Making Computer Algebra More Symbolic

Stephen M. Watt

Abstract

This paper is a step to bring closer together two views of computing with mathematical objects: the view of "symbolic computation" and the view of "computer algebra." Symbolic computation may be seen as working with expression trees representing mathematical formulae and applying various rules to transform them. Computer algebra may be seen as developing constructive algorithms to compute algebraic quantities in various arithmetic domains, possibly involving indeterminates. Symbolic computation allows a wider range of expression, while computer algebra admits greater algorithmic precision. We examine the problem of providing polynomials symbolic exponents. We present a natural algebraic structure in which such polynomials may be defined and a notion of factorization under which these polynomials form a UFD.

1 Introduction

For the purposes of this paper, it is useful to make a distinction between "symbolic computation" and "computer algebra."

By "symbolic computation," we mean computation with expression trees, or "terms," representing mathematical objects. In these trees may appear symbols denoting operations, such as "+," "×" or "sin", numbers and variables. Computation consists of combining or transforming these trees to, for example, expand multiplications or multiple angle formulae. There are problems related to expression equivalence, simplification and computing canonical forms. A given expression might represent values belonging to various mathematical domains, depending on the interpretation. For example, is "I" an identity matrix, an indexed Bessel function, an imaginary unit, etc. Algebraic algorithms are typically well-defined on subclasses of the space of all expressions. Working with new classes of objects requires defining new operators and transformations on expressions containing them.

By "computer algebra," we mean computations using the arithmetic from particular algebraic constructions. The values used are elements of mathematically defined sets, such as polynomial rings, algebraic extensions, quotients and so on. The elements might be represented in any one of a number of ways. Certain algebraic domains, such as polynomials, may include indeterminates. Algorithms are defined over particular algebraic input domains and yield well-defined results. Working with new classes of objects requires defining new algebraic domains and determining their properties.

A useful problem solving environment should provide some way to do both symbolic computation and computer algebra, and there are different ways to do this. We may view, for example, Maple as being a symbolic computation system with computer algebra built as a layer on top. On the other hand, we may view Axiom as a computer algebra system, with symbolic computation provided as a top layer. In both cases we suffer because there is a gap between the symbolic and the algebraic semantics.

This gap is particularly evidenced when we work problems by hand. We do not hesitate to work with vectors of dimension n, or write polynomials of degree d with coefficients from some ring k of characteristic p. We then take facts about d, n, k or p into account when we do calculations. As important as it is, there is relatively weak support for this sort of computation in our symbolic mathematical software.

We are motivated to explore how to bring the symbolic and algebraic views closer together, providing a more robust conceptual framework and providing tools to address an important family of practical problems. We may view this as defining algebraic domains for wider classes of symbolic expressions. That is, we wish to work in algebraic domains that lie beyond the well-studied algebraic constructions of classical algebra. As a start, we examine the problem of working with polynomials with symbolic exponents.

2 Symbolic Polynomials

We wish to work with polynomials where the exponents are not known in advance, such as $x^{2n} - 1$. There are various operations we will want to be able to do, such as squaring the value to get $x^{4n} - 2x^{2n} + 1$, or differentiating it to get $2nx^{2n-1}$. This is far from a purely academic problem. Expressions of this sort arise frequently in practice, for example in the analysis of algorithms, and it is very difficult to work with them effectively in current computer algebra systems.

We may think of these values as sets of polynomials, one for each value of n, or we may think of them as single values belonging to some new ring. We wish to perform as many of the usual polynomial operations on these objects as possible. Many computer algebra systems will allow one to work with polynomials with symbolic exponents. They do this, however, either by falling back on some form of general expression manipulation or by treating all symbolic powers as algebraically independent. They thus miss many of the important properties we wish to reflect. The relationship between exponents may be non-trivial. We would like, for example, to compute factorizations such as

$$x^{n^4-6n^3+11n^2-6(n+2m-3)} - 1000000^m$$

$$\begin{aligned} &= x^{-12m} \times \left(x^{p_1} + 10^m x^{p_2+2m} + 10^{2m} x^{4m}\right) \times \left(x^{p_2} + 10^m x^{2m}\right) \\ &\quad \times \left(x^{p_1} - 10^m x^{p_2+2m} + 10^{2m} x^{4m}\right) \times \left(x^{p_2} - 10^m x^{2m}\right) \end{aligned}$$

$$\begin{aligned} p_1 &= x^{1/3n^4-2n^3+11/3n^2-2n+6} \\ p_2 &= x^{1/6n^4-n^3+11/6n^2-n+3} \end{aligned}$$

