*J. Symbolic Computation* (1990) **10**, 39–51

# Complexity of Computation on Real Algebraic Numbers

MARIE-FRANÇOISE ROY† AND AVIVA SZPIRGLAS‡

† *Institut de Recherche Mathématique de Rennes, Université de Rennes I, Campus de Beaulieu, F-35042 Rennes Cedex, France*

‡ *CSP, Université Paris Nord, F-93430 Villetaneuse, France*

(*Received* 14 *January* 1988)

This paper is devoted to a precise algorithmical and complexity study of a new polynomial time method for formal computations with polynomial inequalities and real algebraic numbers.

## 1. Introduction

A new method for coding the real algebraic numbers, based on the use of Thom's lemma and the study of simultaneous inequalities from Ben-Or, Kozen & Reif (1986), has been introduced by Coste & Roy (1988). This leads to various applications in the field of computational real algebraic geometry: study of the topology of a real algebraic curve (Roy, 1987), or of the analytic branches of a real algebraic curve (Cucker *et al.*, 1987).

In this paper, we give improved versions of the algorithms in Coste & Roy (1988) and we study their complexity.

In the second section we introduce some basic tools, based on the techniques of computer algebra (mainly results on subresultants). In the third section, we study simultaneous inequalities at the real roots of a polynomial and in the fourth section, we consider the coding of real algebraic numbers.

## 2. Basic Tools and Notations

In the paper, for $P = a_0X^p + \ldots + a_p$, a polynomial with integer coefficients, we define the norm of $P$, $N(P)$, by $N(P) = (a_0 + \ldots + a_p^2)^{1/2}$. The size of $P$ is the log of $N(P)$. The length of an integer is the log of the integer. The degree of $P$ is denoted by $\deg(P)$.

Our algorithms are based on a generalization of Sturm theorem, hence on divisions of polynomials, taking care of signs. We therefore require various notations for the signed remainders.

In this section $P$ denotes a polynomial with integer coefficients of degree $p$ and $Q$ denotes a polynomial with integer coefficients of degree $q$ with leading coefficient $b_0 \neq 0$, $q \leq p$.

### 2.1. SIGNED PSEUDO-REMAINDERS

We denote by $rem(P, Q)$, the remainder of $P$ and $Q$ in the Euclidian division process: so $rem(P, Q)$ has rational coefficients.

0747–7171/90/070039+13 $03.00/0

In order to perform division of polynomials with only integer coefficients, one uses classically the pseudo-division process.

The pseudo-remainder of $P$ and $Q$ $prem(P, Q)$ is the remainder of the quotient of $b_0^{p-q+1}P$ by $Q$.

The signed pseudo-remainder of $P$ and $Q$ $sprem(P, Q)$ is the remainder of the quotient of $|b_0^{p-q+1}|\ P$ by $Q$.

The pseudo-remainder sequence of $P$ and $Q$ is by definition the following sequence:

$$prem_0(P, Q) = P$$
$$prem_1(P, Q) = prem(Q, P)$$
$$prem_j(P, Q) = prem(prem_{j-2}(P, Q), prem_{j-1}(P, Q)) \text{ for } j \leq j_0$$

where $j_0$ is such that $prem_{j0}(P, Q)$ is of degree 0 (so $j_0 \leq p-2$).

The signed pseudo-remainder sequence $spremj(P, Q)$ is obtained by replacing in the preceding definition each pseudo-remainder by the corresponding signed pseudo-remainder.

## 2.2 SIGNED SUBRESULTANT SEQUENCE

One problem with the pseudo-division process is the growth of the size of the coefficients. It is well known (see for example Loos, 1982) that the subresultants techniques allow integer computations and a good control of the size of intermediate results.

Let us recall the definition of the subresultants. We shall always consider here a polynomial $P$ of degree $p$, and a polynomial $Q$ of degree $q < p$.

For $0 \leq k < q < p$, the $k$th subresultant of $P$ and $Q$, $S_k$ is defined as follows: denote the $(p+q-2k, p+q-k)$-matrix by $M_k$ whose lines are

$$(x^{q-k-1}P, \ldots, P, x^{p-k-1}Q, \ldots, Q)$$

expressed in the basis $x^{p+q-k-1}, \ldots, x, 1$. Then the $k$th subresultant of $P$ and $Q$, $S_k$ is

$$\sum \det M_{k,j} x^{p+q-k-j},$$

where $M_{k,j}$ is the minor of $M_k$ obtained by taking the first $p+q-2k-1$ columns and the $j$th column (so $j$ varies from $p+q-2k$ to $p+q-k$).

The subresultant sequence of $P$ and $Q$ is obtained by considering $Q$ as a polynomial of degree $p-1$ (the first $p-1-q$ coefficients being equal to 0) and by taking for $S_p$ the polynomial $P$, for $S_{p-1}$ the polynomial $Q$ and for $S_k$ the $k$th subresultant of $P$ and $Q$ $(0 \leq k \leq p-2)$.

Let us recall the subresultant theorem, which indicates the relation between the subresultant sequence and the pseudo-remainder-sequence:

SUBRESULTANT THEOREM: If the degree of $S_{j+1}$ is equal to $j+1$ and the degree of $S_j$ is equal to $s$, then:

(*i*) $S_{j-1} = \ldots = S_{s+1} = 0$ for $-1 \leq s < j < p-1$

(*ii*) $(R_{j+1})^{j-s}S_s = lc(S_j)^{j-s}S_j$ for $0 \leq s \leq j \leq p-1$

(*iii*) $(-1)^{j-s}(R_{j+1})^{j-s+2}S_{s-1} = prem(S_{j+1}, S_j)$

where $R_{j+1}$ is the leading coefficient of $S_{j+1}$ except for $j = p-1$ $(R_p = 1)$ and $lc(S_j)$ the leading coefficient of $S_j$. For the proof see Loos (1982).

