


CHAPTER 4

Mathematical Modeling with MAPLE

4.1 Just the Basics

MAPLE is an extremely versatile and robust software package. In addition to providing its' users with many “built-in” functions and procedures, it contains its' own programming language. The version of MAPLE used in the bulk of the text is MAPLE V. The most current version of MAPLE is MAPLE 9. It is completely downward compatible. Features requiring a newer version of MAPLE will be indicated.

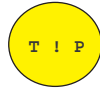
MAPLE may be started by pressing the  button on the WinEdt toolbar. With it a user can readily obtain symbolic solutions to complex algebraic and differential equations, easily create and export publication quality graphics and animations, and can export entire *worksheets* in L^AT_EX or HTML format!


To begin using MAPLE a user should realize that like all programs and programming languages, it has its' own syntax and grammar.¹ The basics are very simple however, and most problems can be avoided if a user pays close attention to the following set of rules:

- i) All executable statements must end with either a semi-colon(;) or colon(:).
- ii) All name assignments should take the form, **name:=expression:.**
- iii) Upon pressing the **enter** key a statement is executed.
- iv) To enter more than a single statement, a user must press **shift-enter**.
- v) All MAPLE commands and names (intrinsic and user-defined) are **C_aS_e sensitive**.

When using the semi-colon the individual statement is also echoed to the screen. The subtlety here can be frustrating, and while printing the commands to the screen may be a good idea when a user wishes to verify the entry, printing the output to the screen can be time intensive. Generally it is a good idea to use the colon and suppress the screen printing. Note however that to view the output from a command “on-screen,” one must use the semi-colon.

¹While MAPLE does provide GUI “palettes” to input many of its functions, all of the examples discussed here will utilize command-line arguments. To access the palettes, choose the View-Palettes option in the drop-down menus.



To avoid printing output to the screen use a semi-colon (;) instead of a colon (:). In the event of a very long calculation or during one that is output intensive, start over by pushing the  button located on the toolbar, and change a semi-colon to a colon.

It is extremely important for new users to remember that all MAPLE commands are case sensitive. While this may seem a trivial point, many minutes (or hours!) can be spent tracking-down a simple error. MAPLE makes available a large number of built-in functions and commands, and the difference between one name and another can be frustrating. As an example of how subtle this problem can be, consider the following MAPLE session:

```
f:=sqrt(pi^2-a^2);
ans:=eval(f,a=4.0);
evalf(ans);
```

$$f := \sqrt{\pi^2 - a^2}$$

$$ans := \sqrt{\pi^2 - 16}$$

$$\sqrt{\pi^2 - 16.00}$$

Notice here that the **eval** function puts the value of a into the expression, and the **evalf** function changes the 16 to a real-valued number, but the entire expression is not evaluated as expected.

The problem here is that “pi” is the name given to the symbol, π and “Pi” is the name given to the floating-point estimation of the numerical constant. To see the difference consider a slightly modified version of the same:

```
f:=sqrt(Pi^2-a^2);
ans:=eval(f,a=4.0);
evalf(ans);
```

$$f := \sqrt{\pi^2 - a^2}$$

$$ans := \sqrt{\pi^2 - 16.00}$$

$$2.475963569 I$$

The lesson here is clear. By using the appropriate name a numerical solution is obtained and MAPLE even knows complex values!

In addition to these basic rules, there are a few commands that are very useful for the beginning user to know. These are given in table 4.1 and will be illustrated throughout the text.

Command	Result
?name	Get help with name.
restart;	Restart the MAPLE Kernel.
quit;	Quit MAPLE . (closes worksheet)
%*	Recall value of previous result.
assume(name,property);	Assign a property to a named expression.
eval(name, assignment);	Substitute an assigned value for a named quantity.
evalf(name);	Obtain a numerical answer for a named expression.

* The %% and %%% operators are also available.

Table 4.1: Basic MAPLE Commands

4.1.1 Text Mode, Math Mode, and Command Mode

As illustrated in figure 4.1 below, MAPLE is a unified working environment that allows a user to enter plain-text, mathematical formulae, and MAPLE commands.

