Experiment 101: Introduction to Matlab

Aims

The objects of this experiment are to introduce Matlab as a calculational tool and to help you visualise the geometric significance of operations on complex numbers.

Introduction

In the laboratory part of the Analytical Techniques courses, we shall use computers to help us become familiar with the material introduced in lectures. Analytical techniques are powerful in their generality and abstraction, but it can sometimes be difficult to visualise the significance of complicated mathematical objects. An important use of computers is to provide insight into mathematical concepts by helping us to explore and visualise what these concepts describe in a few specific and well-chosen examples. From these examples, we try to train our mathematical intuition to infer what is likely to happen in more general situations.

For example, by drawing graphs of a few quadratic expressions, we learn that they always have a parabolic shape. The computer can help relieve the drudgery of plotting the graphs by hand. Using a combination of graphical and analytical tools, we soon learn how to predict the position of the vertex of the parabola and various other properties of such graphs. After we gain sufficient familiarity with the ideas in a particular field, it is often no longer necessary to use a computer as we can "see" what the mathematics is telling us.

In order to use computers in this exploratory way, we need to learn how to make a computer carry out tasks such as the basic arithmetic operations, evaluating functions and plotting various types of graph. Sometimes, we shall also want to program the computer to carry out long and repetitive sequences of operations. We have chosen to use the Matlab package (available from www.mathworks.com) for this purpose, as it provides a programming language that is very well sutied to expressing mathematical ideas in a compact form. We shall be introducing how to use Matlab and how to write computer programs as we proceed, so do not be too concerned if you have not written computer programs before.

Remember: The examples you encounter in these experiments are not ends in themselves, but a means of training your mathematical intuition. Always feel free to investigate further, and please **ask** the demonstrators if you'd like help.

Starting Matlab

Before you can use any of the computers, you will need to login to the system using your UPI and password. Once you have reached the Desktop and the computer is ready to use, you can start up Matlab by clicking on the Start button at the bottom-left and selecting Programs, then Maths & Stats and finally MATLAB. A window should appear titled "MATLAB Command Window". On the lefthand side of this window there should be a `>>` prompt followed by a flashing vertical line cursor indicating where commands will appear when they are typed in. Ask for help if you cannot get to this stage.

After you have finished the experiment, you should shut down Matlab and Windows to prepare the computer for the next user. In order to do this, type `exit` at the Matlab prompt to leave Matlab. Then click on the Start button and select Shut Down. You should end up back at the login prompt that you originally started from.

Using Matlab as a calculator

Perhaps the simplest way of using Matlab is just to treat it as a calculator. If we enter an arithmetic expression at the `>>` prompt, Matlab will print the result of the expression in the command window immediately

after we press the Enter key. Use Matlab to evaluate the following expressions, and write down the results. Check them by hand as well.

(1) `3*7+12*5-3^2`

(2) `3*(7+12)*(5-3)^2`

(3) `-4^4+0.5^-8`

(4) `(-4)^4+0.5^-8`

From these and other examples of your own choice, work out the order (called the *precedence* rank) in which Matlab performs the basic arithmetic operations if parentheses are not used to change the order. Addition and subtraction have the same rank and are performed from left to right in an expression. Similarly multiplication and division have the same rank. What can you say about the relative ranking of unary negation (a leading minus sign) and raising to a power?

Matlab also recognises a large number of mathematical functions such as the square root and trigonometric, exponential and logarithmic functions. For a list of these, type `help elfun` (short for 'elementary functions') at the command prompt. For further help, type `help` and the name of the function (e.g. `help sqrt` for help on the square root function). In order to apply any function to an expression, the argument of the function is enclosed in parentheses, so that we can construct more complicated expressions.

(5) Use Matlab to calculate the square root of π and the square root of $-\pi$. Note that π is represented as `pi`.

Matlab can manipulate complex numbers just as easily as real numbers. The symbol `i` is used to represent the square root of -1. The functions `abs` and `angle` are used to find the modulus and argument of complex numbers.

(6) Enter `(25+25*i)/(3+4*i)` to perform a complex division. Check the result by working the problem by hand.

