Certified Numerical Real Root Isolation for Bivariate Polynomial Systems*

Jin-San Cheng
KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences
School of Mathematical Sciences, University of Chinese Academy of Sciences
Beijing, China
jcheng@amss.ac.cn

Junyi Wen
KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences
School of Mathematical Sciences, University of Chinese Academy of Sciences
Beijing, China
wenjunyi15@mails.ucas.ac.cn

ABSTRACT

In this paper, we present a new method for isolating real roots of a bivariate polynomial system. Our method is a subdivision method which is based on real root isolation of univariate polynomials and analyzing the local geometrical properties of the given system. We propose the concept of the orthogonal monotone system in a box and use it to determine the uniqueness and the existence of a simple real zero of the system in the box. We implement our method to isolate the real zeros of a given bivariate polynomial system. The experiments show the effectivity and efficiency of our method, especially for systems with high degrees and sparse terms. Our method also works for non-polynomial systems.

CCS CONCEPTS

• **Computing methodologies** → *Symbolic and algebraic algorithms*; Hybrid symbolic-numeric methods;

KEYWORDS

Real root isolation; orthogonal monotone system; bivariate polynomial system; uniqueness and existence.

ACM Reference Format:
Jin-San Cheng and Junyi Wen. 2019. Certified Numerical Real Root Isolation for Bivariate Polynomial Systems. In *International Symposium on Symbolic and Algebraic Computation (ISSAC '19), July 15–18, 2019, Beijing, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3326229.3326237

1 INTRODUCTION

Real root isolation of zero-dimensional polynomial systems is a fundamental problem in mathematics and engineering applications. One can compute the real roots of a zero-dimensional polynomial system by symbolic methods (such as the Gröbner basis method, the Ritt-Wu characteristic set method, the resultant method, the generic position method and so on), numeric methods (such as the homotopy continuation method) and subdivision methods and so on.

In this paper, we consider only the problem of real root isolation of polynomial systems for bivariate case. The problem was considered by numerous authors [2, 3, 6, 7, 9, 11–13, 16, 19, 30, 38]. Some methods project the systems to two directions (x-axis, y-axis) by resultant and then determine whether a root pair (one x-coordinate and one y-coordinate) is a true root or not [2, 12, 16, 19, 30]. In [6, 11], they project the roots of the bivariate system to x-axis, using a matrix formulation, and lifted them up to recover the roots of the original system. The multiplicities of the roots are also considered. A local generic position method is proposed in [7]. They transform the system to a new system which is in a generic position and then project the new system to x-axis again. Finally, they recover these roots of original system from the projections of the roots of the new system to the x-axis. Moreover, the multiplicities of the roots are preserved. The method is improved and extended to general zero-dimensional polynomial systems in [9]. In [3], The authors presented new algorithms for computing linear separating forms, RUR (Rational Univariate Representations) decompositions and isolating boxes of the solutions for real root isolation of bivariate polynomial systems. All these methods are symbolic methods and they works for systems with integer or rational number coefficients. There are also some numeric methods [10]. The authors use the level set sweeping method to tackle this problem and it can handle the large system. Also, there exists nice work to isolate real roots of non-polynomial systems [35–37], besides that, the commercial software Mathematica also has this capability.

The subdivision methods [4, 5, 15, 17, 23, 28, 33] are useful to compute the real roots of polynomial systems. The subdivision methods use an exclusion test to check if a system has roots or not in the given domain. If the domain contains no root then it is thrown away, otherwise the domain is subdivided. This process is repeated until the domain contains no root or satisfies the given termination precision. However, the boxes obtained by such subdivision methods may not be isolating boxes of the given system. There are also methods to check the existence and uniqueness of the root in the given domain and to get the isolate domain. Miranda theorem [15, 24] is used for checking the existence of the real zeros.

*This work is supported by NSFC Grants 11471327.

Jacobian test [1, 21, 23] is used for checking a system has at most one real zeros. The interval Newton method [20, 26, 32] and α-theory [34] can work for testing the uniqueness of the complex zeros.

Our method is different from those methods for isolating the real zeros of bivariate polynomial systems. The resultant computation is avoided in our method and it can be regarded as a subdivision method. We use the upper and lower bounding polynomials related to an interval [8, 22] of the given bivariate system to compute candidate boxes [9] of the real zeros. This step can be regard as an exclusion test. Compared to interval evaluation, the bounding polynomials involves only the real root isolation of univariate polynomial equations, and interval evaluation involves two variables which makes the result interval much wider than the real evaluation. In [15, 28], they use the Bernstein basis to obtain the range of a polynomial in a box. The Bernstein basis has many nice properties. However, for a polynomial with high degree and sparse terms, its Bernstein representation need to compute many coefficients like a dense polynomial but the terms of the bounding polynomials are still sparse.