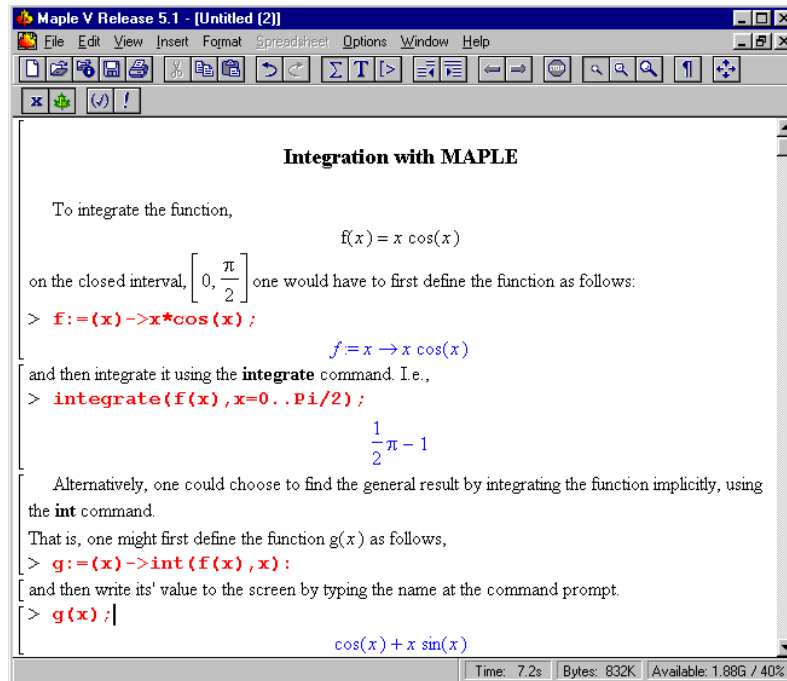
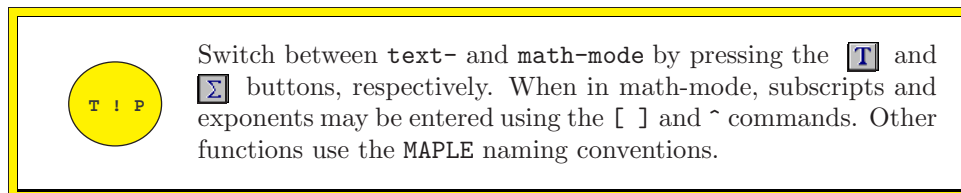


Figure 4.1: MAPLE can be used to enter plain-text, mathematical symbols and formulae and commands.

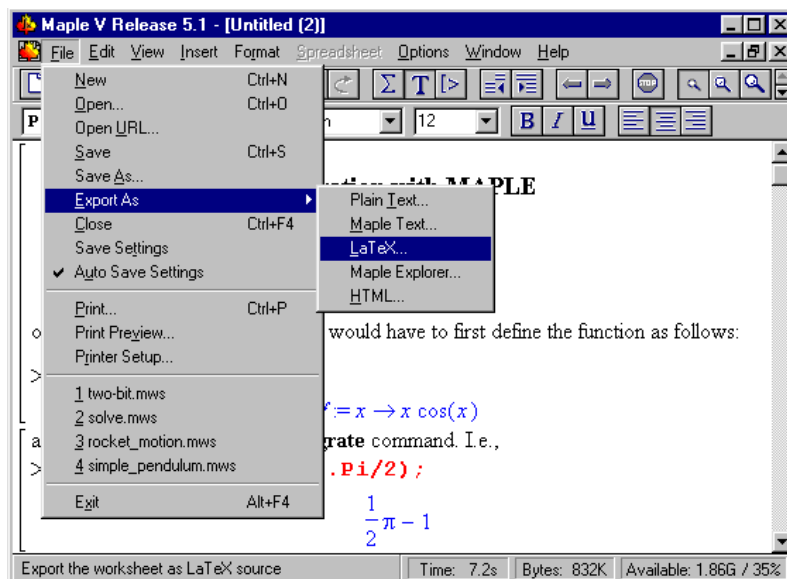
These three “modes” make MAPLE extremely useful. To access the *text*, *math*, and *command* modes a user can press the Σ , **T**, and \triangleright buttons respectively². Alternatively the **Insert** menu options or shortcut keys may be used. i. e., use **Ctrl+T**, **Ctrl+M**, to change to *text*- and *math*-modes, and use **Ctrl+K** or **Ctrl+J** to insert an execution group either before or after the cursor location.



4.1.2 Exporting MAPLE Worksheets

As mentioned above, MAPLE allows its users to export worksheets in many different formats. To do this, a user would simply choose the **File-Export As** option from the drop-down menus, as shown below:

²The command mode is also referred to as an execution group.



While some of these options are MAPLE-specific and of limited general use, the LaTeX... and HTML... options will produce high-quality documents that are highly portable and platform-independent.

Exporting $\text{\LaTeX} 2_{\epsilon}$ Documents

The power and flexibility of \LaTeX was well discussed in chapter 2 and is motivation enough to exploit MAPLE's built-in ability to produce documents in the ".tex" format. The real question is of course, how this is accomplished. While the details of the algorithm are not important, a user should understand that the bulk of the process involves the creation of newcommands and newenvironments as discussed in section 2.7; these are provided by the maple2e package, and the appropriate style files.³

Upon choosing to export a MAPLE worksheet in a $\text{\LaTeX} 2_{\epsilon}$ format, a user is presented with the dialogue box shown in figure 4.2, requesting the width (in inches) that equations should be formatted to. Note that by default, equation numbers are omitted. A user can however choose to edit the .tex file where appropriate.