(7) Determine whether the trigonometric functions `sin`, `cos` and `tan` accept their arguments in degrees or in radians (by trying out a few familiar angles). Use Matlab to verify explicitly that $\sin\theta/\cos\theta = \tan\theta$ when $\theta = 60°$, and write down the steps you performed in order to do this.

In Matlab, it is also possible to store numbers (including the results of previous calculations) in named variables. Valid names start with a letter, and can contain letters, digits and the underscore (_). Thus if we enter `a=1728` at the prompt, the variable `a` is assigned the value 1728 and subsequently contains that value, and we can then enter expressions such as `a*a`, `a/12` or `sqrt(a)`.

(8) After typing `a=1728` at the prompt (and pressing Enter), type the line

```
b=sqrt(a)
```

followed by the line

```
c=sqrt(a);
```

(note the semicolon at the end). Describe the difference in what happens. Type `c` on a line by itself, and confirm that the value is what you would expect.

The value in a variable remains until it is changed by a further assignment. The command `clear a` may be used to clear the value of the variable `a` and `clear all` may be used to clear all the variables.

When an expression is evaluated, and the result is *not* assigned to a named variable, its value is assigned to the special variable `ans`. This may be used in the subsequent expression.

Optional: Try entering some expression that have undefined values, such as `1/0`, `-5/0`, `0/0` and `log(0)`, and notice what Matlab returns.

Plotting Graphs

Consider the steps we go through to plot the graph of a function by hand. For example, consider graphing

$$y = x^2 - 5x + 6$$

We first choose a range of values of x of interest. Suppose we decide to use $x = 0$ to $x = 5$. We then pick some values in the range and make up a table, calculating the value that the function takes for each of the chosen values of x. In this example, the table may take the form:

x	0	1	2	3	4	5
y	6	2	0	0	2	6

We then mark the points (0,6), (1,2), ..., (5,6) on a piece of graph paper and draw a curve through these points.

Using Matlab we follow a similar procedure, except that we deal with the collection of the x values as a single object. At the Matlab prompt, enter the line

```
x = 0:1:5
```

After pressing Enter, notice that Matlab responds by stating that the variable `x` now contains the row of values `0 1 2 3 4 5`. In Matlab, variables can contain more complicated objects than single numbers. In order to obtain a row of equally spaced numbers, the colon notation `start:increment:end` is used. It the increment is omitted, it is assumed to be `1`, so the above could also have been entered as `x = 0:5` . Next enter the line

```
y = x.^2-5*x+6
```

Don't leave out the full stop preceding the `^` sign. You should find that the result of this calculation is another row of values, namely `6 2 0 0 2 6`, and that this row has been assigned to the variable `y`. In order to understand what has happened, enter the two lines `x.^2` and `5*x` one after another. Notice that each of these is a row of values, the former being `0 1 4 9 6 25`, whose elements are the squares of the elements of `x`, and the latter `0 5 10 15 20 25`, whose elements are five times those of the corresponding elements of `x`. When we type `x.^2-5*x+6` the rows are added element by element to give the final result that is assigned to `y`.

In order to plot the graph, enter the line `plot(x,y,'x')` to see the individual points, or the line `plot(x,y)` to see the points joined by straight line segments.

(9) In order to get a smooth graph, it is necessary to evaluate the function at a large number of points in the interval. Repeat the above steps using points spaced 0.1 apart over the range 0 to 5 for x, and print out your graph of $y = x^2 - 5x + 6$. With the aid of `help` as needed, use the `xlabel`, `ylabel` and `title` commands to label your graph. (Whenever you print a graph as part of these laboratories, please include both your ID number and the number of the question you are answering in the title.)

(10) The lines of code

```
t = 0:0.01:2*pi;
s = sin(t);
plot(t,s,t,s.^2);
```

plot graphs of $\sin t$ and $\sin^2 t$ (i.e. $(\sin t)^2$) on a single set of axes for t taking on values spaced 0.01 apart in the range $[0, 2\pi]$. Starting from this, show how one can plot graphs of $\sin^2 t$, $\cos^2 t$ and $\sin^2 t + \cos^2 t$ on a single set of axes for t taking on the same values as before. Print out your plot and label the curves appropriately.