and perhaps operations on symbolic integers

$$16^n - 81^m = (2^n - 3^m)(2^n + 3^m)(2^{2n} + 3^{2m}).$$

We can imagine a number of models for symbolic polynomials that have these properties. Most generally, we could say that any set S, which under an evaluation map ϕ gives a polynomial ring $R[x_1, ..., x_v]$, represents symbolic polynomials. This would allow such forms as

$$\gcd(x^n - 1, x^m - 1) - x^{\mathrm{lcm}(n,m)} + 1$$

or

$$(x - 1) \sum_{i=0}^{n} x^i.$$

Having a more obvious ring structure will be useful to us, so we begin by generalizing to symbolic exponents only. First we recall the concept of a "group ring." A *monoid ring* is a ring formed from a ring R and monoid M with elements being the finite formal sums

$$\sum_i r_i m_i, r_i \in R, m_i \in M.$$

A monoid ring has a natural module structure, with basis M, and addition defined in terms of coefficient addition in R. Multiplication is defined to satisfy distributivity, with $r_1 m_1 \times r_2 m_2 = (r_1 r_2)(m_1 m_2)$. When the monoid M is a group, then the algebraic structure is called a *group ring*. For example, the Laurent polynomials with complex coefficients may be constructed as the group ring $\mathbb{C}[\mathbb{Z}]$, viewing $\mathbb{Z}$ as an additive group.

We now define a useful class of symbolic polynomials.

Definition 2.1. *The ring of* symbolic polynomials *in $x_1, ..., x_v$ with exponents in $n_1, ..., n_p$ over the coefficient ring R is the ring consisting of finite sums of the form*

$$\sum_i c_i x_1^{e_{i1}} x_2^{e_{i2}} \cdots x_n^{e_{in}}$$

where $c_i \in R$ and $e_{ij} \in \mathrm{Int}(\mathbb{Z})[n_1, n_2, ..., n_p]$. Multiplication is defined by

$$c_1 x_1^{e_{11}} \cdots x_n^{e_{1n}} \quad \times \quad c_2 x_1^{e_{21}} \cdots x_n^{e_{2n}} = c_1 c_2 x_1^{e_{11}+e_{21}} \cdots x_n^{e_{1n}+e_{2n}}$$

We denote this ring $R[n_1, ..., n_p; x_1, ..., x_v]$.

We make use of the notion of "integer-valued polynomials," $\mathrm{Int}(D)[n_1, ...n_p]$. For an integral domain D with quotient field K, univariate integer-valued polynomials may be defined as

$$\mathrm{Int}(D)[X] = \{f(X) \,|\, f(X) \in K[X] \text{ and } f(a) \in D, \text{ for all } a \in D\}$$

For example $\frac{1}{2}n^2 - \frac{1}{2}n \in \mathrm{Int}(\mathbb{Z})[n]$. Integer-valued polynomials have been studied by Ostrowski [2] and Pólya [3], and we take the obvious multivariate generalization.

Our definition of symbolic polynomials is isomorphic to the group ring $R[\mathrm{Int}(\mathbb{Z})[n_1, ..., n_p]^v]$. We view $\mathrm{Int}(\mathbb{Z})[n_1, ...n_p]$ as an abelian group under addition and use the identification

$$X_1{}^{e_1} X_2{}^{e_2} \cdots X_v{}^{e_v} \cong (e_1, \ldots, e_v) \in \mathrm{Int}(\mathbb{Z})[n_1, \ldots, n_w]^v$$

We note that $R[; x_1, ..., x_v] \cong R[x_1, ..., x_v]$. Under any evaluation $\phi : \{n_1, ..., n_p\} \to \mathbb{Z}$, we have

$$\phi : R[n_1, ..., n_p; x_1, ..., x_v] \to R[x_1, ..., x_v, x_1^{-1}, ..., x_v^{-1}].$$

That is, ϕ evaluates symbolic polynomials to Laurent polynomials. It would be possible to construct a model for symbolic polynomials that, under evaluation had no negative variable exponents, but this would require keeping track of cumbersome domain restrictions on the exponent variables.

By definition, these symbolic polynomials have a ring structure. What is more interesting is that they also have a useful unique factorization structure that can be computed effectively.

3 Factorization

We now show the multiplicative structure of our symbolic polynomials, and describe two algorithms for factorization. For simplicity we first show the case when $R = \mathbb{Z}$ and exponents are in $\mathbb{Z}[n_1, ..., n_p]$.