Generally speaking, the structure of these files is the same as any other $\text{\LaTeX} 2_{\epsilon}$ file. The preamble of the document may contain lines similar to the following,

```

%% Created by Maple V Release 5.1 (IBM INTEL NT)
%% Source Worksheet: Untitled (2)
%% Generated: Thu Oct 04 10:56:33 2001
\documentclass{article}
\usepackage{maple2e}
\def\emptyline{\vspace{12pt}}
\DefineParaStyle{Maple Output}
\DefineCharStyle{2D Comment}
\DefineCharStyle{2D Math}
\DefineCharStyle{2D Output}

```

If additional graphics, or symbols, etc. are needed the standard packages may be added to the preamble. Compilation of the file is accomplished as always, and the output may be converted to either PS format or PDF. The postscript version generated from the MAPLE worksheet shown in figure 4.1 is shown below.

³These files are not part of the standard \LaTeX distribution, but have been included with the accompanying software.



Figure 4.2: Users can specify the width of equations when exporting $\text{\LaTeX} 2_{\epsilon}$ documents.

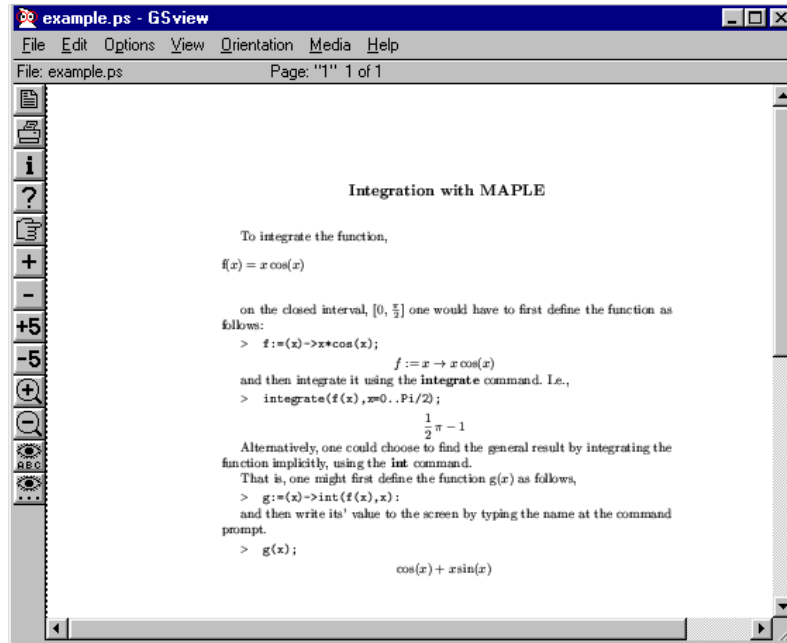


Figure 4.3: MAPLE can be used to export $\text{\LaTeX} 2_{\epsilon}$ documents using the `maple2e` package.

Notice that the the MAPLE commands are preceded by the `>` symbol and that the colors have been eliminated.

Due to the wealth of MAPLE-specific commands and environments that are used, the $\text{\LaTeX} 2_{\epsilon}$ code may appear unintelligible at first glance. For example, the following $\text{\LaTeX} 2_{\epsilon}$ code fragment was generated by MAPLE, when exporting the worksheet:

```

...
\begin{center}
\textbf{\large Integration with MAPLE}
\end{center}
\emptyline
To integrate the function,
\begin{center}
\mapleinline{inert}{2d}
  {f(x) = x*cos(x);}
  {\mathrm{f}(x)=x\,\mathrm{cos}(x)}
\end{center}
on the closed interval,
\mapleinline{inert}{2d}{{[0, Pi/2]};}
  {[0, \,\frac{\pi}{2}]$}
one would have to first define the function as follows:
\begin{mapleinput}
\mapleinline{active}{1d}{f:=(x)->x*cos(x);}
\end{mapleinput}
\mapleresult
\begin{maplelatex}
\mapleinline{inert}{2d}
  {f := proc (x) options operator, arrow; x*cos(x) end;}
  {[f := x\rightarrow x\,\mathrm{cos}(x)\]}
\end{maplelatex}
...

```

Here nearly all of the environments are new and the `\[... \]` has been used instead of

the $\$. . \$$ to indicate a math-mode element.⁴ With careful attention to the syntax, a user should feel more than comfortable adding or subtracting code as desired.

Exporting HTML Documents

When a user chooses to export a MAPLE worksheet in HTML format they should be aware of the fact that MAPLE will create an HTML page with two frames.⁵ If the document name chosen is, “example.html”, then the left-hand frame will contain a page called “example-toc.html”, which will provide links to the pages that make up the entire worksheet.⁶ The worksheet content will be broken into sections according to MAPLE’s own internal conversion algorithms, unless a user inserts sections manually.⁷ This may result in several pages being created. These pages would have names of the form, example1.html . . . example.html.