Manipulating tables of numbers

In Matlab, it is possible to store an entire table of numbers (called a *matrix*) in a variable. The following notation is used to enter a table of numbers, consisting in this case of three rows and four columns:

```
A = [1,2,3,4;5,6,7,8;9,10,11,12]
```

or

```
A = [1 2 3 4;5 6 7 8;9 10 11 12]
```

Type this into the computer and notice that Matlab responds with

```
A =
    1  2  3  4
    5  6  7  8
    9 10 11 12
```

Note that commas or spaces are used to separate columns and semicolons to separate rows. Later in the course we shall see in more details what matrices may be used to represent. For the present, think of them simply as a convenient way of collecting numbers together. For example, while plotting graphs above, we found it convenient to store all the x values together as a row and all the y values together as another row.

Arithmetic operations can be carried out between tables of the same shape. Addition and subtraction are represented by `+` and `-` as usual, but multiplication, division and raising to a power are represented by `.*`, `./` and `.^` respectively. The full-stops indicate that the operations are to be carried out element by element. Thus

```
[1,2,3].*[4,5,6]
```

produces

```
4 10 18
```

If you leave out the full-stop, an error results, since the symbol `*` by itself represents a different operation, called *matrix multiplication*, which cannot be performed between these two matrices. We will cover matrix multiplication later in the course.

We can however use plain `*` and `/` (without the leading full-stop) if one of the quantities involved is a single number rather than a table. In this case, the operation is performed on each element of the table combined with the single number.

Once we have a table of numbers, such as `A` in the above example, we can examine or change individual elements within the table. We refer to an element in a table by specifying its row followed by its column. For example, if we now type `A(2,3)` Matlab will produce the result `7`, since that is the element in the second row and third column of `A`.

(11) Ensure that you have entered the table of numbers `A` given above. Enter `A(2,3)=15`, and explain what Matlab has done as a result of this command.

Writing programs in Matlab

A program (note the use of US-style spelling for this meaning of the word) is simply a sequence of instructions for the computer to perform. So far we have been issuing instructions to Matlab one by one, by typing them at the command prompt. Every time we press the 'Enter' key the instruction is obeyed and the prompt reappears. When we write a program, we prepare a sequence of instructions and store this in a *file*. Writing a program is a good idea if the computer is going to take a long time to complete the calculation, or if we want to repeat a very similar set of instructions again and again. By preparing the instructions beforehand and storing them in a file, we can carry out the entire sequence just by typing the name of the file at the Matlab prompt.

Follow these steps to create a Matlab program (also called a script or an M-file):

(a) Select the 'New M-file' item from the Matlab window's File menu. This will open another window labelled 'MATLAB Editor/Debugger' that contains a new document called Untitled1.

(b) Within this new document, type in the instructions for the program. You can enter any of the commands that are allowed at the command prompt.

For this example, enter the following lines, which should plot the graphs $\sin t$, $\cos t$ and $2\sin t\cos t$ on the same set of axes when the program is run.

```
%This is a script file to plot three graphs
t = 0:0.01:2*pi;
s = sin(t); c = cos(t);
plot(t,s,t,c,t,2*s.*c);
```

Note that the `%` character introduces a comment; any characters following a `%` are for the reader's benefit and will be ignored by Matlab. While you are typing commands into the editor rather than at the prompt, Matlab does not obey them immediately. They are obeyed after the file has been saved and the name of the file is typed at the prompt.

(c) Select the 'Save As' item from the Editor window's File menu. Enter the filename `script1.m` and click on the 'Save' button. This will create the file in your personal file space.

Return to the Command window and type `script1` at the `>>` prompt. This will cause the commands in the file `script1.m` to be obeyed. If instead you type `help script1` any comments at the start of the file are printed out, while entering `type script1` causes the whole contents of the file to be displayed.

(12) Modify the file `script1` so that it plots out the graphs $2\sin t\cos t$ and $\cos^2 t - \sin^2 t$ on the same axes for t in the interval 0 to 2π. Include commands to label the horizontal axis "Time" and the vertical axis "Value". Print out your graph and include it in your report (don't forget to include your ID and the question number in the graph title!).

