Getting your hands dirty with MATLAB

Dr Kartik Bulusu
Research Asst Professor
MAE Dept., GWU
Email: bulusu@gwu.edu

September 4, 2014

1 What is MATLAB ?

The name MATLAB stands for MATrix LABoratory. It is a software package for numerical computation and visualization with many built-in functions for technical computation, graphics and animation. Most of these functions are state-of-the art algorithms which provide excellent tools for linear algebra computations, data analysis, signal processing, numerical solution to ODEs etc.

The basic building block of MATLAB is the matrix. The fundamental data-type is the *array*. Special cases of this basic data-type are vectors, scalars, real and complex matrices. In MATLAB, you almost never have to declare the dimensions of a matrix. A novice MATLAB programmer has to have a grasp over the basics of matrix algebra, because MATLAB quite simply and precisely works with matrix operations.

2 Basics of MATLAB

In this section we look at the general structure of MATLAB environment. MATLAB works through three basic windows.

2.1 Types of Windows

1. **Command Window:** When you launch the MATLAB program, it pulls up this window. It is characterized by the MATLAB command prompt " $\gg$ ". All commands, including user-written programs can be executed from this prompt.

2. **Graphics Window:** The output of all graphics commands and plotting functions are shown in the graphics window or *Figure* window.

3. **Edit Window:** MATLAB provides a built-in editor where you can write, edit, create and save your programs in files called "*M-files*". Consequesntly, you can use any text editor to carry out these tasks. All programs should have a "**.m*" extension. For example, a sample file name can be "*myprogram.m*"

2.2 Types of Files

1. **M-files:** These are standard ASCII text files, with a"**.m*" extension. These file types are further classified into *script files* and *function files.* A script file is an "**.m*" with a valid set of MATLAB commands in it. It is executed by typing the name of the file (without the "**.m*" extension) in the command window. A function file is also an "**.m*" are much like programs or subroutines in Fortran, procedures in Pascal, and functions in C. Once you get to know MATLAB well, you are likely to spend time writing and refining your own function files.

2. **Mat-files:** These are binary data-files with a "**.mat*" extension to the filename. They are created when you save data with "*save*" command in the command window. The data is written in a special format that only MATLAB can read.

3. **Mex-files:** Mex-files are MATLAB- callable Fortran and C programs. They have "**.mex*" extension. We do not intend Mex-files in this course. Use of these files requires a good amount of experience with MATLAB.

3 Lesson 1: Creating and Working with Arrays of Numbers

An *array* is a list of numbers or expressions arranged in horizontal rows and vertical columns. When an array has one row or column, it is called a *vector.* An array with m rows and n columns is a called a *matrix* of size $m \times n$. (Pratap, 1999)

There are many mathematical concepts associated with vectors and matrices that will not be discussed in this course. But if you have some background in linear algebra, you will appreciate the fact that MATLAB can do almost any matrix computation like inverse, determinant, rank etc.

In this lesson we deal with one-dimensional arrays or vectors only. In later exercises, we will introduce two-dimensional arrays, i.e., matrices. Follow the instructions below and type them in the command window as you read along.

Note that any text following % symbol is a comment. While creating M-files or simply typing the script on the command window any text following the % is taken in MATLAB as comment which will not be executed in the program. It is a good practice to comment certain line of the programs for it will be easy for the programmer to understand the chain of thought that went into the program, when revisited.

We will now get a hang of

- **Creating row and column vectors**

  ```
  ≫ x = [2 4 5]          % x is a row vector with 3 elements
  x =
  2 4 5

  ≫ y = [2; 4; 5]          % y is a column vector with 3 elements
  y =
  2
  4
  5
  ```

- **Doing simple arithmetic operations on vectors**

  ```
  ≫ z = [3 4 6];          % z is a row vector with 3 elements
  ≫ a = x + z          % two vectors of the same size can be added or subtracted
  a =
  5 8 11

  ≫ b = x + y
  ???  Error using ==> +
  Matrix dimensions must agree.
          % a row vector cannot be added to or subtracted from a column vector and vice versa
  ```

- **Doing *array* operations**

  - **Term by term multiplication (.*)**

    ```
    ≫ a = x.*z          % multiply or divide two vectors of the same size term by term with
    the array operator .* (or ./)
    a =
    6 16 30

    ≫ b = 2*a          % to multiply or divide a vector with a scalar array operators are not
    required; no dot before * (or /)
    b =
    12 32 60
    ```

  - **Term by term division (./)**
  - **Term by term exponentiation (.^)**

- **Creating a vector of *n* numbers equally spaced between two given numbers *a* and *b***

  ```
  ≫ x = linspace(0,10,5)
  % creates a vector x with 5 elements linearly spaced between 0 and 10
  ```

x =
0 2.5000 5.000 7.5000 10.000

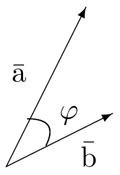
- **Using trigonometric functions and other elementary math functions with array arguments**