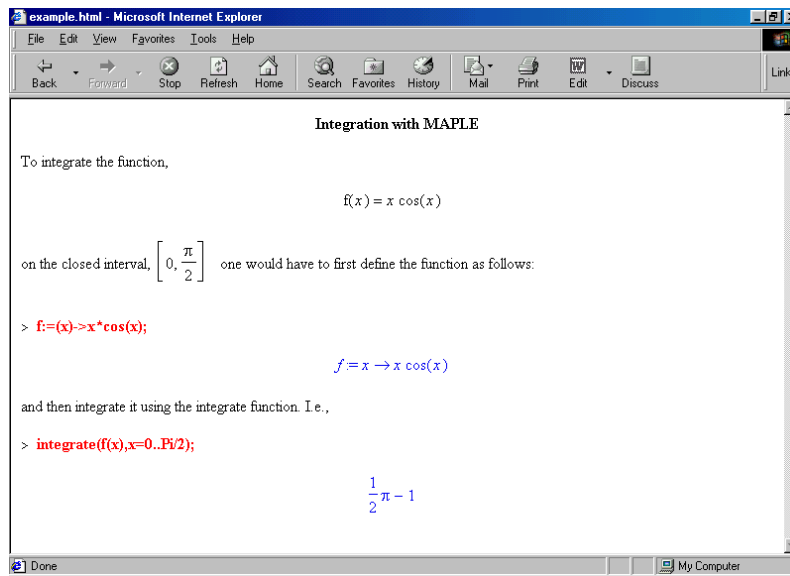


Figure 4.4: MAPLE can be used to export HTML documents.

In figure 4.4 the page generated from the worksheet shown in figure 4.1 is shown. Notice that the colors have been preserved and that all of the mathematical equations appear with appropriate formatting. These equations are not typeset, but are image files. MAPLE’s conversion algorithm takes all math-mode and output and converts them to the Graphic Image Format (GIF). All of the image files are placed in the subdirectory, `./images`, and appropriate links are created within the “.html” file.

In contrast to the $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ conversion, the HTML file produced by MAPLE, should be very readable. Indeed, the following code fragment was generated by MAPLE, and edited only to add indentation:

```

<html>
<head>
<title>example.html</title>
<!-- Created by Maple V Release 5.1, IBM INTEL NT -->
</head>
<body bgcolor="FFFFFF">
  <a name="MapleAutoBookmark1">

```

⁴This alternative notation was not mentioned previously to avoid confusion.

⁵See section 3.3.1 for more on frames.

⁶i. e., the content of the page is similar to that of `navigate.html` discussed on page 65.

⁷For more information on how to insert menus, type `?sections` at the command prompt.

```

<p align=center>
  <b><font color=#000000>Integration with MAPLE</font></b>
</p>
<p align=left>
  <font color=#000000>To integrate the function,</font>
</p>
<p align=center>
  
</p>
<p align=left>
  <font color=#000000>on the closed interval, </font>
  
  <font color=#000000>
    one would have to first define the function as follows:
  </font>
</p>
<p align=left>
  <tt>&gt; </tt>
  <b><font color=#FF0000>f:=(x)-&gt;x*cos(x);</font></b>
</p>
<p align=center>
  
</p>
<p align=left>
  <font color=#000000>
    and then integrate it using the integrate function. I.e.,
  </font>
</p>
<p align=left>
  <tt>&gt; </tt>
  <b><font color=#FF0000>integrate(f(x),x=0..Pi/2);</font></b>
</p>
<p align=center>
  
</p>
...
</body>
</html>

```

Notice that the main difference between the code generated by MAPLE, and code that a programmer may write by hand, is that no default attribute values have been assumed. (e.g., the left-alignment attribute of the <P> tags.) As with the L^AT_EX 2_ε code, users should feel comfortable editing the code as desired.

4.2 Functions and Procedures

4.2.1 Intrinsic Functions

MAPLE recognizes many mathematical functions. In addition to the standard Mathematical functions, such as the trigonometric and exponential functions, MAPLE has “built-in” routines

that evaluate special functions of interest in a wide variety of applications. Some of the most useful functions known to MAPLE are shown in the table below. Most of these are functions of a single variable and are called as such. I. e., `sin`= $\sin(x)$.

<code>sin, cos, tan, etc.</code>	Trigonometric Functions
<code>arcsin, arccos, arctan, etc.</code>	Inverse Trigonometric Functions
<code>sinh, cosh, tanh etc.</code>	Hyperbolic Trigonometric Functions
<code>exp</code>	Exponential Function
<code>ln</code>	Natural Logarithmic Function
<code>log[10]</code>	Logarithmic Function Base 10
<code>sqrt</code>	Algebraic Square Root
<code>round</code>	Round to the Nearest Integer
<code>trunc</code>	Truncate to the Integer Part
<code>erf, erfc</code>	Error and Complementary Error Function
<code>Dirac</code>	The Dirac Delta Function
<code>hypergeom</code>	Hypergeometric Function


Table 4.2: MAPLE Functions

The standard mathematical *operators* and logical *relations* are available with MAPLE.⁸ I. e.,

