PH2130 Mathematical Methods
Lab 1

Introduction to *MATHEMATICA*

Overview

Mathematica is designed as an integrated scientific computer environment. The facilities that it offers include: • computer algebra (and calculus), • arithmetic to arbitrary precision, • high quality graphical displays in two and three dimensions, • scientific text editing, including equations.

The aim of this Module is to familiarize you with the facilities of *Mathematica*, so that you will appreciate its power in the solution of problems. When you have completed the module it is hoped that you will want to use *Mathematica* when working on exercises and course-work from other courses.

Your *Mathematica* session is displayed and stored in the form of a *Mathematica notebook*. As well as input and output expressions it can contain comment text and it can be structured into sections and subsections. You don't need to gain mastery of this to get started using *Mathematica* but at the very least, the insertion of comments is useful. Some instructions on working with Mathematica notebooks are included at the end of this script. It is suggested that when you have got to the end of the exercises you study the notebook instructions and tidy-up and beautify your notebook.

Computer algebra

Central to the *Mathematica* system is computer algebra. You are used to the idea that computers can be used to carry out numerical calculations. Thus entering the sum `2+3` (in an appropriate computer language) will return the answer `5`. You may not be aware that computers can also be used to perform algebraic calculations. Thus you might be surprised that, using appropriate software, entering the expression `(a+b)^2` can give the response `a^2+2*a*b+b^2`, or even $a^2 + 2ab + b^2$. However when you realize that most mathematical operations follow clearly defined rules, and that a computer can be programmed to carry out any specified sequence of operations, then it is perhaps surprising that the use of computers in symbolic algebra and calculus is not more widely known.

Computer algebra stands today in a similar position to that of numerical calculators (computer arithmetic) of twenty years ago. And it is likely that in another twenty years pocket calculators will have a wide range of algebra functions supplementing the usual arithmetic facilities. Those of us who use computer algebra routinely, cannot imagine tackling the same tasks without it; others, who do not use it, feel that for students to be made aware of it is not quite playing the game. You should be able to judge for yourself when you have completed the *Mathematica* module.

Although ample documentation on *Mathematica* is available, both on-line and in printed form, you are strongly recommended to compile your own instruction manual. This should summarize the commands you use, covering the pitfalls you have encountered.

Give examples of input and output for cases where you got what you expected and for those cases where you were surprised. As you become experienced in using *Mathematica* you will discover useful facilities, sometimes by chance. Be sure to write these down for your future reference.

Starting up *Mathematica*
Mathematica is loaded, on the College PCs, from the Physics network server. When you have logged-on to your computer, from the Start button follow the route:

Start → Programs → Academic Applications → Physics →
→ Mathematica 4.0 → Mathematica 4.0.

You might be asked for *Mathematica*'s password. If so you should type in the following number:

55898-10669-47239-49844-53548-616

When you start *Mathematica* running you will see a menu bar at the top of the screen and a new notebook will open up on the left hand side. You might also see the Basic input palette on the right hand side of the screen; you can ignore this for the present.

As with most Windows programs, *Mathematica* has a sophisticated on-line help system. By clicking on “Help” on the menu bar you can gain access to extensive information. You scan through the “Getting Started/Demos” tutorial, or you can find anything you wish in “The *Mathematica* Book”. Other things are available as well as you will see, such as information on various available Add-ons. For a simple query you can just type ? followed by a the topic name in your notebook, but only limited information is returned in this way.

A Brief Tour of *Mathematica*

In order to familiarise yourself with *Mathematica* you should work through the following exercises.

Type each command into the *Mathematica* Notebook window and terminate it by pressing the *Shift* key together with the *Return* key (alternatively, you may terminate the command by pressing the *enter* key on your computer's numeric keypad. You will see that each line of input (and output) is numbered by *Mathematica*. This will be of use if you want to refer to an earlier result later in a session.

As you work through this “tour” you should make notes by hand of what you do together with the results of the operations. Execute all the operations suggested and try to answer (or at least comment intelligently on) the questions posed. Be careful to distinguish between upper case and lower case characters when typing. *Mathematica* is very particular about this. Commands always begin with a capital letter. You must also use the right kind of brackets otherwise *Mathematica* will not understand what you mean. At the end of the “tour” you should discard the notebook before tackling the larger exercises.

