For instance, there is a function named plot in MATLAB, you can type help plot, to know about its usage and working.
    >> help
    >> help plot
    See Figure 3.

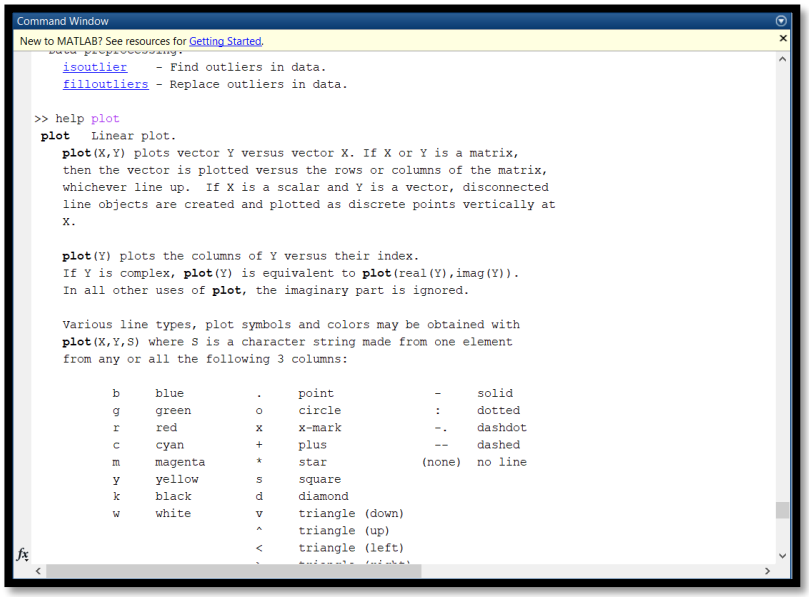


*Figure 3: How to write commands in command window*

4. **Declaring & using variables:** Now type following commands one by one (in command window) and observe what happens. Write answer in front of the command.
    >> a = 2
    >> a
    >> b

```
>> b = 5
>> a+b
>> ans
>> c = a + b
>> c
>> ans
```

What is variable **ans** and what is the datatype of declared variables?

5. **Vectors:** In MATLAB, the basic objects are matrices, i.e. arrays of numbers. Vectors can be thought of as special matrices. A row vector is recorded as a 1×n matrix and a column vector is recorded as a m×1 matrix. To enter a row vector in MATLAB, type the following in the command window one by one (using ENTER):

```
>> v = [0 1 2 3]
>> u = [9; 10; 11; 12; 13]
>> u(2)
>> u(2) = 47
>> u(2:4)
>> w = v'
>> A = [1 2; 3 4]
>> A
>> B = [2 3; 4 7];
>> B
```

What does this " **'** " operator is called? Observe how vector is being changed.

How can we create a column and row vector in MATLAB? In above examples which one is column and row vector in ***v*** and ***u***?

What happens with entering "**;**" in the end of command **B = [2 3; 4 7]**?

Now type following commands and see what happens.

```
>> x = -1:.1:1
>> y = linspace(0, 1, 11)
```

Explain your results.

6. **Formatting line spaces:** Type following commands and write your observation for both **compact** and **long** separately**.**

```
>>format compact
>> j = -1:.1:1
>>format long
>> j = -1:.1:1
```

Explain your results.

7. **Plotting data & tables:** We have some results on viscosity of a liquid which is a function of temperature as shown in table 1.

*Table 1: Viscosity of a liquid as a function of temperature*

| T (C) | 5 | 20 | 30 | 50 | 55 |
|---|---|---|---|---|---|
| U | .08 | .015 | .009 | .006 | .0055 |

Type following commands in command window:

```
>> x = [5      20     30     50     55]
>> y = [0.08   0.015  .009   .006   .0055]
>>x
>>y
>> plot(x,y)
>> plot(x,y, '*')
>> plot(x,y, 'o')
>> plot(x,y, '.')
```

Explain your results.

8. **Built in functions:** MATLAB has certain built in functions, which ease our life. Input goes into parenthesis (just like other programming languages). Type following commands for some of the available functions

```
>> sin(pi)
>> exp(0)
>> exp(1)
>> exp(2)
>> exp(10)
```

Observe:

```
>> x = linspace(0, 2*pi, 40)
>> y = sin(x)
>> plot(x, y)
```

Explain your results. How graph is created?

In the same way, you can use cos( ), tan( ), sinh( ), cosh( ), log( ) (natural log), log10( ) (log base 10), asin( ) (inverse sine), acos( ), atan( ). You can use help command to find more available functions.