We deduce immediately the following corollary for signed pseudo-remainders.

COROLLARY. *For each* $j$, $j_0 \leqslant j \leq p-2$, *there exists* $i_j$ $0 \leq i_j \leq p$ *and* $C_j \in \mathbb{Z}$ *such that* $C_j S_{ij} = sprem_j(P, Q)$.

With the notations of the corollary the signed subresultant sequence is by definition,

$$ssub_j(P, Q) = \varepsilon_j S_{ij} \text{ for } j_0 \leq j \leq p \text{ where } \varepsilon_j \text{ is the sign of } C_j.$$

### 2.3. GENERALIZED SUBRESULTANT STURM SEQUENCE

The generalized Sturm sequence associated to $P$ and $Q$ is defined by:

$P_0 = P$
$P_1 = rem(P'Q, P)$
$P_j = -rem_j(P, P_1)$.

One denotes $v_{P,Q}(-\infty)$ and $v_{P,Q}(+\infty)$ as the number of sign changes in the sequences of the signs of the $P_i$ at $-\infty$ and $+\infty$, and $v(P, Q)$ as the number $v_{P,P'Q}(-\infty) - v_{P,P'Q}(+\infty)$.

The Sturm sequence of $P$ is the generalised Sturm sequence associated to $P$ and 1.

We have the following property, due to Sylvester (1853).

PROPOSITION 2.4. *Denote by* $c_{>0}(P; Q)$ $(c_{<0}(P; Q))$ *the number of real roots* $\xi$ *of* $P$ *such that* $Q(\xi)$ *is* $>0$ $(<0)$. *With the above definitions and notations one has* $c_{>0}(P; Q) - c_{<0}(P; Q) = v(P, Q)$.

PROOF. It is similar to the classical proof of Sturm theorem (see for example Jacobson, 1974): examine the sign variations in the generalized Sturm sequence when passing through a root of $P$.

In order to get only integer computations with a good control on their size we define the generalized subresultant Sturm sequence associated to $P$ and $Q$, denoted by GSS($P$, $Q$) by:

$P_0 = P$
$P_1 = sprem(P'Q, P)$
$P_j = -ssub_j(P, P_1)$.

It is clear from the definition of the signed subresultant sequence that the number of sign changes in the sequences of the signs of the $P_i$ at $-\infty$ and $+\infty$ are equal to $v_{P,Q}(-\infty)$ and $v_{P,Q}(+\infty)$.

The subresultant Sturm sequence of $P$ is the generalized subresultant Sturm sequence associated to $P$ and 1.

REMARK 2.5. The computation of the generalized subresultant Sturm sequence associated to $P$ and $Q$ takes $O(pq)$ arithmetic operations.

By Hadamard inequality (Mignotte, 1982) the coefficients of all the subresultants of $P$ and $Q$ are bounded by $N(P)^q N(Q)^p$, hence the size of the coefficients of the polynomials in the generalized subresultant Sturm sequence associated to $P$ and $Q$ is in $O((p+q)\log(N(P)) + p\log(N(Q)))$.

## 3. Simultaneous Inequalities

All algorithms are based on proposition 2.4, a result due to Sylvester (1853) revisited by Ben-Or, Kozen & Reif (1986).

Let $P, Q_1, \ldots, Q_k$ be polynomials with integer coefficients and let $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_k)$ be a sequence of sign conditions ($>0$, $<0$, $=0$). The subject of this section is the study of the number of real roots of $P$ where $\varepsilon$ are the signs of $Q_1, \ldots, Q_k$.

One denotes by $c_\varepsilon(P; Q_1, \ldots, Q_k)$, the number of real roots of $P$ where $Q_1, \ldots, Q_k$ take the signs $\varepsilon_1, \ldots, \varepsilon_k$. The sign conditions realized by $Q_1, \ldots, Q_k$ at the real roots of $P$ are the $k$-multiples of signs conditions $\varepsilon$ such that

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } Q_j(x)\varepsilon_j, j = 1, \ldots, k\}$$

is non-empty.

### 3.1. Algorithm SI

In this section, $P$ will denote a polynomial with integer coefficients, of degree $p$; $r$ will denote the number of real roots of $P$, $r \leq p$; $Q_1, \ldots, Q_k$ denote polynomials with integer coefficients.

Algorithm SI is an improved version of algorithm $b_3$ and $b_4$ of Coste & Roy (1988).

Presentation of SI. The aim of the algorithm is to determine the number of real roots of $P$ giving to $Q_1, \ldots, Q_k$ fixed signs.

Let us consider the case $k = 1$. Let $Q$ be a polynomial with integer coefficients, prime to $P$. We want to compute $c_{>0}(P; Q)$ and $c_{<0}(P; Q)$. Using proposition 2.4, we have

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} c_{=0}(P, Q) \\ c_{>0}(P, Q) \\ c_{<0}(P, Q) \end{bmatrix} = \begin{bmatrix} v(P, 1) \\ v(P, Q^2) \\ v(P, Q) \end{bmatrix}$$

We can easily deduce $c_{=0}(P; Q)$, $c_{>0}(P; Q)$ and $c_{<0}(P; Q)$ from $v(P, 1)$, $v(P, Q^2)$ and $v(P, Q)$.

The idea for $k \neq 1$ is to compute the generalized subresultant Sturm sequences associated to the polynomials of the form $Q_1^{f(1)} \ldots Q_k^{f(k)}$, with $f(i) \in \{0, 1, 2\}$ and use the sign variations of these generalized subresultant Sturm sequences and some linear algebra (generalizing $k = 1$) to deduce the values of the $c_\varepsilon(P; Q_1, \ldots, Q_k)$. If done without care, this leads to an exponential algorithm, since we have $3^k$ subsets polynomials, hence $3^k$ generalized Sturm sequences to compute, to get the values $c_\varepsilon$ for the $3^k$ sign conditions $\varepsilon$. To avoid this exponential growth the idea (Ben-Or, Kozen & Reif, 1986) is to remark that the number $r(k)$ of distinct sign conditions realised by $Q_1, \ldots, Q_k$ at the real roots of $P$ is, for any $k$, smaller than the number $r$ of real roots of $P$.

