

---

### Workpackage 3 Complex Numbers (MATLAB Exercise 1)

---

Matlab is an advanced interactive software package specially designed for scientific and engineering numerical computation. It is the one of the software tools available in the Department of Electronic and Electrical Engineering and you will be using it throughout your degree.

In this laboratory you will investigate some of Matlab's basic functions, in application to both real and complex numbers, and also discover some of the help facilities that are available in Matlab.

The aim of this laboratory is not to provide you with a comprehensive description of all Matlab's functions and capabilities. It should, however, provide a starting point which will be built on in later Mathematics work packages. We also want to help you to learn how to help yourself by introducing the Matlab online help functions (Matlab's Help Window and Command Line Help) that provide full details on how to use Matlab, all the Matlab functions and much more.

The basic approach is that you will teach yourself the fundamentals of MATLAB as you follow the exercises, by using MATLAB to solve a series of problems associated with the Mathematics course, and at the same time strengthen your understanding of a selection of topics in Mathematics.

Virtually any book on MATLAB is suitable for additional reading, and a number of books are available in the library. For example:

Getting started with MATLAB by Rudra Pratap

MATLAB for engineers by Adrian Biran and Moshe Breiner

Mastering MATLAB 5 a comprehensive tutorial and reference, by D Hanselman and B Littlefield

Introduction to MATLAB 6 for Engineers by William Palm

This last book, while expensive, would be a useful source of reference material throughout all years of your course, as in later years many courses assume a knowledge of MATLAB and recommend that it is used in coursework. Also many final year projects are MATLAB based.

Some of these books have several editions, relating to different versions of Matlab, and although you will be using Matlab 6, books for previous versions of Matlab are still very useful.

#### Objectives

By the end of this experiment you should be able to:

- *start and exit Matlab;*
- *perform basic assignments and calculations;*
- *save your work*
- *use Matlab's online help facility;*
- *apply Matlab to various mathematical problems involving complex numbers.*

#### What must I give in and when?

Details of how to save your work and the filenames to use are given below. **All exercises must be completed by the end of Monday 3<sup>rd</sup> November 2003.**

**Be sure that the diary files complex1.txt, complex2.txt and complex3.txt and the workspace files complex1.mat, complex2.mat and complex3.mat have been saved in your personal directory h:\maths1 before the deadline. It is intended to check student's work for copying (plagiarism) and also check the answers automatically.**

