Linear Algebra and its Applications 439 (2013) 3016–3030

Contents lists available at ScienceDirect

Linear Algebra and its Applications

www.elsevier.com/locate/laa

Zero–nonzero and real–nonreal sign determination

Daniel Perrucci [a,b,*,1], Marie-Françoise Roy [c]

[a] Departamento de Matemática, FCEN, Universidad de Buenos Aires, Ciudad Universitaria, 1428 Buenos Aires, Argentina
[b] CONICET, Argentina
[c] IRMAR (UMR CNRS 6625), Université de Rennes 1, Campus de Beaulieu, 35042 Rennes cedex, France

ARTICLE INFO

Article history:
Received 17 May 2013
Accepted 4 September 2013
Available online 25 September 2013
Submitted by R. Brualdi

MSC:
14P10
14Q20

Keywords:
Polynomial equations and inequations systems
Sign determination
Complexity

ABSTRACT

We consider first the zero–nonzero determination problem, which consists in determining the list of zero–nonzero conditions realized by a finite list of polynomials on a finite set $Z \subset \mathrm{C}^k$ with C an algebraic closed field. We describe an algorithm to solve the zero–nonzero determination problem and we perform its bit complexity analysis. This algorithm, which is in many ways an adaptation of the methods used to solve the more classical sign determination problem, presents also new ideas which can be used to improve sign determination. Then, we consider the real–nonreal sign determination problem, which deals with both the sign determination and the zero–nonzero determination problem. We describe an algorithm to solve the real–nonreal sign determination problem, we perform its bit complexity analysis and we discuss this problem in a parametric context.

1. Introduction

Let L be a field and C an algebraically closed extension of L. Consider a finite set $Z \subset \mathrm{C}^k$ and a finite list of polynomials $\mathcal{P} = P_1, \ldots, P_s$ in $\mathrm{L}[X_1, \ldots, X_k]$; the zero–nonzero determination problem is the problem of computing the zero–nonzero conditions of $\mathcal{P}$ which are realized on Z. In order to better explain this, we introduce some notation and definitions.

* Corresponding author.
[1] Partially supported by the following grants: PIP 099/11 CONICET and UBACYT 20020090100069 (2010/2012).

0024-3795/$ – see front matter
http://dx.doi.org/10.1016/j.laa.2013.09.010

For $a \in \mathrm{C}$, its **invertibility** is defined as follows:

$$\begin{cases} \mathrm{inv}(a) = 0 & \text{if } a = 0, \\ \mathrm{inv}(a) = 1 & \text{if } a \neq 0. \end{cases}$$

Let $I \subset \{1, \ldots, s\}$. Given a **zero–nonzero condition** $\sigma \in \{0,1\}^I$, the **realization of** σ **on** Z is

$$\mathrm{Reali}(\sigma, Z) = \left\{ x \in Z \;\middle|\; \bigwedge_{i \in I} \mathrm{inv}\big(P_i(x)\big) = \sigma(i) \right\}$$

and we denote by $c(\sigma, Z)$ the cardinal of $\mathrm{Reali}(\sigma, Z)$. We write $\mathrm{Feas}(\mathcal{P}, Z)$ for the list of $\sigma \in \{0,1\}^{\{1,\ldots,s\}}$ such that $c(\sigma, Z)$ is not zero, and $c(\mathcal{P}, Z)$ for the corresponding list of cardinals. The **zero–nonzero determination problem** is to determine $\mathrm{Feas}(\mathcal{P}, Z)$ and $c(\mathcal{P}, Z)$.

Typically, the set Z is not known explicitly, but given as the complex zero set of a polynomial system; therefore, to solve the zero–nonzero determination problem it is not possible to simply evaluate the polynomials in $\mathcal{P}$ separately at each point of Z, and a more clever strategy is needed.

The zero–nonzero determination problem is analogous to the more classical sign determination problem, which we recall now. Let K be an ordered field and R a real closed extension of K. For $a \in \mathrm{R}$, its **sign** is defined as follows:

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

Consider a finite set $W \subset \mathrm{R}^k$ and a finite list of polynomials $\mathcal{P} = P_1, \ldots, P_s$ in $\mathrm{K}[X_1, \ldots, X_k]$. Given a **sign condition** $\tau \in \{0, 1, -1\}^{\{1,\ldots,s\}}$, the **realization of** τ **on** W is

$$\mathrm{Reali}_{\mathrm{sign}}(\tau, W) = \left\{ x \in W \;\middle|\; \bigwedge_{i=1,\ldots,s} \mathrm{sign}\big(P_i(x)\big) = \tau(i) \right\}$$

and we denote by $c_{\mathrm{sign}}(\tau, W)$ the cardinal of $\mathrm{Reali}_{\mathrm{sign}}(\tau, W)$. We write $\mathrm{Feas}_{\mathrm{sign}}(\mathcal{P}, W)$ for the list of $\tau \in \{0, 1, -1\}^{\{1,\ldots,s\}}$ such that $c_{\mathrm{sign}}(\tau, W)$ is not zero, and $c_{\mathrm{sign}}(\mathcal{P}, W)$ for the corresponding list of cardinals. The **sign determination problem** is to determine $\mathrm{Feas}_{\mathrm{sign}}(\mathcal{P}, W)$ and $c_{\mathrm{sign}}(\mathcal{P}, W)$. Once again, the set W is typically not known explicitly, but given as the real zero set of a polynomial system.

Let $Q \in \mathrm{K}[X_1, \ldots, X_k]$, the **Tarski-query** of Q for W is

$$\mathrm{TaQu}(Q, W) = \sum_{x \in W} \mathrm{sign}\big(Q(x)\big) = \mathrm{card}\big(\{x \in W \mid Q(x) > 0\}\big) - \mathrm{card}\big(\{x \in W \mid Q(x) < 0\}\big).$$

Tarski-queries play a leading role in the most efficient algorithms to solve the sign determination problem [7,3,1]. In fact, these algorithms consist in computing a relevant list of Tarski-queries and solving linear systems with integer coefficients having a specific structure. Suppose that the polynomial system defining W is in $\mathrm{K}[X_1, \ldots, X_k]$. Then, in the mentioned algorithms for sign determination there are three different kind of operations:

- sign comparisons,
- operations in $\mathbb{Q}$, which appear in the linear solving steps,
- operations in K, which appear in the Tarski-query computations.

Regarding the complexity analysis, in [1, Chapter 10] the Tarski-query computation is considered as a blackbox. Indeed, there exist many well-known methods to compute them, and depending on the setting, the application of one method or another is convenient. However, when asymptotically fast methods for computing the Tarski-query are used in the univariate case, the cost of solving the linear system dominates the overall complexity as already noticed in [3, Section 3.3]. In [5], a method for solving the specific linear systems arising in the sign determination algorithm is given, leading to a complexity improvement (see [5, Corollary 2]).

The real–nonreal sign determination problem is a compressed way of dealing with both the sign determination and the zero–nonzero determination problem as we explain now. Let K be an ordered field, R a real closed extension of K and $C = R[i]$. Consider a finite set $Z \subset C^k$, one more time, typically given as the complex zero set of polynomial system, and a finite list of polynomials $\mathcal{P} = P_1, \ldots, P_s$ in $K[X_1, \ldots, X_k]$. We define

$$Z_R = Z \cap R^k,$$
$$Z_{C\setminus R} = Z \cap \left(C^k \setminus R^k\right).$$

The **real–nonreal sign determination problem** is to determine $\mathrm{Feas}_{\mathrm{sign}}(\mathcal{P}, Z_R)$, $\mathrm{Feas}(\mathcal{P}, Z_{C\setminus R})$, $c_{\mathrm{sign}}(\mathcal{P}, Z_R)$ and $c(\mathcal{P}, Z_{C\setminus R})$. By solving the real–nonreal sign determination problem we obtain a complete description of the sign and invertibility of the polynomials in $\mathcal{P}$ on Z.

This paper serves several purposes. First, we give an algorithm for zero–nonzero determination. This algorithm is based on the same principles that sign determination, the role of Tarski-queries being played by invertibility-queries, which we introduce now. Let $Q \in L[X_1, \ldots, X_k]$, the **invertibility-query** of Q for Z is

$$\mathrm{Qu}(Q, Z) = \sum_{x\in Z} \mathrm{inv}\big(Q(x)\big) = \mathrm{card}\big(\{x \in Z \mid Q(x) \neq 0\}\big).$$

Then we perform a complexity analysis of this algorithm as follows: we consider the invertibility-queries as a blackbox, and estimate the *bit complexity* of the zero–nonzero comparisons and the operations in $\mathbb{Z}$. Note that the algorithm we present here is not a straightforward adaptation of the known algorithms for sign determination. In fact, in order to obtain a good bit complexity bound, we introduce some new definitions (i.e. the compression and a matrix summarizing the useful information) which can be used in turn to improve the known algorithms for sign determination (see [2, Chapter 10]).