We present a concept of the orthogonal monotone system in a box at the first time. The differences with other subdivision methods are that we present a new criterion based on properties of the orthogonal monotone system to check the uniqueness and existence of simple real zeros of the original system in the given box and we prove that for any simple root of the given system, we can obtain the isolating box containing only the simple root by our method. Our uniqueness condition is different from the existing ones. It is not easy to satisfy the uniqueness condition for Newton's method since it may not converge in some bad cases. Based on our result, we design an algorithm to isolate the real zeros of a zero-dimensional bivariate polynomial system. For systems which may contain multiple real zeros, our method can not determine the uniqueness of multiple real zeros, thus we give a terminate precision in our algorithm. We also show that our method works for non-polynomial bivariate systems. A nice property of subdivision method is that it can be parallelized, so as our method. We implement our algorithm in Maple. But we didn't realize parallel computing in our implementation. Nevertheless, our experiments show the effectivity and efficiency of our method, especially for systems with high degrees and sparse terms.

The rest of this paper is organized as follows. We introduce some notations and preliminaries in the next section. In Section 3, we present and prove the uniqueness and existence theorem and show how to satisfy the related conditions of the theorems for a given system and a given box. The algorithm of the method is given in Section 4 and some experiment results are given in Section 5. We draw a conclusion in the last section.

2 NOTATIONS AND PRELIMINARIES

2.1 Notations

In this section, we will give some definitions and notations. Let $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$ be a polynomial system, where $\mathbb{R}$ is the field of real numbers. Let $B = [a_1, b_1] \times [a_2, b_2] \subset \mathbb{R}^2$, $m(B) = (\frac{a_1+b_1}{2}, \frac{a_2+b_2}{2})$ and $w(B) = max\{b_1 - a_1, b_2 - a_2\}$. We let $\mathbb{V}(F)$ denote all the real zeros of $F = 0$. We define similarly for $\mathbb{V}(f)$ with $f \in \mathbb{R}[x]$. Let J_F be the Jacobian matrix of F. For a polynomial $f \in \mathbb{R}[x, y]$, we let $f(B) = \{f(p)|p \in B\}$ and we say $f(B) > 0 (< 0)$ if $\forall p \in B, f(p) > 0$ (< 0). Similarly, for an interval $I \subset \mathbb{R}$, we say $I > 0$ (< 0) if $\forall p \in I, p > 0$ (< 0).

2.2 Bounding polynomials

In this subsection, we introduce the bounding polynomial (also called sleeve functions [8] or max-min polynomial [22], similar idea was also used in [39]) of a univariate interval polynomial. In the following of this subsection, we always assume that $x \geq 0$. (Note that we can consider the transformation $x \to -x$ when $x < 0$ [9].)

Given a univariate interval polynomial $k(x) = \Sigma_{i=1}^n [u_i, v_i]x^i$ with $[u_i, v_i] \subset \mathbb{R}$, we call $k_u(x) = \Sigma_{i=1}^n v_i x^i (k_d(x) = \Sigma_{i=1}^n u_i x^i)$ the **upper (lower) bounding polynomial** of $k(x)$.

We call an interval $(s, t), (s < t)$ a **sleeve interval** of $k(x)$, if $s, t \in \mathbb{V}(k_u k_d) \cup \{0, +\infty\}$, and $k_u(x)k_d(x) < 0, \forall x \in (s, t)$. Let I be a sleeve interval of a univariate interval polynomial $k(x)$. An interval $J \supset I$ is called a **candidate interval** of $k(x)$ if $J \cap I' = \emptyset$ for any sleeve interval $I' \neq I$ of $k(x)$. One can find similar definitions in [9].

Given a univariate polynomial $g(x) \in \mathbb{R}[x, y]$ and an interval $I = [s, t]$, we may evaluate $g(I)$ as an interval $[u, v]$ by means of standard interval arithmetic [25]. Hence, for a bivariate polynomial $g(x, y) \in \mathbb{R}[x, y]$ and a box $B = I_1 \times I_2, I_1 \geq 0$, $g(x, I_2)$ may be evaluated as a univariate interval polynomial. Indeed, where $g(x, y) = \Sigma_{i=1}^n g_i(y)x^i$, we have $g(x, I_2) = \sum_{i=1}^n g_i(I_2)x^i = \sum_{i=1}^n [u_i, v_i]x^i$. We denote the upper (lower) bounding polynomial of $g(x, I_2)$ by $g_u(x)(g_d(x))$ and refer to $g_u(x)(g_d(x))$ as the **upper (lower) bounding polynomial** of $g(x, y)$ related to I_2. When $I_2 \geq 0$, similar definitions apply to $g(I_1, y)$. For examples, let $g = (y + 5)x^3 - 11x^2 + (y^2 + 7)x + y - 1$ and $B = I_1 \times I_2 = [0, 1.5] \times [-0.1, 0.1]$. We have $g(x, I_2) = [4.9, 5.1]x^3 + [-11, -11]x^2 + [7, 7.01]x + [-1.1, -0.9]$, thus $g_u(x) = 5.1x^3 - 11x^2 + 7.01x - 0.9$, $g_d(x) = 4.9x^3 - 11x^2 + 7x - 1.1$. See Figure 1. We will introduce some properties of interval polynomials below. One can find more related results in [8].