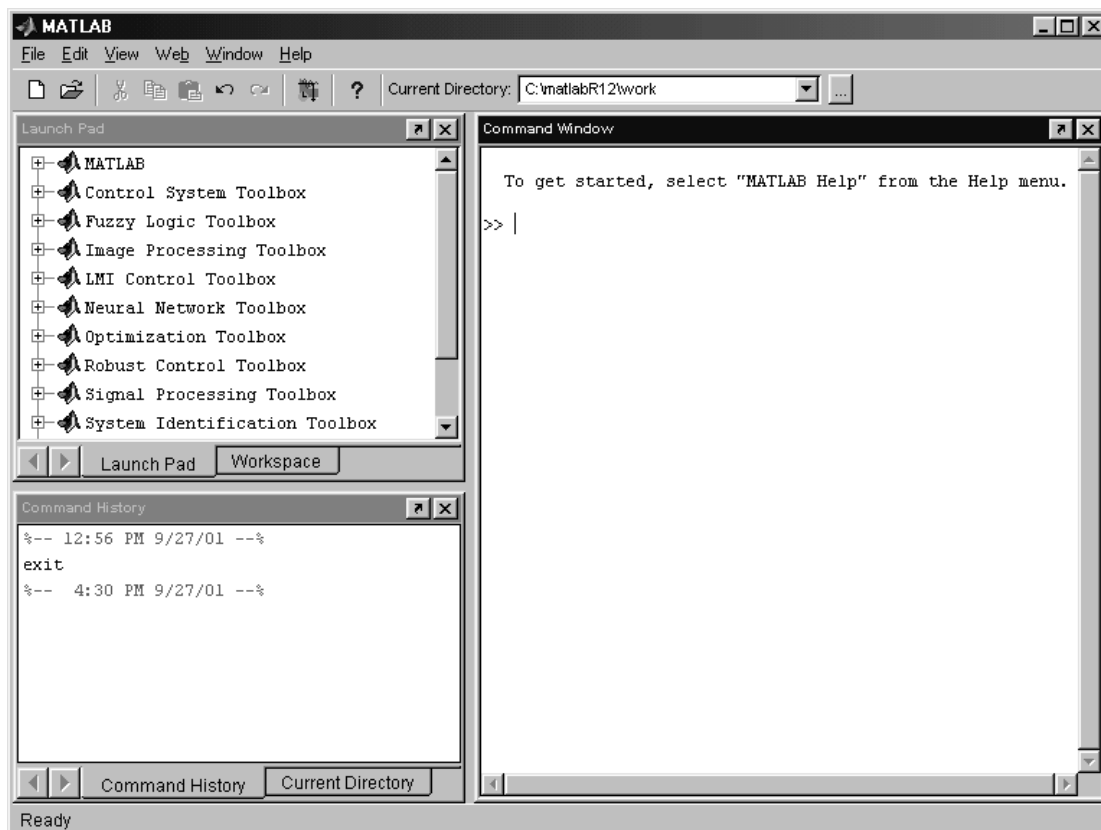
## Laboratory Work

### Getting Started

To start Matlab, click on the Matlab icon



and the following window should appear:



Three windows are displayed: the Launch Pad, the Command History and the Command Window. For now we will only be concerned with the Command Window. To only display this one, from the View menu, select Desktop Layout and then Command Window Only (this can also be done by using the underlined letters; Alt-V, L and C by).

The command prompt, >>, is (somewhat unsurprisingly) where you enter commands. Matlab has lots of demonstrations and help functions. For a quick demonstration of some of the things it can do type *intro* at the command prompt, press *Enter* (↵) and use the buttons to navigate through the demonstration. Many of Matlab's facilities and concepts are introduced by this demonstration some of which you may not be familiar with. If so, do not worry as the key concepts will be covered at a more leisurely pace during the rest of this course.

Matlab can be closed at any time typing *quit* or *exit* at the command prompt, or choosing *exit* on the file menu, either by using the mouse or by *Alt-f* followed by *x*.

### Directory Functions

Matlab has a variety of DOS- and unix-like commands to enable you to keep track of where you are in the directory structure. For example enter

```
>> pwd
```

(print working *directory*) to find out what your current working directory is.

***Ensure that your work is saved in your personal user space. The system should point you at this automatically when you start MATLAB. If it does not, you can set your current directory to your personal H drive by using the command cd h:\***

The contents of the current directory can be found by

```
>> dir
```

It is good practice to have an organised file structure especially if someone else (for example a staff member) is going to examine your files. With this in mind, create a new directory *h:\maths1* by typing

```
>> mkdir maths1
```

and use this for any files that you create during this and future mathematics laboratory sessions. Please note that you are required to use this directory name for the results of this exercise.

### Help Facilities

Matlab has several facilities available to provide help on its commands. Help on any Matlab command can be found by entering the command *help* followed by the name of the command you are interested in. Try

```
>> help dir
```

and investigate the “See also” commands given.

Fuller information can be found from the Help Window which can be accessed by selecting MATLAB Help from the Help menu. In the Help Window, click on the Index tag on the left hand side, type the name of the Command that you are interested in the “Search index for” text box and then click on the result found. Try this for the “dir” command.

The above help facilities are useful if you know the name of the command that you want help for. The *lookfor* command is useful if you do not know the name of the command you want to use. For example, to find out about commands that can contain the word “directory” in their description type

```
>> lookfor directory
```

Alternatively click on the “Search” tab in the Help Window and enter the text in the box.

To see complete lists of Matlab functions click on either the “Contents” tab in the Help Window and then the “Reference” directory under MATLAB and then click on either

- Functions by Category

or

- Alphabetical List of Functions

in the main window on the right.

