International Journal of Research in Engineering, Science and Management
Volume-2, Issue-11, November-2019
www.ijresm.com | ISSN (Online): 2581-5792

"MATLAB" Basics: Commands, Functions & Operations

Vishal V. Mehtre[1], Aman Pal[2]
[1]*Assistant Professor, Dept. of Electrical Engineering, Bharati Vidyapeeth's College of Engineering, Pune, India*
[2]*Student, Dept. of Electrical Engineering, Bharati Vidyapeeth's College of Engineering, Pune, India*

***Abstract*: This paper gives us the knowledge of basic MATLAB and its uses. The software is widely used in Engineering Fields/Education. The software helps us in fast and quick calculations by the help of computational techniques. MATLAB is widely used tool in Electrical Engineering. It helps in control systems, circuit theory and magnetic field instruments. It is much faster and easier to develop code on MATLAB for quick and faster calculations than writing full code on C++ program.**

***Keywords*: MATLAB**

1. Introduction

A. *MATLAB [Matrix Laboratory]*

It can be run on UNIX, WINDOWS, and MACINTOSH. It is a multiparadigm numerical computing programming language developed by MathWorks.Com. The syntax is very similar for the DOS version. MATLAB integrates mathematical computing, visualization, and a powerful language to provide a flexible environment for technical computing. MATLAB allows matrix manipulations, plotting of function and data implementation of algorithms. MATLAB is based on the language C, but is generally much easier to use. Basically, MATLAB has 3 windows:

- COMMAND WINDOW
- GRAPHICS WINDOW
- EDIT WINDOW

COMMAND WINDOW: Used to enter commands and data to display plots and graphs.

GRAPHICS WINDOW: Used to display plots and graphs.

EDIT WINDOW: Used to create and edit m-Files

It is typically used in Numerical computation and algorithm development.

- Symbolic computation
- Modeling, simulation, and prototyping.
- Data analysis and signal processing
- Engineering graphics and scientific visualization

- *Arithmetic Operations:*
  - Addition +
  - Subtraction -
  - Multiplication *
  - Left Division \
  - Right Division /

B. *Built in Functions*

MATLAB has a number of functions for performing computations which includes logarithms, elementary functions and trigonometric functions.

Some common used functions are:

Table 1
Function and their description

Function	Description
Log(x)	Natural logarithm of x base to (e)
Log10(x)	Common logarithm of x base to (10)
Exp(x)	Exponential (e^x)
Sqrt(x)	Square root
Sin (x)	Computes sine of x, x in radians
Round(x)	Round to nearest integer
Abs(x)	Computes the absolute value of x

C. *Commands*

Commands can be used to eliminate variables or to obtain information about variables that have been created.

- First we have to enter the command in Command Window.
- Then we have to press Enter Key.

Table 2
Commands

Function	Description
clc	Clear the command window
Who	Lists the variables currently in the workspace
Help	Lists topic on which help is available
Demo	Runs the demo program
clf	Clear figure window
pwd	Shows the current work directory
Copy file	Copies a file
Quit	Quits MATLAB

D. *Vectors and Matrices*

A simple array is defined using the colon syntax: initial: increment: terminator. For instance:

>> array = 1:2:9 array =

1 3 5 7 9

defines a variable named array (or assigns a new value to an existing variable with the name array) which is an array consisting of the values 1, 3, 5, 7 and 9. That is, the array starts previous value by 2

(the increment value), and stops once it reaches (or to avoid exceeding) 9
(the terminator value).
>>array = 1 : 3 : 9 array =
1 4 7

the increment value can actually be left out of this syntax (along with one of the colons), to use a default value of 1.

>>ari = 1: 5 ari =
1 2 3 4 5

Assigns to the variable named ari an array with the values 1, 2, 3, 4, and 5, since the default value of 1 is used as the incrementer. Indexing is one-based that is the usual convention for matrices in mathematics, although not for some programming languages such as C, C++, and Java.

Matrices can be defined by separating the elements of a row with blank space or comma and using a semicolon to terminate each row. The list of elements should be surrounded by square brackets: []. Parentheses: () are used to access elements and sub arrays (they are also used to denote a function argument list).

>> A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
A=
16 3 2 13
5 10 11 8
9 6 7 12
4 15 14 1
>> A(2,3)
ans = 11

Sets of indices can be specified by expressions such as "2:4", which evaluates to [2, 3, 4]. For example, a sub-matrix taken

from rows 2 through 4 and columns 3 through 4 can be written as:

>> A(2:4, 3:4)
ans = 11 8
7 12
14 1

A square identity matrix of size n can be generated using the function eye, and matrices of any size with zeros or ones can be

generated with the functions zeros and ones, respectively.

>> eye (3 ,3) ans =
1 0 0
0 1 0
0 0 1
>> zeroes (2,3) ans =
0 0 0
0 0 0
>> ones (2,3) ans =
1 1 1
1 1 1

Transposing a vector or a matrix is done either by the function transpose or by adding prime after a dot to the matrix. Without the dot Matlab will perform conjugate transpose.

