

Maple is an extremely powerful and versatile computer program. It is sometimes described as a "computer algebra system," but it is far more than that. Its capabilities include very broad symbolic, numeric, and graphical abilities. Yet its syntax is easy to learn and offers a very friendly learning curve.

Maple can do a wide variety of *symbolic* operations, including algebraic manipulations, symbolic solution of equations, symbolic evaluation of derivatives and integrals, solution of differential equations, matrix algebra, and many other things. It can do *numerical* solutions of equations, numerical integration, and numerical solution of differential equations. It has extensive *graphic* capabilities, including ordinary graphs, three-dimensional graphs, and animation in two and three dimensions.

In this course we will use only a tiny fraction of Maple's capability, but you will get a good taste of the great power that Maple offers. This introduction provides brief samples of these capabilities. More details about specific applications will be given in supplementary notes as they are needed.

In the problem assignments and solutions, and in all the supplementary notes (including these), Maple input statements will always be written in sans serif type, such as "Arial" or "Univers," like this. I'll use a similar type face for the projected class demonstrations of Maple. (In most of the cluster computers, the default type face for Maple input is Courier.)

Getting Started

Carnegie Mellon has a campus-wide site license for Maple, and it is installed on all the cluster computers. The current version of Maple is Maple 8; all the clusters have this version. You can buy your own copy at the Computer Store for about \$130. It's very handy to have Maple on your own computer. If you own or have access to an older version, Maple 4.0 or any later version is adequate for this course.

To start Maple on a Windows machine in a cluster: If there is a Maple icon on the desktop, just double-click on it. If not, click on "Start," then "Programs," then "Math and Stats," then "Maple 8," then again "Maple 8." The Maple window will open.

To start Maple on a UNIX machine: At the command prompt %, type "xmaple&" and Maple will open and run in a new window. If you have problems, ask a CCon.

When you are learning Maple, I recommend setting the "Input Display" under the "Options" menu to "Maple Notation," and setting the "Output Display" to "Typeset Notation." On most machines these will be the default settings, and you won't need to change anything.

Basic Maple Syntax

Maple is *interactive*; you type in a Maple statement, then type `<enter>`; Maple executes the statement and displays the result immediately. We'll often refer to the Maple statement as "input" and the result as "output." When Maple is waiting for input, the prompt `>` appears at the left margin. The default settings are to display the input in red, usually in "Courier" or "Arial" typeface, and the output in blue in Times Roman, in usual mathematical notation.

Every Maple input statement or command *must* end with a semicolon or a colon. If you end a statement with a *semicolon* and then type `<enter>` the result is displayed immediately. If you end the statement with a *colon*, the statement is still executed, but the result isn't displayed. This is sometimes handy if the output has many lines that you don't need to see.

If there is room, you can put two or more input statements on the same line, provided each statement ends with a semicolon or colon. When you type `<enter>`, all the statements on the line are entered and executed. But it is usually better, for the sake of clarity, to put each statement on a separate line. Occasionally you will want to enter several lines but delay execution until you've entered them all. In that case, type `<Shift> + <enter>` for all but the final line. Such a group of commands is called an *execution group*, and it is denoted by a bracket in the left margin.

If you forget and enter a command without a semicolon, Maple will give you an error message; you can then move the cursor back to the end of the input line, add the semicolon, and type `<enter>` again. Alternatively, you can enter the semicolon alone on the next line (`;` `<enter>`). Exception: No semicolon is needed for the "help" command. To get help on a topic, type `?topic <enter>`, no semicolon needed.

Use `+` for addition, `*` for multiplication, `^` or `**` for exponentiation. Four times x to the fifth power is entered as `4*x^5`. (The usual rules of precedence are used; exponentiation is carried out first, then multiplication and division, and finally addition.) If you enter `4x`, Maple will give you an error message. You can then move the cursor back up to the input line containing the error, correct it, and re-enter it correctly as `4*x`. The cursor doesn't have to be at the end of the input line to re-enter the line; it can be anywhere in the line.

Maple generally ignores spaces in algebraic expressions, so feel free to add spaces to improve legibility. `x + 3` is usually more readable than `x+3`.