For example, click on [Functions by Category](#) and then [General Purpose Commands](#) to access information on managing commands and commands for variables, the command window and files.

### Matlab as a Calculator

Matlab has all the mathematical functions that your calculator has (and many more besides). The basic set of arithmetic expressions are

- |    |   |                       |
|----|---|-----------------------|
| 1. | ^ | <b>power</b>          |
| 2. | * | <b>multiplication</b> |
|    | / | <b>division</b>       |
| 3. | + | <b>addition</b>       |
|    | - | <b>subtraction</b>    |

The order of precedence is from 1 to 3 and operations with equal precedence are executed from left to right. Parenthesis can be used to change precedence.

Try using Matlab to evaluate  $2^2 + \frac{3^2}{2}$ ,  $\frac{2^2 + 3^2}{2}$  and  $\frac{(2+3)^2}{2}$ .

Other special arithmetic operators exist, concerned with matrices and you will encounter these later on.

Built-in mathematical functions include  $abs(x)$ ,  $sqrt(x)$ ,  $exp(x)$  etc. in addition to all the trigonometric functions such as  $sin(x)$ ,  $atan(x)$ ,  $cosh(x)$  etc.

Again information on these commands can be found in the Help Window by selecting [Functions by Category](#) then [Operators and Special Characters](#) or selecting [Alphabetical List of Functions](#) and then typing the symbol or function name in the "Find in page" text box.

### Relational and Logical Operators and Functions

Matlab's relational and logical operators are

Relational Operators		Logical Operators	
<	less than	&	and
<=	less than or equal to		or
>	greater than	~	not
>=	greater than or equal to	xor	exclusive or
==	equal to		
~=	not equal to		

In expressions the order of precedence, from highest to lowest, is arithmetic operators, relational operators and finally logical operators. As before, parenthesis can be used to alter precedence.

Try entering the following and hence determine how relational and logical operators present their results (see the Help Window if you require any assistance).

```
>> 6>4
>> 6>8
>> 6>6
>> 6>=6
```

### Variables in Matlab

When evaluating the expressions in the previous section you should have seen something like

```
ans =
```

```
1
```

appear on your screen. In this case the variable *ans* has been set equal to the value of the expression just evaluated. Its value can be viewed by entering *ans* at the command prompt. *ans* is one of several predefined variables in Matlab, that included *pi* and *inf*. You can create your own variables by assigning them a value

```
»variable = expression
```

For example

```
»length = 16.5
```

To stop Matlab repeating the result of the assignment a semi-colon can be used

```
»length = 16.5;
```

Once assigned, variables can be used in subsequent expressions

```
»half_length = length/2;
```

You can view a list of current variables using the *who* command and a more detailed list by *whos* (short for *whosize*) which is particularly useful if you are working with matrices. To remove a particular variable enter

```
»clear variable_name
```

or just *clear* to remove all variables

To repeat and/or edit commands that you have previously entered the arrow keys (↑,↓) can be used. Another useful trick is to type the first few letters of a previously used command and then use the ↑ to find it. For example, type

```
»he
```

and then ↑ to find previous commands starting with *he*, such as *help*.

### Saving Your Work

It is often useful, for example when you have to give in assignments or to save data for future reference, to be able to save your work. Two simple ways of doing this are the *diary* and the *save* commands.

The *diary* command simply records a log of keyboard input and system responses (essentially what you see on the Matlab command window) in an ASCII file **after** the diary has been activated. After it has been created the file can be edited using any text editor such as Notebook or Word. However, as it is only a text file it cannot be input back into Matlab except by copying and pasting.

The *save* command is used for saving all the current variables in a “.mat” file which can be restored to the Matlab workspace by the *load* command. Use the help facilities to find out more about the diary and save commands. **Make sure that you fully understand how they work as they will be used to record your work below.**

## Complex Numbers

For an introduction to complex numbers see the notes for Workpackage 3 Complex Numbers by Dr. D.A.S Rees and Chapter 9 of Engineering Mathematics by Croft, Davison and Hargraves.