<code>+</code> addition	<code>-</code> subtraction	<code>*</code> multiplication
<code>/</code> division	<code>^</code> exponentiation	<code>=</code> equal to
<code><</code> less than	<code><></code> not equal to	<code><=</code> less than or equal to

MAPLE also allows user to use the standard logical or Boolean Operators, ‘*and*’, ‘*or*’, and ‘*not*’. Generally speaking the operators are used to “build” functions (as discussed in the next section) and the logical and Boolean Operators are used to form conditional statements, as in the following *if-then* construct:⁹

```
if ( a <> 0 and f(a)/a < eps ) then
    writeline("value reached");
fi:
```

Ex: 4.1 Open MAPLE by pressing the  button on the WinEdt toolbar. Using commands similar to those shown in figure 4.1 integrate the function $f(x) = x^2 \sin(x) + 2 \cos(x)$ over the closed interval, $[0, \pi]$, obtaining both numerical and symbolic(general) results. Save your worksheet and add some text and mathematics to describe what is being done. Export both HTML and $\text{\LaTeX} 2_{\epsilon}$ versions of your worksheet. Hand in a hard copy of the MAPLE worksheet and publish PDF and HTML versions on your local web server.

⁸Note that any expressions involving the `>` or `>=` symbols are automatically converted to equivalent forms using the `<` and `<=` operators.[?]

⁹The *if-then* construct as well as other control constructs will be discussed in greater detail in section 4.5 below.

4.2.2 User-Defined Functions and Procedures

Using the intrinsic functions to “build” a function is very straight-forward. For example, to define the function $f(t) = At \cos(at)$ a user would enter the following commands:(Note the comments.)

```
A:=1.0: T:=1.0: omega:=2*Pi/T:      # define parameters
f:=A*t*cos(omega*t);                # define the function
val:=eval(f,t=Pi);                  # evaluate the function
ans:=evalf(val);                     # obtain a numerical result
```

$$f := 1.0t \cos(2.000000000\pi t)$$

$$val := 1.0\pi \cos(2.000000000\pi^2)$$

$$ans := 1.978203463$$

While the above example illustrates how simple it can be to use MAPLE , it is rather limited. If the values of the parameters change, then each of the parameter definitions would have to be changed, or a new session began. In this particular case, the changes would be easy to make, but a user may choose to define a *procedure* instead.

There are two ways to define a procedure in MAPLE . The first of these uses the general syntax,

```
name:=(list of variables)-> definition:
```

and is referred to as a functional operator.

Using the function $f(x)$ introduced above a procedure which would allow the amplitude(A) and the period(T) to be changed interactively could be written as follows:

```
f:=(A,T,t)->A*t*cos(2*Pi*t/T):      # define the procedure
f(A,T,t);                            # echo its' definition to the screen
ans1:=evalf(f(1,1,Pi));               # obtain a numerical result
ans2:=evalf(f(1,2,Pi));               # obtain a different result
```

$$At \cos\left(\frac{2\pi t}{T}\right)$$

$$ans1 := 1.978203463$$

$$ans2 := -2.835869698$$

The most important things to take note of here are that the **eval** function is no longer needed and that the parameters are automatically assigned real values. Note that the order of the parameters is implicit but in the current example any values may be passed to the procedure. I. e., with the above definition a user could call the function in the following manner:

```
f(b,c,z);
```

$$bz \cos\left(\frac{2\pi z}{c}\right)$$

In many cases it may be important to declare the *type*¹⁰ of the parameter. That is, if an *integer* or *character* value is required a user can declare the type by using the general syntax, **name::type**. To utilize the type declaration facility it is easiest to write a formal procedure, which uses the following syntax:

```
name:=proc( List of Parameters )
    body;
```

¹⁰There are many type declarations available in MAPLE . Common examples include: *positive*, *negative*, and *float*. For a complete list use **?type** at the command line.

end:

As a simple example consider the following procedure, which calculates the energy (in electron volts) of an electron in a Hydrogen atom.

```
energy:=proc(n::integer)
  Digits:=3:
  E:=-13.6/n^2:
end:
```

Here the **Digits** command has been used to limit the number of digits that are in the answer.¹¹ Once this procedure is defined it can be called just as any other function would be. E. g.,

Warning, 'E' is implicitly declared local

```
energy(1);
energy(3);
energy(2.1);
```

-13.6

-1.51

Error, energy expects its 1st argument, n, to be of type integer, but received 2.1

Note that when a non-integer value is entered, MAPLE issues an error message. Beginning users of MAPLE are bound to encounter many errors. Most are due to misspellings, improper syntax, or type mismatches. **The warning may be ignored.**

The usefulness of procedures can not be overstated. Whether instantiated as functional operators or as formal procedures, they can greatly simplify repetitive definitions and provide a way of easily updating parameters. Moreover, a user may write procedures and add them to the MAPLE library, so that they may use them in other worksheets.¹²

Piece-Wise Functions

Quite often in applications a function with multiple definitions over a given interval is required. These functions are usually referred to as *piece-wise* functions and MAPLE contains built-in procedures that make defining such functions very straightforward. As an example consider the following maple session:

```
decay:=(A,B,t)->piecewise(
  t<0 , A*cos(B*t),
  t>=0, A*exp(-B/2*t)):
decay(A,B,t);
```

$$\begin{cases} A \cos(Bt) & t < 0 \\ Ae^{-\frac{B}{2}t} & 0 \leq t \end{cases}$$

A plot¹³ of this function for $A = 1.0$ and $B = 0.1$ is shown in figure 4.5.