We shall give a description of the algorithm and then an example to illustrate the situation. Precisely one wants to determine the sign conditions $\varepsilon(k, 1), \ldots, \varepsilon(k, r(k))$, with $r(k) \leq r$, realized by $Q_1, \ldots, Q_k$ at the real roots of $P$ and the number $c(k, j) = c_{\varepsilon(k,j)}(P; Q_1, \ldots, Q_k)$ of elements of the non-empty set

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } Q_i(x)\varepsilon(k, j)_i, i = 1, \ldots, k\}.$$

The input of the algorithm consists of $P, Q_1, \ldots, Q_k$.

The output SIout$(P, Q_1, \ldots, Q_k)$ of the algorithm is the following:

$(1_k)$ the list $\varepsilon(k) = (\varepsilon(k, 1), \ldots, \varepsilon(k, r(k))$, $r(k) \leq r$ of the $k$-multiples of sign conditions realized by $Q_1, \ldots, Q_k$ at the real roots of $P$,

$(2_k)$ the numbers $c(k, 1), \ldots, c(k, r(k))$,

$(3_k)$ a list of polynomials $(Q_{k,j})_{j=1,\ldots,r(k)}$ which are some product of the $Q_m$s and the $Q_m^2$s and the numbers $v(k, j) = v(P, Q_{k,j})$,

$(4_k)$ an invertible matrix $A(P; Q_1, \ldots, Q_k) = A_k$ of dimension $r(k)$ such that $A_k c_k = S_k$, where $c_k$ is the vector $(c(k, 1), \ldots, c(k, r(k))$ and $V_k$ the vector $(v(k, 1), \ldots, v(k, r(k))$.

DESCRIPTION OF SI. We are going to compute by induction on $l$, $1 \leq l < k$ SIout$(P, Q_1, \ldots, Q_l)$. The case $l = 0$ is given by $r(0) = 0$.

The following procedure SIadd allows us to go from $l$ to $l+1$. The input of SIadd$((P; Q_1, \ldots, Q_l), Q_{l+1})$ is SIout$(P; Q_1, \ldots, Q_l)$, its output is SIout$(P; Q_1, \ldots, Q_{l+1})$.

PROCEDURE SIADD. We compute the $2r(l)$ generalized subresultant Sturm sequences associated to the

$$Q_{l, r(l+j)} = Q_{l+1}^2 Q_{l, j}, j = 1, \ldots, r(l) Q_{l, r(2l+j)} = Q_{l+1} Q_{l, j}, j = 1, \ldots, r(l)$$

and $v(l, m) = v(P, Q_{l, m})$, $m = r(l)+1, \ldots, 3r(l)$. One considers the $3r(l)$ dimension matrix

$$A' = \begin{bmatrix} A_l & A_l & A_l \\ 0 & A_l & A_l \\ 0 & A_l & -A_l \end{bmatrix}$$

and the equality $A' . \mathbf{c}' = \mathbf{V}'$, obtained from $(4_l)$ and proposition 1 where $\mathbf{c}'$ is the vector

$$(c_{\varepsilon(l, j), = 0}, j = 1, \ldots, r(l), c_{\varepsilon(l, j), > 0}, j = 1, \ldots, r(l), c_{\varepsilon(l, j), < 0}, j = 1, \ldots, r(l))$$

(with

$$c_{\varepsilon(l, j), = 0} = c_{\varepsilon(l, j), = 0}(P; Q_1, \ldots, Q_{l+1}), c_{\varepsilon(l, j), > 0} = c_{\varepsilon(l, j), > 0}(P; Q_1, \ldots, Q_{l+1})$$

and

$$c_{\varepsilon(l, j), < 0} = C_{\varepsilon(l, j), < 0}(P; Q_1, \ldots, Q_{l+1}))$$

and $\mathbf{V}'$ the vector $(v(l, 1), \ldots v(l, 3r(l))$.

The matrix $A'$ is invertible. One computes $\mathbf{c}'$ by inverting $A'$.

We define $r(l+1)$ as the number $(\leq r)$ of non-zero elements in $\mathbf{c}'$ and let

$$m_1, \ldots, m_{r(l+1)} (m_1 < \ldots < m_{r(l+1)} \leq 3r(l))$$

be such that the $m_i'$th co-ordinate of $\mathbf{c}'$, for $j = 1, \ldots, r(l+1)$, is different from 0. The sign conditions

$$\varepsilon(l+1, 1), \ldots, \varepsilon(l+1, r(l+1))$$

realized by $Q_1, \ldots, Q_{l+1}$ at the real roots of $P$ are the $m_j'$th elements, $j = 1, \ldots, r(l+1)$ of the list

$$((\varepsilon(l, 1), = 0), \ldots, (\varepsilon(l, r_l), = 0), (\varepsilon(l, 1), > 0), \ldots, (\varepsilon(l, r_l), > 0), (\varepsilon(l, 1), < 0), \ldots, (\varepsilon(l, r_l), < 0)).$$

One can then extract from $A'$ the $m_1'$th column, ..., the $m_{r(l+1)}'$th column, which gives a $3r(l)$, $r(l+1)$matrix $A''$. The $r(l)$ first lines of $A''$ are independant, since at least one of the columns $j$, $r(l)+j$, or $2r(l)+j$ of $A'$ appears in $A''$. An invertible matrix $A_{l+1}$, of dimension $r(l+1)$ can be extracted from $A''$. Let us denote

$$m_1', \ldots, m_{r(l+1)-r(l)}', m_1' < \ldots < m_{r(l+1)-r(l)}'$$

the elements of $\{1, \ldots, 2r(l+1)\}$ such that the $(r(l)+m_i')$th line of $A''$ appears in $A_{l+1}$.

