Available online at www.sciencedirect.com
SCIENCE DIRECT
An International Journal
computers & mathematics with applications
ELSEVIER Computers and Mathematics with Applications 52 (2006) 853–860
www.elsevier.com/locate/camwa

Real Solution Isolation Using Interval Arithmetic

BICAN XIA* AND TING ZHANG
LMAM and School of Mathematical Sciences
Peking University, Beijing 100871, P.R. China

(Received January 2005; revised and accepted June 2006)

Abstract—We propose a complete algorithm for real solution isolation for semialgebraic systems by using interval arithmetic. The algorithm is implemented as a Maple program `Nrealzero` and its performance on several examples is reported.

Keywords—Real solution isolation, Interval arithmetic, Polynomial systems, Semialgebraic systems, Maple.

1. INTRODUCTION

We call

$$[[P], [G_1], [G_2], [H]] \tag{1}$$

a *semialgebraic system* (SAS, for short), where P, G_1, G_2, and H denote

$$\{p_1(x_1, \ldots, x_n) = 0, \ldots, p_s(x_1, \ldots, x_n) = 0\},$$
$$\{g_1(x_1, \ldots, x_n) \geq 0, \ldots, g_r(x_1, \ldots, x_n) \geq 0\},$$
$$\{g_{r+1}(x_1, \ldots, x_n) > 0, \ldots, g_t(x_1, \ldots, x_n) > 0\},$$

and

$$\{h_1(x_1, \ldots, x_n) \neq 0, \ldots, h_m(x_1, \ldots, x_n) \neq 0\},$$

respectively. Here, $n, s \geq 1$, $r, t, m \geq 0$ and p_i, g_j, h_k are all polynomials in $x_1, \ldots, x_n$ with integer coefficients. Furthermore, we always assume that $\{p_1, \ldots, p_s\}$ has only a finite number of common zeros in $\mathbb{C}^s$.

Many problems in both practice and theory, such as the maximum number of limit cycles for polynomial differential system [1–3], the stability of a large class of biological networks [4,5], solving geometric constraints, some problems in computer vision, and automated proving inequality-type theorems [6], to name a few, can be reduced to finding real solutions of a certain SAS.

*Author to whom all correspondence should be addressed.
The work was supported by NKBRPC-2004CB318003, NKBRPC-2005CB321902 and NSFC-60573007 in China.

0898-1221/06/$ - see front matter Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX
doi:10.1016/j.camwa.2006.06.003

Because exact computation is needed, numerical methods are not applicable to these kinds of problems. Therefore, isolating the real solutions of semialgebraic systems becomes an important aspect of research in the field of computational real algebra. There have been some works concerning this issue, see, for example, [6–9]. The method in [6] is not a complete one and we have been trying to improve it. In this paper, we propose a complete algorithm based on interval arithmetic and show that it is faster than our earlier method on the examples in [6].

This paper is organized as follows. Section 2 devotes to some basic concepts of interval arithmetic and some notations needed in this paper. The details of the algorithm `Nrealzero` for isolating the real solutions of semialgebraic systems are given in Section 3. All the examples in [6] are recomputed by our new method in Section 4 and the timings reported.

2. INTERVAL ARITHMETIC

All the concepts and results in this section are classical. We only use some new notation. A subset of $\mathbb{R}$, the real numbers, of the form

$$X = [x_1, x_2] = \{x \mid x_1 \le x \le x_2\}, \qquad x_1, x_2 \in \mathbb{R},$$

is called an *interval*. The set of all intervals is denoted by $\mathrm{I}(\mathbb{R})$. If $x_1 = x_2$, X is called a *point interval*. A subset of $\mathbb{R}$ of the form

$$X = [-\infty, a] = \{x \mid x \le a\}, \qquad a \in \mathbb{R},$$

or

$$X = [b, +\infty] = \{x \mid b \le x\}, \qquad b \in \mathbb{R},$$

is called a *semi-infinity interval*. The set of all semi-infinity intervals is denoted by $\mathrm{SI}(\mathbb{R})$. Note that $\mathrm{I}(\mathbb{R})$ and $\mathrm{SI}(\mathbb{R})$ are disjoint sets.

