Some Maple Questions and Answers

Blair F. Madore

July 6, 1994

This document deals with some commonly asked questions about Maple. It will certainly be useful for newer users of Maple and helpful for people trying to support Maple. Some of the questions are simple, others more advanced. At the end of each question I've indicated where you might find relevant information.

Question 1: *I'm using nested* `for` *loops, and any assignments or other statements I make within the second loop do not appear. At first I didn't think statements in the inner loop were being processed at all but I could force something to print by using* `lprint`*. Why don't they print regularly and how can I make them print?*

Answer: Maple has a global variable called `printlevel`. This controls how much information is printed. The default setting is 1. At this setting only statements one nested level or less away from the top level will echo their results. In our example we would need to raise the printlevel to 2. this can be easily done by assigning 2 to the global variable `printlevel`.

This could cause other information to appear. See the help page for `printlevel` for details on how it affects other information messages and procedures.

Other help pages relevant to this topic include: `print, lprint, loops`.

Question 2: *I have an expression in three variables (x, y, z) and three symbolic parameters (r, s, t). I want to find the coefficients of x^2yz and xy^2 but Maple seems to treat the expression as a polynomial in six variables. I tried using* `collect` *but Maple puts a precedence on the variables. I want them equally weighted. Do I have to write a program to do this?*

Answer: No you don't have to write a program to do this. You're using the right function `collect` but you need to use the `distributed` option. Here's an example:

```
> expr:=56*y*r*z+r^2*s*3*y*t*x+r*s*y^2+r**2*x**2+2*r*x*s*y+
> s*r^3*x*z+2*s*y*t*z+t**2*z**2+2*r*x*t*z+s**2*y**2;
```

```
                          2                2    2  2                 3
expr := 56 y r z + 3 r  s y t x + r s y  + r  x  + 2 r x s y + s r  x z

                   2  2                 2  2
    + 2 s y t z + t  z  + 2 r x t z + s  y
```

```
> collect(expr,[x,y,z]);
     2  2            2                  3           2          2
    r  x  + ((2 r s + 3 r  s t) y + (2 r t + s r ) z) x + (s  + r s) y

                                2  2
        + (56 r + 2 s t) z y + t  z
```

```
> collect(expr,[x,y,z],distributed);
                             2  2    2  2     2          2
     (56 r + 2 s t) z y + r  x  + t  z  + (s  + r s) y

                      2                   3
     +  (2 r s + 3 r  s t) x y + (2 r t + s r ) z x
```

Now it's much easier to see the coefficient of xz is $2rt + sr^3$. Maple has a lot of powerful features that are available as options to standard functions. Always watch for options when reading help pages.

Related help pages include: `collect`, `coeff`, `coeffs`.

Question 3: *Can I see the Maple source code? How?*

Answer: Maple has a kernel written in C and a library written in the Maple programming language. The kernel is not available to the public, but the library is. To see a Maple Library procedure, e.g. `gcd`, you use the `print` command.

```
> print(gcd);
proc(aa,bb,cofa,cofb) ... end
```

To avoid the problem of information explosion procedures are not completely printed at Maple's default settings as shown above. To force full printing you need only reset one of the interface variables.

```
> interface(verboseproc=2);
> print(gcd);
proc(aa,bb,cofa,cofb)
local a,b,inda,indb,inds,ta,tb,res,x,GCD,II,Z;
options `Copyright 1990 by the University of Waterloo`;

      ....omitted about 100 lines to save space.....

end
```

Printing the code in this way does not include any of the programmer's comments. Some Maple procedures are coded completely within the kernel. Their code is not available. If you try to print them you see the following type of special procedure body.

```
> print(diff);
proc() options builtin,remember; 61 end
```

Related help pages include: `interface`, `print`.

Question 4: *I generated a Taylor series using the* `series` *command and now I want to evaluate it at several values. But I can't seem to get rid of the* `O(..)` *term.*

Answer: You need to use the `convert` function. Currently you have a series data structure and you want a polynomial (a sum of products data structure). You could issue the command `convert(`*series*`, polynom)` to remove the order term.

```
> S:=series(1/(1-3*x),x,10);
                        2        3        4         5         6          7
   S := 1 + 3 x + 9 x  + 27 x  + 81 x  + 243 x  + 729 x  + 2187 x


                8         9       10
         + 6561 x + 19683 x  + O(x   )

> p:=convert(S,polynom);
                         2        3        4         5         6          7
   p :=  1 + 3 x + 9 x  + 27 x  + 81 x  + 243 x  + 729 x  + 2178 x

                8          9
         + 6561 x  + 19683 x

> subs(x=2,p);
                                       12093235
```

This is also effective for Laurent series, although the result is not neccessarily a polynomial.

