MATH 1070, Introduction to Computational Science

Lab Session 1 - MATLAB

This lab currently runs on the PCs in the CLC, under the M/S Windows operating systems. If you are not already familiar with: finding and opening files, launching applications, etc. in windows; then you should take a few minutes to familiarize yourself with the computers in the lab – and feel free to ask for help.

Next, you should launch MATLAB, from the START -> APPLICATIONS menu on the lower LHS of the screen. Look for the MATLAB HELP menu, and browse around it, to get a feel for how it's structured. This will be a key resource for you. Also look for, and use, the document file "SimpleMatLabCmds" on the MATH1070 web site. More such information is given at the end. Look at the publisher's web site: www.mathworks.com; and later look into MathCentral for lots of contributes codes and useful info. Later, do a quick google search of "matlab"; look at some of the tutorials & simple exercises available online – eg: MIT, U Sydney, UQ, etc.

In this tutorial you will attempt some simple calculations in MATLAB and learn about the "embedded knowledge" within MATLAB regarding arrays, functions, plots, etc. Later you can attempt to learn what is known as the structural programming aspect of MATLAB, exemplifying the way in which a computer is programmed to alter the execution flow of commands and calculations. Examples, such as repeatedly executing a set of commands a specified number of times, or running them subject to logical conditions or tests, are structural components of a program. Such examples form the building blocks for solving just about any scientific or engineering problem with computers.

One of the advantages of MATLAB is its ability plot and "visualize" data and the results of computations, using the PLOT command. The functions used to plot are built-in and easy to use, not only in 1-D, but also in 2-D and 3-D. This tutorial will attempt to show the basics of plotting in MATLAB with the results from the logistic system as an example.

By the end of this laboratory tutorial you should be comfortable with the following:

- How to execute simple commands in MATLAB
- Built in functions in MATLAB
- basic plotting of functions and numerical data in 2-D; simple "3D" plots and sounds.
- Arrays, Matrices and simple Linear Algebra

In subsequent labs you will learn about:

- writing and using ".m" files;
- writing Loops to perform repetitive calculations in MATLAB;
- using conditional/logical test statements such as "if"; and
    - using the "*for*" command; "*while*" command.

• 'Never Ever' Users

Try the following exercises at the command window.

1. Is MATLAB good at doing maths?

a. Type `2+3` after the >> prompt and press the **Enter** key
If this right, try `2*3`
How about `1/2` and `2^3` ? What do the symbols `*`, `/` and `^` do?

b. What will be MATLAB answer to `0/1` and `1/0` ?

c. The symbol `Inf` is short for infinity. What happens if you type `13+Inf` and `29/Inf` ?

d. Another special symbol is `NaN`: it stands for Not a Number. It is the answer to calculations such as `0/0`.

e. MATLAB knows about special constants – type **pi**.

f. Evaluate 11x(15/11)-15. Is your MATLAB copy playing up?

• Getting started

2. let's do some algebra!

a). Enter the command `a=2`.
`a` is called a **variable**.
`a=2` in programming language is called a **statement**: you are *assigning* the value of `2` to the variable `a,` which is stored in the computer memory.
Notice that MATLAB displays this value as soon as the statement is executed – MATLAB is functioning as an *interpreted* language, rather than executing a compiled program.

If you enter the statement `a=a+7,` followed on a new line by `a=a*10`
what will be the final value of `a`?

b). now enter `b=3;`.
What is the effect of the semi-colon (`;`)?
If you want to know what is the value of `b`, type `b`: MATLAB will return the value currently assigned to `b`. You can also check the value of `a` by typing `a`.
MATLA knows scientific notation: try **y=1.0E-3**.

c). Assign any values you like to two variables `x` and `y`. Can you assign in a **single statement ?**
Assign the sum of `x` and `y` to a third variable `z`?

3. Mathematical functions

MATLAB knows all the functions that you find on a hand-held calculator: `cos`, `sin`, `log`, etc.

a). Calculate the square root of π with the command: `sqrt(pi)`. Note that MATLAB knows the value of π.

b). Trigonometric functions `sin(x)` expect the argument `x` to be in **radians**. What do you need to do to calculate **sin(90˚)** ?

c). The exponential function $\mathbf{e}^{\mathbf{x}}$ is `exp(x)`. Find the value of **e** and **1/e**.

