

MET 301: LAB #1 MATLAB QUICK INTRODUCTION

Matlab is a software package for numerical computation and visualization. Its basic building block is the matrix and the fundamental data type is the array. It can handle Vectors, scalars, real and imaginary matrices automatically, without cumbersome declarations or dimensioning as required by many other scientific programming languages. It provides an interactive environment with hundreds of built in functions for technical computations, graphics and even animation. You can also write your own functions in the MATLAB language which will work as any other built in MATLAB function. This lab is intended to get you started quickly.

Launch MATLAB by clicking on:

start -> all programs -> MATLAB 7.0.1 -> MATLAB 7.0.1

MATLAB windows

You can activate, deactivate, dock, undock, open, close MATLAB windows using the “Desktop” pull-down menu bar at the top of the MATLAB screen. It has several windows, such as, Command window, Graphics window, Edit window, Command History window, Workspace window, Current Directory window, Help window etc. You can also change the layout of these windows on the desktop using this desktop menu.

Command window is the main window and has a command prompt ‘>>’. You enter your inputs here, and MATLAB interactively prints out the resulting outputs, except the graphics. Graphics are shown in a separate **Graphics window** which pops up when necessary.

Edit window is used to write your own program or functions in MATLAB. You can open an **Edit window** from the “File” menu, then new, then Mfile. Mfiles containing your programs are saved with ‘.m’ extension, which can be called anytime in a MATLAB session by typing the file name in the command window without the extension. You can also create any text editor, such as notepad or word, to create your mfiles.

Command History window contains all commands used by you, **Workspace window** shows the variables used and their values, **Current Directory window** shows directory of the program files, **Help window** shows all MATLAB help topics and demos.

LESSON 1: Basic arithmetic operations (10 min)

Arithmetic operators: +, -, *, /, ^, sin, cos, log, asin, pi, sqrt, log, log10, etc

Interactive input/output commands - %, clc, clear, ;, format, quit, diary

```
>>diary('yourname.out')
```

Opens a log file 'yourname.out' for your session and stores all inputs and outputs into this file.

```
>>2+8
```

```
ans =
```

```
10
```

Against command prompt type 2+8 and press enter. The result of the expression is stored in a default variable 'ans' and displayed back by MATLAB.

```
>>x=100-57
```

```
x =
```

```
43
```

You can also assign the value of an expression to variable 'x'.

```
>>y = sin(pi/6)*log(30);
```

```
>>y
```

```
y = 1.7006
```

Notice that π is pi in MATLAB. A semicolon at the end suppresses screen output by MATLAB. However, the value of the expression was calculated and stored to the variable y. Typing y against the command prompt will display the value of the y.

```
>>format long
```

```
>>y
```

```
y =
```

```
1.70059869083108
```

Checkout the format command by typing help format at the command prompt.

```
>>% Matlab ignores anything after %
```

```
>>z = x/10+Y
```

```
??? Undefined function or variable 'Y'.
```

```
>>z = x/10+y
```

```
z =
```

```
6.0006
```

Oops, variable names are case sensitive. It does not know the variable Y, but it knows y. You can use the up and down arrow to access any previous input and correct the mistakes, instead of typing the command again.

```
>>clc
```

```
>>clear
```

```
>>diary off
```

clc command clears the command window. Clear command clears the memory of all variable values. diary off command closes the diary file. You can use diary on – diary off to selectively log your session.

```
>> quit
```

LESSON 2 Creating and Working with Arrays (10 min.)

```
>> clear
```

```
>> x=[3 5 10]
```

Creates x variable as a row vector with 3 elements

```
x =
```

```
3 5 10
```

```
>> y = [6;7;8]
```

Creates y variable as a column vector with 3 elements

```
y =
```

```
6
```

```
7
```

```
8
```

```
>> a = [7 5 0]
```

```
a =
```

```
7 5 0
```

```
>> z = x+a
```

```
z =
```

```
10 10 10
```

You can add or subtract two vectors of same size

```
>> za = a.*z
```

```
za =
```

```
70 50 0
```

You can multiply (or divide) elements of two same sized vectors term by term using .* (./) operators

```
>> za = 2+za
```

```
za =
```

```
72 52 2
```

But scalar operation are automatically done term by term

```
>> m = linspace(0,10,5)
```

```
m =
```

```
0 2.5000 5.0000 7.5000 10.0000
```

Create a vector with 5 elements linearly spaced between 0 and 10

```
>> n = sin(m)
```

```
n =
```

```
0 0.5985 -0.9589 0.9380 -0.5440
```

Trigonometric functions sin, asin, cos etc. as well as elemental math functions sqrt, log etc operate on vectors term by term

```
>> quit
```

LESSON 3 Creating a Simple Plot (5 min.)

You are going to plot a function $f(t) = e^{(t/10)}\sin(t)$, for $0 \leq t \leq 20$

To do this first generate the data (x and y coordinates of, say about 200 points) then plot x versus the y coordinates and then add labels and title to the plot.

