Journal of Computational and Applied Mathematics 235 (2011) 2670–2678

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Condition number based complexity estimate for solving polynomial systems☆

Zhikun She [a,*], Bican Xia [b], Zhiming Zheng [a]

[a] SKLSDE, LMIB and School of Mathematics and Systems Science, Beihang University, Beijing, China
[b] School of Mathematical Sciences, Peking University, Beijing, China

ARTICLE INFO

Article history:
Received 18 January 2009
Received in revised form 26 June 2010

MSC:
primary 68W40
secondary 68W30

Keywords:
Real-root-counting
Newton operator
Condition number
Complexity analysis

ABSTRACT

By modifying and combining algorithms in symbolic and numerical computation, we propose a real-root-counting based method for deciding the feasibility of systems of polynomial equations. Along with this method, we also use a modified Newton operator to efficiently approximate the real solutions when the systems are feasible. The complexity of our method can be measured by a number of arithmetic operations which is singly exponential in the number of variables.

1. Introduction

Solving systems of polynomial equations using computer is of great interest in both industrial and academic areas. In the literature of numerical analysis, there are many classical methods such as the Newton method, the simplex method and the homotopy method for solving polynomial systems numerically. On the other hand, many new algorithms [1–6] with lower complexity have been proposed for deciding the feasibility of systems of polynomial equations and inequalities after the work of Tarski [7], which is well known to have hyper-exponential complexity. In particular, based on the real Turing machine [8,9], Cucker and Smale [5] gave an algorithm with singly exponential complexity for deciding the feasibility of systems of polynomial equations.

Isolating the real solutions of polynomial equation(s) is an important topic in computational real algebra from the viewpoint of symbolic computation. Isolating the real roots of a univariate polynomial relies on algorithms for counting real roots in an interval [10]. Analogously, we can achieve the goal of isolating the real solutions of multivariate polynomial equations by combining the subdivision method with the algorithm for counting the real solutions of the system in a region. Such algorithms for real-root-counting exist (e.g., [11,12]). In this paper, we would like to use the method of Pedersen et al. [11] and provide an upper bound for the number of bisections in terms of a newly defined auxiliary quantity.

When all the real solutions have been isolated, the feasibility of the given system is determined. To approximate the solutions faster, we employ a numerical method—a modified Newton method. The original version of this Newton method

☆ The work was partly supported by Beijing Nova Program, NKBRPC-2005CB321902, NSFC-61003021 and SKLSDE-2010ZX-02.
* Corresponding author.
E-mail address: zhikun.she@buaa.edu.cn (Z. She).

0377-0427/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2010.11.018

was proposed in [5]. It involves approximate computations of many auxiliary quantities such as L_f, $\alpha(f, \boldsymbol{x})$ and $\beta(f, \boldsymbol{x})$. Note that the original definitions of L_f, $\alpha(f, \boldsymbol{x})$ and $\beta(f, \boldsymbol{x})$ are complicated and can be found in [5]. However, in our method, we avoid defining and approximating such complicated auxiliary quantities.

The complexity of our method to decide the feasibility of systems of polynomial equations and to approximate their real solutions is measured by a number of arithmetical operations, which is proved to be singly exponential in the number of variables and an intrinsic quantity (i.e., a newly defined condition number).

This paper is organized as follows. In Section 2, we first describe the algorithm proposed in [11] for counting real solutions of systems of polynomial equations inside a real semi-algebraic constraint region and also an algorithm for isolating the real roots. In Section 3, we introduce a modified Newton method and give an upper bound for the number of bisections. In Section 4, we first propose our real-root-counting based algorithm with complexity analysis for deciding the feasibility and approximating the real solutions of systems of homogeneous polynomial equations, and then extend this algorithm to general cases. In Section 5, we conclude our paper.

2. An algorithm for counting real roots

The number of distinct real roots of a univariate polynomial in an interval can be determined by some results such as Sturm's theorem and Descartes' rule of signs [13,10]. For systems of multivariate polynomial equations, we employ the method proposed in [11] in this section.

Let $f_1, \ldots, f_k \in \mathbb{R}[x_1, \ldots, x_n]$ (i.e., a polynomial ring over $\mathbb{R}$) be real polynomials, $I = \langle f_1, \ldots, f_k\rangle \subseteq \mathbb{R}[x_1, \ldots, x_n]$ be the ideal generated by the given polynomials, and $V_{\mathbb{R}}(I)$ be the set $\{\boldsymbol{x} \in \mathbb{R}^n : f_i(\boldsymbol{x}) = 0, i = 1, \ldots, k\}$, where $\boldsymbol{x} = (x_1, \ldots, x_n)$. Suppose that $A = \mathbb{R}[x_1, \ldots, x_n]/I$ is a finite-dimensional vector space. For any $f \in A$, we may consider the vector space endomorphism induced by multiplication with f, which is denoted by $m_f \in \mathrm{End}_{\mathbb{R}}(A)$. This defines a homomorphism $m : A \mapsto \mathrm{End}_{\mathbb{R}}(A)$, so that $m_f m_g = m_{fg}$. Note that m_f is a real matrix here and please refer to [11] for details.

