

Maple Lecture 23. The assume facility and Simplification

This lecture tries to summarize Chapters 13 and 14 of [1]. We encounter the normalization issue again, but now for expressions more general than polynomials, involving trigonometric and exponential functions. Turning back to polynomials and term rewriting algorithms, we show how to use Gröbner basis for constrained optimization.

23.1 The assume facility

We have encountered the assume already, for instance: with improper integrals, depending on a parameter:

```
[> exint := int(exp(a*t),t=0..infinity);
[> assume(a<0);
[> exint;
```

Here we give a more complete treatment of this facility.

23.1.1 Basics of assume

In the case below Maple asks for an assumption:

```
[> assume(x>0);                # make an assumption
[> about(x);                    # query the assumption
[> x;                          # we see x with a flag
[> additionally(x<2);         # add an assumption
[> about(x);
[> x := evaln(x);             # remove assumptions
```

The last command is of course the same as `x := 'x'`;

Suppose we want to declare something as a constant.

```
[> constants;                 # constants known by Maple
[> constants:=constants,NewConstant; # append a new constant
[> D(f)(x);                   # general derivative
[> D(NewConstant)(x);        # derivative of constant is zero
```

The alternative uses `assume`:

```
[> assume(myConstant,constant); # make an assumption
[> D(myConstant)(x);           # verify the assumption
```

23.1.2 Using properties in arithmetic

The `assume` command creates an assumption, with `additionally` we can add extra assumptions, and with `about` we can see the assumption made on the variable. As illustration we see the use of assumption in connection with the bisection method for finding roots of a function. Suppose we are searching for a root of $\cos(x) - x^2 = 0$.

```
[> f := x -> cos(x) - x^2;
[> f(0); f(Pi);
```

By the mean value theorem we know that there is a root in the interval $[0, \text{Pi}]$:

```
[> assume(a>0,a<Pi);
[> about(a);
[> f(Pi/2);
```

We apply the mean value theorem again:

```
[> additionally(a<Pi/2);
[> about(a);
```

So we can use assumptions to perform some kind of interval arithmetic.

23.1.3 An algebra of properties

With the command `is` we can ask Maple to verify properties.

```
[> assume(n,odd); assume(m,odd);
[> is(n+2,odd);
[> is(n^2,odd);
[> is(n^2+m,odd);
```

23.2 Simplification

We have already seen the `simplify` command when we worked with algebraic numbers. Most of what we will see below applies to trigonometric and exponential functions.

In many cases Maple finds out which transformation rules to apply. Here are some examples:

```
[> abs(-Pi*x);
[> min(a,3,4,cos(b));
```

23.2.1 expand and combine

We have seen `expand` for polynomials, but the command also applies to trigonometric functions:

```
[> expression := exp(x+y) + sin(x+y);
[> expand(expression^2);
```

We may wish to freeze the `x+y` in the argument above.

```
[> frontend(expand,[expression^2]);
```

Sometimes Maple does not do the `expand`:

```
[> expand(ln(x*y));
```

... unless both `x` and `y` are positive:

```
[> assume(x>0); assume(y>0);
[> expand(ln(x*y));
[> x := 'x': y := 'y': # remove the assumptions
```

If as above, we only want a temporary assumption on the variables, then we better use `assuming`:

```
[> expand(ln(x*y)) assuming x>0,y>0;
```

The opposite of `expand` is the command `combine`:

```
[> e1 := expand(cos(a+b));
[> combine(e1);
[> expand((cos(x))^3*(sin(3*x))^2);
[> combine((cos(x))^3*(sin(3*x))^2);
[> expand(%);
```

Be aware that there some combinations are not allowed:

```
[> e2 := ln(x)+ln(y);
[> combine(e2);
[> combine(e2,'symbolic'); # implicit assumptions
[> combine(e2) assuming x>0,y>0; # better to have explicit assumptions
```

23.2.2 simplify

We have used `simplify` in connection with algebraic numbers. Here we see the application of `simplify` to exponentials and trigonometric functions:

```
[> expression := exp(x)*exp(y) + sin(x)^2 + cos(x)^2;
[> simplify(expression,trig);           # using trigonometric identities
[> simplify(expression,exp);           # simplifying exponentials
[> simplify(expression,exp,trig);      # both simplifications
```

Adding extra assumptions, we gain control over the simplification process. With the option `symbolic`, we can enforce the symbolic simplification. We illustrate this case with what was formerly (and formally) known as the “square root bug” in computer algebra. Recall our lecture on complex numbers.

```
[> sqrtx2 := sqrt(x^2);
[> simplify(sqrtx2);
[> simplify(sqrtx2,symbolic);
[> simplify(sqrtx2,delay);
```

The option `delay` has an effect opposite to the `symbolic` option. It delays simplification, unless a completely mathematically sound justification exists.