More generally the syntax for entering piece-wise functions requires a conditional statement followed by a comma and then the functional definition. As shown, multiple conditions and functional definitions are separated by a comma. It is also possible to give compound conditional statements. I. e.,

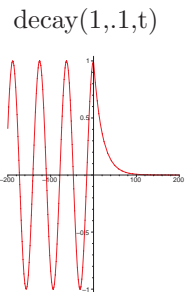


Figure 4.5: With MAPLE users can plot piece-wise functions.

¹¹If the **Digits** command is used outside of a procedure, its effect will continue until a subsequent call to **Digits** is made or a new session is began.

¹²The method for accomplishing this will be discussed in section ??.

¹³See section 4.4 for more on plotting functions.

```
g:=(a,b,c,x)->piecewise(
  x>0 and x<=5, a*x^2+b*x+c,
  x>5           , (125*a+15*b+5*c)/x):
g(a,b,c,x);
```

$$\begin{cases} ax^2 + bx + c & -x < 0 \text{ and } x - 5 \leq 0 \\ \frac{125a + 25b + 5c}{x} & 5 < x \end{cases}$$

While the two examples shown use functions that are continuous, this is not a requirement. It is also possible to leave the value of a function unspecified over a certain domain, as in `g:=piecewise(x>0,10)`. In this case, the value of `g` is 10 for $0 < x < 10$ and 0 *otherwise*. For more information, type `?piecewise` at the command prompt.

Ex: 4.2 The position of a Damped Harmonic Oscillator (DHO) may be modeled with the equation $x(t) = Ae^{-\gamma t} \cos(\omega_0 t)$, where $\omega_0 = \sqrt{k/m}$ is the *natural* frequency of the oscillator, and $\gamma = b/2m$; m represents the mass of the system, k is the ‘spring’ constant, and b characterizes the velocity-dependent, frictional force. Write a MAPLE procedure that will calculate the position of a DHO. Note that while A may be positive or negative, the values of b , k , and m should be positive. Evaluate your procedure for at least five(5) different values and include values that cause errors to be generated.

4.3 Symbolic and Numeric Solution Techniques

One of MAPLE’s real strengths is its’ ability to produce algebraic/analytic and/or numerical solutions to many problems. For instance, MAPLE allows its’ users to find roots(both real and imaginary) of polynomial equations, solve systems of equations, perform differentiation and integration, and even find solutions to differential equations!

4.3.1 Solving for a Single Variable

At the simplest level, a MAPLE user can enter a single equation and solve for a single variable using the `solve` command. To see how straightforward this is, consider using conservation of energy to find the height, h that a ball will rise into the air when given an upward velocity, v . Ignoring air resistance, etc. one finds that the initial kinetic energy of the ball must equal the potential energy of the ball when it reaches its’ final height. I. e., $\frac{1}{2}mv^2 = mgh$, and MAPLE finds the solution with a single line of code.

```
solve(1/2*m*v^2=m*g*h,h);
```

$$\frac{1}{2} \frac{v^2}{g}$$

The `solve` command can also be used to find numerical results as in the following MAPLE session:

```
f:=(x)->2*x+8:
g:=(x)->x^2-x+10:
ans:=solve(f(x)=g(x),x);
```

```
ans := 1, 2
```

MAPLE stores the solutions either as a list or as an array,¹⁴ depending on the nature of the solutions. To extract the two solutions for later use, reference the individual elements as follows:

```
ans[1];
ans[2];
```

```
1
2
```

Performing a check on the results achieved is always a good idea. This can be accomplished using the `eval` command as illustrated below:

```
eval(f(x)=g(x),x=ans[1]);
eval(f(x)=g(x),x=ans[2]);
```

```
10 = 10
12 = 12
```

A Worked Example - Uniform Circular Motion

MAPLE's `solve` command can also be used in conjunction with the `subs` command to solve equations with constraints. To see how this may be done, consider a mass undergoing uniform circular motion on a table-top without friction. The velocity, \vec{v} of the mass is everywhere perpendicular to the circle, as shown in figure 4.6. The normal force, \vec{F}_N and weight, \vec{F}_g cancel one another; accordingly an (inward) centripetal force is developed. The magnitude of this force is given by

$$F = m \frac{mv^2}{r}$$

Assuming that the magnitude of the centripetal force, the mass of the object, and the period of the motion, T , are known, the radius of the circular orbit can be found. To do this one must use the fact that the magnitude of the velocity is constrained by the relation, $v = \frac{2\pi r}{T}$. The following MAPLE session illustrates how this constraint may be used to solve for the radius.

```
eq:= F=m*v^2/r;
c:= subs(v=2*Pi*r/T, eq);
ans:=solve(c,r);
```

```
eq := F =  $\frac{mv^2}{r}$ 
ans :=  $\frac{1}{4} \frac{FT^2}{m\pi^2}$ 
```

Here the `subs` command has been used to substitute the constraining equation into the original equation. More generally, the `subs` command takes the form `subs(c1, ..., cn, expr)`, where `expr` and the `cn` are constraints.