We define a symmetric bilinear form $S : A \times A \mapsto \mathbb{R}$ by $S(f, g) = \mathrm{Tr}(m_f m_g) = \mathrm{Tr}(m_{fg})$, where $\mathrm{Tr}(M)$ denotes the trace of the matrix M. For a given basis $B = \{w_j\}$ of A, the associated matrix M for S with respect to B is given by $M_{i,j} = \mathrm{Tr}(m_{w_i} m_{w_j})$. Similarly, for each polynomial h, we can construct a bilinear form as $S_h(f, g) = S(hf, g) = \mathrm{Tr}(m_{fgh})$. The associated matrix M_h for S_h with respect to B is given by $(M_h)_{i,j} = \mathrm{Tr}(m_{hw_iw_j})$. Obviously, M_h is a real symmetric matrix implying that all its eigenvalues are real.

Recall from [11] that if I is a zero-dimensional ideal, then for any $h \in \mathbb{R}[\boldsymbol{x}]$,

$$\sigma(M_h) = \sharp\{\boldsymbol{x} \in V_{\mathbb{R}}(I) : h(\boldsymbol{x}) > 0\} - \sharp\{\boldsymbol{x} \in V_{\mathbb{R}}(I) : h(\boldsymbol{x}) < 0\}, \tag{1}$$

where $\sigma(M_h)$ denotes the signature of M_h [11,14] and for a set S, $\sharp S$ denotes the number of elements in S.

Let $\mathbb{R}^n = H^+ \cup H^- \cup V_{\mathbb{R}}(h)$, where $H^+ = \{\boldsymbol{x} \in \mathbb{R}^n : h(\boldsymbol{x}) > 0\}$ and $H^- = \{\boldsymbol{x} \in \mathbb{R}^n : h(\boldsymbol{x}) < 0\}$. Using Eq. (1), $\sharp V_{\mathbb{R}}(I) \cap H^+$, $\sharp V_{\mathbb{R}}(I) \cap H^-$ and $\sharp V_{\mathbb{R}}(I) \cap V_{\mathbb{R}}(h)$ can be determined by the following relation

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & -1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} \sharp V_{\mathbb{R}}(I) \cap V_{\mathbb{R}}(h) \\ \sharp V_{\mathbb{R}}(I) \cap H^+ \\ \sharp V_{\mathbb{R}}(I) \cap H^- \end{pmatrix} = \begin{pmatrix} \sigma(M_1) \\ \sigma(M_h) \\ \sigma(M_{h^2}) \end{pmatrix}. \tag{2}$$

As a preprocessing of computing M_h, a monomial basis B of A is constructed by Gröbner basis computing [13,15,16]. As our discussion is restricted to the zero-dimensional case, the complexity of this preprocessing is well known to be a polynomial in degree d and the number m of the input polynomials and singly exponential in the number n of variables. More precisely, the complexity is $m^{O(1)}(dn)^{O(n)}$.

Making use of Eq. (2) and Ben-Or et al.'s trick [17], Pedersen et al. [11] gave the CRZ-algorithm (i.e., "Counting Real Zeros" algorithm) for counting the real zeros of a system in any real semi-algebraic constraint region, whose computational complexity is singly exponential in the number of constraints.

Remark 1. For a detailed description of the CRZ-algorithm, please refer to [11]. Here, we just provide a brief version of the CRZ-algorithm. For this, we assume that a real semi-algebraic constraint region is defined to be

$$P = \{x \in \mathbb{R}^n : h_1(x)\varepsilon_1, \ldots, h_s(x)\varepsilon_s\},$$

where $h_j(x)$ is a polynomial and $\varepsilon_j \in \{>0, =0, <0\}$ for $j \in \{1, \ldots, s\}$. Moreover, for $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_s) \in \{>0, =0, <0\}^s$, let $c_\varepsilon(F, H) = \sharp\{x \in V_{\mathbb{R}}(I) : h_i\varepsilon_i, 1 \le i \le s\}$. The CRZ-algorithm (i.e., "Counting Real Zeros" algorithm) which outputs the set of $\mathrm{CRZ}(F, H) = \{c_\varepsilon(F, H) : \varepsilon \in \{>0, =0, <0\}^s\}$ is briefly described as follows.

- Compute $PM(F, H) = \{m_{h_i} : 1 \le i \le s\}$.
- For $1 \le j \le s$, knowing the $\mathrm{CRZ}(F, H_J)$, where H_J having j elements, compute the $\mathrm{CRZ}(F, H_{J'})$, $H_{J'}$ having j' elements and $H_J \subset H'_J$. □