4. Arrays

Simple variables such as `a` and `b,` above are called **scalar**: their value is a single number.
MATLAB can also handle **vectors,** commonly referred to in MATLAB as **arrays**.

a. The easiest way to define a vector is with a statement such as: `x = 0 : 10;`

Enter the statement (nb. you do not need to leave a space before and after the "**=**" and the "**:**") - now check the value of **x**. What does the ";" do?
Now try **y = [1.0 2.7 3.2];** this is a row vector. What is y(2) ?

b. The nice thing about MATLAB is that you can define new vectors in terms of the vector **x**.
Try: **y = 2 * x** and **z = sin(x)**

5. Basic operations on arrays:

MATLAB understands how to perform these. Some of these are:

ab	vector multiplication
a+c	vector addition
a-c	vector subtraction,
a.c	multiplies each element of a by the corresponding element in c,
a.^2	squares each of the elements of a, one at a time.

Other functions can be applied in the obvious way. Some of these are:

size(a)	returns the number of elements in the vector a,
transpose(a)	returns the transpose of a,
sqrt(a)	returns the square root of every element in a,
sin(a)	returns the sine of every element in a.

6. Plotting

a). Let's plot **sin(x)** - just type the command: **plot(x, sin(x))**
You could also type the command: **plot(x, z)**.The plot is displayed in a separate figure window. Type **help plot** to learn more.

b). The plot is not pretty because we do have enough points between **0** and **10**. Let's make the vector **x** go up in steps of **0.1** instead of **1** ,by typing: **x = 0 : 0.1 : 10;**
The middle number is called the **increment**.

Now enter: **plot(x, sin(x))**. The plot looks more like the real thing!
We can put a grid on the graph to make it easier to read with **plot(x, sin(x)), grid .**
Later you can learn how to put a **legend** and **axis labels** on you graphs
– just look in the **help** menu**.**

c. If you want to see more cycles of the sine graph, use command-line editing to change **sin(x)** to **sin(2*x)**:
use the **up arrow** to access the previous command
use the left arrow to insert **2*** next to **x** in **sin(2*x)**.

f. Draw the graph of **tan(x)** over the same domain. Should you return your copy of MATLAB to its maker?

7. System of linear equations

MATLAB can solve a system of linear equations easily. For instance to solve:

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

you need to type the following:

```
a = [1 2; 2 -1];
b = [4; 3];
x= a \ b
```

which results in: x = 2
1
or **x = 2** and **y = 1**.
a is a matrix or a 2 x 2 array. **b** is a **column vector**. Compare it with a **row vector** `c = [4 3]`.

To change a column vector into a row vector and vice versa, perform the transpose by using the ' symbol as illustrated:
`c = [4 3]'` is equivalent to `c = [4; 3]`.

8. The Mexican hat

The graphics capabilities of MATLAB are very good. Enter the following lines:

```
[x y] = meshgrid(-8 : 0.5 : 8);
r = sqrt(x.^2 + y.^2) + eps;
z = sin(r) ./ r;
mesh(z)
```

Try `surf(z)` to generate a faceted view of the surface. `surfc(z)` or `meshc(z)` draws a 2-D contour plot under the surface.
For better results try the command:

```
surf(z), shading flat
```

9. Handel

If your PC has a speaker (check the volume!), try:

```
load handel
sound(y,Fs)
```

10. MATLAB Earth

May be not Google Earth yet, but …

```
load earth
image(X); colormap(map)
axis image
```

11. MATLAB wisdom

Want to know why, just type **why**

12. Help

Open the help browser and learn from the examples.

Now you are ready to do some real work with MATLAB. For more information look at the publisher's web site: www.mathworks.com; and later look into its MathCentral for lots of contributes codes and useful info. Later, do a quick google search of "matlab"; look at some of the tutorials & simple exercises available online – eg: MIT, U Sydney, UQ, etc.

References:
Brian D Hahn, "Essential MATLAB for Scientists and Engineers" (Elsevier, 2nd Ed. 2002).
E. Part-Enander & A. Sjoberg, "The Matlab 5 handbook", (Addison Wesley, 1988).

There are rows of books in the UQ PSE library – take a walk around and look.

Acknowledgements:
Dr. Nicole Bordes, SPS –for helpful and creative suggestions – so look into MATH2200.