Arithmetic

In your fresh *Mathematica* notebook, at the input prompt type

```
        1 + 2 + 3 + 4
```

followed by pressing the *Shift*+*Return* keys. As this is the first command of the session, *Mathematica* will take a few moments while it loads up the "kernel" which does all the work.

You will see that *Mathematica* labels and numbers your input as

```
        In[1] :=  1 + 2 + 3 + 4
```

and the corresponding output appears as

```
        Out[1] = 10
```

The combination of the *Shift*+*Return* keys tells *Mathematica* that you have finished your instruction. If you forget the *Shift* and you press the *Return* key alone, don't worry; simply press *Shift*+*Return* and the command will be executed. You see how you can enter large instructions spanning more than one line; you can start new lines wherever is convenient.

You will see square 'brackets' at the right hand side of the screen showing how the inputs and outputs are grouped. Ordinarily the brackets are blue. However when *Mathematica* is busy evaluating a command then that command's bracket appears black.
You can perform arithmetic operations on decimal fractions:

```
        In[2] :=  1.2 + 2.3 – 3.1
```

which returns the output

```
        Out[2] = 0.4
```

You should appreciate that *Mathematica* is performing an *exact* calculation here. This becomes apparent if we ask it to perform arithmetic operations on rational fractions such as

```
        In[3] :=  1/2 + 2/3
```

This returns the output

```
        Out[3] = 7/6
```

which demonstrates that the answer is being kept in exact form. If you want the answer in decimal fraction representation, then you will have to ask for a floating point numerical approximation, discussed below.

Multiplication may be denoted either by the symbol *, or simply by a space. Thus

```
3*2
```

and

```
3 2
```

will both give the answer `6`.

Floating point approximations

Everything that *Mathematica* has done so far has been totally accurate. It is possible to use floating point approximations using the `N[…]` function. Thus to find the floating point approximation to the previous result type in

```
N[%]
```

Note that the function starts with a capital letter, and that the argument of the function is enclosed in square brackets. This is an important part of the *Mathematica* syntax.

In this expression the symbol `%` is used as a convenient abbreviation for the output of the previous calculation.

Mathematica allows you to specify the precision of numerical calculations. This is done by adding a second argument to the `N[…]` function. Thus of you enter

```
N[Pi, 30]
```

then the value of π to 30 digits is returned. Similarly you might want to find the square root of 2 to 40 digits:

```
N[Sqrt[2], 40]
```

Last and previous result symbols

You saw above how the `%` symbol could be used to refer to the previous answer. A useful generalisation of this is `%n` which refers to the n^{th} output, which will be labeled Out[*n*] = in your notebook.

Special numbers: *π*, *e* and i

Mathematica knows about π and *e*, and i the square root of –1. In accord with the *Mathematica* syntax, all known objects have names starting with a *capital* letter, thus π and *e* are referred to as `Pi` and `E`. Similarly i is referred to by I. You should be careful not to use these symbols for other purposes.

An alternative way of entering these special numbers is by using the Basic Input Palette which should be sitting on the right hand side of the screen. There you will see little buttons for *e*, *i*, π, ∞ as well as Greek and other symbols.

Some simple examples to test are:

```
Sin[Pi]
```

and

`Log[E]` (note this is the *natural* logarithm; there is no `Ln[..]` command)

Note how *Mathematica* gives the *exact* answer; you recall that you must ask specifically if you want a floating point approximation.

You can also enter these using the symbol palette which you should see on the right hand side of the screen. If the palette is not present you can display it by following the menu path:

File → Palettes → 3BasicInputs.

The symbol palette will also allow you to enter Greek characters and it also contains templates for some mathematical structures such as powers, roots, matrices etc.

Functions

You have encountered some of *Mathematica*'s functions already. One of the powers of Mathematica is the *very* large number of functions it "knows about". You must be clear about the syntax of functions; the name starts with a capital letter and the argument(s) go inside square brackets. Other types of brackets will appear later.

Some functions to try are:

`Sin[Pi/2]`	output will be exact
`N[%]`	will give the floating point approximation
`Exp[4]`	output will be exact
`N[%]`	will give the floating point approximation
`Cosh[1]`	doesn't seem to have done much, but as before
`N[%]`	will give the floating point approximation.