Then, combining sign and zero–nonzero determination, we give an algorithm for real–nonreal sign determination and perform a complexity analysis in a similar way. A final purpose of this paper is to discuss real–nonreal sign determination in a parametric context.

This paper is organized as follows. In Section 2 we give an algorithm for zero–nonzero determination and perform its bit complexity analysis considering the invertibility-query as a blackbox. In Section 3 we give an algorithm for real–nonreal sign determination and perform its bit complexity analysis in a similar way. In Section 4 we explain the various existing methods for computing the Tarski-queries and invertibility-queries in the univariate and multivariate case and we deduce the bit complexity of zero–nonzero and real–nonreal sign determination in the univariate case. Finally, in Section 5 we discuss real–nonreal sign determination in a parametric context.

2. The zero–nonzero determination problem

We recall that L is a field and C an algebraically closed extension of L, $Z \subset C^k$ a finite set, $\mathcal{P} = P_1, \ldots, P_s$ a list of polynomials in $L[X_1, \ldots, X_k]$, I a subset of $\{1, \ldots, s\}$ and $\sigma \in \{0, 1\}^I$ a zero–nonzero condition.

2.1. Definitions and properties

Given $J \subset I \subset \{1, \ldots, s\}$ and $\sigma \in \{0, 1\}^I$, we write $\mathcal{P}^J$ for $\prod_{i\in J} P_i$ and σ^J for $\prod_{i\in J} \sigma(i)$. Note that when $\mathrm{Reali}(\sigma, Z) \neq \emptyset$, the value of $\mathrm{inv}(\mathcal{P}^J(x))$ is fixed as x varies in $\mathrm{Reali}(\sigma, Z)$ and is equal to σ^J. By convention $\mathcal{P}^\emptyset = 1$, $\sigma^\emptyset = 1$, and $\{0, 1\}^\emptyset = \{\emptyset\}$.

For a fixed $I \subset \{1, \ldots, s\}$, we consider the lexicographical order on $\{0, 1\}^I$ (with $0 < 1$), identifying a zero–nonzero condition in $\{0, 1\}^I$ with a bit string of length $\mathrm{card}(I)$. Throughout this paper, all the lists of zero–nonzero conditions we consider are ordered and have no repetitions. By the union of two disjoint lists we mean their ordered union.

Given $I \subset \{1, \ldots, s\}$ and a list $A = [I_1, \ldots, I_m]$ of subsets of I, we define $\mathrm{Qu}(\mathcal{P}^A, Z)$ as the vector with coordinates $\mathrm{Qu}(\mathcal{P}^{I_1}, Z), \ldots, \mathrm{Qu}(\mathcal{P}^{I_m}, Z)$. Also, given a list $\Sigma = [\sigma_1, \ldots, \sigma_n]$ of zero–nonzero conditions in $\{0,1\}^I$, we define $c(\Sigma, Z)$ as the vector with coordinates $c(\sigma_1, Z), \ldots, c(\sigma_n, Z)$. The **matrix of A on Σ** is the $m \times n$ matrix $\mathrm{Mat}(A, \Sigma)$ whose i, j-th entry is $\sigma_j^{I_i}$ for $i = 1, \ldots, m,\ j = 1, \ldots, n$. By convention $\mathrm{Mat}(\emptyset, \emptyset) = \emptyset$ (i.e. the empty matrix with 0 rows and columns) and is invertible.

Let $\mathcal{P}_I = \{P_i \mid i \in I\}$. With this notation, we have the following:

Proposition 1. *Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$ with $\mathrm{Feas}(\mathcal{P}_I, Z) \subset \Sigma$. Then*

$$\mathrm{Mat}(A, \Sigma) \cdot c(\Sigma, Z) = \mathrm{Qu}\left(\mathcal{P}_I^A, Z\right).$$

Proof. The claim is easy if Z has a single element. Indeed, suppose $Z = \{x\}$, and σ_j is the zero–nonzero condition in Σ satisfied by $\mathcal{P}_I$ at x. The coordinates of $c(\Sigma, \{x\})$ are 0 except at place j, where it is 1, so that $\mathrm{Mat}(A, \Sigma) \cdot c(\Sigma, \{x\})$ is the j-th column of $\mathrm{Mat}(A, \Sigma)$. Its i-th coordinate is $\sigma_j^{I_i} = \mathrm{inv}(\mathcal{P}_I^{I_i}(x)) = \mathrm{Qu}(\mathcal{P}_I^{I_i}, \{x\})$.

In the general case, the claim follows by linearity since $c(\Sigma, Z) = \sum_{x \in Z} c(\Sigma, \{x\})$ and $\mathrm{Qu}(\mathcal{P}_I^A, Z) = \sum_{x \in Z} \mathrm{Qu}(\mathcal{P}_I^A, \{x\})$. □

Example 2. When $I = \{i\} \subset \{1, \ldots, s\}$, $A = [\emptyset, \{i\}]$, and $\Sigma = [0, 1]$, the conclusion of Proposition 1 is

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} c(P_i = 0, Z) \\ c(P_i \neq 0, Z) \end{pmatrix} = \begin{pmatrix} \mathrm{Qu}(1, Z) \\ \mathrm{Qu}(P_i, Z) \end{pmatrix}.$$

It follows from Proposition 1 that if $\Sigma \subset \{0,1\}^{\{1,\ldots,s\}}$ contains $\mathrm{Feas}(\mathcal{P}, Z)$ and the matrix $\mathrm{Mat}(A, \Sigma)$ is invertible, we can compute $c(\Sigma, Z)$ from $\mathrm{Qu}(\mathcal{P}^A, Z)$. Therefore, if we take A as the list of the 2^s subsets of $\{1, \ldots, s\}$ and Σ as the list of the 2^s zero–nonzero conditions in $\{0,1\}^{\{1,\ldots,s\}}$, since it is easy to check that $\mathrm{Mat}(A, \Sigma)$ is invertible, we have a naive method for zero–nonzero determination: we solve the $2^s \times 2^s$ linear system from Proposition 1 and we discard the zero–nonzero conditions σ such that $c(\sigma, Z) = 0$. This naive method involve an exponential number of invertibility queries and solving a linear system of exponential size. We want an algorithm with a better complexity bound.

The key fact is to take into account that the number of realizable zero–nonzero conditions does not exceed $\mathrm{card}(Z)$. We are going to consider one by one the polynomials $P_1, \ldots, P_s$ in the list $\mathcal{P}$ and to compute at step i the realizable zero–nonzero conditions for the list $\mathcal{P}_i := P_1, \ldots, P_i$, determining the nonempty zero–nonzero conditions inductively and getting rid of the empty ones at each step. In this way, the size of the data we manipulate is well controlled.

We need some preliminary definitions and results.

Definition 3. Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$. A set A of subsets of I is **adapted to zero–nonzero determination on Σ** if the matrix of A on Σ is invertible.

Note that the set A has to be ordered into a list so that the matrix of A on Σ is unambiguous, but the choice of this ordering does not change the fact that the matrix is invertible.

Example 4.

1. Consider $I = \emptyset$ and $\Sigma = [\emptyset]$, then $A = \{\emptyset\}$ is adapted to zero–nonzero determination on Σ since $\mathrm{Mat}(A, \Sigma) = (1)$ is invertible.
2. Consider $I = \{i\} \subset \{1, \ldots, s\}$, then:
   - if $\Sigma = [0, 1]$, $A = \{\emptyset, \{i\}\}$ is adapted to zero–nonzero determination on Σ, since

     $$\mathrm{Mat}(A, \Sigma) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

     is invertible,

- if $\Sigma = [0]$ or $\Sigma = [1]$, $A = \{\emptyset\}$ is adapted to zero–nonzero determination on Σ, since $\mathrm{Mat}(A, \Sigma) = (1)$ is invertible.

Let $I \subset \{1, \ldots, s\}$. Our aim is to describe a method for determining for each $\Sigma \subset \{0,1\}^I$, a set A of subsets of I adapted to zero–nonzero determination on Σ. First, we introduce some more definitions and notation.

Definition 5. If $J \subset I \subset \{1, \ldots, s\}$, $\sigma \in \{0,1\}^J$ is the **restriction** of $\tau \in \{0,1\}^I$ if $\sigma(j) = \tau(j)$ for every $j \in J$; we also say that τ is an **extension** of σ. If $\Sigma \subset \{0,1\}^I$, we denote by $\Sigma_J \subset \{0,1\}^J$ the list of restrictions of elements of Σ to J.

