# 1 Introduction to Matlab

## 1.1 What is MATLAB?

MATLAB is widely used in all areas of applied mathematics, in education and research at universities, and in the industry. MATLAB stands for Matrix Laboratory and the software is built up around vectors and matrices. This makes the software particularly useful for linear algebra but MATLAB is also a great tool for solving algebraic and differential equations and for numerical integration. It has powerful graphic tools and can produce nice pictures in both 2D and 3D. It is also a programming language, and is one of the easiest programming languages for writing mathematical programs. It also has some tool boxes useful for signal processing, image processing, optimization, etc. MATLAB is available on a number of computing environments: MS windows, DOS, UNIX workstations, Macintosh etc.

## 1.2 How to Start MATLAB

**Mac:** Double-click on the icon for MATLAB.

**PC:** Choose the submenu "Programs" from the "Start" menu. From the "Programs" menu, open the "MATLAB" submenu. From the "MATLAB" submenu, choose "MATLAB".

**Unix:** At the prompt, type `matlab`.

You can quit MATLAB by typing `exit` in the command window.

## 1.3 MATLAB Environment

The MATLAB environment (on most computer systems) consists of menus, buttons and a writing area similar to an ordinary word processor. There are plenty of help functions that you are encouraged to use. The writing area that you will see when you start MATLAB, is called the *command window*. In this window you give the commands to MATLAB. For example, when you want to run a program you have written for MATLAB you start the program in the command window by typing its name at the prompt. The command window is also useful if you just want to use MATLAB as a scientific calculator or as a graphing tool. If you write longer programs, you will find it more convenient to write the program code in a separate window, and then run it in the command window.

In the command window you will see a prompt that looks like >>. You type your commands immediately after this prompt. Once you have typed the command you wish MATLAB to perform, press <enter>. If you want to *interrupt a command* that MATLAB is running, type <ctrl> + <c>.

The commands you type in the command window are stored by MATLAB and can be viewed in the *Command History* window. To repeat a command you have already used, you can simply double-click on the command in the history window, or use the <up arrow> at the command prompt to iterate through the commands you have used until you reach the command you desire to repeat.

## 1.4 Useful Functions and Operations in MATLAB

Using MATLAB as a calculator is easy.

*Example:* Compute 5 sin(2.5$^{3\text{-pi}}$)+1/75. In MATLAB this is done by simply typing

```
5*sin(2.5^(3-pi))+1/75
```

at the prompt. Be careful with parentheses and don't forget to type `*` whenever you multiply.

**Note:** MATLAB is *case sensitive*. This means that MATLAB knows a difference between letters written as lower and upper case letters. For example, MATLAB will understand `sin(2)` but will not understand `Sin(2)`.

## 1.5 Obtaining Help on MATLAB Commands

To obtain help on any of the MATLAB commands, you simply need to type

```
help <command>
```

at the command prompt. For example, to obtain help on the `gamma` function, we type at the command prompt:

```
help gamma
```

Try this now. You may also get help about commands using the "Help Desk", which can be accessed by selecting the MATLAB Help option under the Help menu.

**Note:** The description in MATLAB returns about the command you requested help on contains the command name in ALL CAPS. This does not mean that you use this command by typing it in ALL CAPS. In MATLAB, you almost always use all lower case letters when using a command.

## 1.6 Variables in MATLAB

We can easily define our own variables in MATLAB. Let's say we need to use the value of 3.5 sin (2.9) repeatedly. Instead of typing `3.5*sin(2.9)` over and over again, we can denote this variable as x by typing the following:

```
x = 3.5*sin(2.9)
```

Often, we may not want to have the result of a calculation printed-out to the command window. To supress this output, we put a semi-colon at the end of the command; MATLAB still performs the command in "the background". If you defined x as above, now type as shown below and observe.

```
y = 2 * x;
y
```

In many cases we want to know what variables we have declared. We can do this by typing `whos`. Alternatively, we can view the values by opening the "Workspace" window. This is done by selecting the Workspace option from the View menu. If you want to erase all variables from the MATLAB memory, type `clear`. To erase a specific variable, say `x`, type `clear x`. To clear two specific variables, say `x` and `y`, type `clear x y`, that is separate the different variables with a space. Variables can also be cleared by selecting them in the Workspace window and selecting the delete option.