DEFINITION 2.1. *For $X = [a, b] \in \mathrm{I}(\mathbb{R})$, the width, the midpoint, and the sign of X are defined, respectively, as $W(X) = b - a$, $m(X) = (a + b)/2$, and*

$$\mathrm{sign}(X) = \begin{cases} -1, & b < 0, \\ 0, & a \le 0 \le b, \\ 1, & a > 0. \end{cases}$$

DEFINITION 2.2. *For $X, Y \in \mathrm{I}(\mathbb{R}) \cup \mathrm{SI}(\mathbb{R})$ and $\diamond \in \{+, -, \cdot\}$, we define $X \diamond Y = \{x \diamond y \mid x \in X, y \in Y\}$. For $X = [a, b] \in \mathrm{I}(\mathbb{R})$, if $\mathrm{sign}(X) \ne 0$, we define*

$$X^{-1} = \frac{1}{X} = \left[\frac{1}{b}, \frac{1}{a}\right];$$

if $\mathrm{sign}(X) = 0$ and $W(X) \ne 0$, we define

$$X^{-1} = \frac{1}{X} = \begin{cases} \left[-\infty, \dfrac{1}{a}\right], & b = 0, \\ \left[\dfrac{1}{b}, +\infty\right], & a = 0, \\ \left[-\infty, \dfrac{1}{a}\right] \cup \left[\dfrac{1}{b}, +\infty\right], & a < 0 < b; \end{cases} \tag{2}$$

if $X = [0, 0]$, X^{-1} is undefined. And Y/X is defined to be $Y \cdot X^{-1}$, where $Y/X = Y \cdot [-\infty, 1/a] \cup Y \cdot [1/b, +\infty]$ if $a < 0 < b$.

For $a \in \mathbb{R}$, $X \in \mathrm{I}(\mathbb{R})$ and $\diamond \in \{+, -, \cdot, /\}$, we define $a \diamond X = [a, a] \diamond X$ and $X \diamond a = X \diamond [a, a]$.

DEFINITION 2.3. *(See [10,11].) Let f be an arithmetic expression of a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$. We replace all operands of f as intervals and replace all operations of f as interval operations and denote the result by F. Then, $F : \mathrm{I}(\mathbb{R})^n \to \mathrm{I}(\mathbb{R})$ is called an interval evaluation.*

Let F be an interval evaluation in $D \in \mathrm{I}(\mathbb{R})^n$. If for all $X, Y \subset D$, $X \subset Y$ implies $F(X) \subset F(Y)$, we call F a *monotonic interval evaluation.*

THEOREM 2.1. *(See [10,11].) An interval evaluation of any polynomial in $\mathbb{R}[x_1, \ldots, x_n]$ is a monotonic interval evaluation. Especially, this is true for univariate polynomials.*

3. THE ALGORITHM

Given a SAS in the form of (1), because the ideal generated by $p_1, \ldots, p_s$ is zero dimensional, we can use the Ritt-Wu method, Gröbner basis method or resultant methods to transform the system of equations into one or more systems in triangular form. Therefore, a SAS, $[[P], [G_1], [G_2], [H]]$, in the form of (1) can be transformed into one or more systems in the form of

$$[[F], [G_1], [G_2], [H]], \tag{3}$$

where

$$F = \{f_1(x_1), f_2(x_1, x_2), \ldots, f_s(x_1, x_2, \ldots, x_s)\}$$

is a *normal ascending chain* [12] (or a *regular chain* by [13] or a *regular set* by [14]). We call a system in the form of (3) a *triangular semialgebraic system* (TSA, for short).

Let the leading coefficient and the discriminant of a polynomial, f, with respect to x be denoted by $\mathrm{lc}(f, x)$ and $\mathrm{dis}(f, x)$, respectively. A TSA is *regular* if

(a) $\mathrm{lc}(f_1, x_1) \neq 0$ and $\mathrm{dis}(f_1, x_1) \neq 0$,
(b) each zero of $\{f_1 = 0, \ldots, f_{i-1} = 0\}$ is not a zero of $\mathrm{lc}(f_i, x_i) \cdot \mathrm{dis}(f_i, x_i)$, for $i = 2, \ldots, s$, and
(c) each zero of $\{f_1 = 0, \ldots, f_s = 0\}$ is not a zero of any g_j $(1 \leq j \leq t)$ and h_k $(1 \leq k \leq m)$.

Obviously, a regular TSA can be viewed as a system in the following form:

$$\{f_1 = 0, \ldots, f_s = 0,\ g_1 > 0, \ldots, g_t > 0\}. \tag{4}$$

Xia and Yang [6] gave an algorithm for decomposing any TSA (or SAS) into regular TSAs. So, in the following we only discuss how to isolate the real solutions of a regular TSA in the form of (4).