9. **User-defined anonymous functions:** An anonymous function is a function that is not stored in a program file, but is associated with a variable whose data type is function_handle. Anonymous functions can accept inputs and return outputs, just as standard functions do. However, they can contain only a single executable statement. Let's say you want to create your own function and use it to plot data. Function is **f(x) = 2x^2 – 3x + 1.** Try following commands for defining it.

```
>> f = @(x) 2 * x.^2 – 3 * x + 1
```

Where f is the function_handle.

```
>> y = f (2)
>> y = f(2.23572)
```

Let's plot it for different values of -2 <= x <= 2.

```
>> x= -2:.2:2
>> y = f(x)
>>plot(x,y)
```

Explain your results. How graph is created?

Also, type following commands

```
>> [-2 -1.8 -1.6].^2
>> [-2 -1.8 -1.6]^2
```

What is the difference between these commands?

## 1.2 PART II (CREATING CODE FILES)

If you want to write scripts or functions in MATLAB, you can either write your function in command window or can also create script or functions files with extension "**.m**".

1. Click on the New Script button on top-left corner of MATLAB window

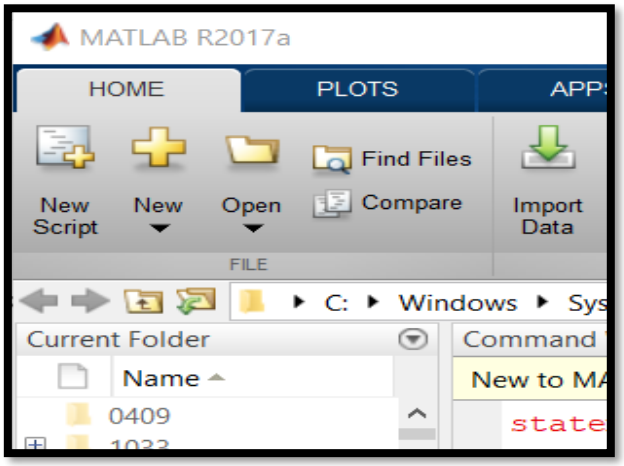


*Figure 4: Create script or function files (.m)*

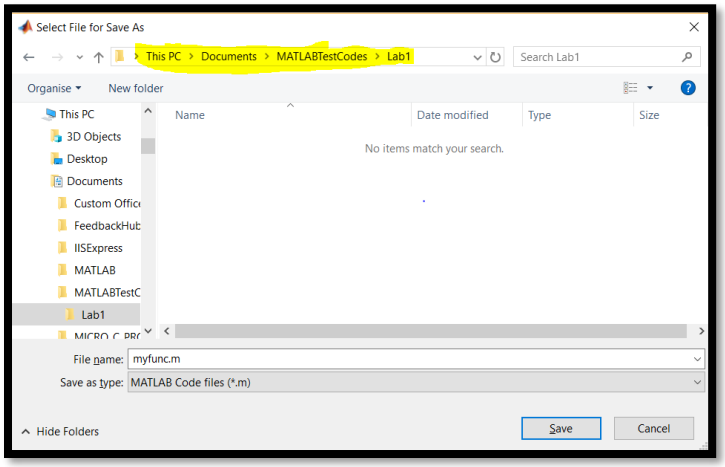


*Figure 5: Saving file to a different path*

2. Enter the code shared in screenshot in file as shown in Figure 6.

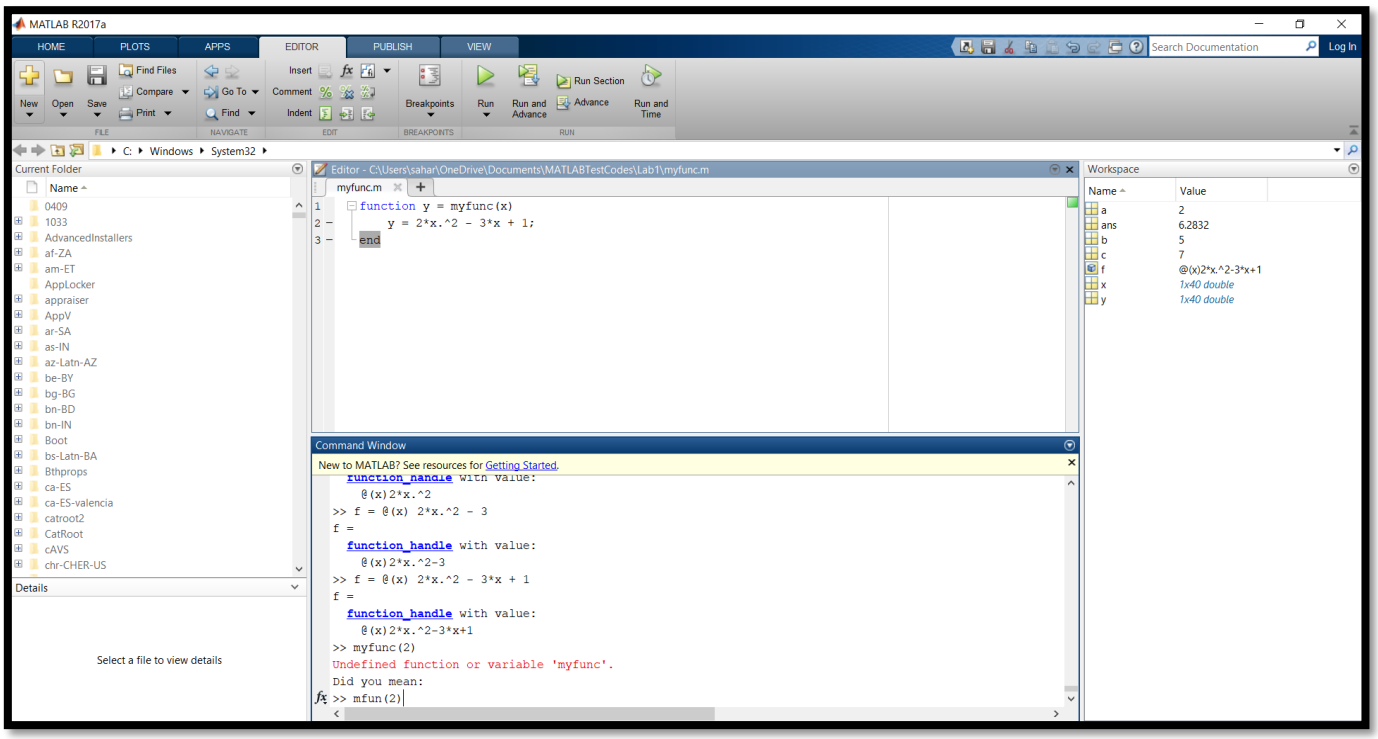


*Figure 6: myfunc.m file*

3. Either save the file in same directory or change the directory as we have done in Figure 5.
4. Now type myfunc(2) in command window. If you have saved your file in same directory then you will see an answer but if you have saved your function in different directory, then error will occur. Observe the error. MATLAB does not recognize your declared function. Why is it so? This is because it does not know any function declared in current directory. You need to change your current directory to the directory, where you have saved your file as shown in Figure 6.