Notation 6. For a set A of subsets of $J \subset \{1, \ldots, s\}$ and $j \in \{1, \ldots, s\}$ bigger than $\max(J)$, we denote by (A, j) the set of subsets of $J \cup \{j\}$ obtained by adding j to all the elements of A.

We are now ready to construct a set of subsets adapted to sign determination.

Definition 7 *(Adapted family).* Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$. The **adapted family** $\mathrm{Ada}(\Sigma)$ is defined by induction as follows:

- If $I = \emptyset$, then, if $\Sigma = \emptyset$, define $\mathrm{Ada}(\Sigma) = \emptyset$, if $\Sigma = [\emptyset]$, define $\mathrm{Ada}(\Sigma) = \{\emptyset\}$.
- If $I \neq \emptyset$, consider $i = \max(I)$, $I' = I \setminus \{i\}$, $\Xi = \Sigma_{I'}$ and Ξ' the list of elements of Ξ having two different extensions in Σ. Define

$$\mathrm{Ada}(\Sigma) = \mathrm{Ada}(\Xi) \cup \big(\mathrm{Ada}\big(\Xi'\big), i\big).$$

From the previous definition it is easy to prove that for $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$, $\mathrm{card}(\mathrm{Ada}(\Sigma)) = \mathrm{card}(\Sigma)$. Before proving that the adapted family $\mathrm{Ada}(\Sigma)$ we defined is adapted to zero–nonzero determination on Σ, we prove some auxiliary results.

Lemma 8. *Let $I \subset \{1, \ldots, s\}$ and $\Sigma_1 \subset \Sigma \subset \{0,1\}^I$, then* $\mathrm{Ada}(\Sigma_1) \subset \mathrm{Ada}(\Sigma)$.

Proof. We prove the claim by induction on $\mathrm{card}(I)$. If $I = \emptyset$, the claim is true. Suppose now $I \neq \emptyset$. Following the notation in Definition 7, and noting that $\Xi_1 = (\Sigma_1)_{I'} \subset \Xi$ and $\Xi_1' \subset \Xi'$, where $\Xi_1' \subset \{0,1\}^{I'}$ is the list of elements of Ξ_1 having two different extensions in Σ_1, the claim follows using twice the induction hypothesis. □

The following result will be useful for the complexity analysis.

Proposition 9. *Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$. For every $J \in \mathrm{Ada}(\Sigma)$, $\mathrm{card}(J) < \mathrm{bit}(\mathrm{card}(\Sigma))$.*

Proof. We will prove by induction on $\mathrm{card}(I)$ that for every $J \in \mathrm{Ada}(\Sigma)$ all the subsets of J belong to $\mathrm{Ada}(\Sigma)$. The proposition follows since $2^{\mathrm{card}(J)} \leqslant \mathrm{card}(\mathrm{Ada}(\Sigma)) = \mathrm{card}(\Sigma)$. If $I = \emptyset$, the claim is true. Suppose now $I \neq \emptyset$. Taking $i = \max(I)$, we consider two cases: if $i \notin J$ then $J \in \mathrm{Ada}(\Xi)$, if $i \in J$, then $J \setminus \{i\} \in \mathrm{Ada}(\Xi')$. Therefore, using twice the induction hypothesis, all the subsets of J not containing i belong to $\mathrm{Ada}(\Xi) \subset \mathrm{Ada}(\Sigma)$ and all the subsets of J containing i belong to $(\mathrm{Ada}(\Xi'), i) \subset \mathrm{Ada}(\Sigma)$. □

Proposition 10. *Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{1,0\}^I$. The set* $\mathrm{Ada}(\Sigma)$ *is adapted to zero–nonzero determination on Σ.*

We adapt the proof in [5, Proposition 6], which provides information about the inverse of $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ useful later for algorithmic matters.

Proof. First, we define the total order $\prec$ on the subsets of $\{1,\ldots,s\}$, which we use whenever we have to order a set of subsets into a list. Given $J \subset \{1,\ldots,s\}$, we associate the natural number $|J| = \sum_{j\in J} 2^{j-1}$ and then we define for $J_1, J_2 \subset \{1,\ldots,s\}$, $J_1 \prec J_2$ if $|J_1| < |J_2|$. Note that $\prec$ extends the partial order of inclusion of subsets.

We will prove by induction on $\mathrm{card}(I)$ that $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ is invertible. If $I = \emptyset$, then the claim is true. Suppose now $I \neq \emptyset$. We follow the notation in Definition 7 and divide Σ in three (possibly empty) sublists:

- Σ_0 consisting of the elements σ of Σ such that $\sigma(i) = 0$ and the restriction of σ to I' is in Ξ',
- Σ_1 consisting of the elements σ of Σ such that $\sigma(i) = 1$ and the restriction of σ to I' is in Ξ',
- $\Sigma_\star$ consisting of the elements of Σ whose restriction to I' is in $\Xi \setminus \Xi'$.

If $\Xi' = \emptyset$ is empty, $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma) = \mathrm{Mat}(\mathrm{Ada}(\Xi), \Xi)$ is invertible by induction hypothesis.

If $\Xi' \neq \emptyset$, we reorder columns in $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ so that the columns corresponding to zero–nonzero conditions in $\Sigma_0 \cup \Sigma_\star$ appear first. Then, $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ gets the following structure

$$M = \left(\begin{array}{c|c} M_{1,1} & M_{1,2} \\ \hline M_{2,1} & M_{2,2} \end{array}\right) \tag{1}$$

with

$$\begin{aligned} M_{1,1} &= \mathrm{Mat}\big(\mathrm{Ada}(\Xi), \Xi\big), \\ M_{1,2} &= \mathrm{Mat}\big(\mathrm{Ada}(\Xi), \Xi'\big), \\ M_{2,1} &= \mathrm{Mat}\big(\big(\mathrm{Ada}\big(\Xi'\big), i\big), \Sigma_0 \cup \Sigma_\star\big), \\ M_{2,2} &= \mathrm{Mat}\big(\mathrm{Ada}\big(\Xi'\big), \Xi'\big). \end{aligned}$$

It is easy to see that $M_{1,2}$ equals the submatrix of $M_{1,1}$ composed by the columns corresponding to zero–nonzero conditions in Σ_0 and also that all the columns in $M_{2,1}$ corresponding to zero–nonzero conditions in Σ_0 are 0.

Suppose, by induction hypothesis, that $\mathrm{Mat}(\mathrm{Ada}(\Xi), \Xi)$ and $\mathrm{Mat}(\mathrm{Ada}(\Xi'), \Xi')$ are invertible. We invert $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ using a Gaussian elimination method by block, i.e. multiplying to the left by block elementary matrices. We call $\tilde{\mathrm{Id}}$ the submatrix of the identity matrix with columns indexed by the list $\Sigma_0 \cup \Sigma_\star$ composed by the columns corresponding to zero–nonzero conditions in Σ_0. Taking into account that $M_{2,1} \cdot \tilde{\mathrm{Id}} = 0$, it is easy to check that

$$\left(\begin{array}{c|c} \mathrm{Id} & -\tilde{\mathrm{Id}} \\ \hline 0 & \mathrm{Id} \end{array}\right)\left(\begin{array}{c|c} \mathrm{Id} & 0 \\ \hline 0 & M_{2,2}^{-1} \end{array}\right)\left(\begin{array}{c|c} \mathrm{Id} & 0 \\ \hline -M_{2,1} & \mathrm{Id} \end{array}\right)\left(\begin{array}{c|c} M_{1,1}^{-1} & 0 \\ \hline 0 & \mathrm{Id} \end{array}\right)$$

is the inverse of the matrix M defined in (1). Therefore $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ is invertible as we wanted to prove. □

A last key observation which leads to a well controlled size of the data we manipulate is the following. Let $\mathcal{P}_i = P_1, \ldots, P_i$ and suppose that we compute inductively $\mathrm{Feas}(\mathcal{P}_i, Z)$ for $i = 0, \ldots, s$. Since $1 = \mathrm{card}(\mathrm{Feas}(\mathcal{P}_0, Z))$ and $\mathrm{card}(\mathrm{Feas}(\mathcal{P}_s, Z)) = \mathrm{card}(\mathrm{Feas}(\mathcal{P}, Z)) \leqslant \mathrm{card}(Z)$, in the sequence

$$\mathrm{card}\big(\mathrm{Feas}(\mathcal{P}_0, Z)\big) \leqslant \cdots \leqslant \mathrm{card}\big(\mathrm{Feas}(\mathcal{P}_i, Z)\big) \leqslant \cdots \leqslant \mathrm{card}\big(\mathrm{Feas}(\mathcal{P}_s, Z)\big)$$

we have at most $\mathrm{card}(Z)-1$ places where a strict inequality holds. For $i=1,\ldots,s$ such that

$$\mathrm{card}\big(\mathrm{Feas}(\mathcal{P}_{i-1},Z)\big)=\mathrm{card}\big(\mathrm{Feas}(\mathcal{P}_i,Z)\big),$$

we have that every zero–nonzero condition in $\mathrm{Feas}(\mathcal{P}_{i-1},Z)$ can be extended to a zero–nonzero condition in $\mathrm{Feas}(\mathcal{P}_i,Z)$ in only one way. Once this information is known, we have that for each point in Z, the invertibility of P_i is determined from the invertibility of the polynomials in $\mathcal{P}_{i-1}$ at this point. With this remark in mind, we introduce the following definitions which will be useful in Algorithm `Zero-nonzero Determination` and its complexity analysis.