Let a regular TSA T in the form of (4) be given. There exist some efficient methods to isolate the real roots of a univariate polynomial [15]. To isolate the real solutions of T, a natural idea is to isolate the real roots of the first equation of the system and substitute each resulting interval in the rest of the equations and then repeat the above computation. Of course, we have to deal with polynomials with "interval coefficients".

DEFINITION 3.1. *Let a polynomial $q \in \mathbb{Z}[x_1, \ldots, x_{i+1}]$ be represented as*

$$q = q_l(x_1, \ldots, x_i)x_{i+1}^l + \cdots + q_1(x_1, \ldots, x_i)x_{i+1} + q_0(x_1, \ldots, x_i),$$

where $q_l(x_1, \ldots, x_i) \not\equiv 0$. For any $X = ([a_1, b_1], \ldots, [a_i, b_i]) \in \mathrm{I}(\mathbb{R})^i$, let Q_j $(0 \leq j \leq l)$ be an interval evaluation of q_j in X and

$$\begin{aligned} Q &= Q_l([a_1, b_1], \ldots, [a_i, b_i])x_{i+1}^l + \cdots + Q_0([a_1, b_1], \ldots, [a_i, b_i]) \\ &= [c_l, d_l]x_{i+1}^l + \cdots + [c_0, d_0]. \end{aligned}$$

We call

$$_{-}q = c_l x_{i+1}^l + \cdots + c_0 \qquad \text{and} \qquad {}^{+}q = d_l x_{i+1}^l + \cdots + d_0 \tag{5}$$

the lower bound polynomial and upper bound polynomial of q in X, respectively.

Let $[q(x)]^{(n)}$ $(n \in \mathbb{N})$ denote the n-order derivative of $q(x)$ with respect to x and $[q(x)]^{(0)} = q(x)$.

PROPOSITION 3.1. *Suppose $X = ([a_1, b_1], \dots, [a_i, b_i])$ is an isolating cube of some zero, x^*, of $\{f_1 = 0, \dots, f_i = 0\}$ in the system T and $_-f_{i+1}$ and $^+f_{i+1}$ are the lower bound and upper bound polynomials of f_{i+1} in X, respectively, then for all $n \in \mathbb{N}$ and all $x_{i+1} \in (0, +\infty)$*

$$[_-f_{i+1}]^{(n)} \leq [f_{i+1}(x^*, x_{i+1})]^{(n)} \leq [^+f_{i+1}]^{(n)}. \tag{6}$$

PROOF. Suppose $_-f_{i+1} = c_l x_{i+1}^l + \cdots + c_0$, $^+f_{i+1} = d_l x_{i+1}^l + \cdots + d_0$, and $f_{i+1}(x^*, x_{i+1}) = e_l x_{i+1}^l + \cdots + e_0$. From Definition 3.1, it is easy to see that $c_j \leq e_j \leq d_j$ for $0 \leq j \leq l$. Therefore, the relations (6) hold for all $x_{i+1} \in (0, +\infty)$. ∎

In fact, the relations (6) hold not only for x^* but also for any $x \in X$.

Now, suppose x^* is a real solution of $\{f_1 = 0, \dots, f_i = 0\}$ and X is an isolating cube such that $x^* \in X$. Let $_-f_{i+1}(x_{i+1})$ and $^+f_{i+1}(x_{i+1})$ be the lower bound and upper bound polynomials of f_{i+1} in X, respectively. By using Proposition 3.1, we want to obtain the isolating intervals of $f_{i+1}(x^*, x_{i+1})$ by isolating the real zeros of $_-f_{i+1}$ and $^+f_{i+1}$.

From Proposition 3.1 (by letting $n = 0$), we have

$$_-f_{i+1} \leq f_{i+1}(x^*, x_{i+1}) \leq {}^+f_{i+1}$$

for $x_{i+1} > 0$. So, we first shift the real roots of $f_{i+1}(x^*, x_{i+1})$ to the positive real roots of $\overline{f_{i+1}}(x^*, x_{i+1}) = f_{i+1}(x^*, x_{i+1} - B)$, where B satisfies that any real root of $f_{i+1}(x^*, x_{i+1})$ is greater than B.

