**Arithmetic operations**

The symbols +, -, \*, /, and ^ (or \*\*) denote addition, subtraction, multiplication, division, and exponentiation (4^2 = 4\*\*2 = 42 = 16). When a string of operations are specified in a command, MAPLE first does exponentiations, then multiplications and divisions, then additions and subtractions. To change the order, use parentheses.

**Editing commands**.

If you make a mistake in a command (like forgetting a semicolon) or want to change a command, you can go back and edit the command with the cursor and mouse as you would a word-processor text.

**Operating on the last result**.

Use a quotation mark (") to refer to the result of the last calculation, and a double quotation mark ("") to refer to the next-to-last result. For example,

* [> **5\*2;**

10

* [> **7 + ";**

17

* [> **" \* "";**

170

**Exact arithmetic and floating-point arithmetic**

MAPLE calculates fractions (exact arithmetic) unless you specify that you want decimals (floating-point arithmetic) with the **evalf** function ("evalf" stands for "evaluate using floating-point arithmetic"):

* [> **25/27 + 3/51;**

452   
459

* [> **evalf(25/27 + 3/51,3);** ( or [> evalf(",3); )

0.985

**Defining variables**

Use the colon-equal symbol (:=) to define variables--that is, to assign values to them. Once you have defined a variable, simply typing its name will show its value, and using the name in a formula will cause the value to be substituted.

* [> **A := 5; B := 2;**

*A* := 5   
*B* := 2

(Notice that you can string commands together on the same line.)

* [> **A;**

5

* [> **4\*A/B;**

10

*Variable names are case-sensitive*: A and a are two completely different variables.

**Specifying the number of digits (significant figures) in your results.**

When you do floating point arithmetic, MAPLE defaults to 10 significant figures:

* [> **evalf(5/2);**

2.500000000

You can change the default by setting a variable "Digits" equal to the desired number of significant figures.

* [> **Digits := 4;** (Make sure you capitalize the word Digits.)

*Digits* := 4

* [> **evalf(5/2);**

2.500

**Built-in functions**

MAPLE has many built-in mathematical functions including sin, cos, tan, exp (exp(x)=ex), ln, and log10. However, if you want to calculate the value of one of these functions, you must use the evalf command.

* [> **ln(10);**

ln(10)

* [> **evalf(ln(10), 4);**

2.303

**Expressions and functions**

An **expression** is a string of constants, variables, and mathematical operators (+   -   \*   /   ^   =   ln   sin   ...). The following are all expressions:

1/2  
3\*x^3 - 4\*ln(x)     [3x3 - 4ln(x)]  
5\*x^2 - 2\*y^2 = 3\*cos(x\*y)     [5x2 - 2y2 = 3cos(xy)]

A **function** is a relationship for a variable (the *dependent variable* in terms of one or more other variables (the *independent variables*). The following are each functions:

y(x) = 3\*x + 2   [--> *y*(1) = 5]  
z(x,y) = 2\*x/y   [--> *z*(5,3) = 10/3]

MAPLE handles expressions and functions in completely different ways, which can lead to a great deal of confusion when relations like this are encountered:

y = 6\*x^2 - 3\*x + 4

This could be an *expression* relating *x* and *y* or the definition of *y* as a *function* of *x*. To do anything to or with this relation (like substituting a value of *x* into it, or solving it for one variable in terms of the other one, or differentiating or integrating it), you must know whether the relation is an expression or a function.

In the next two sections we show how expressions and functions are defined and evaluated. In the remainder of the tutorial we will continue to make the distinction between the two types of relations. If you forget the difference, refer back to this section.

**Defining expressions and substituting values into them**

Use the colon-equal (:=) to define an *expression*, and the **subs** function to substitute a value into it.

* [> **f := x^2;**

f := x2

* [> **f;**

*x*2

* [> **subs(x=5, f);**

25

Typing [> f(5); will not work for the latter calculation. If you want to use functional notation like that, you need to use the next method to define a *function*.

**Defining functions and substituting argument values into them**

Use the colon-equal (:=) and the minus-greater than (->) symbols to define a function, and functional notation to substitute a value into it.

* [> **g := x -> 1/(x+1);**

*g* := *x* -> 1/(*x*+1)

* [> **g(x);**

1/(*x*+1)

* [> **g(4);**

1/5

**Converting expressions into functions and vice versa**

We noted that the previously defined expression f = x2 cannot be treated as a function.

* [> **f(5);**

x(5)^2

You can convert the expression into a function of *x*, however, by using the *unapply* command.

* [> **f := unapply(f,x);**

*f* := (*x -> x)2*

* [> **f(5)**;

25

You can also take the function *f = x*2 and convert it back into an expression.

* [> **f := f(x);**

*f* := *x*2

* [gt; **subs(x=5,f);**

25

**SOLVING ALGEBRAIC EQUATIONS**

**Defining equations**

Equations are expressions, so we can use the colon-equal (:=) to define equations involving one or more variables. (This will be the first step in getting MAPLE to solve the equations.)