## 1.7 Vectors and Matrices in MATLAB

We create a vector in MATLAB by putting the elements within [ ] brackets.

*Example:* `x = [1 2 3 4 5 6 7 8 9 10]`

We can also create this vector by typing `x = 1:10.` The vector (`1 1.1 1.2 1.3 1.4 1.5`) can be created by typing

`x =[1 1.1 1.2 1.3 1.4 1.5 ]` or by typing

`x = 1:0.1:1.5.`

Matrices can be created according to the following example. The matrix $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ is created by typing

```
        A = [1 2 3 ; 4 5 6; 7 8 9],
```

i.e., rows are separated with semi-colons. If we want to use a specific element in a vector or a matrix, study the following example:

*Example:*

```
    x=[10 20 30]
    A=[ 1 2 3 ; 4 5 6 ; 7 8 9]
x(2) represents second element of X vector
A(3,1) represents third row first elememt of Matrix
A
```

Here we extracted the second element of the vector by typing the variable and the position within parantheses. The same principle holds for matrices; the first number specifies the row of the matrix, and the second number specifies the column of the matrix.

If the matrices (or vectors which are special cases of a matrices) are of the same dimensions then matrix addition, matrix subtraction and scalar multiplication works just like we are used to.

*Example:* Type

```
    x =[1 2 3]
    y =[4 5 6]
    a = 2
    x + y
    x - y
    a × x
```

If want to apply an operation such as squaring each element in a matrix we have to use a dot. before the operation we wish to apply. Type the following commands in MATLAB.

```
    x = 1:10
    x .^2
    A = [1 2 3 ; 4 5 6 ; 7 8 9 ]
    A.^2
    A^2
```

## 1.8 The Dot Allows us to do Operations Element Wise

All built-in functions such as `sin`, `cos`, `exp` and so on automatically act element wise on a matrix. Type

```
Y =[0 1/4 1/2 3/4 1]
y = pi*y
sin(y)
```

## 1.9 How to draw a plot with MATLAB

There are different ways of plotting in MATLAB. The following two techniques, illustrated by examples, are probably the most useful ones.

**Example 1:** Plot sin($x^2$) on the interval [–5, 5]. To do this, type the following:

```
x =-5:0.01:5;
y = sin(x.^2);
plot(x,y)
```

**Example 2:** Plot exp(sin(x)) on the interval [–π, π]. To do this, type the following:

```
x=linspace(-pi,pi,101);
y = exp(sin(x));
plot(x,y)
```

The command `linspace` creates a vector of 101 equally spaced values between –π and π (inclusive).

Occasionally, we need to plot values that vary quite differently in magnitude. In this case, the regular `plot` command fails to give us an adequate graphical picture of our data. Instead, we need a command that plots values on a log scale. MATLAB has 3 such commands: `loglog`, `semilogx`, and `semilogy`. Use the `help` command to see a description of each function. As an example of where we may want to use one of these plotting routines, consider the following problem:

**Example 3:** Plot $x^{5/2}$ for $x = 10^{-5}$ to $10^5$. To do this, type the following:

```
x=logspace(-5,5,101);
y=x.^(5/2);
plot(x,y)
```

Now type the following command:

```
loglog(x,y)
```

The command `logspace` is similar to `linspace`, however it creates a vector of 101 points logarithmically equally distributed between $10^{-5}$ and $10^{5}$.

- **Options**

  Plot, loglog, semilogx, semilogy, bar, hist, stem, polar, contour.

- **Graphic Commands**

  Title, xlabel, ylabel, text, grid, axis, clf, ginput, gtext.

- **Line types**

  Dotted .. dashes-- dashdot-.

- **Point Type**

  Point. Plus+ star* circle o xmark x

- **Color**

  Red r green g blue b yellow y magenta m

  Cyan c white w black k

- **Multiple plots**

  Subplot(mnp) options:221 211 121 ….

- **Auto scaling**
- **Zooming** zoom on