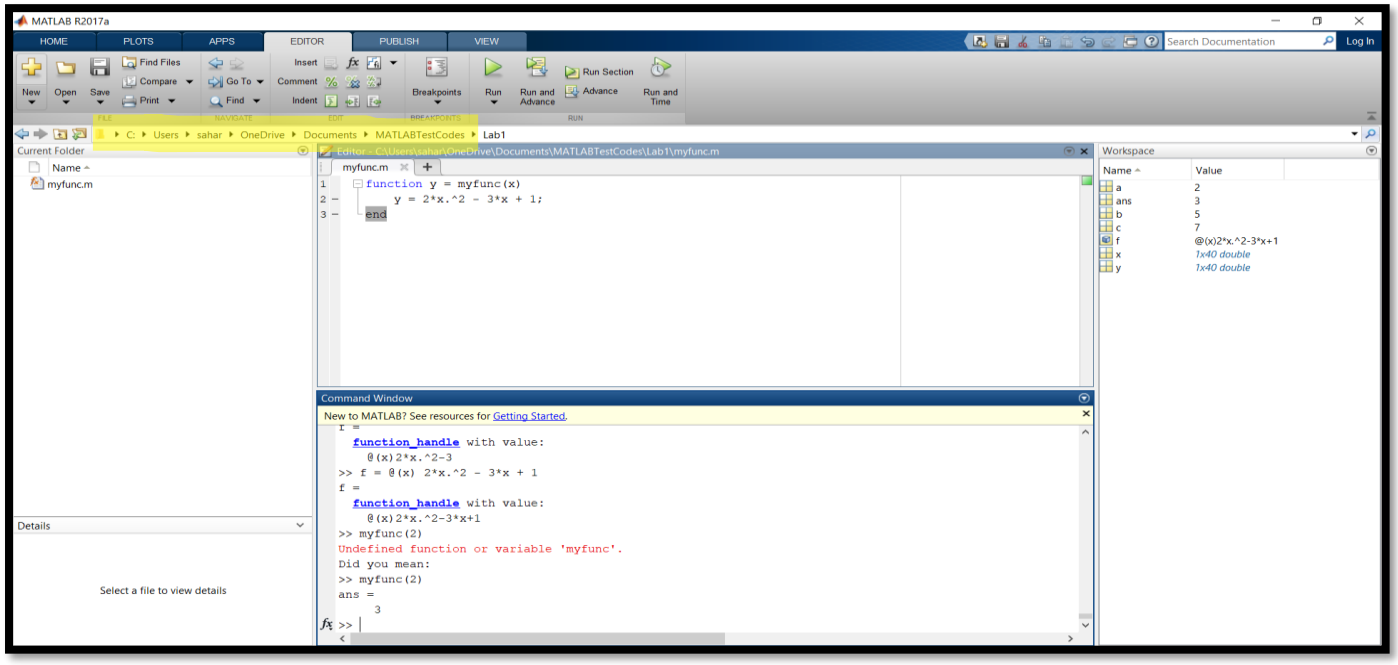


*Figure 7: Changed directory and myfunc is recognized*

5. Now type following commands in command window. Resultant figure should be same as before.
   >> p = -2:.1:2;
   >> y = myfunc(p);
   >> plot(p, y)

   Following observation have been noted.

- Begin with the word function.
- There is an input and an output.
- The output, name of the function and the input must appear in the first line.
- The body of the program must assign a value to the output variable(s).
- The program cannot access variables in the current workspace unless they are input.
- Internal variables inside a function do not appear in the current workspace.
- Functions can have multiple inputs, which are separated by commas. For example:

```
function y = myfunc2d(x,p)
    y = 2*x.^p - 3*x + 1;
end
```

- Functions can have multiple outputs

```
function [x2 x3 x4] = mypowers(x)
    x2 = x.^2;
    x3 = x.^3;
    x4 = x.^4;
end
```

6. Save this file as mypowers.m. Type following commands in command window.
   >> a = -1: .1 : 1
   >> [a2, a3, a4] = mypowers(a);
   >> plot(a, a, 'black', a, a2, 'blue', a, a3, 'green', a, a4, ''red)
   Explain your results. How graph is created?

7. Scripts: You can also create plain code files. There are no input and output. For example, create a file named mygraph.m and write following code in it.

```
x2 = x.^2;
x3 = x.^3;
x4 = x.^4;
plot(x,x,'black',x,x2,'blue',x,x3,'green',x,x4,'red')
```

*Figure 8: mgraph.m file*

8. Now type following commands in command window.
   >> x = -1:.1:1;
   >> mygraph
   Explain your results. How graph is created?

9. **Comments:** Make sure to add comments in your functions. Add information about input, output and details about function. "**%**" is used for commenting in MATLAB. For scripts, add file name on top and details about code. Add comments where required. After this, comment your functions and script files. **This is ground rule and hence will be applied on all other labs without explicitly writing or expecting.**

```
function y = myfunc(x)
    % Computes the function 2x^2 -3x +1
    % Input: x -- a number or vector;
    %             for a vector the computation is elementwise
    % Output: y -- a number or vector of the same size as x
    y = 2*x.^2 - 3*x + 1;
end
```

```
% mygraphs
% plots the graphs of x, x^2, x^3, and x^4
% on the interval [-1,1]

% fix the domain and evaluation points
x = -1:.1:1;

% calculate powers
% x1 is just x
x2 = x.^2;
x3 = x.^3;
x4 = x.^4;

% plot each of the graphs
plot(x,x,'+-',x,x2,'x-',x,x3,'o-',x,x4,'--')
```

10. Now type following commands and see what happens.
    >> help mygraph
    Write the output.

## 2 EXERCISE

1. Find a table of data in an engineering or science textbook or website. Input it as vectors and plot it. Use the insert icon to label the axes and add a title to your graph. Turn in the graph. Indicate what the data is and properly reference where it came from.
2. Find a function formula in an engineering or science textbook or website. Make an anonymous function that produces that function. Plot it on a physically relevant domain. Label the axes and add a title to your graph.

Turn in the graph and write on the page the MATLAB command for the anonymous function. Indicate what the function means and properly reference where it came from.

3. Write a well-commented function program for the function $x^2e^{-x^2}$, using entry-wise operations (such as .* and .^). To get ex use exp(x). Plot the function on [−5,5] using enough points to make the graph smooth. Turn in printouts of the program and the graph.
4. Write a well-commented script program that graphs the functions sinx, sin2x, sin3x, sin4x, sin5x and sin6x on the interval [0,2π] on one plot. (π is pi in MATLAB.) Use a sufficiently small step size to make all the graphs smooth. Turn in the program and the graph.

 