To determine the value of B, we let

$$\widetilde{f_{i+1}}(x_1, \dots, x_i, x_{i+1}) = f_{i+1}(x_1, \dots, x_i, -x_{i+1}).$$

Then the negative zeros of $f_{i+1}(x^*, x_{i+1})$ correspond to the positive zeros of $\widetilde{f_{i+1}}(x^*, x_{i+1})$, and thus, the positive-root-bound of $\widetilde{f_{i+1}}(x^*, x_{i+1})$ is the negative-root-bound of $f_{i+1}(x^*, x_{i+1})$. We shrink X repeatedly until $\mathrm{lc}(_-f_{i+1}) \cdot \mathrm{lc}(^+f_{i+1}) > 0$ (this inequality must hold at a certain step because the TSA T being regular implies $\mathrm{lc}(f_{i+1}, x_{i+1})(x^*) \neq 0$) which guarantees that the greatest real root of $f_{i+1}(x^*)$ is smaller that that of $_-\widetilde{f_{i+1}}(x^*)$ or $^+\widetilde{f_{i+1}}(x^*)$. Then, let $_-\widetilde{f_{i+1}}$ and $^+\widetilde{f_{i+1}}$ be the lower bound and upper bound polynomials of $\widetilde{f_{i+1}}$ in X, respectively, and $B > 0$ the maximum of the root-bounds of $_-\widetilde{f_{i+1}}$ and $^+\widetilde{f_{i+1}}$. We define

$$\overline{f_{i+1}}(x_1, \dots, x_i, x_{i+1}) = f_{i+1}(x_1, \dots, x_i, x_{i+1} - B).$$

Obviously, all the real zeros of $f_{i+1}(x^*, x_{i+1})$ are shifted to the real zeros of $\overline{f_{i+1}}(x^*, x_{i+1})$, which are all in $(0, +\infty)$. Therefore, without loss of generality, we only consider the positive roots of $f_{i+1}(x^*, x_{i+1})$ in our algorithm.

Suppose

$$S_j = \left[\left[\alpha_1^{(j)}, \beta_1^{(j)}\right], \dots, \left[\alpha_{m_j}^{(j)}, \beta_{m_j}^{(j)}\right]\right], \qquad j = 1, 2,$$

and S_1 and S_2 isolate all positive zeros of $_-f_{i+1}(x_{i+1})$ and $^+f_{i+1}(x_{i+1})$, respectively. Because T is regular, $f_{i+1}(x^*, x_{i+1}) = 0$ has no repeated roots. So, by Proposition 3.1, if X is small enough, $m_1 = m_2$ and we can define that

$$S = [[\alpha_1, \beta_1], \dots, [\alpha_{m_1}, \beta_{m_1}]], \tag{7}$$

where for $1 \leq k \leq m_1$,

$$\alpha_k = \min\left(\alpha_k^{(1)}, \alpha_k^{(2)}\right), \qquad \beta_k = \max\left(\beta_k^{(1)}, \beta_k^{(2)}\right). \tag{8}$$

If the widths of X, S_1 and S_2 are all small enough, any two adjacent intervals of S do not intersect. Furthermore, $_{-}f_{i+1}(x_{i+1})$, $^{+}f_{i+1}(x_{i+1})$ and f_{i+1} are all monotonic in each $[\alpha_k, \beta_k]$ of S. In this case, we write $S = S_1 \Delta S_2$ and it is easy to prove that S isolates all positive zeros of f_{i+1}.

REMARK 3.1. Whenever we use the notion $S = S_1 \Delta S_2$ (or $S \leftarrow S_1 \Delta S_2$), we mean that, given S_1 and S_2, S is defined by (7) and (8) and the following three conditions are satisfied:

1. $m_1 = m_2$;
2. $f_{i+1}(x_{i+1})$ is monotonic when x_{i+1} is in each (α_k, β_k);
3. any two adjacent intervals of S do not intersect.

To make the three conditions hold, we may have to shrink X repeatedly (finite many times, of course).

Now, we can describe our algorithm as follows. The finiteness and correctness of the algorithm are guaranteed by the above discussion.

Algorithm: `NREALZERO`

INPUT: A regular TSA T in the form of (4);

OUTPUT: A list of isolating cubes of the positive real solutions of T.

STEP 0. $L_1 \leftarrow \emptyset$, $L_2 \leftarrow \emptyset$, $i \leftarrow 0$.