23.2.3 convert and trigonometric simplification

We can write trigonometric expressions with exponentials and vice versa:

```
[> convert(1/((x-3)*(x^2+4*x+4)), 'parfrac', x);
[> sc := sin(x)*cos(y);
[> expsc := convert(sc,exp);
[> back := convert(expsc,trig);         # is this our original expression?
[> simplify(back);                     # another good use of simplify
```

With `trigsubs` we see various suggestions to substitute a given expression, for example:

```
[> suggestions := trigsubs(sin(x+y));
[> suggestions[1] = suggestions[7];
[> subs(%,(sin(x+y))^2);
```

23.3 Simplification w.r.t. side relations

Suppose $x^2 + y^2 = 1$ (analogous to $\sin^2 + \cos^2 = 1$), then we can simplify

```
[> simplify(x^3+y^3,{x^2+y^2 = 1});
```

The example above is a particular case of a general term rewriting scheme.

Let us look at constrained minimization with Lagrange multipliers, something we have seen in our multivariate calculus class. As example, we consider the problem of finding those points on the unit sphere $x^2 + y^2 + z^2 = 1$ which take minimal or maximal values in the function $f(x, y, z) = x^2 + 2xyz - z^2$,

```
[> g := x^2 + y^2 + z^2 - 1;           # constraint = side relation
[> f := x^2 + 2*x*y*z - z^2;          # locate minima and maxima of f
[> sys := {diff(f,x)-lambda*diff(g,x),
           diff(f,y)-lambda*diff(g,y),
           diff(f,z)-lambda*diff(g,z),g};
```

With a Gröbner basis we rewrite the equations into an equivalent system in triangular form.

```
[> with(grobner);
[> G := gbasis(sys, [x,y,z,lambda], plex);
[> nops(G);
[> solve(G[10], lambda);           # gives the Lagrange multipliers
```

The Gröbner basis reveals lots of other relations between the variables. If we are interested in the solutions, we select those equations that give us x , y , and z in function of the Lagrange multipliers:

```
[> G[1];
[> G[5];
[> G[8];
```

It is a nice exercise to continue the solution of this problem and to find actual values for x , y , and z .

23.4 Assignments

1. Consider $\int_0^{\infty} e^{\alpha x^2} \cos(x) dx$. Give the Maple commands to compute the symbolic value of the integral for any negative value of the parameter α .
2. Execute `assume(apple,"red");` followed by `is(apple,"red")`.
Give the Maple commands to change the remember table of `is` so that `is(apple,"green")` returns true. Do *NOT* use `assume` for this change.
3. Give the Maple commands to show the identity

$$\tan(x) + \tan(y) = \frac{\sin(x+y)}{\cos(x) + \cos(y)}.$$

4. Show that $\ln(\tan(\frac{1}{2}x + \frac{1}{4}\pi)) - \operatorname{arcsinh}(\tan(x)) = 0$, symbolically and numerically.
5. Do $f := 1/(x^2 + y)$ followed by `i1 := int(int(f,x),y)` and `i2 := int(int(f,y),x)`.
Can you show that `i1` and `i2` are the same?
6. Find those points on the surface $x^2 - xy + y^2 - z^2 = 1$ closest to the origin.
Set up the system with Lagrange multipliers and solve.
7. Given the conic section $Ax^2 + 2Bxy + Cy^2 = 1$, where $A > 0$ and $B^2 < AC$. Let M denote the distance from the origin to the furthest point on the conic. Use Maple to show that

$$M^2 = \frac{A + C + \sqrt{(A - C)^2 + 4B^2}}{2(AC - B^2)}.$$

Find a formula for m^2 , when m denotes the distance from the origin to the nearest point on the conic.

References

- [1] A. Heck. *Introduction to Maple*. Springer-Verlag, third edition, 2003.