Proposition 3.1. *$\mathbb{Q}[n_1, ..., n_p; x_1, ..., x_v]$ is a UFD, with monomials being units.*

We sketch the proof: The fact that $x, x^n, x^{n^2}, ...$ are algebraically independent can be used to remove exponent variables inductively. We observe that

$$x_k^{e_{ik}} = x_k^{\sum_j h_{ij} n_1^j} = \prod_j \left(x_k^{n_1{}^j}\right)^{h_{ij}} = \prod_j x_{kj}{}^{h_{ij}}, \qquad h_{ij} \in \mathbb{Z}[n_2, ..., n_p].$$

This gives the isomorphism

$$\mathbb{Z}[n_1, n_2, ..., n_p; x_1, ...x_v] \cong \mathbb{Z}[n_2, ..., n_p; x_{10}, x_{11}, x_{12}, ...x_{1d_1}, ...x_{v0}, x_{v1}, x_{v2}, ...x_{vd_1}]$$

where d_1 is the maximum degree of n_1 in any exponent polynomial and x_{ij} corresponds to $x_i^{n_1{}^j}$. Once all the exponent variables have been removed, we factor in $\mathbb{Z}[x_{10...0}, ..., x_{vd_1...d_1}]$, then reform the exponent polynomials of $x_1, ..., x_v$.

When the exponents come from the integer-valued polynomials $\mathrm{Int}(\mathbb{Z})[n_1, ..., n_p]$, as opposed to $\mathbb{Z}[n_1, ...n_p]$, care must be taken to find the fixed divisors of the exponent polynomials. For example, the fact that $n(n-1)$ is always even implies the factorization

$$x^2 - y^{n^2-n} = (x - y^{n(n-1)/2})(x + y^{n(n-1)/2})$$

Fixed divisors are given by the content when polynomials are written in a binomial basis. That is, for each exponent variable n_i, we use the polynomial basis 1, $\binom{n_1}{1}$, $\binom{n_2}{1}$, ..., $\binom{n_1}{2}$, $\binom{n_1}{1}\binom{n_2}{1}$, $\binom{n_2}{2}$, ..., *etc.* Combining these two ideas, we make the change of variables to $x_k^{\binom{n_1}{i_1}\cdots\binom{n_p}{i_p}}$, to obtain factorization in $\mathbb{Q}[n_1, ...n_p; x_1, ..., x_v]$. The same strategy may, of course, be used to compute greatest common divisors, square-free factorizations and similar quantities.

We have described this transformation as though the exponent polynomials were dense. In this worst case, the number of new variables will be D^p, where D is the degree bound on the n_i. In practice, the number of variables occurring in exponents will be small and the exponent polynomials will be of low degree so the introduction of new variables may be acceptable. In many cases, most of the new variables will occur only trivially. Blindly changing to a factorial basis to make fixed divisors manifest may not, however, be the best strategy. This destroys any sparseness in the input polynomials. A better strategy would be to convert only when necessary.

If the number of exponent variables is large, then another method may be used to manage the complexity of many variables. We may use projections to map the exponents to integers at several points, and combine them via interpolation. Naively, an exponential number of factorizations in $\mathbb{Q}[x_1, ..., x_v]$ will be needed, but this not always necessary. If there are may small factors, then there is a combinatoric problem of factor identification. If the coefficient field is large enough, and the polynomials are not special, then coefficient values may be used to greatly limit the search. We have experimental implementations of both the "change of variables" method and the "projection" method, but it is too early to say which method will be most useful in practice.

4 Generalizations

As mentioned earlier, we may contemplate other algebraic structures to encompass a wider class of expressions. Without going to the most general model of polynomial-valued integer functions, we may consider

- Allowing exponent variables to also appear as regular variables. To do this we can work in $R[n_1, ..., n_p; n_1, ..., n_p, x_1, ..., x_v]$. This is useful if we require formal derivatives.
- Symbolic exponents on coefficients. We discuss this case more below.
- Symbolic polynomials as exponents, or richer structures.
- Other polynomial forms, such as exponential polynomials, *e.g.* [1] [4].
- Other problems, *e.g.* Gröbner bases of symbolic polynomials [5].

Let us examine more closely the question of symbolic exponents on coefficients. Suppose we wish to factor a polynomial of the form $x^{4m} - 2^{4n}$. Assuming m and n may take on only integer values, the factorization over $\mathbb{Q}$ is $(x^{2m} + 2^{2n})(x^m + 2^n)(x^m - 2^n)$. This, however is equivalent to $x^{4m} - 16^n$, which is not manifestly the difference of fourth powers. So how can we approach symbolic integer coefficients?