In order to use the result found by MAPLE to obtain numerical results it is necessary to create a functional relationship similar to those discussed in section 4.2.2. To do this a user could call the `unapply` command, listing the parameters that are expected to vary. For example,

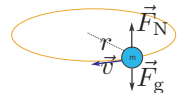


Figure 4.6: A mass m undergoing uniform circular motion in a horizontal plane.

¹⁴See section 4.5.1 for more on lists.

```
radius:=unapply(ans,F,m,T):
radius(F,m,T);
```

$$\frac{1}{4} \frac{FT^2}{m\pi^2}$$

As with all proper functional relationships, evaluation of the `radius(F,m,T)` at various values may be accomplished using only `evalf`. I. e.,

```
Digits:=3:
evalf(radius(4500,454,1.1));
evalf(radius(4500,454,2.0));
evalf(radius(4500,454,3.05));
```

.303

1.00

2.33

The `solve` Command

Occasionally MAPLE will encounter a variable within an equation that it can not find an analytic solution for. In these cases MAPLE will do nothing. That is, no output will be generated- not even an error message! If this occurs a user may wish to see if a numerical solution can be found.



If an analytic solution can not be found using the `solve` command, try using the `fsolve` command to find a numerical result.

As an example of this sort of situation, consider the two functions,

$$f(t) = t^2, \quad g(t) = 1 - \sin(10t)$$

As shown in figure 4.7 these two functions have five(5) points of intersection. If the `solve` command is used to try and find these solutions however none are found. That is, as the following MAPLE session illustrates,

```
eq:=t^2=1-sin(10*t):
solve(eq,t);
```

MAPLE simply makes no response.

On the other hand, if the `fsolve` command is used a root is found near 0.138. I. e.,

```
eq:=(t)->t^2=1-sin(10*t):
fsolve(eq(t),t);
```

.1375905603

Here it is known that multiple roots exist and even though the equation is transcendental¹⁵ a user may wish find them. One method that may be used is to specify the range over which MAPLE should look for a solution. This may be accomplished using the following general syntax:

¹⁵Recall that a transcendental equation is one wherein the variables are related in a non-separable way.

Finding Solutions

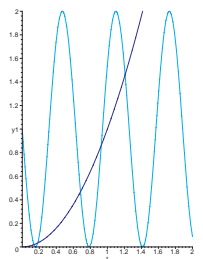


Figure 4.7: With the `fsolve` command users can find the intersection points of two functions.

```
r:=fsolve(equation, variable, range);
```

Note that while a user may wish to use the name “root” to identify these solutions, MAPLE will issue an error message, because root is a protected name.

```
root:=fsolve(eq(t), t, 0..0.15);
```

Error, attempting to assign to ‘root’ which is protected

Choosing a different name and increasing the number of digits of accuracy allows a user to find all of the roots.

```
Digits:=15;
r1:=fsolve(eq(t), t, 0..0.15);
r2:=fsolve(eq(t), t, 0.15..0.2);
r3:=fsolve(eq(t), t, 0.6..0.8);
r4:=fsolve(eq(t), t, 0.8..1.0);
r5:=fsolve(eq(t), t, 1.2..1.4);
```

```
r1 := .137590560278600
r2 := .183037924272541
r3 := .684371575383325
r4 := .928678399216379
r5 := 1.20872348211007
```

A check then shows the amount of error in the individual solutions.

```
eval(eq(t), t=r1);
eval(eq(t), t=r2);
eval(eq(t), t=r3);
eval(eq(t), t=r4);
eval(eq(t), t=r5);

.0189311622777791 = .018931162277778
.0335028817220005 = .033502881722001
.468364453192654 = .468364453192655
.862443569171096 = .862443569171097
1.46101245620429 = 1.46101245620430
```

This error may of course be further diminished by increasing the number of digits in the solutions.

MAPLE may also have trouble finding roots for equations that are not transcendental. For example, in looking for the roots of a polynomial equation, some of the roots may be *real*-valued and some may be *complex*. As shown above, MAPLE does not have a problem “understanding” imaginary numbers, it merely needs to be told to find them. In the following example (adapted from [?]) all of the roots of the fourth-order polynomial, $p(x) = 3x^4 - 16x^3 - 3x^2 + 13x + 16$ are sought. First the `fsolve` command is used directly as follows:

```
Digits:=3;
p:=(x)->3*x^4-16*x^3-3*x^2+13*x+16;
fsolve(p(x)=0, x);
```

Because only two roots are found, a user should expect that additional complex roots may exist.¹⁶ To check for any complex roots, simply add the `complex` command as shown.

```
p:=(x)->3*x^4-16*x^3-3*x^2+13*x+16:
r:=fsolve(p(x)=0, x, complex);
```

```
r := -.6623589786 - .5622795121I, -.6623589786 + .5622795121I,
      1.324717957, 5.333333333
```

Hence, all four all roots are found.

