

## 11. Rational functions and conversions

In this lecture we encounter for the first time the normalization problem and related this to the problem of “expression swell”. We also investigate the best way to evaluate expressions. For dense polynomials we can do not much better than converting to the Horner form, while for rational expressions, we save operations when we convert to continued fractions.

### 11.1 Rational Functions

Besides polynomials, Maple recognizes the type `ratpoly`:

```
[> p := x^3-1; q := x^2 - 1;
[> f := p/q;
[> whattype(p); type(f, ratpoly);
```

Note that Maple does not simplify automatically. We have to normalize with `normal`:

```
[> normal(f);
```

The normalization of rational expressions clears all common factors from the numerator and the denominator. There are many other representations of the same function. For instance, we can factor the numerator:

```
[> f = factor(numer(f))/denom(f);
```

or the denominator :

```
[> f = numer(f)/factor(denom(f));
```

or factor both :

```
[> f = factor(numer(f))/factor(denom(f));
```

Here the simplification happened automatically. Let us try to prevent the normalization:

```
[> nf := factor(numer(f)); df := factor(denom(f));
```

To prevent the cancellation of the common factor we can freeze the numerator:

```
[> ff := freeze(nf)/df;
[> thaw(numer(ff));
```

If you ask for a good reason why Maple does not normalize automatically, then here is an example:

```
[> fd := (x^1000-1)/(x-1);
```

Can you explain now why Maple does not normalize automatically?

Finding a normal form for a symbolic expression is very important to decide whether two expressions are the same, but sometimes a normal form may be computationally too expensive.

## 11.2 Conversions

For efficient evaluation of polynomials or rational expressions, we may wish to convert to the Horner form or to continued fractions respectively.

```
[> p := sum('a[i]*x^i', 'i'=0..5);
[> with(codegen):                # we need the function cost
[> cost(p);                       # cost for evaluation
[> hp := convert(p, 'horner');
[> cost(hp);
```

Observe though that the C code generation does not follow this Horner rule:

```
[> C(p, 'optimized');
```

For rational expressions, we convert to continued fractions:

```
[> q := sum('b[i]*x^i', 'i'=0..5);
[> pq := p/q;
[> cost(pq);
[> #cpq := convert(pq, 'confrac', x); # this did not return...
```

Let us try it with random coefficients :

```
[> for i from 0 to 5 do
[>   a[i] := rand(); b[i] := rand();
[> end do;
[> cpq := convert(pq, 'confrac', x);
[> cost(cpq);
```

With continued fractions, we have replaced multiplications by divisions. Since there are fewer divisions than the original number of multiplications, one would prefer the continued fraction form. However, observe the growth of the numbers...

Finally we compare with cost of using the Horner form on numerator and denominator:

```
[> hpq := convert(numer(pq), 'horner')/convert(denom(pq), 'horner');
[> cost(hpq);
```

We close with another type of conversion. To integrate rational functions, Maple converts to partial fractions:

```
[> convert(pq, 'parfrac', x, complex);
```

Observe that the choice of the number field (in this case complex) drastically changes the outcome.

### 11.3 Assignments

1. Consider the rational expression  $\frac{x^4+x^3-4x^2-4x}{x^4+x^3-x^2-x}$ .

Transform the expression into:

(a)  $\frac{(x+2)(x+1)(x-2)}{x^3+x^2-x-1}$

(b)  $\frac{x^4+x^3-4x^2-4x}{x(x-1)(x+1)^2}$

(c)  $\frac{(x+2)(x-2)}{(x-1)(x+1)}$

(d)  $\frac{x^2}{(x-1)(x+1)} - 4\frac{1}{(x-1)(x+1)}$

2. Consider the same rational expression as in the previous exercise.

- (a) Write it as a continued fraction.  
(b) Compute a partial fraction decomposition.