0
A Tutorial Introduction to Maple

Aims and Objectives

- To provide a tutorial guide to the Maple package.
- To give practical experience in using the package.
- To promote self-help using the on-line help facilities.

On completion of this chapter, the reader should be able to

- use Maple as a mathematical tool;
- produce simple Maple programs;
- access some Maple commands and programs over the Web.

It is assumed that the reader is familiar with either the *Windows* or *Unix* environment.

Commands listed in Sections 0.1 and 0.2 have been chosen to allow the reader to become familiar with Maple in a few hours. These tutorial sheets have been used with great success over a number of years with both mathematics and engineering undergraduate students. Experience has shown that the Maple worksheets can be completed in under two hours, after which students are able to adapt the commands to tackle their own problems. This method of teaching works well with computer laboratory class sizes of no more than 20 students to one staff member. Section 0.3 gives a brief introduction to programming with Maple.

If any problems result, there are several options. For example, there is an excellent help browser in Maple, the 10 most common errors are listed in Section 0.4, and Maple commands and programs with the respective output from this text can be found on the Web at

http://www.birkhauser.com/cgi-win/ISBN/0-8176-4150-5

or

http://www.maplesoft.com/apps/.

The Maple worksheets on the Web may be edited and copied.

Remember to save your Maple files at regular intervals. You could label your first file as *tut1.mws*, for example.

0.1 Tutorial One: The Basics (One Hour)

There is no need to copy the comments; they are there to help you.

Click on the Maple icon and copy the command after the > prompt.

Maple Commands	**Comments**
`> # This is a comment`	`# Helps when writing` `# programs.`
`> 1+2-3;`	`# Simple addition and` `# subtraction.`
`> 2*3/7;`	`# Multiplication and` `# division.`
`> 2*6+3^2-4/2;`	
`> (5+3)*(4-2);`	
`> sqrt(100);`	`# The square root.`
`> n1:=10:`	`# The colon suppresses` `# the output.`
`` > lprint(`n1:=`,n1): ``	`` # Use the ` character `` `# for quotes.`
`> n1^(-1);`	`# Negative powers.`
`> sin(Pi/3);`	`# Use capital P for Pi.`
`> y:=sin(x)+3*x^2;`	`# Equations and` `# assignments.`
`> evalf(sin(Pi/3));`	`# Evaluate as a floating` `# point number.`

```
> diff(y,x);                         # Differentiate y with
                                     # respect to x.
> y:='y':                            # Set y back equal to y.
> diff(x^3*y^2,x$1,y$2);             # Partial differentiation.
> int(cos(x),x);                     # Integration with
                                     # respect to x.
> int(x/(x^3-1),x=0..1);             # Definite integrals.
> int(1/x,x=1..infinity);            # Improper integrals.
> convert(1/((s+1)*(s+2)),parfrac,s);
                                     # Split into partial
                                     # fractions.
> expand(sin(x+y));                  # Expansion.
> factor(x^2-y^2);                   # Factorization.
> limit((cos(x)-1)/x,x=0);           # The limit as x goes
                                     # to zero.
> z1:=3+2*I;z2:=2-I;                 # Complex numbers. Use
                                     # I NOT i.
> z3:=z1+z2;
> z4:=z1*z2/z3;
> modz1:=abs(z1);                    # Modulus of a complex
                                     # number.
> evalc(exp(I*z1));                  # Evaluate as a complex
                                     # number.
> solve({x+2*y=1,x-y=3},{x,y});      # Solve two simultaneous
                                     # equations.
> fsolve(x*cos(x)=0,x=7..9);         # Find a root in a given
                                     # interval.
> S:=sum(i^2,i=1..n);                # A finite sum.
> ?linalg                            # Open a help page.
> with(linalg):                      # Load the linear
                                     # algebra package.
> A:=matrix([[1,2],[3,4]]);          # Defining 2 by 2
> B:=matrix([[1,0],[-1,3]]);         # matrices.
> evalm(B^(-1));                     # Matrix inverse.
> C:=evalm(A+2*B);                   # Evaluate the new
```

```
                                    # matrix.
> AB:=evalm(A &* B);                # Matrix multiplication.
> A1:=matrix([[1,0,4],[0,2,0],[3,1,-3]]);
> det(A1);                          # The determinant.
> eigenvals(A1);                    # Gives the eigenvalues
                                    # of A1.
> ?eigenvects                       # Shows how the eigen-
                                    # vectors are displayed.
> eigenvects(A1);                   # Gives the eigenvectors
                                    # of A1.
> # Use of the help browser - one option.
> ?interp                           # Open a help page for
                                    # interpolation.
>??interp                           # List the syntax for
                                    # this command.
>???interp                          # List some examples.
> # End of Tutorial One.
```

Exit the Maple worksheet by clicking on the **File** and **Exit** buttons, but remember to save your work.