A more useful program will interact with the user while it is running. This is done with the help of the `input` statement. Enter the following program and save it as the file `script2.m`

```
fprintf('This plots a graph of A*cos(w*(t-t0))\n');
A = input('Value of A? ');
w = input('Value of w? ');
t0 = input('Value of t0? ');
t = 0:0.01:2*pi;
plot(t,A*cos(w*(t-t0)));
```

The `\n` in the `fprintf` statement starts a new line.

(13) Use `script2`, entering different values of `A`, `w` and `t0` in order to see their effects on the graph. In your report, write down two different sets of values of `A`, `w` and `t0` that will produce the graph shown on the back of your answer sheet. Explain how you worked out these parameter values.

Visualising complex operations

In the final part of this experiment, we shall run a prewritten Matlab program called `complx1` that is designed to illustrate graphically arithmetic operations on the complex plane (also called the Argand plane).

At the Matlab prompt, type in the command `complx1` (you may need to get the file `complx1.m` from the course website first). When the program starts, you should see a window titled "Argand plane explorer". A complex number is represented by a point on the Argand plane whose horizontal and vertical coordinates are the real and imaginary parts respectively of the complex number. In the window, three complex numbers are shown with lines of different colour joining each to the origin. At the right, each of the three is written in real and imaginary form as well as polar (modulus and argument) form. For example the red complex number is $3.0 - 1.0i$; its modulus is $\sqrt{3^2 + (-1)^2} \approx 3.2$ while its argument is $\tan^{-1}\left(-\frac{1}{3}\right) \approx -0.3$radians. Between the red and green numbers is a collection of five radio buttons, from which you can select one of the arithmetic operations of addition, subtraction, multiplication, division and exponentiation. The selected operation is performed on the red and green complex numbers, producing the result that is shown by the blue complex number.

You can change the red and green complex numbers by using the mouse. Point to the current position of the number in the Argand diagram, then click and drag (using the left mouse button) to move the number to any other place within the diagram.

(14) As an example, set the red complex number to $2 + 3i$ and the green one to $1 + i$. Select each of the operations except exponentiation in turn and write down the result (i.e. the blue number). Show that the results are as you expect by also computing the answers by hand. (Remember that to divide by a complex number, you should multiply top and bottom lines by the divisor's complex conjugate, so that the bottom line becomes a real number.)

(15) Notice that the operation of complex addition on the Argand plane is equivalent to using the parallelogram method for adding two-dimensional vectors. The real parts and imaginary parts are separately added together to give the resulting complex number. Ensure that the "plus" operation is selected. Set the green complex number to $-2 + i$ and tick the checkbox "Enable trails". Click on the red complex number and *slowly* drag it around to form a closed figure of some arbitrary shape. Describe how the path traced out by the blue number can be related to that of the red number.

(16) Turn off the checkbox "Enable trails", and select the operation "minus". Set the red number to $3 + i$ and turn "Enable trails" back on. Slowly drag the green number around some path. Describe how the path traced out by the blue number can be related to that of the green number.

(17) Turn "Enable trails" off, and select the operation "times". Set the green complex number to i and move the red one to various positions. Note that if the red number is $x + iy$, the blue number is $-y + ix$. How do the modulus and argument of the blue number relate to those of the red number? Enable trails, and slowly drag the red number around some path. Describe how the path traced out by the blue number can be related to that of the red number.

(18) Turn trails off, and set the green complex number to $1 + 2i$, leaving the operation set to "times". Reenable trails, and slowly drag the red number around some path. Describe how the path traced out by the blue number can be related to that of the red number. Compare them in terms of both their relative sizes and their orientation.

(19) Turn trails off, and set the red complex number to 2.8, which is close to the value of $e \approx 2.71828\ldots$. Select the operation "to the power of", and move the green complex number slowly along the imaginary axis from $-5i$ to $+5i$. Describe what happens to the blue number, explaining the behaviour of its modulus, argument, real part and imaginary part.

(20) With the red number still set to 2.8, move the green number along a line parallel to the imaginary axis, with a constant real part of 1. Again explain what happens to the blue complex number.