  ≫ y = sin(x);

  ≫ y = sqrt(x).*y
  x =
  0 0.9463 -2.1442 2.5688 -1.7203

  % *Trigonometric functions sin, cos etc., as well as elementary math functions sqrt, exp, log, etc., operate on vectors term by term,*

- **Calculating the angle (φ) between any two vectors $\bar{a}$ and $\bar{b}$**



The governing equation for computing the angle between the vectors shown in the figure above is as follows:

$$\bar{a} \cdot \bar{b} = ab\cos(\varphi)$$

The term on the left hand side of the above equation is known as the *dot product* of vectors $\bar{a}$ and $\bar{b}$. In the MATLAB sense, dot products are nothing more than sum of a term by term multiplication of vectors.

MATLAB command for dot products is quite simply

dot(A,B)

where A and B are any two vectors of same dimension; also meaning that both A and B have the same number of elements. You can try creating vectors in which the dimensions don't match and use *dot(A,B)*.

The error generated will be as follows:
```
???  Error using ==> dot
A and B must be same size
```

The magnitude of vectors ā and $\bar{\text{b}}$ is defined as the square root of the sum of squares of elements in a vector. The magnitude of a vector is also known as the *Norm of a vector*

$$\begin{aligned} \text{a} = |\bar{\text{a}}| &= \sqrt{a_1^2 + a_2^2 + ...} \\ \text{b} = |\bar{\text{b}}| &= \sqrt{b_1^2 + b_2^2 + ...} \end{aligned}$$

MATLAB command for the norm of a vector is again, quite simply

norm(X)

where X is a vector. You can try finding the norms of all the vectors generated so far.

In order to find the angle between the vectors ā and $\bar{\text{b}}$, the dot product of and the vectors is divided by the product of their magnitude. This leads us to the final equation.

$$\varphi = \arccos\left(\frac{\bar{a}.\bar{b}}{ab}\right)$$

MATLAB line of program for the above equation is as follows:

someangle = acos(dot(a,b)/(norm(a)*norm(b)))

Note that the angle thus calculated will be in radians and can be converted into degrees.

- **Walk through example**
  Note that the script following ≫ can be typed directly in the MATLAB command window and the reader is urged to do so. As usual, the text following the % symbols are comments and need not be typed. In this example, like a few others in the handout, the output for each line of program is printed below each command line just as one would see in the MATLAB command window.

```
≫ a = [sqrt(3),1]          % creating a row vector ā
a =
1.7321 1.0000

≫ b = [sqrt(3),-1]          %creating a row vector b̄
b =
1.7321 -1.0000

≫ dot_product = dot(a,b)          %dot product of vectors ā and b̄
dot_product =
2.0000

≫ norm_product = norm(a)*norm(b)          %product of norms of vectors ā and b̄
norm_product =
4.0000

≫ radians = acos(dot_product/norm_product)          %angle between vectors ā and b̄
in radians
radians =
1.0472

≫ degrees = radians*180/pi          %angle between vectors ā and b̄ in degrees
degrees =
60.0000
```

There will be times when the user need not see the outputs from each line of program. In order to suppress the output, a semicolon (;) should be added at the end of the command line script.

4 Exercise

1. **Angle between vectors:** Create a vector ā with elements 2, 4, 5, 9 and a vector b̄ of the same dimension as ā with elements equally spaced between 0 and 10. Compute the angle (in degrees) between the vectors ā and b̄.

2. **Multiply, Divide and exponentiate vectors:** Create a vector t̄ with 10 elements equally spaced between 1 and 10. Compute the following quantities:

   (a) $\mathrm{x} = \frac{t-1}{t+1}$

   (b) $\mathrm{y} = \frac{\sin(t^2)}{t^2}$

3. **Points on a circle:** Create a column vector θ with values 0, $\pi/4$, $\pi/2$, $3\pi/4$, π and $5\pi/4$.

   (a) Compute $\mathrm{x} = \mathrm{r}\cos\theta$ and $\mathrm{y} = \mathrm{r}\sin\theta$, taking r = 2.

   (b) Compute x^2 and y^2. (Yeah ! You need to use x.*x to compute x^2)

   (c) Now check whether x and y satisfy the equation of a circle, by computing the radius

$$\mathrm{r} = \sqrt{(x^2 + y^2)}$$

References

[1] Pratap, R., *Getting Started with MATLAB 5 - A Quick Introduction for Scientists and Engineers*, Oxford University Press, 1999.