STEP 1. $(i = 0)$

$L_1 \leftarrow$ the isolating intervals of the positive zeros of f_1; $i \leftarrow i + 1$;

STEP 2. $(0 < i < s)$

FOR $X = ([a_1, b_1], \ldots, [a_i, b_i]) \in L_1$ DO
$L_1 \leftarrow L_1 \setminus \{X\}$;
Compute $_{-}f_{i+1}$ and $^{+}f_{i+1}$ in X;
$S_1 \leftarrow$ the isolating intervals of the positive zeros of $_{-}f_{i+1}$;
$S_2 \leftarrow$ the isolating intervals of the positive zeros of $^{+}f_{i+1}$;
$S \leftarrow S_1 \Delta S_2$;
$L_2 \leftarrow L_2 \cup \{([a_1, b_1], \ldots, [a_i, b_i], [c, d]) \mid [c, d] \in S\}$;
END FOR;
If $L_1 = \emptyset$ and $L_2 = \emptyset$, then RETURN($\emptyset$);
If $L_1 = \emptyset$ and $L_2 \neq \emptyset$, then $L_1 \leftarrow L_2$, $L_2 \leftarrow \emptyset$, $i \leftarrow i + 1$;

STEP 3. $(i = s)$

For each $X \in L_1$, compute $G_j(X)$ $(1 \leq j \leq t)$, where G_j is an interval evaluation of g_j. If $\mathrm{sign}(G_{j_0}(X)) < 0$ for some j_0 $(1 \leq j_0 \leq t)$, delete X from L_1; If $\mathrm{sign}(G_{j_1}(X)) = 0$ for some j_1 $(1 \leq j_1 \leq t)$, shrink X repeatedly until either $\mathrm{sign}(G_{j_1}(X)) < 0$ or $\mathrm{sign}(G_{j_1}(X)) > 0$. Return the remaining elements in L_1.

In Step 3 and the loop of Step 2 ($S \leftarrow S_1 \Delta S_2$), we may have to shrink X repeatedly. The following subalgorithm is for this end.

Subalgorithm: `NSHR`

INPUT: A cube $X = ([a_1, b_1], \ldots, [a_i, b_i])$ from `NREALZERO` and the regular TSA T;

OUTPUT: A cube $X' \subset X$ such that $x^* \in X'$, where $x^* = (x_1^*, \ldots, x_i^*)$ is the only solution of $\{f_1 = 0, \ldots, f_i = 0\}$ in X.

STEP 0. $j \leftarrow 0$.

STEP 1. $(j = 0)$

By the intermediate value theorem, we obtain an interval $[a_1', b_1'] \subset [a_1, b_1]$ such that $x_1^* \in [a_1', b_1']$ and $W([a_1', b_1']) \leq (1/2)W([a_1, b_1])$. Then, let $j \leftarrow j + 1$, $X' \leftarrow ([a_1', b_1'])$.

STEP 2. $(0 < j < i)$

Compute ${}_-f_{j+1}$ and ${}^+f_{j+1}$ with respect to X';
By the intermediate value theorem, compute an interval $[a, b] \subset [a_{j+1}, b_{j+1}]$ such that $[a, b]$ contains the zero of ${}_-f_{j+1}$ in $[a_{j+1}, b_{j+1}]$ and $W([a, b]) = (1/8)W([a_{j+1}, b_{j+1}])$;
Similarly, compute an interval $[c, d] \subset [a_{j+1}, b_{j+1}]$ such that $[c, d]$ contains the zero of ${}^+f_{j+1}$ in $[a_{j+1}, b_{j+1}]$ and $W([c, d]) = (1/8)W([a_{j+1}, b_{j+1}])$;
$a'_{j+1} \leftarrow \min(a, c)$, $b'_{j+1} \leftarrow \max(b, d)$;
$X' \leftarrow (X', [a'_{j+1}, b'_{j+1}])$, $j \leftarrow j + 1$.

STEP 3. $(j = i)$ Output X'.