>> A = [1 ; 2], B= A. ', C = transpose(A) A= 1
2
B= 12
C= 12
>>D = [0 3 ; 1 5], D.' D =
0 3
1 5
ans =
0 1
3 5

Most MATLAB functions can accept matrices and will apply themselves to each element. For example, mod(2*J,n) will multiply every element in "J" by 2, and then reduce each element modulo "n". MATLAB does include standard "for" and "while" loops, but (as in other similar applications such as R), using the vectorized notation often produces code that is faster to execute. This code, excerpted from the function magic.m, creates a magic square M for odd values of n (MATLAB function meshgrid is used here to generate square matrices I and J containing 1:n). [1][3]

[J, I] = meshgrid (1: n);
A = mod (I + J - (n + 3) / 2, n); B= mod (I + 2 * J - 2, n);
M = n * A + B + 1

2. Newton Raphson Method

The method is used for finding the approximate roots of Polynomial Equation by programming in MATLAB. This provides the quick and faster calculations for finding the root of polynomial equation.

The Newton- Raphson method starts with a function of defined over the real numbers x the function derivative f' and an initial guess x0 for a root of the function f. If the function satisfies the assumption made in the derivation of the formula and the initial guess is close, then a better approximate x_1 is,

$$X_{n+1}=X_n-(\boldsymbol{Xn})/\boldsymbol{f}'(\boldsymbol{Xn})$$

A. Derivation

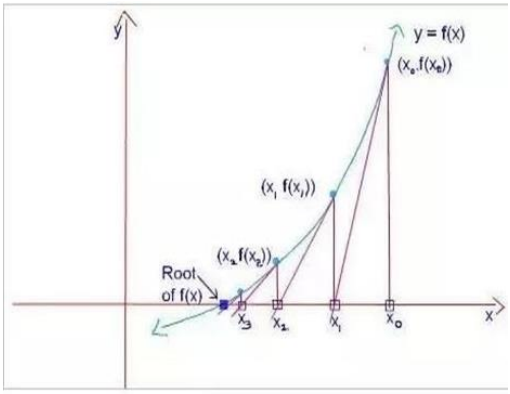

Fig. 1. Graphical Representation

f(x)=0
$x_1=x_0+h$
using Taylor's Series for:
$f(x_1)=f(x_0)+(h)f'(x_0)+(h^2)f''(x_0)/2!+$.....

h is small neglecting higher order terms
$f(x_1)=f(x_0)+(h)f'(x_0)=0$

$h=-f(x_0)/f'(x_0)$

substituting values of h in equation: $x_1=x_0 - f(x_0)/f'(x_0)$

$x_2=x_1- f(x_1)/f'(x_1)$

In general,

$$X_{n+1}=X_n-(\boldsymbol{Xn})/\boldsymbol{f}'(\boldsymbol{Xn})$$

Example:

Given equation is: x^3-4x-9 f= @(x)
x^3-4*x-9
df= @(x) 3*x^2-4

```
%change the lower limit a and upper limit b a=0; b=0;
x=a;
for i=1:1:100
        x1=x-(f(x)/df(x)); x=x1;
end sol =x;
fprintf ('Approximate root is %.15f', sol) a=2 ; b=3;
x=2; er(4)=0;
for i=1:1:4
        x1=x-(f(x)/df(x)); x=x1;
   er(i)=x1-sol; end
plot(er)
xlabel ('Number of iterations') ylabel
('Error')
title ('error Vs. Number of iterations')
f = @(x)x^3-4*x-9
df = @(x)3*x^2-4
```

Approximate root is 2.706527954497935>> MATLAB software also helps in analysis through graph. A MATLAB program produce the three-dimensional graphics using the functions surf, plot or mesh [2], [4].

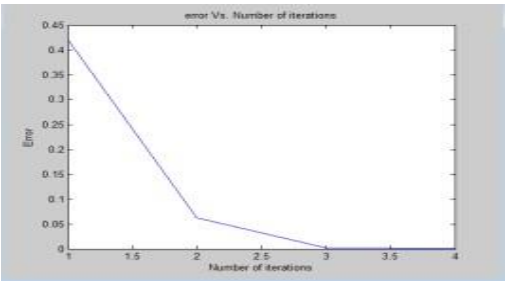

Fig. 2. Graph for no. of iterations vs. errors

3. Conclusion

The analysis carried out through this paper is that commands, Functions and Operations performed on BASIC MATLAB. Which can be used to perform computational techniques. It helps in various Engineering Fields to perform calculations. To find the roots of linear and non-linear equation. Newton – Raphson method is discussed which can be used in MATLAB software.

Acknowledgment

We would like to express our special thanks of gratitude to Dr. D. S Bankar, Head of Department of Electrical Engineering for their able guidance and support for completing my research paper. I would also like to thank the faculty members of the Department of Electrical Engineering would helped us with extended support.

References

[1] Rao V. Dukkipati, "Analysis and design of control systems using MATLAB," New Age International Publishers.
[2] www.mathworks.in
[3] David Houcque, "Introduction to Matlab for Engineering Students."
[4] Autar Kaw, "Newton Raphson Method for solving Nonlinear equations," Saylor,.org
[5] R. S. Maciel, A. Padilha-Feltrin, "Newton Raphson Method for Equations," IEEE 2006.
[6] Saba Akram, "Newton Raphson Method," IJSER, July 2015.
[7] Stephen J. Chapman, "MATLAB Programming for Engineers."
[8] A. V. Uppuluri, R. J. Jost, "MATLAB Processing Software", IEEE 2004.