Inverse functions are entered as

`ArcSin[y]`	observe the positions of the capital letters

Naming expressions

An expression is given a name, by which it may be referred to later. Thus if you type

```
x = 32.5
```

then *x* will continue to have this value unless it is subsequently changed. You can use longer names if you want to, and generally it makes sense to give your expressions descriptive names. A good practice is to use lower case letters, particularly for the first character to avoid the possibility of a clash with any of *Mathematica*'s names. For example:

```
height = 4
```

Functions of *x* may be found, such as

```
Log[x])
```

or

```
y = Log[x]
```

if you want to name it.

If x has been defined then `Log[x]` return the logarithm of that definition. You can clear this definition by entering `Clear[x].` Test this.

Sums and sequences

The `Sum[…]` function will evaluate sums such as 1 + 2 + 3 + 4 + 5. The syntax for this is

```
Sum[n, {n, 1, 5}]
```

This is telling *Mathematica* to evaluate the sum of *n* from 1 to 5. Observe that the range information is placed in curly brackets. Curly brackets contain *lists*, sequences of elements in a particular order. A vector in component form would be a good example of a list. The sequence of numbers 1, 2, 3, 4, 5 is another example of a list. *Mathematica* would generate such a list using the `Table[…]` function:

```
Table[n, {n, 1, 5}]
```

which gives the output

```
{1, 2, 3, 4, 5}
```

in curly brackets, as expected.

Try printing out the sequence 1^2, 2^2, 3^2, 4^2, 5^2 and evaluating its sum.

Bracket types

There are three types of brackets, which you must be careful to distinguish.

Square brackets […] enclose the arguments of a function.

Round brackets (…) are used in the conventional manner for algebraic expressions, such as

```
Cos[(3 Pi)/4]
```

Curly brackets {…} enclose lists. You can have lists within lists, such as `{1, 2, {3,4},5}`. Thus, for example, a 2×2 matrix would be represented by `{{a,b},{c,d}}`. You can also enter matrices using the Basic Input Palette; try it.

You will want to extract elements from lists. If we specify *list* by

$$\texttt{lis = } \{1^2,\ 2^2,\ 3^2,\ 4^2,\ 5^2\}$$

then the 4th element is found using

```
lis[[4]]
```

which will return `16`

Algebra

It was mentioned above that one of the important features of *Mathematica* is algebraic or *symbolic* calculation. The manipulation of polynomials is an example of this. If you define the polynomial *poly* by

```
poly = x^2 + 5x + 6
```

(actually you don't need the space between the 5 and the *x* for multiplication since *Mathematica* objects cannot start with a number – so multiplication is understood) !

You can factorise the polynomial by typing

```
Factor[poly]
```

which returns the result

```
(2+x) (3+x)
```

This example might be trivial, but in general *Mathematica* does a lot of work behind the scenes in performing factorisation. You will see examples of this when using *Mathematica* for solving equations below.

If you give *Mathematica* a factored polynomial such as

```
facs = (3+x)(5+x)
```

then it will be returned in factored form. If you want it expanded you must explicitly request it:

```
    Expand[%]
```

And this will return

$$15 + 8\,x + x^2$$

Substitution of values into mathematical expressions is a very important aspect of practical calculations. *Mathematica* has a slightly odd syntax for this, but it is worth becoming familiar with this. You have already defined the variable *poly* as $x^2 + 5x + 6$. If you want to substitute $x = -3$ into *poly* (perhaps to test whether $x + 3$ is a factor of the polynomial) then the *Mathematica* input is

```
    poly /. x ->  -3
```

The *substitute* command is `/.` and this is followed by what you want to substitute, but note you must use the arrow `->` (built up from a – and a >) rather than an equals sign =.

Plotting in 2d

To plot the previously defined function you may type either

```
    Plot[poly, {x,0,2}]
```

or

```
    Plot[x^2 + 5x + 6, {x,0,2}]
```

Try them both. The graph appears in your notebook underneath the command. Observe the structure of the `Plot[…]` command. Its first argument is the function to be plotted. The second argument is a *list* enclosed in curly brackets. This list contains information about the plot range. In the above example it is saying that *x* should vary from 0 to 2. This is the standard way of indicating ranges in *Mathematica*; you will encounter it in many other places.

The plot command can take a number of optional extra arguments. One of these is `PlotRange`. You will find this to be useful in the exercises, so investigate what *Help* tells you about it.

Differentiation and integration

Differentiation and indefinite integration are performed with the operators `D` and `Integrate`. To differentiate log *ax* with respect to *x* type in

```
    D[Log[a*x],x]
```

You see that `D[ …]` takes two arguments. The first is the function to be differentiated and the second argument is the differentiation variable. Try some other examples.