0.2 Tutorial Two: Plots and Differential Equations (One Hour)

There is no need to copy the comments, they are there to help you.

Click on the Maple icon and copy the command after the > prompt.

Maple Commands **Comments**

```
> ?plot                             # Open a help page.
> with(plots):                      # Load the plots package.
> plot(cos(2*x),x=0..4*Pi);         # Plot a trigonometric
                                    # function.
> plot(x*(x^2-1),x=-3..3,y=-10..10,
> title=`A cubic polynomial`);      # Plot a cubic polynomial
                                    # and add a title.
> plot(tan(x),x=-2*Pi..2*Pi,y=-10..10,
```

```
> discont=true);                        # Plot a function with
                                        # discontinuities.

> plot({x*cos(x),x-2},x=-5..5);        # Plot two curves on one
                                        # graph.

> c1:=plot(sin(x),x=-2*Pi..2*Pi,
> linestyle=1):
> c2:=plot(2*sin(2*x-Pi/2),x=-2*Pi..2*Pi,
> linestyle=3):
> display({c1,c2});

> points:=[[n,sin(n)]$n=1..10]:        # Plot points and lines
> pointplot(points,style=point,        # joining the points on
> symbol=circle);                       # two separate graphs.
> pointplot(points,style=line);

> implicitplot(y^2+y=x^3-x,x=-2..3,
> y=-3..3);                             # Implicit plots.

> animate(sin(x*t),x=-4*Pi..4*Pi,t=0..1,
> color=red);                           # 2-D animation.

> plot3d(sin(x)*exp(-y),x=0..Pi,y=0..3,
> axes=boxed);                          # 3-D plots. You can
                                        # rotate the figure
                                        # with the left mouse
                                        # button.

> cylinderplot(z+3*cos(2*theta),
> theta=0..Pi,z=0..3);

> animate3d(t*y^2/2-x^2/2+x^4/4,x=-2..2,
> y=-2..2,t=0..2);                      # 3-D animation.

> ?DEtools                              # Open a help page.

> with(DEtools):                        # Load the differential
                                        # equations package.

> dsolve(diff(y(x),x)=x,y(x));          # Solve a differential
                                        # equation.

> dsolve({diff(v(t),t)+2*t=0,v(1)=5},
> v(t));                                # Solve an initial value
                                        # problem.

> dsolve(diff(x(t),t$2)+8*diff(x(t),t)
> +25*x(t)=0,x(t));                     # Solve second-order
                                        # differential equations.

> dsolve(diff(x(t),t$2)+8*diff(x(t),t)
> +25*x(t)=t*exp(t),x(t));
```

```
> deqn:=diff(y(x),x$2)=x^3*y(x)+1;

> DEplot(deqn,y(x),x=-3..2,          # Plot a solution curve.
> [[y(0)=0.5,D(y)(0)=1]]);

> # Differential equations will be considered in more detail
> # in Chapter 1.

> # End of Tutorial Two.
```

0.3 Simple Maple Programs

Programming in Maple is much simpler than programming in some other languages. The Maple language contains powerful commands, which means that some complex programs may contain only a few lines of code. Of course, the only way to learn programming is to sit down and try it yourself. The aim of this section is to introduce simple programming techniques by example. The programs are kept short to aid in understanding; the output is also included.

Procedures. You can create your own procedures. For example, the command **norm3d** below gives the norm of a three-dimensional vector.

```
> # The norm of a vector in three-dimensional space.
> norm3d:=proc(a,b,c)
>         sqrt(a^2+b^2+c^2);
>         end;
>
          norm3d := proc(a, b, c) sqrt(a^2 + b^2 + c^2) end

> norm3d(3,4,5);
                              5 sqrt(2)
```

The for..do..od loop. This type of command is used in most languages.

```
> # A program to sum the natural numbers from 1 to imax.
> # Note that the do must be ended with an od:
> i:='i':total:=0:
>     for i from 0 to 100 do
>     total:=i+total:
>     od:
> total;

                                 5050
```

Conditional statements. If, then, elif, else, etc.

```
> # A simple program - note that if must be ended with a fi:
> p:=4:
> if p<2 then lprint(`p is less than 2`):
>        elif p>=2 then lprint(`p is not less than 2`):
>        fi:
```

```
p is not less than 2
```

Arrays and sequences. Set up an array; F in this case can hold up to 10001 elements. This is a simple program to evaluate the first fifteen terms of the Fibonacci sequence.

```
> # The Fibonacci sequence.
> F:=array(0..10000):F[0]:=0:F[1]:=1:imax:=14:
>    for i from 2 to imax do
>    F[i]:=F[i-1]+F[i-2]:
>    od:
>
> seq(F[i],i=0..imax);

               0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377
```

To conclude this section, some options within plots will be highlighted.