An alternative way to remove the order term is shown below. The result is still a series data structure but it now represents an exact series. Exact series and polynomials appear exactly the same. Substitutions and evaluation can be done on an exact series just as if it were a polynomial. Due to the nature of its data structure this form may be preferable for some applications.

```
> p2:=subs(O(1)=0,S);
                           2        3        4         5         6          7
      p2 := 1 + 3 x + 9 x  + 27 x  + 81 x  + 243 x  + 729 x  + 2187 x

                     8          9
              + 6561 x  + 19683 x

> subs(x=2,p2);
                                       12093235
```

Related help pages include: `convert[polynom]`, `subs`, `order`, `type[series]`.

Question 5: *Now I've converted my series to a polynomial. I've decided to drop the last few terms, those of degree higher than 5. I tried to assign zero to t^6 but that gave me a syntax error. Then I tried to substitute zero for t^6 but that only zeroed the t^6 term.*

Answer: Maple only allows assignment to names. `subs` does not work in this case since it is strictly syntactic. The easiest thing to do is to use `rem` as follows:

```
> p;
                   2        3        4         5         6          7
       1 + 3 x + 9 x  + 27 x  + 81 x  + 243 x  + 729 x  + 2178 x

                 8           9
          + 6561 x  + 19683 x

> rem(p,x^6,x);
                                      2        3        4         5
                       1 + 3 x + 9 x  + 27 x  + 81 x  + 243 x
```

Related help pages include: `rem`.

Question 6: *I have assigned a matrix to* `A`*. If I just type* `A;` *all I get is the name* `A`*. I don't see the matrix. Why is this different from a polynomial and how can I see the matrix?*

Answer: Matrices are a special type of an array. Arrays (and tables) have special evaluation rules in Maple. If they are named, the name only evaluates to itself but, it points to the matrix data structure. Think of it this way: the name of a polynomial is a variable which *contains* the polynomial while the name of a matrix is a variable which *points to* the matrix but *contains* its own name.

There are several ways to "see" the matrix. `eval(A)` returns the matrix data structure which is interpreted by the prettyprinter and printed the way you want to see it. `print(A)` will also display the matrix.

```
> A:=linalg[matrix](2,2,max);
                                           [ 1  2 ]
                                      A := [      ]
                                           [ 2  2 ]

> A;
                                            A

> eval(A);
                                         [ 1  2 ]
                                         [      ]
                                         [ 2  2 ]

> print(A);
                                         [ 1  2 ]
                                         [      ]
                                         [ 2  2 ]
```

Related help pages include: `print, eval, matrix, array, op`.

Question 7: *Can I print Maple graphs (2D) even if I don't have a graphics terminal?*

Answer: Yes. Maple supports graphic output for many printers and on many platforms, including imagen 300, DEC ln03, pic, PostScript and Macintosh plot (for use with MacMaple).

Even if you're working at an ASCII terminal you can produce two dimensional character plots. The most common way of producing a high-quality paper copy of a graph is through PostScript. Here's an example of how to do this.

```
> interface(plotdevice=postscript);
> interface(plotoutput=`myfile.ps`);
> K:=(n,t) -> 1/(n+1)*(sin((n+1)*t/2)/sin(t/2))^2;
                                                   2
                                 sin(1/2 (n + 1) t)
                   K := (n,t) -> -------------------
                                                   2
                                 (n + 1) sin(1/2 t)

> plot({seq(K(n,t),n=1..8)},t=-Pi..Pi,title=`Fejer Kernel`);
```

A PostScript version of the plot is placed in the file *myfile.ps*. It can then be printed on any PostScript printer. The actual command to send the file to the printer varies from system to system.

The resulting plot, is a plot of the first 8 functions in the Fejer summability kernel (important in theoretical Fourier analysis).

Related help files include: `plot`, `interface`, `plotsetup`, `plot[device]`, `seq`.

Question 8: *How do I find out what algorithms Maple uses for a specific function?*

Answer: The help pages for some functions include details of which algorithms are used and some may include references.