The following MATLAB command window does the job. Comments are placed after %.

```
>> x = 0: 0.1: 20; % another way of creating a linearly spaced vector with 201 elements
% don't forget the semicolon, otherwise your command window will be filled by a large
%array. If you want to see the array you can use 'more on' command.
>> y = exp(0.1*x).*sin(x); %calculate y by doing term by term array operation (.* )
>> plot(x,y) %plot a simple graph between x and y
>> title('A Simple 2D Plot by xxxxx') % Add a title
>> ylabel('Response amplitude') %label y-axis
>> xlabel('Time (t) in Seconds') %label x-axis
```

You can draw two or more plots superimposed on one other.

```
>> z= 2*y; % z is an array representing a function  $f(t) = 2*e^{(t/10)}\sin(t)$ 
>> plot(x,y, x,z, x,z,'o'); %plot contains x vs. y, x vs. z, with default line and x vs. z plot
% with character 'o'. You can use any character to draw the plot
>>quit
```

LESSON 4 Solving a Matrix Equation (5 min.)

Solving a set of linear equations is very common computation required in engineering. It is easy to do in MATLAB. We will solve x, y, and z from the following set of three independent equations.

$$\begin{aligned}2*x + 3*y - z &= -13 \\x + 9*y + 4*z &= 10 \\0*x + y + 4*z &= 43\end{aligned}$$

The above equations can be written in matrix form $[A]\{x\} = \{b\}$, where $[A]$ is the coefficient matrix, $\{x\}$ is the vectors of unknowns and $\{b\}$ is the vector containing the right hand side constants.

In MATLAB solution of the unknown $\{x\}$ will be simply achieved by using the backslash operator $x = A \setminus b$ (not the forward slash '/')

```
>> A = [2 3 -1; 1 9 4; 0 1 4]           %create the matrix A
A =

     2     3    -1
     1     9     4
     0     1     4

>> b = [-13; 10; 43]                   %create the column vector b
b =

    -13
     10
     43

>> x = A \ b                           %get the solution vector x
x =

     7
    -5
    12

>> c = A * x                           %check the solution. c is same as b.
c =

    -13
     10
     43
```

LESSON 5 Finding the roots of a non-linear polynomial equation (5 min.)

MATLAB's 'roots' function finds the roots of a polynomial equation. To use the roots function, first a vector is created with the coefficients of the polynomial arranged in the standard form that is:

$$C_1x^n + C_2x^{n-1} + C_3x^{n-2} + \dots + C_{n-1}x + C_n = 0$$

Then the roots function is supplied with this coefficient vector to find the roots.

We will find the roots of the following non-linear equation,

$$x^5 - 3x^3 + x^2 - 9 = 0$$

which should have 5 roots that satisfy the equation.

```
>> c=[1 0 -3 1 0 -9]      % compose a vector with the coeffs. of the polynomial.

c =

    1     0    -3     1     0    -9

>> x = roots(c)          % invoke the roots function

x =
    1.9316
    0.5898 + 1.1934i      % notice four roots are complex conjugate roots.
    0.5898 - 1.1934i
   -1.5556 + 0.4574i
   -1.5556 - 0.4574i
% CHECK THE ACCURACY OF THE ROOTS by using each root into the poly.
>> format compact
>> c1 = 1*x(1)^5+ 0*x(1)^4 -3*x(1)^3+ 1*x(1)^2 +0*x(1)-9
c1 =
    8.8818e-014
>> c2 = 1*x(2)^5+ 0*x(2)^4 -3*x(2)^3+ 1*x(2)^2 +0*x(2)-9
c2 =
    2.3093e-014 -2.8866e-015i
>> c3 = 1*x(3)^5+ 0*x(3)^4 -3*x(3)^3+ 1*x(3)^2 +0*x(3)-9
c3 =
    2.3093e-014 +2.8866e-015i
>> c4 = 1*x(4)^5+ 0*x(4)^4 -3*x(4)^3+ 1*x(4)^2 +0*x(4)-9
c4 =
    4.2633e-014 +2.8644e-014i
>> c5 = 1*x(5)^5+ 0*x(5)^4 -3*x(5)^3+ 1*x(5)^2 +0*x(5)-9
c5 =
    4.2633e-014 -2.8644e-014i %ALL VALUES ARE EXTREMELY CLOSE TO ZERO
>> quit
```

HOMEWORK ON MATLAB
Due on September 12, 2005

Use MATLAB to complete the following exercise. Save your MATLAB session using the diary function. Submit your work by printing the diary file.

1. Let: $t = 1.8$

Solve the value of $3.5 * e^{-t} [\cos(\pi/8) + \sin(\pi/6)]$

2. Compute the quantity $3 \frac{\sqrt{5}-1}{(\sqrt{5}+1)^2} - 1$

3. Plot $f(t) = t^{3.5}$ against t for $t = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$

Also plot $df(t)$ and $\int f(t)dt$ against t for the above values of t on the same plot.

Use different symbols for the plots. Label x and y axes appropriately and give a title of the plot with your name in it.

Print the plot separately and submit with your homework.

4. Solve the following set of simultaneous equations

$$2x + 3y - 4z = 20$$

$$x + 2y + 3z = -4$$

$$3x - y - z = 7$$

After you get the solution for x , y and z , check the accuracy of your solution by substituting back the values of x , y and z in the left hand side of each of the equations.

5. Solve the quadratic equation $x^2 - 226100x + 9005400000 = 0$. Use the roots function. After you find the solutions, check each solution for its accuracy.

Reference books:

1. Getting started with Matlab by Rudra Pratap
2. Check NJIT Library holdings with MATLAB title. There are many introductory books on MATLAB available.