Maple is case sensitive; you can't interchange capital and lowercase letters. Example: the cosine of x is `cos(x)`, not `COS(x)`. Most functions are lower case; an exception is the gamma function, usually written as $\Gamma(x)$. Maple's notation is `GAMMA(x)`. The constant π **MUST** be spelled `Pi`, and the imaginary unit (the square root of -1) is `I`, not `i`. The base of natural logs is `exp(1)`, not `e` or `E`.

Here are Maple notations for a few common functions, to illustrate the general pattern.

sine of x :	$\sin(x)$	(x in radians, not degrees)
inverse sine of x :	$\arcsin(x)$	
tangent of x :	$\tan(x)$	
square root of x :	\sqrt{x}	
exponential of x :	$\exp(x)$	
natural logarithm of x :	$\ln(x)$	or $\log(x)$

The various delimiters such as $()$, $[]$, and $\{\}$ are *not* interchangeable in Maple. Usually functions and algebraic expressions require $()$ parentheses. $[]$ denotes a *list* or a *sequence* (with a definite order); $\{\}$ denotes a *set*. These can be lists or sets of constants, expressions, functions, equations, plot structures, matrices, or other entities.

A range of values of a variable is denoted with two dots. If x ranges from a to b , then type $x = a..b$, *not* $x = a, b$

When you are editing Maple code, you may sometimes want to insert a line of Maple input between two existing lines. To insert a blank line *after* the line the cursor is on, type $\langle\text{ctrl}\rangle + j$; to insert a blank line *before* the cursor, type $\langle\text{ctrl}\rangle + k$.

Often you'll want to clear Maple's memory of all previous input and output. This is done easily with the command `restart;` $\langle\text{enter}\rangle$. You will use this command frequently; get in the habit of using it *first*, before you begin a new calculation.

In the following sections, don't just read through the material. Sit down at a computer, log on, get into Maple, try *all* the input statements, and note carefully what happens.

While you are learning Maple, *make up your own manual*. Write down things you have trouble remembering or that cause confusion. And don't be afraid to ask questions; don't get hung up on trivial but basic things.

Arithmetic

Let's begin by using Maple as a simple calculator. Maple can do both integer and floating-point arithmetic: If the input numbers are all integers or rational fractions, the output will be also. Try these examples:

```
restart;
3 + 4;
3/4 + 2/7;
2^5;
2^100;
(2/5)^15;
```

Now try entering some of these without the semicolon at the end; you will get an error message because Maple is waiting for you to end the statement. You can either type `;<enter>` alone on the next line, or you can move the cursor back to where the missing semicolon belongs, put it in, and type `<enter>`.

In arithmetic operations, if one or more of the numbers are expressed in decimal or floating-point form, the output will also be floating-point. The default number of digits is 10, but it can be changed. Try these examples:

```
2.0^100;  
(3.14/4)^24;
```

You can *force* Maple to do floating-point evaluation even with integer input by using the command `evalf` followed by the expression in parentheses. Try these examples:

```
evalf(1/3);  
evalf(2^100);  
evalf(2^(1/2));  
evalf((2/5)^15);
```

In the last two examples, note carefully how the parentheses are nested, and make sure you understand them.

Thus there are two ways to force floating-point evaluation: using floating-point numbers or using the `evalf` command. Compare the following:

```
(3/2)^4;  
(1.5)^4;  
evalf((3/2)^4);
```

In floating-point calculations, the default number of digits is 10. If you want a different number, you can specify it in the `evalf` command by adding a comma followed by the number of digits you want (inside the final parenthesis):

```
evalf((2/5)^15, 5);  
Pi;  
evalf(Pi, 100);
```

There's an alternative: Enter the command `Digits := 20;` (Note the capital D.) Then all succeeding floating-point calculations will be done and displayed with 20-digit precision until you enter `restart;` or another `Digits` command.

Assignment Statements

The symbol `:=` is used to give something a symbolic name. Example: `f := x^2 + 3;` is *not* an equation; it is an *assignment statement*. It means that I am giving the name `f` to the expression `x^2 + 3` (i.e., assigning the value `x^2 + 3` to the variable named `f`). A name may be given to an expression, a function, an equation, a plot structure, a set or list of any of these, a matrix, etc. *There must be no space between the colon and the equals*. Otherwise, Maple generally ignores spaces in algebraic expressions.