Definition 11 *(Compressed set of indices).* Let $I\subset\{1,\ldots,s\}$ and $\Sigma\subset\{1,0\}^I$, $\Sigma\neq\emptyset$. The **compressed set of indices** $\mathrm{comp}(\Sigma)$ is defined by induction as follows:

- If $I=\emptyset$ define $\mathrm{comp}(\Sigma)=\emptyset$.
- If $I\neq\emptyset$, consider $i=\max(I)$, $I'=I\setminus\{i\}$ and $\Xi=\Sigma_{I'}$; then
  - if $\mathrm{card}(\Sigma)=\mathrm{card}(\Xi)$, define $\mathrm{comp}(\Sigma)=\mathrm{comp}(\Xi)$,
  - if $\mathrm{card}(\Sigma)>\mathrm{card}(\Xi)$, define $\mathrm{comp}(\Sigma)=\mathrm{comp}(\Xi)\cup\{i\}$.

We define also the **compressed list of zero–nonzero conditions**, $\mathrm{Comp}(\Sigma)=\Sigma_{\mathrm{comp}(\Sigma)}$.

Example 12. If $I=\{1,2,3,4,5\}$ and Σ is the list of zero–nonzero conditions

$$[10110,\ 10111,\ 11011,\ 11100,\ 11101]$$

then $\mathrm{comp}(\Sigma)$ is $\{2,3,5\}$ and $\mathrm{Comp}(\Sigma)$ is

$$[010,\ 011,\ 101,\ 110,\ 111].$$

Remark 13. Let $I\subset\{1,\ldots,s\}$ and $\Sigma\subset\{0,1\}^I$. Following Definition 7 and Definition 11, it is easy to prove by induction in $\mathrm{card}(I)$ that $\mathrm{Ada}(\Sigma)=\mathrm{Ada}(\mathrm{Comp}(\Sigma))$.

2.2. Algorithms and complexity

Given $I\subset\{1,\ldots,s\}$ and $\Sigma\subset\{0,1\}^I$, we represent Σ with a $\mathrm{card}(\Sigma)\times\mathrm{card}(I)$ matrix filled with 0 and 1, each row representing a zero–nonzero condition in $\{0,1\}^I$. We consider this representation even when $\Sigma=\emptyset$ or $I=\emptyset$. Similarly, we represent a list A of subsets of I with a $\mathrm{card}(A)\times\mathrm{card}(I)$ matrix filled with 0 and 1, each row representing an element of A, and the bit 0 (resp. 1) in each column indicating that the corresponding element does not belong (resp. belongs) to the subset of I. When we speak of Σ (resp. A) we mean either the list Σ (resp. A) or its representation as a matrix as convenient.

For a matrix M of size $m\times n$, given ordered lists of integers with no repetitions ℓ and ℓ', with the entries of ℓ (resp. ℓ') in $\{1,\ldots,m\}$ (resp. $\{1,\ldots,n\}$), we denote by $M(\ell,\ell')$ the submatrix of M obtained by extracting from M the rows in ℓ and the columns in ℓ'. We use this notation even when one of the lists ℓ and ℓ' is empty. For a vector v of size m, analogously we denote by $v(\ell)$ the subvector formed by the entries with index in ℓ.

Up to the end of the subsection, we follow the notation in Definition 7 and Proposition 10. We introduce an auxiliary definition.

Definition 14. Let $I\subset\{1,\ldots,s\}$ and $\Sigma\subset\{0,1\}^I$. The matrix $\mathrm{Info}(\Sigma)$ is defined by induction as follows:

- If $I=\emptyset$ define $\mathrm{Info}(\Sigma)$ as the matrix with as many rows as elements in Σ (possibly 0 or 1) and 0 columns.

- If $I \neq \emptyset$, we consider the list ℓ_0 (resp. ℓ_1, $\ell_\star$) formed by the indices of zero–nonzero conditions in Σ which belong to Σ_0 (resp. Σ_1, $\Sigma_\star$) and define $\mathrm{Info}(\Sigma)$ as follows:

$$\begin{cases} \mathrm{Info}(\Sigma)(\ell_0 \cup \ell_\star, [1, \ldots, \mathrm{card}(I) - 1]) = \mathrm{Info}(\Xi), & \\ \mathrm{Info}(\Sigma)(\ell_1, [1, \ldots, \mathrm{card}(I) - 1]) = \mathrm{Info}(\Xi'), & \\ \mathrm{Info}(\Sigma)(j, \mathrm{card}(I)) = 0 & \text{if } j \in \ell_0, \\ \mathrm{Info}(\Sigma)(j, \mathrm{card}(I)) = 1 & \text{if } j \in \ell_1, \\ \mathrm{Info}(\Sigma)(j, \mathrm{card}(I)) = \star & \text{if } j \in \ell_\star. \end{cases}$$

Example 15. Continuing Example 12, $\mathrm{Info}(\Sigma)$ is

$$\begin{pmatrix} \star & 0 & \star & \star & 0 \\ \star & 0 & \star & \star & 1 \\ \star & 1 & 0 & \star & \star \\ \star & \star & 1 & \star & 0 \\ \star & 1 & \star & \star & 1 \end{pmatrix}.$$

The availability of the information provided by the matrix $\mathrm{Info}(\Sigma)$ is very important to obtain the complexity bound in the algorithms in this subsection. We consider then the following auxiliary technical algorithm.

Algorithm `Get Info`

- **Input:** A list $\Sigma \subset \{0, 1\}^I$ with $I \subset \{1, \ldots, s\}$.
- **Output:** The matrix $\mathrm{Info}(\Sigma)$.

It is easy to give a procedure for Algorithm `Get Info` with bit complexity $O(\mathrm{card}(\Sigma)\mathrm{card}(I)^2)$. Since the list Σ is ordered, this procedure takes $O(\mathrm{card}(\Sigma)\mathrm{card}(I))$ bit operations to compute the lists ℓ_0, ℓ_1 and $\ell_\star$ and then does recursive calls to itself to compute the matrices $\mathrm{Info}(\Xi)$ and $\mathrm{Info}(\Xi')$.

The following algorithm computes the adapted family for a given list of zero–nonzero conditions.

Algorithm `Adapted Family`

- **Input:** A list $\Sigma \subset \{0, 1\}^I$ with $I \subset \{1, \ldots, s\}$ and the matrix $\mathrm{Info}(\Sigma)$.
- **Output:** The set $\mathrm{Ada}(\Sigma)$ as a list.
- **Procedure:** Extract from each row of $\mathrm{Info}(\Sigma)$ the subset of indices with entry 1.

Lemma 16. *Given a list $\Sigma \subset \{1, 0\}^I$ with $I \subset \{1, \ldots, s\}$ and the matrix $\mathrm{Info}(\Sigma)$, Algorithm* `Adapted Family` *computes the set $\mathrm{Ada}(\Sigma)$ as a list. The bit complexity of this algorithm is $O(\mathrm{card}(\Sigma)\mathrm{card}(I))$.*

Proof. The correctness of the algorithm follows from Definition 7 and Definition 14. The bound on the bit complexity is clear since $\mathrm{Info}(\Sigma)$ is read once. □

Example 17. Continuing Example 15, the representation of $\mathrm{Ada}(\Sigma)$ as a list obtained by Algorithm `Adapted Family` is

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Finally $\mathrm{Ada}(\Sigma) = \{\emptyset, \{5\}, \{2\}, \{3\}, \{2, 5\}\}$.