The indefinite integral of log *ax* is found, in a similar way, by typing

```
    Integrate[Log[a x],x]
```

Since differentiation and indefinite integration are the inverse of each other, the result of an indefinite integral may be checked by differentiation, to see if the original expression is obtained. This may be done using the command

```
D[%,x]
```

Definite integration

A simple extension of the previous notation gives the expression for definite integration. Thus to find the integral of exp(*ax*) from $x = 0$ to $x = 1/a$ one types

```
Integrate[Exp[a x], {x,0,1/a}]
```

Here observe that the second argument is the list `{x,0,1/a}`, specifying the range of the integration.

These examples are all perfectly straightforward. Apart from not making careless errors, the real power of a package such as *Mathematica* is in performing *difficult* calculations. For instance try

```
Integrate[Sin[a/x]- Cos[x^2],x]
```

You see this gives quite a complicated result. Although at this stage of your scientific career you don't know about the cosine integral function or the Fresnel function, they can be looked up in a book if needed. However part of the power of *Mathematica* is that it *does* know about these functions – both mathematical properties and numerical values.

Infinity

Mathematica knows about infinity. This can be used in integral limits, such as

```
Integrate[1/(x^2+a^2), {x,0,Infinity}]
```

The result of this input is presented in a rather complicated form. See if you can decipher what *Mathematica* is telling you.

In accord with the *Mathematica* syntax, all known objects have names starting with a capital letter, thus `Infinity`. Similarly, as mentioned already, π and e are referred to as `Pi` and `E`. You can also input ∞ from the basic Input Palette.

Solving equations

We shall now see how *Mathematica* can be used for solving equations. First a simple one:

```
Solve[x^2-5x+6==0,x]
```

You see that `Solve` takes two arguments: the equation to be solved and the variable to be solved for. Note that the equals sign of the equation uses the double symbol `==` to indicate *equality* rather than *assignment*. (Other languages use different conventions to distinguish these; Pascal uses := for assignment and = for equality whereas Fortran uses = for assignment and *.eq.* for equality) Remember, in the above example, that 5*x* may be written without either a * or a space between.

Now try this more complicated equation (make sure you type it correctly)

```
Solve[x^6-21x^5+175x^4-735x^3+1624x^2-1764x+720==0,x]
```

The solution is simple; you should get

```
{{x→1}, {x→2}, {x→3}, {x→4}, {x→5}, {x→6}}
```

But in general the solutions to higher order polynomial equations are impossible or very complicated. Try the very similar (use *Windows* cut and paste facility to save typing all again):

```
Solve[x^6-21x^5+175x^4-735x^3+1624x^2-1764x+719==0,x]
```

The solution is *possible*, but very complicated.

You should spend some time investigating this solution. It will help to inspect a numerical approximation to the roots. The important point is to see how the roots of this equation differ from the roots of the rather similar previous equation.

You might start investigating by using the command N[%]. You will observe the appearance of I, the square root of –1. But you might be suspicious as the complex roots don’t occur in complex conjugate pairs. Maybe the (very small) imaginary parts of the roots have come from rounding errors. A better way to look for the numerical solution for the roots is to use the NSolve[..] command. What do you conclude?

Simultaneous equations

The following line will result in the solution of a pair of simultaneous equations.

```
Solve[{x+2y==2, 3x-y==4}, {x,y}]
```

Note that `Solve[…]` still has two arguments. In this case the arguments are *lists*, enclosed in curly brackets. The first list is the set of equations to be solved and the second list is the set of variables to be solved for.

Look carefully at the results you get for this set of equations. Is it what you expect? If not then why not? You see that early on you had assigned *y* the value sin*x*. You can remove this assignation by typing

```
Clear[y]
```

Try this and then solve the simultaneous equations again.

You will now realize that it is a good idea to keep track of your variables. When in doubt, a new problem should be started in a fresh notebook.

It is also possible to solve some nonlinear systems of equations. You can demonstrate this ability on the following set

```
Solve[{x^2 + y^2 == 25, x^2-5 == y}, {x,y}]
```

but in general the solution of such equations can be very complicated.