Display. This command can be used to show multiple plots with text. See the examples below.

```
> # A program to plot two functions on one graph.
> with(plots):
> p1:=plot(x^2,x=-4..4,color=blue):
> p2:=plot(4-x^2,x=-4..4,color=red):
> t1:=textplot([2.6,6,`y=x^2`],align=RIGHT):
> t2:=textplot([-3,-6,`y=4-x^2`],align=RIGHT):
> display({p1,p2,t1,t2},font=[TIMES,ROMAN,20],tickmarks=[3,3]);
```

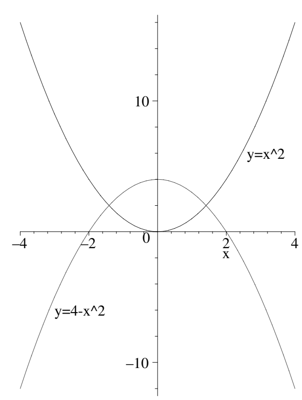

Figure 0.1: Multiple plot with text. (Unfortunately, the colors are missing here.)

```
> # Two solution curves on one graph.
> with(DEtools):with(plots):
> deqn1:=diff(x(t),t$2)=-2*diff(x(t),t)-25*x(t);
```

```
> p1:=DEplot(deqn1,x(t),t=0..10,[[x(0)=1,D(x)(0)=0]],stepsize=0.1,
> linestyle=1,linecolor=black):
> deqn2:=diff(x(t),t$2)=-25*x(t);
> p2:=DEplot(deqn2,x(t),t=0..10,[[x(0)=1,D(x)(0)=0]],stepsize=0.1,
> linestyle=7,linecolor=black):
> t1:=textplot([10,1,`Harmonic motion`],align=RIGHT):
> t2:=textplot([10,0,`Damped motion`],align=RIGHT):
> display({p1,p2,t1,t2},font=[TIMES,ROMAN,15],labels=[`t`,`x`]);
```

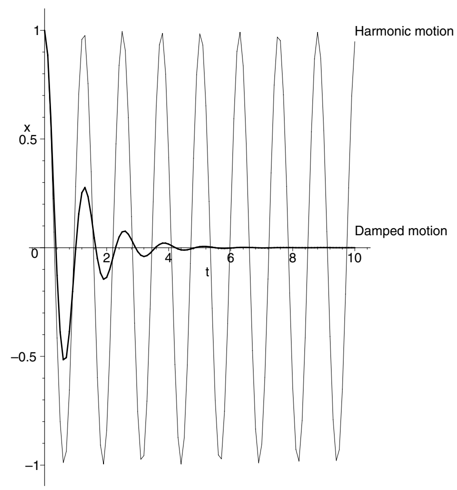

Figure 0.2: Solution curves for differential equations deqn1 and deqn2.

0.4 Common Errors

Do not forget to end the command with either a semicolon (to see the output) or a colon (to suppress the output). Remember to check the help pages within Maple and the Web sites given at the beginning of the chapter if this page does not help you.

The Error	**The Command and Error Message**

`1. Omission of a bracket.`	`>(5+3)*4-2);`
	`syntax error:`
	`` `)` unexpected``

`2. Omission of a colon.`	`>n1=10:` `No error message.`

`3. Negative powers.`	`>n1^-1;`
	`syntax error:`
	`` `-` unexpected``

4.	Use a capital P for the number Pi and a small p for the letter.	>sin(pi/3); No error message.
5.	Quotes - ` or '.	>lprint('n1:=`,n1); syntax error: `:=` unexpected
6.	Omission of multiplication sign.	>y:=sin(x)+3x^2; syntax error missing operator or `;`
7.	Omission of a dot.	>int(x/(x^3-1),x=0.1); error (in int) wrong number (or type) of arguments
8.	Complex numbers - use I not i.	>z1:=3+2*i; No error message.
9.	Brackets.	>A:=matrix([1,2],[3,4]); error (in matrix) 1st and 2nd arguments must be nonnegative integers
10.	Matrix multiplication.	>AB:=evalm(A*B); error (in evalm/evaluate) use the &* operator for matrix/vector multiplication

The programs throughout the book should all compile under both Maple V and Maple 6. If you experience difficulties with Maple 6 see the *updstrc* text files, which are located in the directory in which Maple 6 is installed.