The list $(Q_{l+1, j})_{j=1, \ldots, r(l+1)}$ is obtained by adding to the list $(Q_{l, j})_{j=1, \ldots, r(l)}$ the polynomial $Q_{l+1}^2 Q_{l, j}$ if $m_j' \leq r(l)$ or $Q_{l+1} Q_{l, j}$ if $m_j' > r(l)$ for $j \in \{1, \ldots, r(l+1)-r(l)\}$.

It is easy to verify $(1_{l+1}), \ldots, (4_{l+1})$.

REMARK 3.2. The list $(Q_{l+1,j})_{j=1,\ldots,r(l+1)}$ coincides with list $(Q_{l,j})_{j=1,\ldots,r(l)}$ if $r(l) = r(l+1)$. Since there are at most $r$ values of $l$ such that $r(l+1) > r(l)$, this implies that the polynomials $Q_{l,j}$ are always products of a number smaller or equal to $r$ of polynomials among $Q_1^2, \ldots, Q_k^2, Q_1, \ldots, Q_k$.

EXAMPLE 3.3. Let us consider

$P = (X^3-1)(X^2-9)$
$Q_1 = X$
$Q_2 = X+1$
$Q_3 = X-4$
$Q_4 = X-2$
$Q_5 = X+2$
$Q_6 = X^4-X.$

We have $v(P, 1) = 3$: $P$ has 3 real roots.

Also, $v(P, Q_1^2) = 3$ and $v(P, Q_1) = 1$: $P$ has 0 roots with $Q_1 = 0$, $P$ has 2 roots with $Q_1 > 0$ and 1 with $Q_1 < 0$. So we have $r_1 = 2$; the list of polynomials is $(1, Q_1)$ and $A_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

Let us add $Q_2$:

$v(P, Q_2^2) = 3,$
$v(P, Q_1 Q_2^2) = 1,$
$v(P, Q_2) = 1,$ and
$v(P, Q_1 Q_2) = 3,$

hence $P$ has 2 roots with $Q_1 > 0$ and $Q_2 > 0$ and 1 root with $Q_1 < 0$ and $Q_2 < 0$. So we have $r_2 = r_1$ and the list of polynomials is just $(1, Q_1)$, also $A_2 = A_1$.

Let us add $Q_3$:

$v(P, Q_3^2) = 3,$
$v(P, Q_1 Q_3^2) = 1,$
$v(P, Q_3) = -3,$ and
$v(P, Q_1 Q_3) = -1,$

hence $P$ has 2 roots with $Q_1 > 0$, $Q_2 > 0$ and $Q_3 < 0$ and 1 root with $Q_1 < 0$, $Q_2 < 0$ and $Q_3 < 0$. So we have $r_3 = r_2 = r_1$ and the list of polynomials is just $(1, Q_1)$, also $A_3 = A_1$.

Let us add $Q_4$:

$v(P, Q_4^2) = 3,$
$v(P, Q_1 Q_4^2) = 1,$
$v(P, Q_4) = -1,$ and
$v(P, Q_1 Q_4) = 1,$

hence $P$ has 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 > 0$ and $Q_4 > 0$; 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 < 0$ and $Q_4 < 0$; 1 root with $Q_1 < 0$, $Q_2 < 0$, $Q_3 < 0$ and $Q_4 < 0$. So we have $r_4 = 3$ and the list of polynomials is $(1, Q_1, Q_4)$, also

$$A_4 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & -1 \\ 1 & 1 & -1 \end{bmatrix}.$$

We add $Q_5$:

$v(P, Q_5^2) = 3,$
$v(P, Q_1 Q_5^2) = 1,$

$v(P, Q_4Q_5^2) = -1$,
$v(P, Q_5) = 1$,
$v(P, Q_1Q_5) = 3$, and
$v(P, Q_4Q_5) = 1$,

hence $P$ has 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 < 0$, $Q_4 > 0$ and $Q_5 > 0$; 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 < 0$, $Q_4 < 0$ and $Q_5 > 0$; 1 root with $Q_1 < 0$, $Q_2 < 0$, $Q_3 < 0$, $Q_4 < 0$ and $Q_5 < 0$.

So we have $r_5 = 3$ and the list of polynomials is just $(1, Q_1, Q_4)$, also $A_5 = A_4$.

We add $Q_6$:

$v(P, Q_6^2) = 2$,
$v(P, Q_1Q_6^2) = 0$,
$v(P, Q_4Q_6^2) = 0$,
$v(P, Q_6) = 0$,
$v(P, Q_1Q_6) = 2$, and
$v(P, Q_4Q_6) = 2$.

Hence $P_1$ has 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 < 0$, $Q_4 > 0$, $Q_5 > 0$ and $Q_6 > 0$ and 1 root with $Q_1 < 0$, $Q_2 < 0$, $Q_3 < 0$, $Q_4 < 0$, $Q_5 < 0$ and $Q_6 < 0$; $P_2$ has 1 root with $Q_1 > 0$, $Q_2 > 0$, $Q_3 < 0$, $Q_4 < 0$, $Q_5 > 0$ and $Q_6 = 0$.

COMPLEXITY OF THE COMPUTATION. Let us denote by

$P$, a polynomial with integer coefficients, of degree $p$;
$r$, the number of real roots of $P$, $r \leq p$;
$Q_1, \ldots, Q_k$, a finite list of polynomials with integer coefficients of respective degrees $q_1, \ldots, q_k$;
$d$, the maximum of $p, q_1, \ldots, q_k$;
$N$, the maximum of $N(P), N(Q_1), \ldots, N(Q_k)$; and
$n$, an integer bigger than $d, \log(N), k$.