From now on, for every $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{0,1\}^I$, we considered the set $\mathrm{Ada}(\Sigma)$ ordered into a list as obtained by Algorithm `Adapted Family`.

We give now a specific method for solving the linear systems arising in the algorithm for zero–nonzero determination.

Algorithm `Linear Solving`

- **Input:** A list $\Sigma \subset \{0,1\}^I$ with $I \subset \{1, \ldots, s\}$, the matrices $\mathrm{Info}(\Sigma)$ and $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ and an integer vector v of size $\mathrm{card}(\Sigma)$.
- **Output:** The vector $c = \mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)^{-1} v$.
- **Procedure:**
  1. If $I = \emptyset$ output $c = v$. So from now we suppose $I \neq \emptyset$.
  2. Extract from $\mathrm{Info}(\Sigma)$ the lists ℓ_0, ℓ_1 and $\ell_\star$ (cf. Definition 14), define $\ell = \ell_0 \cup \ell_\star$, $m^0 = \mathrm{card}(\ell_0)$ and $m^\star = \mathrm{card}(\ell_\star)$.
  3. Compute $t(\ell) = \mathrm{Mat}(\mathrm{Ada}(\Xi), \Xi)^{-1} v(\ell)$, doing a recursive call to Algorithm `Linear Solving`.
  4. If $m^0 \neq 0$ and $m^\star \neq 0$, compute $t(\ell_1) = -\mathrm{Mat}((\mathrm{Ada}(\Xi'), i), \Sigma_\star) t(\ell_\star) + v(\ell_1)$.
  5. If $m^0 \neq 0$:
     (a) Compute $c(\ell_1) = \mathrm{Mat}(\mathrm{Ada}(\Xi'), \Xi')^{-1} t(\ell_1)$, doing a recursive call to Algorithm `Linear Solving`.
     (b) Compute $c(\ell_0) = t(\ell_0) - t(\ell_1)$.
     (c) Define $c(\ell_\star) = t(\ell_\star)$.
  6. Output c.

Remark 18. Let $I \subset \{1, \ldots, s\}$ and $\Sigma \subset \{1,0\}^I$. Following the steps of the algorithm, it is easy to prove by induction in $\mathrm{card}(I)$ that if v is an integer vector of size $\mathrm{card}(\Sigma)$, then $c = \mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)^{-1} v$ is also an integer vector.

Proposition 19. *Given a list $\Sigma \subset \{0,1\}^I$ with $I \subset \{1, \ldots, s\}$, the matrices $\mathrm{Info}(\Sigma)$ and $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ and an integer vector v of size $\mathrm{card}(\Sigma)$, Algorithm* `Linear Solving` *computes the vector*

$$c = \mathrm{Mat}\big(\mathrm{Ada}(\Sigma), \Sigma\big)^{-1} v.$$

If $Z \subset \mathrm{C}^k$ is a finite set with r elements, $\mathcal{Q}$ is a finite set of polynomials indexed by I and $\mathrm{Feas}(\mathcal{Q}, Z) \subset \Sigma$ then $v = \mathrm{Qu}(\mathcal{Q}^{\mathrm{Ada}(\Sigma)}, Z)$ implies $c = c(\Sigma, Z)$ and the bit complexity of Algorithm `Linear Solving` *is $O(\mathrm{card}(\Sigma)\mathrm{card}(I) + \mathrm{card}(\Sigma)^2 \mathrm{bit}(r))$.*

Proof. The correctness of the algorithm follows from the formula for the inverse of $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ coming from the proof of Proposition 10. The fact that $v = \mathrm{Qu}(\mathcal{Q}^{\mathrm{Ada}(\Sigma)}, Z)$ implies $c = c(\Sigma, Z)$ follows from Proposition 1. Now we deal with the complexity analysis. Note that the matrices $\mathrm{Info}(\Xi)$, $\mathrm{Mat}(\mathrm{Ada}(\Xi), \Xi)$, $\mathrm{Info}(\Xi')$ and $\mathrm{Mat}(\mathrm{Ada}(\Xi'), \Xi')$, which are necessary to do the recursive calls, and the matrix $\mathrm{Mat}((\mathrm{Ada}(\Xi'), i), \Sigma_\star)$, which is necessary at Step 4 can be extracted from the matrices $\mathrm{Info}(\Sigma)$ and $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$.

A rough description of Step 2 is the following: consider first all the lists we want to determine as empty and then, reading the last column of $\mathrm{Info}(\Sigma)$ one row at a time, actualize the right lists and their lengths. Taking into account that increasing n times the quantity 0 by 1 takes $O(n)$ bit operations, this step takes at most $C_1 \mathrm{card}(\Sigma)$ bit operations for some constant C_1.

At Step 3, the recursive call to Algorithm `Linear Solving` is done for the list Ξ and the vector $\mathrm{Qu}(\mathcal{Q}^{\mathrm{Ada}(\Xi)}, Z)$ and since $\mathrm{Feas}(\mathcal{Q}, Z) \subset \Sigma$, we have that $\mathrm{Feas}(\mathcal{Q}', Z) \subset \Xi$ where $\mathcal{Q}'$ is the set obtained from $\mathcal{Q}$ by removing its element of index i.

Step 4 takes first 2 bit operations to decide if $m^0 \neq 0$ and $m^\star \neq 0$. Since $v(\ell_1) = \mathrm{Qu}(\mathcal{Q}^{(\mathrm{Ada}(\Xi'), i)}, Z)$, all its entries are nonnegative integers less than or equal to r. The product $\mathrm{Mat}((\mathrm{Ada}(\Xi'), i), \Sigma_\star) t(\ell_\star)$ is computed by reading if the elements in $\mathrm{Mat}((\mathrm{Ada}(\Xi'), i), \Sigma_\star)$ are 0 or 1 and adding the corresponding elements from $t(\ell_\star)$. Since $t(\ell_\star)$ is a subvector of $c(\Xi, Z)$, all its entries are nonnegative integers

and their sum is less than or equal to r. So, we conclude that this step takes at most $2+C_2m^0m^\star\text{bit}(r)$ bit operations for some constant C_2.

The beginning of Step 5 takes 1 bit operation to decide if $m^0 \neq 0$. It is easy to see that at Step 5(a), the recursive call to Algorithm `Linear Solving` is done for the list Ξ' and the vector $\text{Qu}(\mathcal{Q}^{\text{Ada}(\Xi')}, Z^1)$ where

$$Z^1 = \bigcup_{\sigma\in\Sigma_1} \text{Reali}(\sigma, Z).$$

Also, by definition of Z^1, we have that $\text{Feas}(\mathcal{Q}', Z^1) \subset \Xi'$. On the other hand, it is also easy to see that Step 5(b) takes $C_3m^0\text{bit}(r)$ bit operations for some constant C_3.

After this analysis, the bound on the bit complexity can be proved by induction in $\text{card}(I)$. □

Notation 20. Let $J \subset \{1,\dots,s\}$ and $j \in \{1,\dots,s\}$ bigger than $\max(J)$. For a zero–nonzero condition $\sigma \in \{0,1\}^J$, we denote by $\sigma \wedge 0 \in \{0,1\}^{J\cup\{j\}}$ (resp. $\sigma \wedge 1$) the zero–nonzero condition obtained by extending σ with $\sigma(j) = 0$ (resp. $\sigma(j) = 1$). For a list $\Sigma = [\sigma_1,\dots,\sigma_n] \subset \{0,1\}^J$ of zero–nonzero conditions, we denote by $\Sigma \wedge \{0,1\}^{\{j\}} \subset \{0,1\}^{J\cup\{j\}}$ the list of zero–nonzero conditions $[\sigma_1\wedge 0, \sigma_1\wedge 1,\dots,\sigma_n\wedge 0, \sigma_n\wedge 1]$ and by $\Sigma \wedge \{0\}^{\{j\}} \subset \{0,1\}^{J\cup\{j\}}$ (resp. $\Sigma \wedge \{1\}^{\{j\}}$) the list of zero–nonzero conditions $[\sigma_1\wedge 0,\dots,\sigma_n\wedge 0]$ (resp. $[\sigma_1\wedge 1,\dots,\sigma_n\wedge 1]$).

We are now ready for our main algorithm. This algorithm determines iteratively, for $i = 0,\dots,s$ the list $\Sigma_i := \text{Feas}(\mathcal{P}_i, Z)$ of the zero–nonzero conditions realized by $\mathcal{P}_i$ on Z, and the corresponding list $c_i := c(\mathcal{P}_i, Z)$ of cardinals. In order to do so within a good complexity bound, the algorithm also computes at each step the compressed set of indices $\text{comp}_i := \text{comp}(\Sigma_i)$, the matrix $\text{Info}_i := \text{Info}(\text{Comp}(\Sigma_i))$, the list $\text{Ada}_i := \text{Ada}(\Sigma_i)$ and the matrix $\text{Mat}_i := \text{Mat}(\Sigma_i, \text{Ada}(\Sigma_i))$.