You can assign a numerical value to a variable. For example

```
x := 4;
```

In this case, no operation is performed, and Maple simply repeats your statement. But now type

```
x^2 + 3;
```

and see what happens. Also, you can assign a name to an expression. Try this:

```
y := x^2 + 3;
```

Maple already has a value for `x`, so it returns a value for `y`.

Maple remembers all such assignments indefinitely. If you want to erase the record and start from the beginning, use the command `restart`; Or if you want to unassign an individual variable `x`, type `x := 'x'`;

A variable name can have any combination of letters (capital or lower-case) and numbers, but it must start with a letter. Punctuation marks aren't allowed, but underscores are OK. Some combinations of letters, such as `cos`, `exp`, and `diff`, have special meanings as part of Maple commands. These names are *protected* and thus aren't available for general use. If you try to assign to one of these, Maple gives you an error message. Usually it's a good idea to choose variable names that suggest the physical significance of the quantities they represent.

The percent sign `%` is used to refer to the previous result. For example, `f := cos(%);` means that we give the name `f` to the cosine of whatever appeared as output on the preceding line. `%%` refers to the second-to-last line. And so on, up to `%%%`. However, I strongly recommend that when you're first learning Maple, you avoid the use of the `%` completely and instead give *everything* a name if there's any chance you might want to refer to it later.

In Maple, *expressions*, *functions*, and *equations* are three distinct kinds of entities; the distinction will emerge as you use Maple. For example, `f := a*x^2 + b*x + c;` defines an *expression* called `f`; `g := x -> a*x^2 + b*x + c;` defines a *function* called `g`; `h := a*x^2 + b*x + c = 0;` defines an *equation* called `h`. The arrow (made with a hyphen and a "greater than") shows explicitly that a function is a *mapping*, not simply an expression containing the particular variable `x`. We'll discuss this distinction further on pages 1-8 and 2-1.

Algebra

Maple can do a wide variety of algebraic manipulations. Here is a small sample.

Suppose we have an algebraic expression $x^2 + 2x - 3$. We choose to give this expression the name `expr`:

```
expr := x^2 + 2*x - 3;
```

Maple can factor this expression:

```
pieces := factor(expr);
```

The factors are $(x - 1)$ and $(x + 3)$. Then if we want to, we can re-combine them:

```
expand(pieces);
```

Suppose we want to find the roots of the equation $x^2 + 2x - 3 = 0$. Let's give this equation the name "eq":

```
eq := x^2 + 2*x - 3 = 0;
```

Then we use the Maple command `solve`:

```
solve(eq);
```

Alternatively,

```
solve(x^2 + 2*x - 3 = 0);    or    solve(expr = 0);
```

But look at what happens when we solve a similar equation $x^2 + 2ax - 3a^2 = 0$.

```
eq := x^2 + 2*a*x - 3*a^2 = 0;
```

```
solve(eq);
```

We get something unexpected and possibly puzzling. The problem is that Maple doesn't know which symbol is the unknown, x or a . We must tell it to solve for x :

```
solve(eq, x);
```

Now we get what we expected!

Three other commands that are useful for algebraic manipulation are `simplify`, `collect`, and `isolate`. For explanation of these commands, see the Maple Help files.

Sometimes the roots of an equation can't be expressed in terms of familiar functions but have to be obtained by numerical approximation. An example is $\tan(x) = x + 1$. The Maple commands `eq := tan(x) = x + 1;` and `solve(eq, x);` just give you back a restatement of the problem. Instead, use the command `fsolve(eq, x);` or `fsolve(tan(x) = x + 1, x);` The command `fsolve` tells Maple to obtain a floating-point *numerical* approximation of the root. In this particular case, the equation has infinitely many roots; to find a root in a particular range, say $x = 0$ to 2 , use the command `fsolve(tan(x) = x + 1, x = 0..2);`

Maple can solve *sets* of simultaneous equations. Consider the simple pair of equations

$x + y = 5$, $x - 2y = -6$. We give each equation a name:

```
eq1 := x + y = 5;      eq2 := x - 2*y = -6;
```

Maple requires a set to be enclosed in curly brackets. Here the pair of equations form one set, and the unknowns x and y form another. Thus the Maple command is:

```
solve({eq1, eq2}, {x, y});
```

There are many other useful commands that do algebraic manipulations. A few are listed in the summary of Maple commands, starting on page 1-9.