Let us prove the correctness of Algorithm NSHR. Only Step 2 of NSHR needs some further description. Let us denote ${}_-f_{j+1}$ and ${}^+f_{j+1}$ with respect to X' by ${}_-f_{j+1}(X')$ and ${}^+f_{j+1}(X')$, respectively. By Definition 3.1 and Theorem 2.1, the following relations hold for all $x_{j+1} \in (0, +\infty)$:

$$
\begin{aligned}
{}_-f_{j+1}(X) \leq {}_- f_{j+1}(X') \leq f_{j+1}(x_1^*, \ldots, x_j^*, x_{j+1}) \leq {}^+f_{j+1}(X') \leq {}^+f_{j+1}(X), \\
{}_-f'_{j+1}(X) \leq {}_- f'_{j+1}(X') \leq f'_{j+1}(x_1^*, \ldots, x_j^*, x_{j+1}) \leq {}^+f'_{j+1}(X') \leq {}^+f'_{j+1}(X).
\end{aligned}
$$

On the other hand, from NREALZERO, ${}_-f_{j+1}(X)$ is monotonic on $[a_{j+1}, b_{j+1}]$ and has only one zero in it. So does ${}^+f_{j+1}(X)$. Then, by the above relations, ${}_-f_{j+1}(X')$ and ${}^+f_{j+1}(X')$ are both monotonic on $[a_{j+1}, b_{j+1}]$ and each of them has only one zero in the interval. The correctness of NSHR is thus proved.

REMARK 3.2. In Step 2 of Algorithm NSHR, we use an empirical factor 1/8. Theoretically speaking, the factor can be any rational number between zero and one.

4. EXAMPLES

Xia and Yang [6] proposed an incomplete algorithm for isolating the real solutions of a given SAS and the algorithm is implemented as a Maple program `realzero`. Our new algorithm, NREALZERO, has also been implemented as a Maple program which is called `Nrealzero`. In general, for a SAS, the computation of `Nrealzero` consists of three main steps. First, transform the system of equations into one or more systems in triangular form. Second, transform each component into regular TSAs if necessary. Third, apply NREALZERO to each resulting regular TSA.

By the new program, we recomputed all the six examples in [6]. For readers' convenience, we list the six systems in the appendix but refer the reader to [6] for detail. The following table reports the performance of `Nrealzero` on those six examples.

Table 1.

Example No.	Ex. 2	Ex. 3	Ex. 4	Ex. 5	Ex. 6	Ex. 7
Triangular Form	0.151	0.571	2.855	0.621	0.040	0.231
Regular TSA	0.	7.470	3.655	2.424	0.010	1.161
NREALZERO	**0.120**	**0.511**	**0.**	**0.020**	**0.260**	**14.571**
REALZERO	0.396	15.382	2.889	3.07	0.45	33.840

The first two steps of `Nrealzero` and `realzero` are the same. So, the data in the fourth and fifth rows of the table show the difference in efficiency of these two algorithms. Obviously, `Nrealzero` is faster than `realzero`.

APPENDIX

EXAMPLE 2. (Chemical reaction)

$$\begin{aligned} h_1 &= 2 - 7x_1 + x_1^2 x_2 - \frac{1}{2}(x_3 - x_1) = 0, \\ h_2 &= 6x_1 - x_1^2 x_2 - 5(x_4 - x_2) = 0, \\ h_3 &= 2 - 7x_3 + x_3^2 x_4 - \frac{1}{2}(x_1 - x_3) = 0, \\ h_4 &= 6x_3 - x_3^2 x_4 + 1 + \frac{1}{2}(x_2 - x_4) = 0. \end{aligned}$$

EXAMPLE 3. (Neural network)

$$\begin{aligned} f_1 &= 1 - cx - xy^2 - xz^2 = 0, \\ f_2 &= 1 - cy - yx^2 - yz^2 = 0, \\ f_3 &= 1 - cz - zx^2 - zy^2 = 0, \\ f_4 &= 8c^6 + 378c^3 - 27 = 0, \\ & c > 0, \qquad 1 - c > 0. \end{aligned}$$

EXAMPLE 4. (Cyclic 5)

$$\begin{aligned} p_1 &= a + b + c + d + e = 0, \\ p_2 &= ab + bc + cd + de + ea = 0, \\ p_3 &= abc + bcd + cde + dea + eab = 0, \\ p_4 &= abcd + bcde + cdea + deab + eabc = 0, \\ p_5 &= abcde - 1 = 0. \end{aligned}$$

EXAMPLE 5.

$$\begin{aligned} p_1 &= 2x_1(2 - x_1 - y_1) + x_2 - x_1 = 0, \\ p_2 &= 2x_2(2 - x_2 - y_2) + x_1 - x_2 = 0, \\ p_3 &= 2y_1(5 - x_1 - 2y_1) + y_2 - y_1 = 0, \\ p_4 &= y_2(3 - 2x_2 - 4y_2) + y_1 - y_2 = 0, \\ x_1 \geq 0, \qquad & x_2 \geq 0, \qquad y_1 \geq 0, \qquad y_2 \geq 0. \end{aligned}$$