Algorithm `Zero-nonzero Determination`

- **Input:** a finite subset $Z \subset \mathbb{C}^k$ with r elements and a finite list $\mathcal{P} = P_1,\dots,P_s$ of polynomials in $\mathbb{L}[X_1,\dots,X_k]$.
- **Output:** the list $\text{Feas}(\mathcal{P}, Z)$ of the zero–nonzero conditions realized by $\mathcal{P}$ on Z, and the corresponding list $c(\mathcal{P}, Z)$ of cardinals.
- **Blackbox:** for a polynomial $Q \in \mathbb{L}[X_1,\dots,X_k]$, the invertibility-query blackbox $\text{Qu}(Q, Z)$.
- **Procedure:**
  1. Compute $r = \text{Qu}(1, Z)$ using the invertibility-query blackbox. If $r = 0$, output $\text{Feas}(\mathcal{P}, Z) = \emptyset$ and $c(\mathcal{P}, Z) = \emptyset$. So from now we suppose $r > 0$.
  2. Initialize $\Sigma_0 = [\emptyset]$, $c_0 = [r]$, $\text{comp}_0 = \emptyset$, Info_0 as the matrix with 1 row and 0 columns, $\text{Ada}_0 = [\emptyset]$ and $\text{Mat}_0 = (1)$.
  3. For i from 1 to s:
     (a) Compute $\text{Qu}(P_i, Z)$ using the invertibility-query blackbox.
     (b) Using the equality
     $$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} c(P_i = 0, Z) \\ c(P_i \neq 0, Z) \end{pmatrix} = \begin{pmatrix} \text{Qu}(1, Z) \\ \text{Qu}(P_i, Z) \end{pmatrix}$$
     compute $c(P_i = 0, Z)$ and $c(P_i \neq 0, Z)$.
     (c) If $c(P_i = 0, Z) = 0$ (resp. $c(P_i \neq 0, Z) = 0$), $\Sigma_i = \Sigma_{i-1} \wedge \{1\}^{\{i\}}$ (resp. $\Sigma_{i-1} \wedge \{0\}^{\{i\}}$), $c_i = c_{i-1}$, $\text{comp}_i = \text{comp}_{i-1}$, $\text{Info}_i = \text{Info}_{i-1}$, $\text{Ada}_i = \text{Ada}_{i-1}$ and $\text{Mat}_i = \text{Mat}_{i-1}$.
     Else if $c(P_i = 0, Z) > 0$ and $c(P_i \neq 0, Z) > 0$:
        i. Compute $v' = \text{Qu}(\mathcal{P}_i^{(\text{Ada}_{i-1}, i)}, Z)$ using the invertibility-query blackbox.
        ii. Take the auxiliary list $\Sigma = \text{Comp}(\Sigma_{i-1}) \wedge \{0,1\}^{\{i\}}$ and determine $\text{Info}(\Sigma)$ and $\text{Mat}(\text{Ada}(\Sigma), \Sigma)$. Consider the integer vector v of size $\text{card}(\Sigma)$ having in its odd entries the entries of $\text{Qu}(\mathcal{P}_i^{\text{Ada}_{i-1}}, Z)$ (which has already been computed at previous steps) and in its even entries the entries of v'. Compute $c = \text{Mat}(\text{Ada}(\Sigma), \Sigma)^{-1}v$, using Algorithm `Linear Solving`.

iii. Compute c_i removing from c its zero components. Compute also Σ_i going trough c by pairs of elements (note that each pair will have at least one element different from zero). If both elements are different from zero, the corresponding zero–nonzero condition in Σ_{i-1} is extended both with a 0 and a 1 in Σ_i. If only the first (resp. second) element of the pair is different from zero, the corresponding zero–nonzero condition in Σ_{i-1} is extended only with a 0 (resp. 1) in Σ_i. At the same time, compute the lists ℓ_0, ℓ_1 and $\ell_\star$.

iv. If $\mathrm{card}(\Sigma_i) = \mathrm{card}(\Sigma_{i-1})$, $\mathrm{comp}_i = \mathrm{comp}_{i-1}$, $\mathrm{Info}_i = \mathrm{Info}_{i-1}$, $\mathrm{Ada}_i = \mathrm{Ada}_{i-1}$ and $\mathrm{Mat}_i = \mathrm{Mat}_{i-1}$.
Else if $\mathrm{card}(\Sigma_i) > \mathrm{card}(\Sigma_{i-1})$:
- $\mathrm{comp}_i = \mathrm{comp}_{i-1} \cup \{i\}$.
- Define $\Xi'_i = \Sigma_i(\ell_1, \mathrm{comp}_{i-1})$.
- Compute $\mathrm{Info}(\Xi'_i)$, using Algorithm `Get Info`, and extract $\mathrm{Ada}(\Xi'_i)$, using Algorithm `Adapted Family`.
- Compute Info_i, from Info_{i-1} and $\mathrm{Info}(\Xi'_i)$ and Ada_i from Ada_{i-1} and $\mathrm{Ada}(\Xi'_i)$.
- Compute $\mathrm{Mat}((\mathrm{Ada}(\Xi'_i), i), \Sigma')$ with $\Sigma' = \Sigma_i(\ell, \mathrm{comp}_i)$ and finally Mat_i.

4. Output $\mathrm{Feas}(\mathcal{P}, Z) = \Sigma_s$ and $\mathrm{c}(\mathcal{P}, Z) = c_s$.

Theorem 21. *Given a finite subset $Z \subset \mathrm{C}^k$ with r elements and a finite list $\mathcal{P} = P_1, \ldots, P_s$ of polynomials in $\mathrm{L}[X_1, \ldots, X_k]$, Algorithm* `Zero-nonzero Determination` *computes the list* $\mathrm{Feas}(\mathcal{P}, Z)$ *of the zero–nonzero conditions realized by $\mathcal{P}$ on Z and the corresponding list $\mathrm{c}(\mathcal{P}, Z)$ of cardinals. The complexity of this algorithm is $O(sr^2\mathrm{bit}(r))$ bit operations plus $1 + sr$ calls to the to the invertibility-query blackbox which are done for products of at most* $\mathrm{bit}(r)$ *products of polynomials in $\mathcal{P}$.*

Proof. The correctness of the algorithm follows from Proposition 1, Proposition 10 and Remark 13.

We prove first the bound on the number of bit operations.

Step 3(b) and evaluating the conditions $c(P_i = 0, Z) = 0$ and $c(P_i \neq 0, Z) = 0$ at the beginning of Step 3(c) take $O(\mathrm{bit}(r))$ bit operations.

At Step 3(c)ii, Σ is obtained by duplicating each row in $\Sigma_{i-1}([1, \ldots, \mathrm{card}(\Sigma_{i-1})], \mathrm{comp}_{i-1})$ and adding a final new column with a 0 in the odd rows and a 1 in the even rows, and $\mathrm{Info}(\Sigma)$ is obtained in a similar way. The matrix $\mathrm{Mat}(\mathrm{Ada}(\Sigma), \Sigma)$ is obtained from Mat_{i-1} following the formula (1) in the proof of Proposition 10. Since $\mathrm{card}(\mathrm{Comp}(\Sigma_{i-1})) \leqslant r$ and $\mathrm{card}(\mathrm{comp}_{i-1}) \leqslant r - 1$, by Proposition 19, the computation of c takes $O(r^2\mathrm{bit}(r))$ bit operations.

Step 3(c)iii takes $O(r)$ bit operations.

Since $\mathrm{card}(\mathrm{comp}_{i-1}) \leqslant \min\{s, r-1\}$ for every i, at Step 3(c)iv the computation of $\mathrm{Info}(\Xi'_i)$ takes $O(\mathrm{card}(\Xi'_i)sr)$ bit operations using Algorithm `Get Info`. By Lemma 16, the computation of $\mathrm{Ada}(\Xi'_i)$ takes $O(\mathrm{card}(\Xi'_i)r)$ bit operations. The computation of Info_i and Ada_i is then done following Definition 14 and using the already computed matrices Info_{i-1}, $\mathrm{Info}(\Xi'_i)$, Ada_{i-1} and $\mathrm{Ada}(\Xi'_i)$ and the lists ℓ_0, ℓ_1 and $\ell_\star$. By evaluating the product σ^J for every $\sigma \in \Sigma'$ and $J \in (\mathrm{Ada}(\Xi'_i), i)$, the computation of $\mathrm{Mat}((\mathrm{Ada}(\Xi'_i), i), \Sigma')$ takes $O(\mathrm{card}(\Xi'_i)sr)$ bit operations. The matrix Mat_i is obtained following the formula (1) in the proof of Proposition 10 using the already computed matrices Mat_{i-1} and $\mathrm{Mat}((\mathrm{Ada}(\Xi'_i), i), \Sigma')$.

