date

ALGEBRAIC TRANSFORMATIONS OF POLYNOMIAL EQUATIONS, SYMMETRIC POLYNOMIALS AND ELIMINATION

Marc Giusti *
Centre de Mathématiques
Ecole Polytechnique
91128 Palaiseau Cedex
Unité associée au CNRS No. 169
and
GRECO de Calcul Formel No. 60
UUCP: ... mcvax!inria!cmep!giusti

Daniel Lazard *
LITP (tour 45-55)
4, Place Jussieu
75252 Paris Cedex 05
Unité associée au CNRS No. 248
and
GRECO de Calcul Formel No. 60
UUCP: ... mcvax!inria!litp!dl

Annick Valibouze
LITP (tour 45-55)
4, Place Jussieu
75252 Paris Cedex 05
Unité associée au CNRS No. 248
and
GRECO de Calcul Formel No. 60
UUCP: ... mcvax!inria!litp!avb

May 28, 1993

INTRODUCTION

Let k be a field, and K an algebraic closure. To every univariate polynomial f with coefficients in k is associated the algebra homomorphism f^* "inverse image" (i.e. $f^*(P) = P \circ f$, which is in this particular case the polynomial whose roots are the inverse images by f of the roots of P in K).

As usual defining a "direct image" is more difficult, but examples occur frequently in Mathematics :

(i) In the Graeffe method giving the real roots of a polynomial P, a basic step consists in computing the polynomial whose roots are the square of the roots of the given polynomial P. This is one of the simplest example of transforming a polynomial by an algebraic morphism.

(ii) Given a monic univariate polynomial P, we consider the polynomial whose roots are all the differences of two roots of P in K. Its constant term is the discriminant of P.

(iii) Given two monic univariate polynomial P and Q, we form the polynomial whose roots are the all the differences of a root of P and a root of Q in K. Its constant term is nothing else than the resultant of P and Q.

*Partially supported by a grant from the PRC "Mathématiques et Informatique"

We shall define below a general transformation of polynomials, and study the following concrete problem : how to perform such a transformation using a standard system of Computer Algebra, providing the usual algebraic tools.
Note that in the first example described above, a first way is to compute the resultant $R(y) = Res_x(P(x), y - x^2)$. A second method is to remark that the coefficients of the required polynomial are symmetric functions of the roots of P, hence can be expressed in terms of the elementary symmetric functions which are known as the coefficients of P up to the sign.
As indicated by the previous remarks, we will show that the classical deep relationships between the problems :

($\mathcal{T}$) transforming polynomial equations by an algebraic morphism

($\mathcal{R}$) elementary elimination theory by resultants

($\mathcal{S}$) change of bases for symmetric polynomials.

can be illustrated in Computer Algebra. Actually they are algorithmically equivalent because every one is a particular case of another one, as shown by the following diagram :

$$\begin{tikzcd}
 & \mathcal{R} \arrow[dl] & \\
\mathcal{T} \arrow[rr] & & \mathcal{S} \arrow[ul]
\end{tikzcd}$$

where an arrow linking two problems means that one can describe an algorithm solving the second one if such holds for the first one.

This paper is an extended abstract presenting some results of a paper in preparation. We refer to the preliminary version [GLV] for details.

1 The problem ($\mathcal{T}$) of transforming equations by a morphism

1.1 Notations

Let k be a field, and K an algebraic closure of k. We will denote by $\mathbf{N}$ the set of natural (i.e. positive) integers.

Given an integer p, consider an element $I = (i_1, i_2, \ldots, i_p)$ of $\mathbf{N}^p$. Its *weight* $\mid I \mid$ is the sum $i_1 + \cdots + i_p$. Let R_I be the polynomial algebra $k[x_1^{(1)}, \ldots, x_{i_1}^{(1)}, \ldots, x_1^{(p)}, \ldots, x_{i_p}^{(p)}]$. A *transformation of type* I is nothing else than a polynomial f of R_I, to which is associated a mapping :

$$f : K^{i_1} \times K^{i_2} \times \ldots \times K^{i_p} \longrightarrow K$$

which will be also denoted by f.

The set $\mathbf{N}^p$ is naturally partially ordered. If J is larger than I, it gives rise to a natural inclusion $R_I \subseteq R_J$.

The *multidegree* D of a p-uple of univariate polynomials $(P_1, \ldots, P_p)$ is the sequence $(d_1, \ldots, d_p)$ of their degrees.