If the coefficient ring is a principal ideal domain, then we may extend our definition to allow symbolic exponents on prime coefficient factors:

Definition 4.1. *The ring of* symbolic polynomials *with exponents in $n_1, n_2, ..., n_p$ over the coefficient ring R, a PID with quotient field K, is the ring consisting of finite sums of the form*

$$\sum_i k_i \cdot \prod_j c_j^{d_{ij}} \cdot x_1^{e_{i1}} x_2^{e_{i2}} \cdots x_n^{e_{in}}$$

where each product has a finite number of nonzero d_{ij}, $k_i \in K$, c_j are primes $\in R$, $d_{ij} \in \mathrm{Int}(\mathbb{Z})[n_1, n_2, ..., n_p] \backslash \mathbb{Z}$ and $e_{ij} \in \mathrm{Int}(\mathbb{Z})[n_1, n_2, ..., n_p]$. Multiplication is defined by

$$k_1 c_1^{d_{11}} \cdots c_m^{d_{1m}} x_1^{e_{11}} \cdots x_n^{e_{1n}} \quad \times \quad k_2 c_1^{d_{21}} \cdots c_m^{d_{2m}} x_1^{e_{21}} \cdots x_n^{e_{2n}} = k_1 k_2 c_1^{d_{11}+d_{21}} \cdots c_m^{d_{1m}+d_{2m}} x_1^{e_{11}+e_{21}} \cdots x_n^{e_{1n}+e_{2n}}$$

Let us consider the case of integer coefficients. We note that, for any base, any set of logarithms of distinct primes is linearly independent over $\mathbb{Q}$. This is easily seen, for the equation $\sum_i n_i \log(p_i) = 0$, holds with p_i distinct primes and $n_i \in \mathbb{Z}$, only if $\prod_i p_i^{n_i} = 1$, which requires $n_i = 0$. This implies that

$$\sum_i \alpha_i \log p_i \neq 0$$

for any non-zero algebraic numbers α_i. We can write any product of integers to symbolic powers as an exponential of a linear combination of logarithms of primes, *e.g.*

$$6^m \times 7^{n^2+1} = \exp(m \log 2 + m \log 3 + (n^2+1) \log 7)$$

where exp and log use the same base. We can therefore treat 2^n, $2^{\binom{n}{2}}$, ... as new variables for factoring, etc.

As stated, this approach would require factoring each integer that appears with a symbolic exponent. In practice we do not want to factor the constant coefficients. Instead, we can form, for any particular problem, an easier to compute basis, *e.g.* from $\{70^n, 105^n\}$ the set $\{2^n, 3^n, 35^n\}$ which does not require factoring of 35. This can be done using only integer gcd's and extracting integer roots.

5 Conclusions

We see a mathematically rich and practically important middle ground between the usual approaches of "symbolic computation" and "computer algebra." Rather than working with loosely defined expressions, or strictly with classical polynomial and matrix algebras, there is room to work in other well-defined algebraic contexts. These can provide the structure to make operations well-defined, while at the same time allowing more symbolic treatment in mathematical computations. In this light, we have explored how to usefully work with symbolic polynomials — polynomial-like objects where the exponents can themselves be

polynomials. These are able to represent the kinds of symbolic polynomials we have seen in practice. The algebraic structure allows us to perform arithmetic on these objects, to simplify and transform them. We find, moreover, a UFD structure that admits algorithms for factorization, gcd, *etc.* This encourages us to look at more algebraic treatment of other symbolic structures, such as matrices of unspecified size.

References

[1] de Prony, Baron Gaspard Riche. Essai éxperimental et analytique: sur les lois de la dilatabilité de fluides élastique et sur celles de la force expansive de la vapeur de l'alkool, à différentes températures. Journal de l'École Polytechnique, volume 1, cahier 22, 24-76 (1795).

[2] Ostrowski, A., Über ganzwertige Polynome in algebraischen Zahlköpern, J. Reine Angew. Math., 149 (1919), 117-124.

[3] Pólya, G., Über ganzwertige Polynome in algebraischen Zahlköpern, J. Reine Angew. Math., 149 (1919), 97-116.

[4] C.W. Henson, L. Rubel, and M. Singer, Algebraic Properties of the Ring of General Exponential Polynomials. Complex Variables Theaory and Applications, **13**, 1989, 1-20.

[5] Kazuhiro Yokoyama. On Systems of Algebraic Equations with Parametric Exponents. pp 312-319, ISSAC '04, July 4-7, 2004, Santander, Spain, ACM Press.

Stephen M. Watt
Ontario Research Centre for Computer Algebra
University of Western Ontario, MC 375
London ON, Canada N6A 5B7

watt@orcca.on.ca
http://www.orcca.on.ca/~watt

Errata *11 June 2006, 8 July 2006*

- Page 4, lines 1-3: $\mathbb{Z}$ changed to read Int($\mathbb{Z}$).
- Page 5, lines 2–4; Page 6, line 18: Use of notation $n^{\underline{j}}$, inconsistent with Knuth *et al*, changed to $\binom{n}{j}$.