It is easy to prove that $\sum_{i=1,\ldots,s} \mathrm{card}(\Xi'_i) = \mathrm{card}(\mathrm{Feas}(\mathcal{P}, Z)) - 1 \leqslant r - 1$. From this, we conclude the number of bit operations of this algorithm is $O(sr^2\mathrm{bit}(r))$.

Finally we prove the assertion on the invertibility-queries to compute. At Step 3, for $i = 1, \ldots, s$, there are $\mathrm{card}(\Sigma_{i-1}) \leqslant r$ new invertibility-queries to determine. Therefore, the total number of calls to the invertibility-query blackbox is bounded by $1 + sr$. Since by Proposition 9 the elements of Ada_{i-1} are subsets of $\{1, \ldots, s\}$ with at most $\mathrm{bit}(r) - 1$ elements, these calls are done for polynomials which are product of at most $\mathrm{bit}(r)$ products of polynomials in $\mathcal{P}$. □

Definition 22. We denote by $\mathrm{Used}(\mathcal{P}, Z)$ the list of subsets of $\{1, \ldots, s\}$ constructed inductively as follows:

- $\mathrm{Used}(\mathcal{P}_0, Z) = \emptyset$.

- For $1 \leqslant i \leqslant s$,

$$\begin{cases} \text{Used}(\mathcal{P}_i, Z) = \text{Used}(\mathcal{P}_{i-1}, Z) \cup \{i\} \\ \quad \text{if } c(P_i = 0, Z) = 0 \text{ or } c(P_i \neq 0, Z) = 0, \\ \text{Used}(\mathcal{P}_i, Z) = \text{Used}(\mathcal{P}_{i-1}, Z) \cup (\text{Ada}(\text{Feas}(\mathcal{P}_{i-1}, Z)), i) \quad \text{otherwise.} \end{cases}$$

Remark 23. It is easy to see that if $Z \neq \emptyset$, $\text{Used}(\mathcal{P}, Z)$ is exactly the list of subsets J of $\{1, \ldots, s\}$ such that the invertibility-query of P^J has been computed during the execution of Algorithm `Zero-nonzero Determination`. It is also clear that $\text{Used}(\mathcal{P}, Z)$ can be determined from $\text{Feas}(\mathcal{P}, Z)$. As mentioned in Theorem 21, the elements of $\text{Used}(\mathcal{P}, Z)$ are subsets of $\{1, \ldots s\}$ with at most $\text{bit}(r)$ elements.

3. The real–nonreal sign determination problem

We recall that K is an ordered field, R a real closed extension of K and $\mathrm{C} = \mathrm{R}[i]$, $Z \subset \mathrm{C}^k$ a finite set and $\mathcal{P} = P_1, \ldots, P_s$ a list of polynomials in $\mathrm{K}[X_1, \ldots, X_k]$.

Since $\text{Feas}(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}}) \subset \text{Feas}(\mathcal{P}, Z)$ and for every $\sigma \in \text{Feas}(\mathcal{P}, Z)$ we have that

$$c(\sigma, Z_{\mathrm{C}\setminus\mathrm{R}}) = c(\sigma, Z) - c(\sigma, Z_{\mathrm{R}}) = c(\sigma, Z) - \sum_{\tau \in \Gamma_\sigma} c_{\text{sign}}(\tau, Z_{\mathrm{R}})$$

where

$$\Gamma_\sigma = \left\{\tau \in \text{Feas}_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}}) \;\middle|\; \{i \mid \tau(i) = 0\} = \{i \mid \sigma(i) = 0\}\right\},$$

it is easy to combine algorithms for sign and zero–nonzero determination to solve the real–nonreal sign determination problem as follows.

Algorithm `Real-nonreal Sign Determination`

- **Input:** a finite subset $Z \subset \mathrm{C}^k$ with r elements and a finite list $\mathcal{P} = P_1, \ldots, P_s$ of polynomials in $\mathrm{K}[X_1, \ldots, X_k]$.
- **Output:** the lists $\text{Feas}_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$ and $\text{Feas}(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ of sign and zero–nonzero conditions realized by $\mathcal{P}$ on Z_{R} and $Z_{\mathrm{C}\setminus\mathrm{R}}$ respectively, and the corresponding lists $c_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$ and $c(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ of cardinals.
- **Blackbox 1:** for a polynomial $Q \in \mathrm{K}[X_1, \ldots, X_k]$, the Tarski-query blackbox $\text{Qu}(Q, Z_{\mathrm{R}})$.
- **Blackbox 2:** for a polynomial $Q \in \mathrm{K}[X_1, \ldots, X_k]$, the invertibility-query blackbox $\text{Qu}(Q, Z)$.
- **Procedure:**
  1. Compute $\text{Feas}_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$ and $c_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$, as explained in [2, Chapter 10].
  2. Compute $\text{Feas}(\mathcal{P}, Z)$ and $c(\mathcal{P}, Z)$, using Algorithm `Zero-nonzero Determination`.
  3. For $\sigma \in \text{Feas}(\mathcal{P}, Z)$ compute $c(\sigma, Z_{\mathrm{C}\setminus\mathrm{R}}) = c(\sigma, Z) - \sum_{\tau \in \Gamma_\sigma} c_{\text{sign}}(\tau, Z_{\mathrm{R}})$.
  4. Define $\text{Feas}(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ as the list of $\sigma \in \text{Feas}(\mathcal{P}, Z)$ such that $c(\sigma, Z_{\mathrm{C}\setminus\mathrm{R}}) \neq 0$ and $c(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ as the corresponding list of cardinals.

From the results in Section 2 and [2, Chapter 10], we deduce the following result.

Theorem 24. *Given a finite subset $Z \subset \mathrm{C}^k$ with r elements and a finite list $\mathcal{P} = P_1, \ldots, P_s$ of polynomials in $\mathrm{K}[X_1, \ldots, X_k]$, Algorithm* `Real-nonreal Sign Determination` *computes the lists $\text{Feas}_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$ and $\text{Feas}(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ of sign and zero–nonzero conditions realized by $\mathcal{P}$ on Z_{R} and $Z_{\mathrm{C}\setminus\mathrm{R}}$ respectively, and the corresponding lists $c_{\text{sign}}(\mathcal{P}, Z_{\mathrm{R}})$ and $c(\mathcal{P}, Z_{\mathrm{C}\setminus\mathrm{R}})$ of cardinals. The complexity of this algorithm is $O(sr^2\text{bit}(r))$ bit operations plus $1 + 2sr$ calls to the to the Tarski-query blackbox which are done for products of at most $\text{bit}(r)$ products of polynomials in $\mathcal{P}$ or squares of polynomials in $\mathcal{P}$, plus $1 + sr$ calls to the to the invertibility-query blackbox which are done for products of at most $\text{bit}(r)$ products of polynomials in $\mathcal{P}$.*

4. How to compute the queries?

For completeness, in this section we summarize the main methods to compute the invertibility-queries and Tarski-queries; all these methods are in fact closely related between them. We refer to [1] for notations, proofs and details.

We deal first with the univariate case. Suppose that $Z \subset \mathrm{C}$ is given as the zero set of a polynomial $P \in \mathrm{K}[X]$. For simplicity, we assume P to be monic. The invertibility-queries and Tarski-queries can be computed as follows.