Matlab easily accommodates complex numbers. For example enter

```
»c1 = 2+3i  
»c2 = 3-5i
```

and then *whos* to display the variables. The symbol *i* or *j* is used to represent  $\sqrt{-1}$ , although in Matlab *i* is the default used. Try

```
»c3 = -2-4j
```

Problems can sometimes occur if *i* or *j* is overridden and used as a variable (e.g. by *i*=12 or *j*=-3) and you try to define a complex number by *c4* = -2-4\**i* or by *c5* = -3+*i*. To avoid this problem always use the format above (e.g. *c5* = -3+1*i*) or the *complex* command,

```
»c6 = complex(8,2)
```

Normal arithmetic operations can be applied to complex data, try the following examples

```
»c1 + c2  
»c1 - c2  
»c1 * c2  
»c1 | c2
```

The *conj* command can be used to form the **complex conjugate**. Its operation is straightforward, see the Help Window and try the commands

```
»conj(c1)  
»conj(c2)
```

You should now be able to use Matlab to check the solutions to the questions in Exercise 3.1 of your Mathematics notes, thus enabling you to identify any errors in your working that may have occurred. In particular use Matlab to solve the following questions, **using the *diary* command to record the commands used in the file h:\maths1\complex1.txt and the *save* to save the variables in the file h:\maths1\complex1.mat.** Use the variable name Q3\_1a, Q3\_1b etc.

Q3.1

- (a)  $j^3$
- (e)  $j^{1998}$
- (g)  $(1+j)(1-j)$
- (i)  $(2+j)(2+3j)-(2-j)(2-3j)$
- (k)  $(3+j)/(4+3j)$
- (m)  $(1+j)^5$

Matlab recognises when the data or variables that it is dealing with is complex and performs accordingly. This is a powerful feature of Matlab: the same commands can perform in a different manner depending on the format of the data (or operands) that they are presented with. This feature will be more apparent when we look

at arrays and matrices in future workpackages. If you need to know whether the data produced is real or complex the *isreal* command can be used.

The **modulus** and **argument** of a complex number can be found by the *abs* and *angle* commands.

```
»abs(c1)
»angle(c1)
```

To find out more about these commands, just enter them in the “Search index for:” text box in the Help Window.

Note that some relational operators use only the real part of their operands and some use both. Check out the Help Window for full details.

The *abs* and *angle* commands can be used to convert complex numbers from **Cartesian** to **polar** forms. Another way of doing this is to use the *cart2pol* command, in its two-dimensional form.

If the complex number  $c1 = 2+3i$ , then its polar form is given by

```
»[theta r] = cart2pol(2,3)
```

To find the real and imaginary parts of  $c1$  for the above example the commands *real* and *imag* can be used:

```
»[theta r] = cart2pol(real(c1),imag(c1))
```

To ensure that the argument is correct both *angle* and *cart2pol* use the *atan2* function rather than *atan*; see the example in the Help Window for *atan2*.

To convert from polar to Cartesian forms either (1) enter the  $re^{j\theta}$  form on Matlab’s command line. Taking the example  $2e^{j\pi/4}$  from page 28 of your Mathematics notes gives

```
»c7 = 2*exp(1i*pi/4)
```

or (2) use the *pol2cart* command

```
»[x ,y] =pol2cart(pi/4,2)
»c8 = complex(x,y)
```

Note that in the above example *pi* is a function that returns the floating-point number nearest the value of the ratio of a circle's circumference to its diameter.

With the knowledge you have gained you should now be able to check the solutions to Exercises 3.2 and 3.3 of your Mathematics notes. **As before, use the *diary* command to store the commands used to solve the following questions in the file `h:\maths1\complex2.txt`. Use the *save* command to save your variable in the file `h:\maths1\complex2.mat`.** Use the naming convention Q3\_2a, Q3\_2b etc for your variables.

Q3.2

- (a)  $2+3j$
- (c)  $40+9j$

Q3.3

- (a)  $e^{j\pi/3}$
- (b)  $\sqrt{2}e^{3\pi j/4}$