Suppose equations relating two variables are

2*x*1 - 5*x*2 = 12   
12*x*1 + 4*x*2 = 17

To enter and store these equations for subsequent processing, enter the commands

* [> **eq1 := 2\*x1 - 5\*x2 = 12:**
* [> **eq2 := 12\*x1 + 4\*x2 = 17:**

(Don't forget the multiplication symbols.) We used colons instead of semicolons in these commands, so MAPLE does not reproduce the equation definitions.

**Solving sets of linear algebraic equations.**

Suppose you wish to solve the pair of equations defined above for the variables *x1* and *x2.* Use the **solve** (exact arithmetic) or **fsolve** (floating point arithmetic) command as follows: (*Type these commands carefully*; if you make the slightest mistake, like omitting a comma or typing the wrong kind of brace, MAPLE will give you nothing but heartache.)

* [> **sol := solve({eq1, eq2}, {x1,x2});**

http://www4.ncsu.edu/unity/lockers/users/f/felder/public/tutorials/maple1img1.gif

* [> **sol := fsolve({eq1, eq2}, {x1,x2});**

*sol* := {*x2* = -1.618, *x1* = 1.956}

(There does not seem to be any way to predict which variable MAPLE will print first.)

The argument of the **solve** or **fsolve** command consists of two sets of curly braces enclosed in parentheses. (If you only have one equation in one unknown, you don't need the braces.) The first set of braces contains the equations to be solved or the symbols for them, separated by commas; the second set contains the set of unknown variables to be solved for.

**Extracting components of a solution.**

The solution set *sol* generated with the fsolve command in the previous step contains two components: the equation *x1* = 1.9559, and the equation *x2* = -1.6176. You can extract either of these equations or only their right-hand sides as follows (the example shows the procedure for the first element).

* [> **sol[2];**

*x*1 = 1.956

* [> **rhs(sol[2]);**

1.956

You could also use the **lhs** function to extract the left-hand side of the equation.

**Specifying search ranges for unknown variables in nonlinear equations.**

The procedure for solving nonlinear equations with **solve** or **fsolve** is the same as that outlined above for linear equations. A difference in the possible outcome is that the procedure used by MAPLE (or any other nonlinear procedure) may not converge to a solution, even if one exists, or it may converge to a solution that is mathematically correct but physically meaningless (like a negative mass fraction).   
To improve the odds of success in solving nonlinear equations, specify in the **solve** or **fsolve** command a range of values for each unknown variable within which the desired solution must fall. The more you can narrow the search range, the better the chances for convergence.

For example, if a variable *x* is the solution of the equation *x = e-x*, and you know the solution must be a positive fraction (0<*x*<1), you can proceed as follows:

* [> **eq := x = exp(-x);**

*eq* := *x* = e(-*x*)

* [> **sol := fsolve(eq, x, x=0..1);**

*sol := .5671*

(Since there is only one variable, you could have omitted the "x=" in front of the range 0..1.)

The procedure is the same if you have simultaneous nonlinear equations, except that now you need to enclose the equations, the variables, and ranges for each variable within curly braces. For example, suppose you want to find the solution of the equations

*sin(x+y) -yex* = 0   
*x2 - y* = 2

and you know that because of the physical meanings of *x* and *y* the solutions must fall in the ranges -1<*x*<1 and -2<*y*<0. The solution program would be as follows (*Type carefully!*):

* [> **f := sin(x+y) - y\*exp(x) = 0:**
* [> **g := x^2 - y = 2:**
* [> **solfg := fsolve({f,g}, {x,y}, {x = -1..1, y = -2..0});**

*solfg* := {*x* = -.6687, *y* = -1.553}

If you omitted the search ranges for *x* and *y*, the procedure would have converged to a different solution that satisfies the equations but has no physical significance. (Try it and see.)

If you want to specify a semi-infinite range for a search, you can use **infinity** or **-infinity** as one of the limits of the range.

**Finding roots of polynomials**.

Suppose you want to solve the equation

6*x*4 *-* 7*x*3 *+* 6*x*2 *-* 1 *=* 0

You know that the equation has four roots (solutions), which may be real numbers or complex conjugate pairs. You can find the roots either exactly or with floating-point arithmetic using the solve or fsolve command. In what follows, *I* is the MAPLE symbol for http://www4.ncsu.edu/unity/lockers/users/f/felder/public/tutorials/maple1img3.gif.

* [> **p := 6\*x^4 - 7\*x^3 + 6\*x^2 - 1 = 0;**

*p* := 6*x*4 - 7*x*3 + 6*x*2 - 1 = 0

* [> **sol := solve(p, x);**

http://www4.ncsu.edu/unity/lockers/users/f/felder/public/tutorials/maple1img4.gif

* [> **solf := fsolve(p, x, complex);**

*solf :=* -.3333, .5000 - .8660*I*, .5000, .5000 + .8660*I*

If you had left off "complex" in the last command the program would have generated only the real roots.

**PLOTTING**

**Accessing the MAPLE plotting package**

* [> **with(plots);**

This command makes a wide range of plotting functions available for use. Their names should appear on the screen in response to this command. We will use the "plot", "textplot", and "display" routines in the package.