- Through the Sturm sequence $\mathrm{Stu}(P, Q)$, which is a slight modification of the sequence of remainders of P and $P'Q$:

$$\mathrm{Qu}(Q, Z) = \deg(P) - \deg\big(\gcd\big(P, P'Q\big)\big),$$
$$\mathrm{TaQu}(Q, Z_{\mathrm{R}}) = \mathrm{var}\big(\mathrm{Stu}(P, Q); -\infty, \infty\big),$$

  where var is the number of sign variations in the corresponding sequence. Note that $\gcd(P, P'Q)$ is the last element in $\mathrm{Stu}(P, Q)$.
- Through the subresultant sequence $\mathrm{sRes}(P, R)$ of P and $R = \mathrm{Rem}(P, P'Q)$:

$$\mathrm{Qu}(Q, Z) = \deg(P) - \deg\big(\gcd\big(P, P'Q\big)\big),$$
$$\mathrm{TaQu}(Q, Z_{\mathrm{R}}) = \mathrm{PmV}\big(\mathrm{sRes}(P, R)\big),$$

  where PmV is a generalization of the difference between the number of positive elements and negative elements in the corresponding sequence. Note that $\gcd(P, P'Q)$ is the last nonzero element in $\mathrm{sRes}(P, R)$.
- Through the Bezoutian matrix $\mathrm{Bez}(P, R)$ of P and $R = \mathrm{Rem}(P, P'Q)$:

$$\mathrm{Qu}(Q, Z) = \mathrm{rank}\big(\mathrm{Bez}(P, R)\big),$$
$$\mathrm{TaQu}(Q, Z_{\mathrm{R}}) = \mathrm{sign}\big(\mathrm{Bez}(P, R)\big).$$

Now we deal with the multivariate case (including the univariate case). Suppose that $Z \subset \mathrm{C}^k$ is given as the zero set of a polynomial system in $\mathrm{K}[X_1, \ldots, X_k]$. The invertibility-queries and Tarski-queries can be computed as follows.

- Through the Hermite matrix $\mathrm{Her}(P, Q)$:

$$\mathrm{Qu}(Q, Z) = \mathrm{rank}\big(\mathrm{Her}(P, Q)\big),$$
$$\mathrm{TaQu}(Q, Z_{\mathrm{R}}) = \mathrm{sign}\big(\mathrm{Her}(P, Q)\big).$$

For the univariate case, we deduce the following result:

Theorem 25.

1. *Let $P \in \mathrm{L}[X]$ and $\mathcal{P} = P_1, \ldots, P_s$ be a finite list of polynomials in $\mathrm{L}[X]$ such that the degree of P and the polynomials in $\mathcal{P}$ is bounded by d. The complexity of zero–nonzero determination is $\tilde{O}(sd^2)$ bit operations plus $\tilde{O}(sd^2)$ arithmetic operations in L. If P and the polynomials in $\mathcal{P}$ lie in $\mathbb{Z}[X]$ and the bit size of the coefficients of all these polynomials is bounded by τ, the complexity of zero–nonzero determination is $\tilde{O}(\tau sd^3)$ bit operations.*
2. *Let $P \in \mathrm{K}[X]$ and $\mathcal{P} = P_1, \ldots, P_s$ be a finite list of polynomials in $\mathrm{K}[X]$ such that the degree of P and the polynomials in $\mathcal{P}$ is bounded by d. The complexity of real–nonreal sign determination is $\tilde{O}(sd^2)$ bit operations plus $\tilde{O}(sd^2)$ arithmetic operations in K. If P and the polynomials in $\mathcal{P}$ lie in $\mathbb{Z}[X]$ and the bit size of the coefficients of all these polynomials is bounded by τ, the complexity of real–nonreal sign determination is $\tilde{O}(\tau sd^3)$ bit operations.*

Proof. To prove item 1, we estimate the complexity of computing the invertibility-queries. At each call we have to compute the product between an already computed product of P' and at most $\mathrm{bit}(d)-1$ polynomials in $\mathcal{P}$ and a polynomial in $\mathcal{P}$; and then we have to compute the gcd between this polynomial and P. The complexity of these computations is $\tilde{O}(d)$ operations in L; in the case of integer coefficients, the complexity of these computations is $\tilde{O}(\tau d^2)$ bit operations (see [8, Chapters 8 and 11]). The result then follows directly from Theorem 21.

Item 2 is proved in a similar way, the Tarski-queries being computed following the subresultant sequence approach as in [4]. The Tarski-query of a product of at most $2\mathrm{bit}(d)$ polynomials in $\mathcal{P}$ for the real zero set of P can be computed within $\tilde{O}(d)$ operations in K; in the case of integer coefficients, the complexity of this computation is $O(\tau d^2)$ bit operations (see [6] and [4]). The result then follows directly from Theorem 24. □

5. Real–nonreal sign determination with parameters

Let $P \in \mathrm{K}[Y_1,\ldots,Y_m,X]$ with P monic with respect to X and $\mathcal{P} = P_1,\ldots,P_s \subset \mathrm{K}[Y_1,\ldots,Y_m,X]$, where $Y_1,\ldots,Y_m$ are parameters and X is the main variable. Our goal is to describe a family of polynomials in $\mathrm{K}[Y_1,\ldots,Y_m]$ such that the result of the real–nonreal sign determination problem after specialization of the parameters is determined by a sign condition on this family. First, we introduce some notation.

Notation 26. We denote by $\mathrm{Prod}_{\mathrm{bit}(p)}(\mathcal{P})$ the family of polynomials which are of the form

$$\mathcal{P}^\alpha = \prod_{1\leqslant i\leqslant s} P_i^{\alpha_i}$$

with $\alpha = (\alpha_1,\ldots,\alpha_s) \in \{0,1,2\}^{\{1,\ldots,s\}}$ such that $\mathrm{card}\{i \mid \alpha_i \neq 0\} \leqslant \mathrm{bit}(p)$. For $Q \in \mathrm{K}[Y_1,\ldots,Y_m,X]$, we denote by $\mathrm{HMin}(P,Q)$ the subset of $\mathrm{K}[Y_1,\ldots,Y_m]$ formed by the principal minors of $\mathrm{Her}(P,Q)$. Finally, we denote by

$$\mathrm{HElim}(P,\mathcal{P}) = \bigcup_{Q\in \mathrm{Prod}_{\mathrm{bit}(p)}(\mathcal{P})} \mathrm{HMin}(P,Q).$$

Now we can state our result.

Theorem 27. *For $y \in \mathrm{R}^m$, let $Z(y) \subset \mathrm{C}$ be the zero set of $P(y_1,\ldots,y_m,X)$ and $\mathcal{P}(y,X)$ the list in $\mathrm{K}[X]$ obtained from $\mathcal{P}$ after specialization of $Y_1,\ldots,Y_m$ at $y_1,\ldots,y_m$. Let τ be a sign condition on $\mathrm{HElim}(P,\mathcal{P})$. The lists $\mathrm{Feas}_{\mathrm{sign}}(\mathcal{P}(y,X),Z(y)_{\mathrm{R}})$ and $\mathrm{Feas}(\mathcal{P}(y,X),Z(y)_{\mathrm{C}\setminus\mathrm{R}})$ and their corresponding lists $c_{\mathrm{sign}}(\mathcal{P}(y,X),Z(y)_{\mathrm{R}})$, and $c(\mathcal{P}(y,X),Z(y)_{\mathrm{C}\setminus\mathrm{R}})$ of cardinals are fixed as y varies in $\mathrm{Real}_{\mathrm{sign}}(\tau,\mathrm{R}^m)$.*

Proof. For every $P,Q \in \mathrm{K}[X]$, the signs of the principal minors of $\mathrm{Her}(P,Q)$ determine the rank and signature of $\mathrm{Her}(P,Q)$ (see [1, Chapter 9]). Therefore, by Remark 23 and the analogous result in [1, Chapter 10], as y varies in $\mathrm{Real}_{\mathrm{sign}}(\tau,\mathrm{R}^m)$, all the calls to the Tarski-query blackbox and invertibility-query blackbox done in Algorithm `Real-nonreal Sign Determination` provide the same result; from this, we conclude that the output of this algorithm is the same. □

References

[1] Saugata Basu, Richard Pollack, Marie-Françoise Roy, Algorithms in Real Algebraic Geometry, second edition, Algorithms Comput. Math., vol. 10, Springer-Verlag, Berlin, 2006.
[2] Saugata Basu, Richard Pollack, Marie-Françoise Roy, Algorithms in real algebraic geometry, current online version, available at http://perso.univ-rennes1.fr/marie-francoise.roy/.
[3] John Canny, Improved algorithms for sign determination and existential quantifier elimination, Comput. J. 36 (5) (1993) 409–418.
[4] Thomas Lickteig, Marie-Françoise Roy, Sylvester–Habicht sequences and fast Cauchy index computations, J. Symb. Comput. 31 (3) (2001) 315–341.

[5] Daniel Perrucci, Linear solving for sign determination, Theor. Comput. Sci. 412 (35) (2011) 4715–4720.
[6] Daniel Reischert, Asymptotically fast computation of subresultants, in: ISSAC '97, pp. 233–240.
[7] Marie-Françoise Roy, Aviva Szpirglas, Complexity of computation on real algebraic numbers, J. Symb. Comput. 10 (1) (1990) 39–51.
[8] Joachim von zur Gathen, Jürgen Gerhard, Modern Computer Algebra, Cambridge University Press, New York, 1999.