1.2 Definition

Now we can define the *direct image* or *transformation* :

$$f_* : k[x]^p \longrightarrow k[x]$$

which associates to any ordered p-uple of monic polynomials $P = (P_1, \ldots, P_p)$, of multidegree D larger than I, the monic polynomial $f_*(P)$ obtained in the following way : informally speaking, it is the univariate polynomial whose roots are the elements of K obtained by substituting in f the variables $x_i^{(j)}$'s by the different roots of P_j. We give now a precise definition of $f_*(P)$:

Since the multidegree D is larger than I, we can associate to every polynomial P_j of degree d_j the set $r(P_j) = (a_1^{(j)}, \ldots, a_{d_j}^{(j)})$ of its d_j roots in K ordered in an arbitrary way $(1 \leq j \leq p)$. Choose an evaluation map $E_a : R_D \longrightarrow K$ which is the algebra homomorphism which sends the variable $x_i^{(j)}$ to $a_i^{(j)}$. The product S_D of symmetric groups $S_{d_1} \times \cdots \times S_{d_p}$ acts naturally on R_D : by the natural inclusion $R_I \subseteq R_D$, f becomes an element of R_D ; let $O_{S_D}(f)$ be its orbit under S_D.

Eventually $f_*(P)$ is the polynomial whose roots are the images by the evaluation map E_a of the elements of the orbit of f, i.e. :

$$f_*(P)(x) = \prod_{g \in O_{S_D}(f)} (x - E_a(g))$$

1.3 Properties

By construction, the definition of $f_*(P)$ does not depend on the choice of the evaluation map. The proof is straightforward.

Note also that $f_*(P)$ has a priori its coefficients in the closure K, but it is easy to see that they are actually in k.

2 Representation and manipulation of the algebra of symmetric polynomials

A standard computer algebra system must provide the usual algebraic tools allowing to handle with a polynomial algebra. This implies on the first hand a data structure and an internal representation, on the second hand algorithms and the corresponding implementations of the algebra operations.

In applications of computer algebra, we frequently have to manipulate objects which are invariant under some group of permutation of the indeterminates, for example symmetric polynomials. But we are quickly stucked in a space problem if we use the standard previous tools, since the symmetric group S_n contains $n!$ permutations. So we need to represent this algebra of invariants in the most contracted way : we don't want to list the elements of an invariant subset, but only a fundamental domain. For example, a monomial form is the sum of the elements of the orbit of a single monomial, and we need only to store an element and the operation "+". Note immediately that we have to pay something since we assume implicitely that we are able to compute the isotropy subgroup of a monomial ...

To complete this data structure, we need two functions inverse of each other, one of contraction and the other one of explosion, passing from the usual representation of polynomials in the system to the contracted one and conversely.

Furthermore algorithms for the internal operations in the algebra must be available, with the corresponding implementations. Eventually if the algebra admits several bases as algebra or vector spaces, we need the corresponding algorithms allowing to expand on a new basis a symmetric polynomial given on an old one.

All these goals are attained in the work of A. VALIBOUZE, and implemented in the extension SYM to MACSYMA [V1] [V2].

3 Equivalence of the three problems

3.1 $(\mathcal{R}) \Longrightarrow (\mathcal{T})$

3.1.1 Relationship with elimination

Let us consider the affine algebraic subvariety V of $K^{|I|+1}$ defined by the $\mid I \mid +1$ equations :

$$\begin{aligned} P_1(x_1^{(1)}) &= \cdots = P_1(x_{i_1}^{(1)}) = 0 \\ &\ldots \\ P_p(x_1^{(p)}) &= \cdots = P_p(x_{i_p}^{(p)}) = 0 \\ y &= f(x_1^{(1)}, \ldots, x_{i_1}^{(1)}, \ldots, x_1^{(p)}, \ldots, x_{i_p}^{(p)}) \end{aligned}$$

Let y_0 be a root of $f_*(P)$; then there exists common roots of the previous equations in the variables $x_i^{(j)}$ if we set y to y_0.
Hence y_0 belongs to the projection of V on the y-axis. Thus solving the problem of transforming equations is part of an *elimination* problem of the variables $x_i^{(j)}$ among the previous equations.

The variety V is obviously a complete intersection of dimension 0 and degree $d_1^{i_1} \ldots d_p^{i_p}$. The projection of V on the y-axis is again of same dimension, hence is defined set-theoretically by a univariate polynomial with well-defined roots, whose multiplicity varies unfortunately with the various definitions of elimination.

Here it is one of the seldom cases where the elimination can be done by the use of two by two resultants, since every variable $x_i^{(j)}$ occurs only in two equations. We have the following theorem :

3.1.2 Theorem :

Let $R(y)$ be the polynomial obtained by eliminating two by two the variables $x_i^{(j)}$ among the equations defining V. If $G_{S_I}(f)$ is the isotropy subgroup fixing f, the polynomial $f_*(P)^{\# G_{S_I}}$ divides R.

Note that the degree of this factor F is the degree of $f_*(P)$ times the cardinal of $G_{S_I}(f)$, i.e. the cardinal of S_D divided by the cardinal of $G_{D \setminus I}$, i.e. eventually $\prod_{r=1}^{p}(d_r - i_r + 1)$ to be compared to the degree of $R(y)$, which is $\prod_{r=1}^{p} d_r^{i_r}$.