0.5 Maple Exercises

1. Evaluate the following:

   (a) $12 + 4 - 5$;

   (b) 2^{10};

   (c) $\sin(0.1)$;

   (d) $((2 + 3)(4 - 3(9 - 5)))$.

2. Find the derivatives of the following functions:

   (a) $y = 3x^3 + 2x^2 - 5$;

   (b) $y = \sqrt{1 + x^4}$;

   (c) $y = e^x \sin x \cos x$.

3. Evaluate the following definite integrals:

(a) $\int_{x=0}^{1} 3x^3 + 2x^2 - 5\,dx$;

(b) $\int_{x=1}^{\infty} \frac{1}{x^2}\,dx$;

(c) $\int_{-\infty}^{\infty} e^{-x^2}\,dx$.

4. Evaluate the following limits:

(a) $\lim_{x\to 0} \frac{\sin x}{x}$;

(b) $\lim_{x\to\infty} \frac{x^3+3x^2-5}{2x^3-7x}$;

(c) $\lim_{x\to\pi} \frac{\cos x+1}{x-\pi}$.

5. Given that $z_1 = 1+i$, $z_2 = 2+3i$, and $z_3 = 4-2i$, evaluate the following:

(a) $z_1 + z_2 - z_3$;

(b) $\frac{z_1 z_2}{z_3}$;

(c) e^{z_1};

(d) $\ln(z_1)$;

(e) $\sin(z_3)$.

6. Load the linear algebra package by typing **with(linalg):**. Given that

$$A = \begin{pmatrix} 1 & 2 & -1 \\ 0 & 1 & 0 \\ 3 & -1 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 2 \\ 0 & 1 & 2 \end{pmatrix}, \quad C = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 1 & -1 \\ 4 & 2 & 2 \end{pmatrix},$$

determine the following:

(a) $2A - BC$;

(b) B^{-1};

(c) the eigenvalues and eigenvectors of C.

7. Load the plots package by typing **with(plots):**. Graph the following:

(a) $y = 3x^3 + 2x^2 - 5$;

(b) $y = e^{-x^2}$ for $-5 \le x \le 5$;

(c) $x^2 - 2xy - y^2 = 1$;

(d) $z = 4x^2 e^y - 2x^4 - e^{4y}$ for $-3 \le x \le 3$ and $-1 \le y \le 1$.

8. Load the differential equations package by typing **with(DEtools):**. Solve the following differential equations:

(a) $\frac{dy}{dx} = \frac{x}{2y}$, given that $y(1) = 1$;

(b) $\frac{d^2x}{dt^2} + 5\frac{dx}{dt} + 6x = 0$, given that $x(0) = 1$ and $\dot{x}(0) = 0$.

9. Carry out 100 iterations on the recurrence relation

$$x_{n+1} = 4x_n(1 - x_n),$$

given that (a) $x_0 = 0.2$ and (b) $x_0 = 0.2001$. List the final 10 iterates in each case.

10. Type **?while** to read the help page on the while command. Use a **while-do-od** loop to program Euclid's algorithm for finding the greatest common divisor of two integers. *Hint:* Ese the **irem** command. Use the program to find the greatest common divisor of 12348 and 14238.

Recommended Textbooks

[1] K. M. Heal, M. Hansen, and K. Rickard, *Maple* 6 *Learning Guide*, Waterloo Maple, Toronto, 2000.

[2] M. B. Monagan, K. O. Geddes, K. M. Heal, G. Labahn, S. M. Vorkoetter, and J. McCarron, *Maple* 6 *Programming Guide*, Waterloo Maple, Toronto, 2000.

[3] M. Abell and J. Brasellon, *Maple* V *by Example*, Second ed., Academic Press, New York, 1998.

[4] K. M. Heal, M. Hansen, and K. Rickard, *Maple* V *Learning Guide for Release* 5, Springer-Verlag, Berlin, New York, Heidelberg, 1997.

[5] M. Kofler, *Maple: An Introduction and Reference*, Addison–Wesley, Reading, MA, 1997.

[6] M. B. Monagan, K. O. Geddes, G. Labahn, and S. Vorkoetter, *Maple* V: *Programming Guide*, Springer-Verlag, Berlin, New York, Heidelberg, 1996.