PROPOSITION 3.3. *The complexity of SI is in* $O(kr^4d^4 \log(N)^2)$, *that is* $O(n^{11})$, *or using "fast multiplication", in* $O(n^8 \log n \log\log n)$.

PROOF. In SIadd, which is done $k$ times, the number of generalized subresultant Sturm sequences to compute is bounded by $2r$. Each computation of a generalized subresultant Sturm sequence takes $O(rd^2)$ arithmetic operation, since the $Q_{i,j}$ are of degree $\leq p+2rd$.

So, the total number in SI of arithmetic operations is in $O(kr^2d^2)$.

A bound for $N(P'Q_{i,j})$ is $N(P')(N)^{2r}$, hence a bound for $\log(N(P'Q_{i,j}))$ is $\log(p)+\log(N(P))+2r\log(N)$. So applying the size of the coefficients of the polynomials computed in SIadd is in $O(rd \log(N))$.

So, using classical arithmetic, our total computation is in $O(kr^4d^4 \log(N)^2)$, that is $O(n^{11})$. Using the "fast multiplication" (Aho, Hopcroft & Ullman, 1974) we get

$$O(kr^3d^3 \log(N)\log(rd \log(N))\log\log(rd \log(N)),$$

that is $O(n^8 \log n \log\log n)$.

REMARK 3.4. We get a complexity which is linear in the number $k$ of polynomials and where the number of real roots $r$ plays an important role.

## 4. Real Algebraic Numbers

Algorithms proposed before Coste & Roy (1988) to work on real aglebraic numbers were semi-numerical: one characterizes a real algebraic number by means of a square-free defining polynomial $P$ with integer coefficients and an interval that isolates $\xi$ from all other roots of $P$ (Collins, 1975). The new approach is purely formal and relies on Thom's lemma. We shall present a slight improvement to $b_5$ in Coste & Roy (1988).

### 4.1. Real Algebraic Numbers Coding

In this section let us denote by $P$, a polynomial with integer coefficients, of degree $p$, $r$, the number of real roots of $P$, $r \leq p$; and $n$, an integer bigger than $p, \log(N(P))$.

Principle of the coding is based on the following proposition.

Proposition 4.2. *Let $P$ be a polynomial of degree $p$ with integer coefficients. Let $\xi$ and $\xi'$ be two real roots of $P$. Suppose the signs $\varepsilon(\xi)_{p-l}$ and $\varepsilon(\xi')_{p-l}$ of $P^{(i)}(\xi)$ and $P^{(i)}(\xi')$, $i = 1, \ldots, p-1$, are given. Then:*

*(i) If $\varepsilon(\xi)_i = \varepsilon(\xi)_i$ for all $i = 1, \ldots, p-1$ then $\xi = \xi'$.*

*(ii) In the other case one can decide whether $\xi < \xi'$ or $\xi > \xi'$:*
*let $k$ be the smallest integer such that $\varepsilon(\xi)_k$ and $\varepsilon(\xi')_k$ are different;*

*(a) $\varepsilon(\xi)_{k-1} = \varepsilon(\xi')_{k-1}$ is different from $= 0$,*

*(b) if $\varepsilon(\xi)_{k-1} = \varepsilon(\xi')_{k-1}$ is $> 0$ one has $\xi > \xi'$ if and only if $P^{(p-k)}(\xi)$ is bigger than $P^{(p-k)}(\xi')$ (which is known by looking at $\varepsilon(\xi)_k$ and $\varepsilon(\xi')_k$),*

*(c) if $\varepsilon(\xi)_{k-1} = \varepsilon(\xi')_{k-1}$ is $< 0$ one has $\xi > \xi'$ if and only if $P^{(p-k)}(\xi)$ is smaller than $P^{(p-k)}(\xi')$ (which is known by looking at $\varepsilon(\xi)_k$ and $\varepsilon(\xi')_k$),*

Proof. See Coste & Roy (1988).

Let us remark that $P$ does not need to be irreducible.

Algorithm RAN. As seen in Proposition 3.3, a real root $\xi$ of $P$ is characterized by the sequence of signs it gives to the derivatives $P^{(i)}$ of $P$. So a possible coding of $\xi$ is this sequence of signs. The algorithm $b_5$ given in Coste & Roy (1988) for coding the real roots of a polynomial was simply, after Proposition 4.2, $b_4$ applied to $P; P', \ldots, P^{(p-1)}$.

We could here use SI (which is an improvement of $b_4$) applied to $P; P', \ldots, P^{(p-1)}$.

The algorithm RAN is an improvement of this strategy, based on the following remarks:

Remark 4.3. (a) It is better to introduce the derivatives in the order $P^{(p-1)}, \ldots, P'', P'$, since after proposition 4.2 (ii), if condition (*, $l$) holds: "for all $l$-multiples $\varepsilon'$ of sign conditions on $P^{(p-1)}, \ldots, P^{(p-l)}$ we have $c_{\varepsilon'}(P; P^{(p-1)}, \ldots, P^{(p-l)})$ equal to 0 or 1" then we have

(i) an individual code of length $\leqslant p$ of all the roots of $P$,

(ii) if we code all real roots of $P$ as indicated in (i) we can still compare them using proposition 4.2.

This will allow us to stop the computation in step $l$ as soon as (*, $l$) is fulfilled.

(b) Moreover, using Thom's lemma for $P^{(p-l)}$ condition (**, $l$) holds: "for all $l$-multiples $\varepsilon'$ of sign conditions on $P^{(p-1)}, \ldots, P^{(p-l)}$ with $P^{(p-l)} = 0$ we have $c_{\varepsilon'}(P; P^{(p-1)}, \ldots, P^{(p-l)})$ equal to 0 or 1".