Notice also that the complex roots occur in pairs and are the complex-conjugates of one another. As discussed above, these roots may be checked with the `eval` command. I. e.,

```
eval(p(x),x=r[1]);
eval(p(x),x=r[2]);
eval(p(x),x=r[3]);
eval(p(x),x=r[4]);
```

```
-.2 10-8I
 .2 10-8I
  0
 .1 10-7
```

And the accuracy can be increased if necessary by increasing the number of digits.

Ex: 4.3 If an object is fired straight up with a velocity of v_0 and is acted upon by gravity and a frictional force proportional to its' velocity, then the velocity as a function of time is given by

$$v(t) = -\frac{g}{k} + \left(\frac{g}{k} + v_0\right) e^{-kt}.$$

Create a functional relation of the form, $v(g,k,v_0,t)$ and find an algebraic solution for the time required for the velocity to decrease to an arbitrary percentage of its' initial value. I. e., you may use

```
t:=solve(v(g,k,v0,t)=v0*p,t);
```

where $0 < p < 1$ is the percentage. Using the `unapply` command, create a functional relationship for the time in the form `tp(g,k,v0,p)`. Evaluate this function using $g = 32$ ft/s², $k = 0.1$ 1/s, an initial velocity of $v_0 = 100$ ft/s and for $p = 0.5$ and $p = 0$. While the former value will be the time required for the object to slow to half its' original value, the latter value will be the time required for the object to reach its' maximum height! **Verify your results by performing a check.** You may plot the velocity as a function of time using `plot(v(32,0.1,100,t),t=0..5);`.

¹⁶Alternatively, the roots may be degenerate. I. e., they may occur more than once.

4.3.2 Solving Systems of Equations

When more than one equation is being solved, the following general form should be used for the `solve` command.

```
name:=solve({list of equations},{list of unknowns});
```

As a first example of how this is done, consider a one-dimensional elastic collision between two masses, as illustrated below.

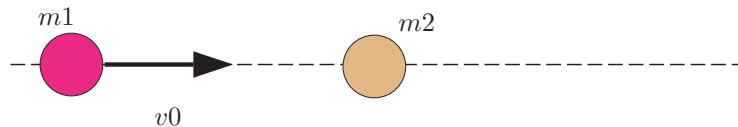


Figure 4.8: A One-Dimensional Collision

While the initial velocity of $m1$ is $v0$, the velocity of $m2$ is zero. Using conservation of momentum and conservation of energy one can write down the two equations needed to solve for the final velocities of the masses. Letting $v1$ and $v2$ represent the final velocity of $m1$ and $m2$ respectively, one could enter these equations into MAPLE to find general solutions.

```
pcon:=m1*v0=m1*v1+m2*v2:
econ:=m1*v0^2=m1*v1^2+m2*v2^2:
velocity:=solve({pcon,econ},{v1,v2});
```

$$velocity := \{v2 = 0, v1 = v0\}, \{v2 = \frac{2m1v0}{m1+m2}, v1 = \frac{(-m2+m1)v0}{(m1+m2)}\}$$

Here the fact that there are two sets of solutions is not a surprise, as the original equations were quadratic. In the present case however, the first set of solutions may be eliminated on physical grounds.¹⁷

To extract the second set of solutions a user will need to use the `rhs` command. This command along with its “partner” command, `lhs` are very useful. Abbreviations for “right-hand-side” and “left-hand-side” these functions may be used to assign the right- or left-hand-sides of an equation a new name. This name in turn may be used to form functional relationships. The final velocity of the $m1$ may be found and evaluated with specific values for $m1$, $m2$, and $v0$, using the following code:

```
v1:=unapply(rhs(velocity[2,2]),m1,m2,v0):
v1(m1,m2,v0);
evalf(v1(10,25,100));
```

$$\frac{(-m2+m1)v0}{m1+m2}$$

-42.9

In a similar manner a user could extract the velocity of $m2$. **It is very important to notice that the order of the solutions may be different from the order used in the solve command.** Furthermore, notice that to “single out” the *second* element of the *second* solution, `velocity[2,2]` was used. This is an example of how MAPLE uses **Arrays** to store information. Arrays will be dealt with in greater detail in section 4.5.1. For now consider that a new technique for forming functional relationships involving multiple algebraic solutions may be expressed in the following manner:

```
newname:=unapply(rhs(oldname[i,j]), parameter list):
```

¹⁷That is, the first set of solutions correspond to the situation *before* the collision and necessarily satisfy the equations.