**Plotting tabular data**

The specific volumes of saturated steam at several temperatures are listed below:

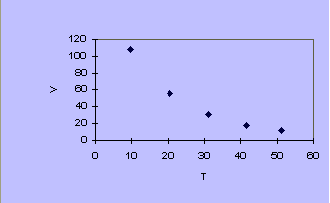
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **T(oC)** | 9.7 | 20.4 | 31.0 | 41.5 | 51.1 |
| **V(m3/kg)** | 108.7 | 56.4 | 31.1 | 18.1 | 11.5 |

To use MAPLE to generate a plot of *V* vs. *T*, we use the following commands:

* [> **VTdata := [[9.7, 108.7], [20.4, 56.4], [31.0, 31.1], [41.5, 18.1], [51.1, 11.5]];**

*VTdata :=* [[*9.7, 108.7*]*,* [*20.4, 56.4*]*,* [*31.0, 31.1*]*,* [*41.5, 18.1*]*,* [*51.1, 11.5*]]*;*

* [> **plot(VTdata, T=0..60, V=0..120, style=point);**



The first command generated a *list* (an array enclosed in square brackets) consisting of the [T,V] data points, each point enclosed in its own square brackets. The second command told MAPLE to plot the points, specifying the labels and ranges for the abscissa and ordinate axes, and to show but not connect the data points. (If the "style=point" option had been omitted, the points would have been connected with line segments.)

There are ways to change the axis labels, the symbols and colors of the data points, and other characteristics of the plot. We will hold off on giving you some of these options until we discuss plotting functions.

The command to generate plots of two functions on the same graph has the form   
  
[> plot( [f1(x), f2(x)], x=a..b, options); (*Don't type this now*)   
  
where   
  
*f1* and *f2* are either MAPLE functions or functions you defined using (:= ->)  
*a* and *b* are the desired boundaries of the abscissa range  
*options* are statements of how you want the plots to appear (details follow).

If you only want to plot one function, you can omit the square brackets enclosing the function labels.

For example, to generate plots of *f(x) = sin(x)* and *g(x) = 0.25x-1* vs. *x* from *x*=0 to *x*=4, the commands would be

* [> **f := x -> sin(x);**

*f* := *sin*

* [> **g := x -> 0.25\*x - 1;**

*g* := *x* -> .25*x* - 1

* [> **plot( [f(x), g(x)], x=0..4, color=[red,blue], style=[point, line]);**

(We are omitting the MAPLE output here but you should see it on your terminal screen.) You could also have added a range for the ordinate axis (e.g. y=-1..1); by omitting it, you allowed MAPLE to do its own scaling.

**Plot options**

Following are some but not all of the options you can include in a plot command. For a more detailed list, enter **[> ? plot** and click on the "plot(options)" link.

*What follows is for your information only. Don't do anything with it now.*

**color = *c***, where *c* may be any of most common colors and some not so common (coral, sienna, etc.) To see a list of colors, enter [> ? plot,color (don't do it now).

**style = *s***, where *s* may be "point" or "line" (line is default). "Point" shows closely spaced points, "line" shows line segments. When a function is plotted in the line style, the line segments are usually small enough to make the curve look continuous.

**labels** = [*xlabel, ylabel*], where the two arguments in square brackets are the labels you want for the abscissa and ordinate. The defaults are the variable names.

**linestyle = *n***, where n is usually 0, 2, or 3 but may be any other integer. The value of *n* specifies the type of line you want plotted when style=line. n=0 (default) is a solid line, n=2 is dots and dashes, and n=3 is dashes.

**symbol = *s*,** where *s* may be circle, POINT, box, cross, or diamond (default).

**title = *t*,** where *t* is a character string. The default is no title.

**Inserting text on a plot**

You can use the "textplot" command to insert text on a plot. In this case, we will label the two curves on the graph we just generated by inserting "f(x)" centered at the point (x=0.8, f=0.5) and "g(x)" just above and to the left of the point (x=2, g=-0.5). We first define a variable to represent the graph.

* Click just to the left of the word "plot" in the plot command.
* Type

**fggraph :=** [so that the command reads "fggraph := plot([f(x),g(x)],..."]

and replace the final semicolon with a colon. Hit "Return."

In the textplot commands that follow, enclose the texts to be inserted [f(x) and g(x)] in forward apostrophes (`) (located under the ~ character on most keyboards), as opposed to ordinary apostrophes (').

* **[> labelf := textplot( [0.8, 0.5, `f(x)`]);**     (Don't forget to use forward apostrophes.)  
    
  *labelf* := textplot([.8,   .5,   *f(x)*])
* **[> labelg := textplot( [2, -0.5,   `g(x)`],   align={ABOVE,LEFT});**     
    
  *labelg*   := textplot(2,   -.5,   *g(x)*],  *align* = {*ABOVE, LEFT*})
* **[> display ([fggraph, labelf, labelg]);**

You should see the same graph with the plot labels inserted. Notice how f(x) is centered on the point specified in the first textplot command (this is the default alignment), and the lower right corner of g(x) is located at the point specified in the second textplot command. It usually takes several trials to get the text exactly where you want it.