Numerical solutions

Try solving the following equation

```
Solve[Cos[x]-x/10==0, x]
```

You will see that an exact (analytic) solution is beyond *Mathematica*'s powers. But does this equation *have* a solution? Let us plot it and see. The graph is given by

```
Plot[Cos[x]-x/10, {x,-15,15}]
```

and the solutions correspond to where the curve crosses the *x* axis. The plot range `{x,-15,15}` is sufficient to see that there are actually 7 roots. No analytic solution is available, but *Mathematica* can find numerical solutions (to whatever accuracy we desire) for the roots of an equation. The command we use is `FindRoot[…]`. Try

```
FindRoot[Cos[x]-x/10==0, {x,1} ]
```

This finds a numerical value for a root of the equation starting at *x*=1. You see that `FindRoot` takes two arguments. The first is the equation to be solved and the second argument is a list detailing where to look for a solution. Experiment with different starting values to find all roots of the equation. You see the value of visualising the equation to be solved; immediately you know how many roots there are.

Use *Windows'* help facility in *Mathematica* to explore some of the other options in specifying the list detailing where to look for a solution.

Plotting in three dimensions

Mathematica has a sophisticated collection of graphics facilities. Three dimensional plots are created with the command `Plot3D[…]`. The following is a simple example

```
Plot3D[x^3-3x y^2, {x,-1,1}, {y,-1,1}]
```

be sure that understand that the function here is $x^3 - 3xy^2$; there is no space between the 3 and the *x*, but there must be a space between the *x* and the *y*. Why?

By this stage the structure of the *Mathematica* commands should be familiar to you.

It is possible to change the orientation of 3d plots in *Mathematica*. You can give `Plot3D[…]` an optional argument `ViewPoint -> {x, y, z}` where `x, y, z` specify the position of the viewpoint. Thus you might type in

```
 Plot3D[x^3-3x y^2, {x,-1,1}, {y,-1,1}, ViewPoint->{1.3, -
2.4, 2}]
```

which actually gives the default viewpoint. The help browser gives more information on `ViewPoint`'s arguments, but there is a tool to allow the viewpoint to be selected graphically. You access the *3D ViewPoint Selector* from the Input icon of the *Mathematica* menu bar. You can drag the wireframe box around to the required orientation. Then clicking on Paste will insert the ViewPoint option into your notebook – so you should be sure the cursor is in the correct position before you paste. There are many other graphics display options which you might also like to investigate.

Mathematica's ViewPoint selector is rather clumsy in use. See if you can work out a convenient and simple way to make use of it.

End

That completes the brief tour of *Mathematica*'s facilities. You should appreciate that *Mathematica* is a very powerful system and you have seen only a small part of what it can do. Further information is available from the Help menu and in the documentation. It is recommended that you familiarise yourself with the Help before you proceed to the set of exercises.

Using *Mathematica* Notebooks

For formatting notebooks (into sections), adding comment text etc. it is convenient to display and use the formatting toolbar. This is turned on by following the menu tree Format → Show ToolBar.

To insert some text commentary above or below *Mathematica* input or output cells:

1 Click cursor at end (beginning) of the line below (above) which you want the text to appear.

2 Press arrow arrow right → (left ←) key.
Or move the cursor vertically between the two cells where you want to insert the text. The cursor will change from a vertical bar to a horizontal bar. Click when it is horizontal.

3 A line will appear separating the lines between which you want to place your text.

4 Select "Text" or "SmallText" from the text style window or from the Format → Style menu tree.

5 Type in required text, including returns if required.

6 Pressing the arrow right → key will move you to a new cell and the style will be set to "Input" by default.

When you become a "power user" you will want to use the Alt + *n* keys to change the styles.

Alt + 1	Title
Alt + 2	Subtitle
Alt + 3	Subsubtitle
Alt + 4	Section
Alt + 5	Subsection
Alt + 6	SubSubsection
Alt + 7	**Text**
Alt + 8	Smalltext
Alt + 9	Input

You can change the style of a section if you first select it by the brackets at the right hand side of the screen, and then pick the style you want. You can't do it by selecting the body of the text with the cursor.

To insert math formulae in a text cell:

Math and various other symbols may be entered as "typeset" objects in text cells by using the various palettes, displayed through the File → Palettes menu tree. The BasicInput palette is of greatest use.

To insert math formulae (*Mathematica* book pp. 194-5):

1 In the text cell press ctrl + (

2 Enter the mathematical formulae

3 Terminate the mathematics by typing ctrl +) or press the arrow right → key.