EXAMPLE 6. (Solving geometric constraints)

$$\begin{aligned} f_1 &= \frac{1}{100} - 4s(s-1)(s-b)(s-c) = 0, \\ f_2 &= \frac{1}{5} - bc = 0, \\ f_3 &= 2s - 1 - b - c = 0, \end{aligned}$$

$$b > 0, \qquad c > 0, \qquad b + c - 1 > 0, \qquad 1 + c - b > 0, \qquad 1 + b - c > 0.$$

EXAMPLE 7.

$$\begin{aligned} h_1 &= x^2 + y^2 - xy - 1 = 0, \\ h_2 &= y^2 + z^2 - yz - a^2 = 0, \\ h_3 &= z^2 + x^2 - zx - b^2 = 0, \\ h_4 &= a^2 - 1 + b - b^2 = 0, \\ h_5 &= 3b^6 + 56b^4 - 122b^3 + 56b^2 + 3 = 0, \end{aligned}$$

$$x > 0, \qquad y > 0, \qquad z > 0, \qquad a - 1 \geq 0, \qquad b - a \geq 0, \qquad a + 1 - b > 0,$$

which is a special case of the "P3P" problem in computer vision.

REFERENCES

1. D. Wang, A class of cubic differential systems with 6-tuple focus, *J. Differential Equations* **87**, 305–315, (1990).
2. E.M. James and N.G. Lloyd, A cubic system with eight small-amplitude limit cycles, *IMA J. of Appl. Math.* **47**, 163–171, (1991).
3. S. Ning, S. Ma, K.H. Kwek and Z. Zheng, A cubic system with eight small-amplitude limit cycles, *Appl. Math. Lett.* **7** (4), 23–27, (1994).
4. D. Angeli, J.E. Ferrell, Jr. and E.D. Sontag, Detection of multistability, bifurcations, and hysteresis in a large class of biological positive-feedback systems, *Proc. Nat. Acad. Sci. USA* **101**, 1822–1827, (2004).
5. D. Wang and B. Xia, Stability analysis of biological systems with real solution classification, In *Proc. ISSAC 2005*, (Edited by M. Kauers), pp. 354–361, ACM Press, New York, (2005).
6. B.C. Xia and L. Yang, An algorithm for isolating the real solutions of semi-algebraic systems, *J. Symb. Comput.* **34**, 461–477, (2002).
7. P. Pedersen, M.-F. Roy and A. Szpirglas, Counting real zeros in the multivariate case, In *Computational Algebraic Geometry*, (Edited by F. Eyssette and A. Galligo), pp. 203–224, Birkhäuser, Basel, (1993).
8. L. Gonzalez-Vega, F. Rouillier, M.-F. Roy and G. Trujillo, Symbolic recipes for real solutions, In *Some Tapas of Computer Algebra*, (Edited by A. M. Cohen, H. Cuypers and H. Sterk), pp. 121–167, Springer-Verlag, Berlin, (1999).
9. G.E. Collins, J.R. Johnson and W. Krandick, Interval arithmetic in cylindrical algebraic decomposition, *J. Symb. Comput.* **34**, 145–157, (2002).
10. G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, New York, (1983).
11. D.R. Wang, L.S. Zhang and N.Y. Deng, *Interval Algorithms for NonLinear Equations*, (in Chinese), Shanghai Science and Technology Press, Shanghai, (1983).
12. L. Yang, J.Z. Zhang and X.R. Hou, A criterion of dependency between algebraic equations and its applications, In *Proceedings of International Workshop on Mathematics Mechanization 1992*, (Edited by W.-T. Wu and M.-D. Cheng), pp. 110–134, International Academic Publishers, Beijing, (1992).
13. M. Kalkbrener, A generalized Euclidean algorithm for computing triangular representations of algebraic varieties, *J. Symb. Comput.* **15**, 143–167, (1993).
14. D. Wang, Computing triangular systems and regular systems, *J. Symb. Comput.* **30**, 221–236, (2000).
15. G.E. Collins and R. Loos, Real zeros of polynomials, In *Computer Algebra: Symbolic and Algebraic Computation*, (Edited by B. Buchberger, G.E. Collins and R. Loos), pp. 83–94, Springer-Verlag, New York, (1982).