Now to conclude it is enough to compute a squarefree decomposition of R, and to throw away the other factors, which can be assumed known by induction. Actually if we consider the collapsed transformations obtained by equaling two or more variables corresponding to the same component of I, the parasite factors are some power of such transformations of P.

3.2 $(\mathcal{T}) \Longrightarrow (\mathcal{S})$

It will be sufficient to show that we can compute change to and from a particular basis, i.e. the symmetric elementary polynomials.

Given $X = (x_1, \ldots, x_n)$. we want to compute a monomial form $M_I(X)$ (i.e. the sum of the monomials of the orbit of X^I under the action of S_n) as a function of the elementary symmetric functions $e_1, e_2, \ldots, e_n$ of the x_i's using $(\mathcal{T})$. Let us call P the polynomial $x^n + \sum_{i=1}^{n}(-1)^i e_i x^{n-i}$. If we choose $p = 1$ and f of type $lg(I) = l$ as follows :

$$f(x_1, x_2, \ldots, x_l) = x_1^{i_1} x_2^{i_2} \ldots x_l^{i_l}$$

the opposite of the coefficient of y^{n-1} in $f_*(P)(y)$ is exactly M_I. This is an immediate consequence of the definition of $f_*(P)$.

3.3 $(\mathcal{S}) \Longrightarrow (\mathcal{R})$

Let P and Q be two univariate monic polynomials of degree respectively p and q, with coefficients in k. Call $x_1, \ldots, x_p$ the roots of P in K. Then the resultant R of P and Q is the product $Q(x_1) \ldots Q(x_p)$, which is a symmetric function of $Q(x_1), \ldots, Q(x_p)$, so can be expressed with the symmetric elementary functions of the x_i's, i.e. the coefficients of P up to the sign.

This transformation has been implemented in `SYM`. This gives an algorithm for the resultant wich is experimentally much better than the subresultant method as implemented in `MACSYMA`, when the coefficients of the polynomial depend on many parameters : for generic polynomial of degrees 3 and 6, this algorithm is 4 times faster than the subresultant algorithm. More generally, the computation of the resultant of generic polynomials of degrees 3 and n needs about 2^n, to be compared to about 5^n by the subresultant algorithm. For higher degree, not enough memory was available.

4 Example

To illustrate different ways to solve $\mathcal{T}$, it may be useful to consider the following example proposed by P.Cartier, which led us to this theory. Consider the polynomial $P(x) = x^7 - 7x + 3$. How to compute the degree 35 polynomial whose roots are the sums of 3 distinct roots of P ?

4.1 Solving the problem through resultants

Here we may take $f(x, y, z) = x + y + z$. We want the direct image $f_*(P)$. Theoretically the problem may be solved by computing

```
Q1 :  Resultant(P(x), x + y + z - u, x)
Q2 :  Resultant(P(y), Q1, y)
Q3 :  Resultant(P(z), Q2, z)
Q : sqfr(Q3)
```

But pratically this fails for lack of memory.

Here is a more efficient program in `MACSYMA` wich gives the result in 1500 seconds (including 700 seconds of garbage collection) on a VAX 780 :

```
(c2) sum3(pol):=
block([a,b,c,d,sol],
a : resultant(pol,ev(pol,x=y-x),x),
b : rat(a/ev(pol,x=y/2),y),
c : sqfr(b),
```

```
d : part(c,2,1),
sol : resultant(ev(pol,x=z-y),d,y),
part(sqfr(sol),1,1,1));
```

4.2 A direct solution

Theoretically, the problem may be solved by use of symmetric functions, computing the above resultants by algorithm 3.3.. But, in this special case, there is a direct way.
The coefficients of the wanted polynomial are symmetric functions of its roots. These can be expressed as sum of the k-th power of roots, $p_1, p_2, \ldots, p_{35}$. But $\sum(x_{i_1}x_{i_2}x_{i_3})^k$ is easily expressed as a symmetric function of $x_1, x_2, \ldots, x_7$ with the multinomial formula and thus as a function of the coefficients of P. All these computations are done by Valibouze's system `SYM` in 2 mn 30 s on the same VAX 780.

Thus the problem may be solved in two different ways : the computation through the symmetric functions is the most efficient one.

REFERENCES

[GLV] M. GIUSTI, D. LAZARD, A. VALIBOUZE, Symmetric polynomials and elimination, Notes informelles de Calcul Formel IX, Prépublication du Centre de Mathématiques de l'Ecole Polytechnique, M810.0987, 1988.

[V1] A. VALIBOUZE, Fonctions Symétriques et changements de bases, to be published in Proceedings of the European Conference on Computer Algebra EUROCAL 87 (Leipzig, RDA), 1987.

[V2] A. VALIBOUZE, Manipulations de fonctions symétriques, Thèse de l'Université Paris VI, 1988.