Calculus

Maple can do symbolic derivatives and integrals. Try these: For an expression

```
expr := x^6 - a*x^2;  the derivative with respect to x is given by
```

```
diff(expr, x);      or      diff(x^6 - a*x^2, x);
```

Note that you have to tell Maple *what* to differentiate (in this case, *expr*) and the variable (in this case, x) *with respect to which* the derivative is to be calculated. (Otherwise Maple doesn't know whether the variable is a or x .) Try omitting the x in the above command `diff(expr,x)`, and see what happens.

For higher derivatives, use a dollar sign. For the fourth derivative of *expr* with respect to x ,

```
diff(expr, x$4);      or      diff(x^6 - a*x^2, x$4);
```

For the derivative of a *function*, the command is different. If `fcn := x -> x^6 - a*x^2`,

the command is simply `D(fcn)`; with no variable specified Try it!

(Note that the derivative of an expression is another expression, and the derivative of a function is another function.)

For the indefinite integral of *expr* (with respect to x), type

```
int(expr, x);      or      int(x^6 - a*x^2, x);
```

Note that you have to specify the variable (x), as you do with derivatives. (Try entering this without the final x , and see what happens.) For a *definite* integral, attach the limits to the variable:

```
int(expr, x = 2..3);      or      int(x^6 - a*x^2, x = 2..3);
```

Now try replacing `diff` with `Diff` and `int` with `Int` in the above commands. These are called *inert operators*; they indicate the operation to be performed but don't actually carry it out. When you are looking for syntax errors such as unmatched delimiters, it is often helpful to make this replacement. For other inert operators, read the help file `?inert`.

Plots

Suppose we want to plot a graph of the expression $\sin x$ from $x = 0$ to 2π . Easy!

```
plot(sin(x), x = 0..2*Pi);
```

Note the spelling of Pi and the parentheses in the sine function. Also note that you have to specify the range of values of the independent variable x . Try omitting this range, and see the rather mysterious error message Maple gives you. An alternative method is:

```
expr := sin(x);
plot(expr, x = 0..2*Pi);
```

To plot a *function* (as distinguished from an *expression*), the commands are similar but don't include a variable name. For example, if $\text{fcn} := x \rightarrow \sin(x)$, the correct plot command is

```
plot(fcn, 0..2);    or    plot(sin, 0..2);
```

As another example, suppose we want to plot $x^2 + 3x$. Then

```
expr := x^2 + 3*x;    fcn := x -> x^2 + 3*x;
```

The following `plot` commands are OK:

```
plot(expr, x = 0..2);    plot(x^2 + 3*x, x = 0..2);
plot(fcn, 0..2);    plot(x -> x^2 + 3*x, 0..2);    plot(fcn(x), x = 0..2);
```

But these don't work:

```
plot(expr, 0..2);    plot(fcn, x = 0..2);
```

Be sure you understand crucial distinction between *functions* and *expressions*. Also see the discussion on page 2-1.

Now see what happens when we plot $\text{expr} = 1/x$.

```
expr := 1/x;
plot(expr, x = -1..1);
```

The infinite discontinuity at $x = 0$ doesn't cause Maple distress, as you might expect; it just quits plotting when `expr` reaches some large value. To get a more useful graph, we may want to restrict the range of values on the vertical axis. To do this we include a second variable range in the `plot` command. Maple likes to call the generic vertical coordinate y . Try this:

```
plot(expr, x = -1..1, y = -5..5);
```

There are many options for the `plot` command that can be used to label axes, change the color of the graph, change line or point style, and so on. To explore these, type

```
?plot[options]    or    ?plot, options
```

Be sure to use square brackets, not parentheses; no semicolon is needed at the end.

Inspirational Message

Like all computer languages, Maple is fussy about details of syntax. If you don't speak Maple's language, it can't help you. Unmatched or omitted delimiters (), [], or { }, are a very common error. Note the comment at the bottom of page 1-7 concerning finding unmatched delimiters. If you try to get Maple to do something, and it won't do what you want because of some trivial but crucial syntax error that you can't find, don't sit and spin your wheels for hours -- ask somebody for help! The more you learn about Maple, the more you will appreciate its enormous power and versatility. You may feel like you're driving a Ferrari; I hope so.