Question 9: *How can I make Maple pull constants outside the integration?*

Answer: Currently there is no predefined function that will do this but it can be easily accomplished given Maple's design. Greg Fee of the Maple Symbolic Computation Lab suggests you define a function `` `expand/Int` `` as follows:

```
> `expand/Int` := proc(a,r)
>  local x,f,y,c;
>      if type(r,name) then x := r
>      elif type(r,`=`) then x := op(1,r)
>      else ERROR(`bad 2nd arg`)
>      fi;
>      if type(a,`*`) then
>          c := 1;
>          y := 1;
>          for f in a do  if has(f,x) then y := y*f else c := c*f fi od;
>          c*Int(y,r)
>      elif type(a,`+`) then map(procname,a,r)
>      else Int(a,r)
>      fi
>  end;
```

Maple is designed so that users may add their own special routines for dealing with special functions. Functions such as `expand`, `simplify`, `combine`, `int` and others have a specially designed interface to make this easy. For example, when `expand` is passed an argument such as `Int(`*something*`)`, Maple will invoke the routine `` `expand/Int` ``, if it exists, on *something*.

Here's an example of how the above `` `expand/Int` `` works:

```
> Int(a*x^n+a*f(x)*sin(1)*Pi,x);
                                  /
                                 |     n
                                 |  a x  + a f(x) sin(1) Pi dx
                                 |
                                /

> expand(");
                                /                          /
                               |   n                      |
                          a    |  x  dx + a sin(1) Pi     |  f(x) dx
                               |                          |
                              /                          /
```

Related help files include: `expand, combine, Int, simplify`

Question 10: *I want to compute the definite integral* $\int_0^\infty e^{-ut} ln(t)^2 \mathrm{d}t$ *. How can I tell Maple that u is positive? In general, how can I declare a name to be positive, real, integral, even, odd, etc.*

Answer: In Maple version V, you can declare a name to be positive or negative as follows.

```
> signum(u) := 1; # Tell Maple that u > 0

                                   signum(u) := 1

> integrate( exp(-u*t) * ln(t)^2, t=0..infinity );

                        2                          2         2
                   ln(u)       ln(u) gamma        Pi    gamma
                   ------ + 2 ----------- + 1/6 --- + ------
                     u             u              u       u
```

Note, you must make the declaration BEFORE you try to compute the integral because Maple remembers the results of previous integrations. There is no ability to make any other kinds of declarations. Such a facility that supports more general types of declarations is presently being developed and will be available in the next release of Maple.

Question 11: *I want to plot the following piecewise defined function in Maple but I always seem to get an error. Let* $f(x, y) = \mathrm{erf}(x - y)$ *if* $x > y$ *and* $\mathrm{erf}(y - x)$ *if* $x < y$. *I defined* f *as a Maple procedure thus*

```
f := proc(x,y) if x>y then erf(x-y) else erf(y-x) fi end;
```

But when I try to plot this I get

```
> plot3d( f(x,y), x=-2..2, y=-2..2 );
Error, (in f) cannot evaluate boolean
```

Answer: The plot functions have normal evaluation rules. What happens then is that before `plot3d` is called, $f(x, y)$ is evaluated at *symbolic* x and y. I.e. we get

```
> f(x,y);
Error, (in f) cannot evaluate boolean
```

Maple gives an error because it can't decide if $x < y$ when x and y have no values. There are two ways of using the plotting commands `plot` and `plot3d` as illustrated in the following example for plotting the binomial function.

```
plot3d( binomial(n,k), n=-3..3, k=-3..3 );
plot3d( binomial, -3..3, -3..3 );
```

Most users are familiar with the first way, which is to plot a formula in two variables, in this case n and k. The second way allows you to plot a function given by a Maple procedure or an operator expression. Therefore the solution to the problem is

```
plot3d( f, -2..2, -2..2 );
```

Alternatively, one can use quotes to prevent $f(x, y)$ from being evaluated before the plot routine is called thus

```
plot3d( 'f(x,y)', x=-2..2, y=-2..2 );
```

Note the frustrated user tried to get around the problem by trying to express $f(x, y)$ this way

```
f := proc(x,y) if signum(x-y)=1 then erf(x-y) else erf(y-x) fi end;
```

This avoids the error but produced a different surprise. This ends up always plotting $\mathrm{erf}(y-x)$ because

```
> signum(x-y);
                              signum(x - y)
```

cannot be evaluted to ± 1 for symbolic x and y hence the `if` test always returns `false` !!

```
> if signum(x-y) = 1 then true else false fi;

                                  false

> f(x,y);
                               - erf(x - y)
```

Note Maple simplified $\mathrm{erf}(y - x)$ to $-\mathrm{erf}(x - y)$. The solution again is to simply plot the function f directly `plot3d( f, -2..2, -2..2 );` Of course, since $\mathrm{erf}(-x) = -\mathrm{erf}(-x)$ we could have expressed $f(x, y)$ as a formula

```
plot3d( signum(x-y)*erf(x-y), x=-2..2, y=-2..2 );
```