It is clear that if a root $\xi$ of $P$ is a root of some $P^{(p-l)}$ the information given by its code as a root of $P^{(p-l)}$ is enough to compare it to other roots of $P$, so that we do not need to determine the signs of $P^{(p-l')}$ at $\xi$ for $l' > l$.

So the code of a root $\xi$ of $P$ will be in general a list of length $l(\xi)$ ($\leqslant d$) of the signs of $P^{(p-1)}(\xi), \ldots, P^{(p-l(\xi))}$, and hence will be shorter in some cases than the code given by $b_5$ in Coste & Roy (1988).

Let us take an example to illustrate this situation.

EXAMPLE 4.4. Let us consider $P = X^5 - X$. The polynomial $P$ has three real roots $\xi_1$, $\xi_2$ and $\xi_3$. If we apply $b_5$ in Coste & Roy (1988) we shall have to compute the signs of $P'$, $P''$, $P^{(3)}$ and $P^{(4)}$ at $\xi_1$, $\xi_2$ and $\xi_3$.

Now, we apply SI to $P$ and $P^{(4)} = X$ (up to a positive constant). We compute

$v(P) = 3$,
$v(P, X^2) = 2$ and
$v(P, X) = 0$.

The polynomial $P$ has three real roots, $\xi_1$ with $X > 0$, $\xi_2$ with $X < 0$, and $\xi_3$ with $X = 0$.

So we have an individual coding for each real root of $P$ just using $P^{(4)}$. This coding is sufficient to compare $\xi_1$, $\xi_2$ and $\xi_3$, since $P^{(5)}$ is constant and positive: we have $\xi_2 < \xi_3 < \xi_1$.

PRESENTATION OF RAN. We want to determine the following output of RAN applied to $P$, RANout($P$), which will consist of:

(i) a list $\varepsilon = (\varepsilon(i))_{i=1,\ldots,r}$ of distinct $l(i)$-multiples of sign conditions, such that every root $\xi$ of $P$ verifies exactly one of the conjunction of sign conditions $P^{(p-1)}(\xi)\varepsilon(i)_1, \ldots, P^{(p-l(i))}(\xi)\varepsilon(i)_{l(i)}$,
(ii) a list of polynomials $(P_j)_{j=1,\ldots,r}$ which are some product of the $P^{(j)}$s and the $P^{(j)^2}$s and the numbers $v(j) = v(P, P_j)$,
(iii) an invertible matrix $A$ of dimension $r$ such that $A\,\mathbf{c} = \mathbf{V}$ where $\mathbf{c}$ is the vector $(1, \ldots, 1)$ and $\mathbf{V}$ the vector $(v(1), \ldots, v(r))$.

DESCRIPTION OF RAN. We define the algorithm by induction on $k$. We suppose that we have computed the output of RAN applied to $P$ at step $k$ RANout($P$, $k$), which consists of:

($1_k$) a list $\varepsilon(k) = (\varepsilon(k, i))_{i=1,\ldots,r(k)}$ of distinct $l(k, i)$-multiples of sign conditions ($l(k, i) \leq k$), and the numbers $c(k, i)$ of elements of the non-empty set

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } P^{(p-j)}\varepsilon(k, i)_j, j = 1, \ldots, l(k, i)\};$$

let $s(k)$ be the biggest integer $i$ such that the last component of $\varepsilon(k, i)$ is $= 0$,

($2k$) a list of polynomials $(P_{k,i})_{i=1,\ldots,r(k)}$ which are some product of the $P^{(j)}$s and the $P^{(j)^2}$s and the numbers $v(k, i) = v(P, P_{k,i})$,

($3_k$) an invertible matrix $A_{r(k)}$ of dimension $r(k)$ such that $A_{r(k)}\mathbf{c}_k = \mathbf{V}_k$ where $\mathbf{c}_k$ is the vector $(c(k, i))_{i=1,\ldots,r(k)}$ and $\mathbf{V}_k$ the vector $(v(k, 1), \ldots, v(k, r(k))$. Moreover, $A_k$ is of the form $\begin{bmatrix} C'_k B'_k \\ 0\ A'_k \end{bmatrix}$ with $A'_k$ a square invertible matrix of dimension $r(k) - s(k)$ and $C'_k$ a matrix with no identically null line.

We suppose the computation completed for step $k$ and we add the polynomial $P^{(p-k-1)}$ by the following procedure RANadd.

PROCEDURE RANADD. The input of RANadd is RANout$(P, k)$ and its output is RANout$(P, k+1)$.

If for every $\varepsilon(k) = (\varepsilon(k, i))_{i=1,\ldots,r(k)}$ we have $c(k,i) = 1$, the algorithm returns RANout$(P)$ = RANout$(P, k)$ and stops. If not we have $s(k) < r(k)$ (using Remark 4.3b).

Let $A''$ be the matrix $\begin{bmatrix} C'_k & B'_k & B'_k & B'_k \\ 0 & A'_k & A'_k & A'_k \\ 0 & 0 & A'_k & A'_k \\ 0 & 0 & A'_k & -A'_k \end{bmatrix}$,

$\varepsilon'(k, j)_{j=1,\ldots,r(k)+2(r(k)-s(k))}$ be the list

$$(\varepsilon(k, i)_{i \le s(k)}, (\varepsilon(k, i), = 0)_{s(k)<i\le r(k)}, (\varepsilon(k, i), > 0)_{s(k)<i\le r(k)}, (\varepsilon(k, i) < 0)_{s(k)<i\le r(k)});$$

$c'(j)$ be equal to $c(k, j)$ if $i \le s(k)$ and be equal to the cardinal of

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } P^{(d-j)}\varepsilon'(k, i)_j, j = 1, \ldots, l(k, i)+1\} \text{ if } s(k) < i \le r(k);$$

$c'$ be the list $c'(j)_{j=1,\ldots,r(k)+2(r(k)-s(k))}$;

$V'$ be the list $V_k, v(P, P_{k,l}P^{(d-k-1)2}), v(P, P_{k,l}P^{(d-k-1)})$.