During this course, Maple will save us from enormous amounts of dogwork, and it will enable us to solve a lot of problems (such as numerical solutions of nonlinear differential equations) that would be hopeless without it. Don't ignore it in the hope that it will go away; it won't. And it will become one of your best friends if you let it!

Summary of Maple Commands

The following three pages list several Maple commands that we'll find useful during the course. Don't sit down and memorize them all at once; learn them a few at a time as we come to applications during the course. And remember that you can always get help for any command from Maple's excellent help files. To get help for the command `fiddle`, simply type `?fiddle` (no semicolon needed).

In this list, `expr` is an expression, `fcn` is a function, and `x` is a variable.

Algebra

```

expand(expr);          factor(expr);          simplify(expr);
sort(expr);           collect(expr);        combine(expr);
eval(expr);           evalf(expr);          evalf(expr, 20);
isolate(eq, expr),    normal(expr);         evalc(expr);
solve(expr = 0, x);   solve({equation set}, {unknowns});
fsolve(expr = 0);     fsolve(expr = 0, x = 3..4);
subs(x = 3, expr);    Digits := 20;         x := 'x';
numer(expr);          denom(expr);          lhs(eq);              rhs(eq);
Re(expr);             Im(expr);             conjugate(expr);      sqrt(expr);
add(expr, n = a..b);  sum(expr, n = a..b);  seq(expr, n = a..b);
do loop: for k from a by b to c do ... ; end do;
f := proc(n) .... ; end;

```

Series

```

series(expr, x = 3, n);
taylor(expr, x = 3, n);
convert(series, polynom);

```

Miscellaneous

```

restart;
list:  L := [a, b, c];      L[1] = a          set:  S := {a, b, c};      S[2] = b
Pi     exp(1)              I

```

Calculus

```

diff(expr, x);          diff(expr, x$6);          Diff(expr,x);
D(fcn);                (D@@6)(fcn);          expr := fcn(x);
fcn := x -> expr(x);   fcn := (x,y) -> expr(x, y);
fcn := unapply(expr, x); fcn := unapply(expr, x, y);
int(expr, x);          int(expr, x = a..b);   Int(expr, x = a..b);
int(fcn(x), x);       int(fcn(x), x = a..b); int(fcn, a..b);
evalf(int(expr, x = a..b)); evalf(int(fcn(x), x = a..b));

```

Plots

```

plot(expr, x = a..b);   plot({expr1, expr2}, x = a..b);
plot([expr1, expr2, t = a..b]); (parametric)
plot([expr1, expr2, x = a..b], coords = polar);
plot(fcn, a..b);       plot(fcn(x), x = a..b);
plot(expr, x = a..b, numpoints = 100, color = black, title = Mygraph);
with(plots, polarplot); polarplot(expr, t = a..b);
polarplot([expr1, expr2, t = a..b]);
plot3d(expr, x = a..b, y = c..d); plot3d(proc, a..b, c..d)
plot([ [x1, y1], [x2, y2], [x3, y3] ]);
pointlist := seq([n, x[n]], n = 0..N); plot([pointlist]);
plotstructure := plot(expr, x = a..b); with(plots, display);
display(plotstructure); display({plotstructure1, plotstructure2});
with(plots, odeplot); odeplot(solution, a..b);
odeplot(solution, [x(t), diff(x(t), t)], a..b);
with(DEtools); dfieldplot({eq1, eq2}, [x(t), v(t)], t = 0..10, x = a..b, v = c..d);

```

Animate

```
with(plots, animate);      with(plots, animate3d);  
animate(sin(x - t), x = 0..a, t = 0..b);  
animate({expr1, expr2}, x = 0..a, t = 0..b);  
animate3d(expr, x = 0..a, y = 0..b, t = 0..c);
```

Matrix Algebra

```
with(linalg);  
A := matrix(3, 3, [1, 2, 3, 3, 4, 5, 5, 6, 9]);  
A + B;      A - B;      A &* B;      multiply(A, B);  
evalm(matrixexpr);      inverse(A);      det(A);  
transpose(A);      htranspose(A);  
eigenvalues(A);      eigenvectors(A);      normalize(vectorlist);
```