We have the equality $A''\mathbf{c}' = \mathbf{V}'$. The vector $\mathbf{V}'$ is easily computed by Sturm sequences. The matrix $A''$ is invertible. One computes $\mathbf{c}'$ by inverting $A''$.

We define $r(k+1)$ as the number $(\le r)$ of non zero elements in $\mathbf{c}'$ and let

$$m_1, \ldots, m_{r(k+1)} (m_1 < \ldots < m_{r(k+1)} \le r(k)+2(r(k)-s(k))$$

be such that the $m'_i$th co-ordinate of $\mathbf{c}'$, for $j = 1, \ldots, r(k+1)$, is different from 0. The sign conditions $\varepsilon(k+1, 1), \ldots, \varepsilon(k+1, r(k+1))$ are the $m_j$th elements $j = 1, \ldots, r(k+1)$ of the list $\varepsilon'$. Let $s(k+1)$ the biggest integer $i$ such that the last component of $\varepsilon(k+1, i)$ is $= 0$. We have always $s(k+1) \le r(k)$.

One can then extract from $A''$ the $m'_1$th column, ..., the $m'_{r(k+1)}$th column, which gives a $r(k)+2(r(k)-s(k))$, $r(k+1)$ matrix $B''$. The $s(k+1)$ first lines of $B''$ are independant and among the $2(r(k)-s(k))$ last lines of $B''$ there are $r(k+1)-s(k+1)$ independant lines. An invertible matrix $A_{k+1}$, of dimension $r(k+1)$, with as $s(k+1)$ first lines the $s(k+1)$ first lines of $B''$ and of the form $\begin{bmatrix} C'_{k+1} & B'_{k+1} \\ 0 & A'_{k+1} \end{bmatrix}$ can hence be extracted from $B''$.

Let us denote $m'_1, \ldots, m'_{r(k+1)-s(k+1)}$, $m'_1 < \ldots < m'_{r(k+1)-s(k+1)}$ the elements of $\{s(k+1), \ldots, r(k)+2((r(k)-s(k))\}$ such that the $(m'_i)$th line of $B''$ appears in $A_{k+1}$.

The list $(P_{k+1,j})_{j=1,\ldots,r(k+1)}$ is obtained by adding to the list $(P_{k,j})_{j=1,\ldots,s(k+1)}$ the polynomial $P_{k,l}(P^{(d-k-1)^2})$ if $l$ is of the form $m'_j$, with $s(k+1) < m'_j \le 2r(k)-s(k)$ or $P_{k,l}P^{(d-k-1)}$ if $l$ is of the form $m'_j$, with $m'_j > 2r(k)-s(k)$ for $j \in \{1, \ldots, r(k+1)-s(k+1)\}$.

It is easy to verify $(1_{k+1}), \ldots, (3_{k+1})$.

REMARK 4.5. The list $(P_{k+1,j})_{j=1,\ldots,r(k+1)}$ coincides with the list $(P_{k,j})_{j=1,\ldots,r(k)}$ in the case $r(k+1) = r(k)$, $s(k+1) = s(k)$. Since $s(k+1) > s(k)$ means that a root of $P$ which is not a root of $P^{(p-1)}, \ldots, P^{(p-k)}$ is a root of $P^{(p-k-1)}$ there are at most $r$ values of $k$ with $s(k+1) > s(k)$. Also since $r(k) \le r$ there are at most $r$ values of $k$ with $r(k+1) > r(k)$. This means that in each step of the computation the $(P_{k,j})$ consist in at most $2r$ products of derivatives (or square of derivatives) of $P$.

COMPLEXITY OF ALGORITHM RAN.

PROPOSITION 4.6. *The complexity of algorithm RAN is in* $O(r^4p^5E^2)$, *with* $E = p+\log(N(P)$ *that is* $O(n^{11})$ (*using classical arithmetic*), *or, using "fast multiplication", in* $O(n^8 \log n \log\log n)$.

PROOF. Instead of the signs given by the real roots of $P$ to $P^{(i)}$s we shall consider equivalently the signs given by the real roots of $P$ to $\underline{P}^{(i)} = P^{(i)}/i!$ ($i!$ is a common divisor of the $P^{(i)}$s coefficients). Remark that $N(\underline{P}^{(i)} \leq 2^i N(P)$.

A bound for the products of $2r$ $N(\underline{P}^{(i)})$ is $2^{2pr}N(P)^{2r}$, hence a bound for their log is $O(r(p+\log(N(P))))$. So the size of coefficients of the polynomials computed is bounded by $O(pr(p+\log(N(P))))$.

After proposition 4.2 the total complexity using classical arithmetic is in $O(r^4p^5E^2)$, with $E = p+\log(N(P)$ and using "fast multiplication" $O(r^3p^4E \log E \log\log E)$.

So the complexity of RAN is in $O(n^{11})$ using classical arithmetic, $O(n^8 \log n \log\log n)$ using "fast multiplication".

## 4.7. SIMULTANEOUS INEQUALITIES AT REAL ALGEBRAIC NUMBERS

ALGORITHM RANI. One wants to compute the signs taken by $Q$ at any real root of $P$. We denote by:

$P$, a polynomial with integer coefficients, of degree $p$;
$r$, the number of real roots of $P$;
$Q$, a polynomial with integer coefficients of degree $q$;
$d$, the maximum of $p$ and $q$;
$n$, an integer bigger than $d$, $\log(N(P))$, $\log(N(Q))$;
$\xi_i$, for $1 \leq i \leq r$ the $r$ real roots of $P$.

We suppose that the roots of $P$ are already coded by RAN.

DESCRIPTION OF RANI. The input of RANI is RANout($P$). It output is for every real root $\xi$ of $P$ the sign of $Q$ at $\xi$. With the notations used in the description of RANout, RANI consists just in

(i) computing the $v(P, P_iQ^2)_{i=1,\ldots,r}$ and $v(P, P_iQ)_{i=1,\ldots,r}$,
(ii) forming $\mathbf{V}' = (V, v(P, P_iQ^2)_{i=1,\ldots,r}, v(P, P_iQ)_{i=1,\ldots,r})$,

(iii) solving the system with $3r$ unknowns $\begin{bmatrix} A & A & A \\ 0 & A & A \\ 0 & A & -A \end{bmatrix} \mathbf{c}' = \mathbf{V}'$

which gives the vector $\mathbf{c}'$ such that

$c'(i)$, $1 \leq i \leq r$, is the number of elements of

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } P^{(d-j)}(x)\varepsilon(i)_j, j = 1, \ldots, l(i), Q(x) = 0\},$$

$c'(r+i)$, $1 \leq i \leq r$, is the number of elements of

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } P^{(d-j)}(x)\varepsilon(i)_j, j = 1, \ldots, l(i), Q(x) > 0\},$$

$c'(2r+i)$, $1 \leq i \leq r$, is the number of elements of

$$\{x \in \mathbb{R} | P(x) = 0 \text{ and } P^{(d-j)}(x)\varepsilon(i)_j,\ j = 1, \ldots, l(i),\ Q(x) < 0\},$$

The non-zero $c'(i)$s give the answer to the question.

COMPLEXITY OF THE ALGORITHM RANI

PROPOSITION 4.8. *The complexity of RANI is in* $O(rp(rp+q)E^2)$ *with*

$$E = (rp+q)(p+\log N(P)) + p(p^2 + p \log N(P) + \log N(Q))$$

*that is in* $O(n^{10})$ *using classical arithmetic, and in* $O(n^7 \log n \log\log n)$ *using "fast multiplication".*

PROOF. Easy, using the preceding results.

ALGORITHM RANSI. We denote by:
$P$, a polynomial with integer coefficients, of degree $p$;
$r$, the number of real roots of $P$;
$Q_1, \ldots, Q_k$ polynomials with integer coefficients, of degree less than $q$;
$d$, the maximum of $p$ and $q$;
$n$, an integer bigger than $d$, $\log(N(P))$, $\log(N(Q_i))$.
One wants to compute the signs taken by $Q_1, \ldots, Q_k$ at any real root of $P$.

DESCRIPTION OF RANSI. The input of RANSI is RANout($P$). Since each root of $P$ is individually coded, RANSI consists just of $k$ applications of RANI.

COMPLEXITY OF RANSI

PROPOSITION 4.9. *The complexity of RANSI is in* $O(krp(rp+q)E^2)$ *with*

$$E = (rp+q)(p+\log N(P)) + p(p^2 + p \log N(P) + \log N(Q))$$

*that is in* $O(n^{11})$ *using classical arithmetic and in* $O(n^7 \log n \log\log n)$ *using "fast multiplication".*

PROOF. Easy, using proposition 4.8.

AN APPLICATION OF RANSI: THE COMPARISON OF TWO REAL ALGEBRAIC NUMBERS. Algorithm RANSI is fundamental in the computation of topology of curves for example (see Roy (1987)). We give below another application.

We want to compare two real algebraic numbers $\xi_1$ and $\xi_2$ coded by RAN as real roots of the polynomial $P_1$ of degree $n_1$ and $P_2$ of degree $n_2$: we apply RANSI to $P_1$ and the derivatives of $P_1P_2$, then RANSI to $P_2$ and the derivatives of $P_1P_2$. Then, the coding of $\xi$ and $\xi'$ as real roots of $P_1P_2$ is known and those two algebraic numbers can be compared by proposition 4.2 as real roots of $P_1P_2$.

## References

Aho, A., Hopcroft, J., Ullman, J. (1974). *The Design and Analysis of Computer Algorithms.* Reading MA: Addison Wesley.

Ben-Or, M., Kozen, D., Reif, J. (1986). The Complexity of Elementary Algebra and Geometry. *J. Comp. Syst. Sci.* **32**, 251–264.

Bochnak, J., Coste, M., Roy, M. F. (1987). *Géométrie algébrique réelle*. Ergebnisse der Mathematik. Berlin: Springer-Verlag.

Collins, G. (1975). Quantifier elimination for real closed fields by cylindric algebraic decomposition. In Second GI Conference on Automata Theory and Formal Languages. *Lecture Notes in Computer Sciences*, vol. 33, pp. 134–183, Berlin: Springer-Verlag.

Coste, M., Roy, M.-F. (1988). Thom's lemma, the coding of real algebraic numbers and the topology of semi-algebraic sets. *J. Symb. Comp.* **5**, 121–129.

Cucker, F., Pardo, L. M., Raimondo, M., Recio, T., Roy, M.-F. (1989). On the computation of the local and global analytic branches of a real algebraic curve. *A.A.E.C.C.-5 Conference of Minorca. Springer Lecture Notes in Computing Science* **356.**

Jacobson, N. (1974). *Basic Algebra*. San Francisco: Freeman.

Lang, S. (1965). *Algebra*. Reading MA: Addison Wesley.

Loos, R. (1982). *Generalized polynomial remainder sequences*. In *Computer Algebra, Symbolic and Algebraic Computation*. Berlin: Springer Verlag.

Mignotte, M. (1982). *Some useful bounds*. In *Computer Algebra, Symbolic and Algebraic Computation*. Berlin: Springer-Verlag.

Roy, M.-F. (1990). *Computation of the topology of a real algebraic curve. Computational Geometry and Topology, Congress in Sevilla* (to appear).

Sylvester, J. J. (1953). *On a theory of syzygetic relations of two rational integral functions, comprising an application to the theory of Sturm's function. Trans. Roy. Soc. London.*