Let $g(x) \in \mathbb{R}[x]$ be a univariate polynomial. We say $g(x)$ is monotonically increasing (decreasing) in an interval I, if $\forall x_1, x_2 \in I$, $x_1 < x_2$, we have $g(x_1) \leq g(x_2)$ $(g(x_1) \geq g(x_2))$. A well-know fact is that $g(x)$ is monotonically increasing (decreasing) in an interval I if and only if $\frac{\partial g}{\partial x}(I) \geq 0$ (≤ 0).

Lemma 2.1. *Let $g(x, y) \in \mathbb{R}[x, y]$ and $B = I_1 \times I_2, I_1 \geq 0$. $g_u(x), g_d(x)$ are the upper and lower bounding polynomials of $g(x, y)$ respectively. We have:*

(a) If $g_u(I_1) < 0$, then $g(B) < 0$.

(b) If $g_d(I_1) > 0$, then $g(B) > 0$.
(c) If $g_u(x)$ is monotonically decreasing in I_1, then for any $\tilde{y} \in I_2$, $g(x, \tilde{y})$ is monotonically decreasing in I_1.
(d) If $g_d(x)$ is monotonically increasing in I_1, then for any $\tilde{y} \in I_2$, $g(x, \tilde{y})$ is monotonically increasing in I_1.

Proof. We only prove (a) and (c). (b) and (d) can be proved in a similar way. Let $g = \Sigma_{i=1}^{n} g_i(y)x^i$ and $g_u(x) = \Sigma_{i=1}^{n} v_i x^i$. $\forall \tilde{y} \in I_2$, we have $g_i(\tilde{y}) \leq v_i$, thus $\forall \hat{x} \in I_1 \geq 0$, $g(\hat{x}, \tilde{y}) \leq g_u(\hat{x}) < 0$. Therefore, we get $g(B) < 0$. Similarly, for any $\tilde{y} \in I_2$, we have $\frac{\partial g(x,\tilde{y})}{\partial x}(\hat{x}) = \Sigma_{i=1}^{n-1} i \cdot g_{i-1}(\tilde{y})\hat{x}^{i-1} \leq \Sigma_{i=1}^{n-1} i \cdot v_{i-1}\hat{x}^{i-1} = \frac{\partial g_u}{\partial x}(\hat{x}) \leq 0, \forall \hat{x} \in I_1$. Therefore $g(x, \tilde{y})$ is monotonically decreasing in I_1. □

Therefore, we can check $g(B)$ is positive or negative by checking whether two univariate polynomials g_u, g_d are positive or negative in B. By isolating the real roots of g_u, g_d, we can easily know g_u, g_d are positive or negative in an interval. By [8], for a box $B = I_1 \times I_2$ and a polynomial $g(x, y) \in \mathbb{R}[x, y]$, we have $g_u(I_1) - g_d(I_1) \to 0$ when $w(B) \to 0$. i.e., $g_u(I_1) \to g(p), g_d(I_1) \to g(p)$ when $B \to p$, $p \in \mathbb{R}^2$ is a point. This property guarantees that we can successfully check $g(B)$ is positive or negative by bounding polynomials and subdivision.

Given a system $F = (f_1, f_2)$ and a box $B = I_1 \times I_2$, let T_1, T_2 denote the set of the candidate intervals of $f_1(I_1, y)$, $f_2(I_1, y)$ respectively, and let $\{t_1 \cap t_2 \cap I_2 | t_1 \in T_1, t_2 \in T_2\} = \{J_1, \ldots, J_m\}$. We call $\{I_1 \times J_1, \ldots, I_1 \times J_m\}$ the **candidates** of F in B. It is obvious that all the real zeros of F in B are in the candidates. One can find more details in [9]. If the width of I_1 is large, we can split it into n parts and compute the candidates separately. We can write it as an algorithm **Candidates**(F, B, n). Notice that F may not have a root in these candidate boxes (we can't even ensure that $S_1 = \mathbb{V}(f_1)$ and $S_2 = \mathbb{V}(f_2)$ both pass through these boxes), so we may need to subdivide them. The main purpose of computing candidates is to delete these many boxes which S_1 or S_2 do not pass through. Thus, only a few candidate boxes need to be computed, this improves the speed of our algorithm. It is different from other subdivision methods.

Here is an example to show how to compute the candidates of a system in a box.

Example 2.2. Let $F = (x^2 + y^2 - 2, x^2 - y)$, $B = [-2, 2] \times [-2, 2], n = 6$. See Figure 2, we split B into six parts and compute the upper and lower polynomials in every small box, and finally select C_1, C_2 as the candidate boxes.

3 UNIQUENESS AND EXISTENCE

We will give the uniqueness and existence conditions of a simple zero of a system in a box.

Let $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$, $p \in \mathbb{R}^2$ a zero of F. We say $p \in \mathbb{V}(F)$ is a **simple zero** of $F = 0$ if $\det(J_F(p)) \neq 0$, otherwise we say p is a **singular** or **multiple zero** of $F = 0$.

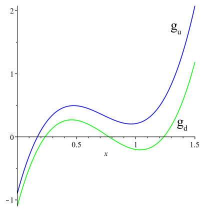

Figure 1: An example for the bounding polynomials.

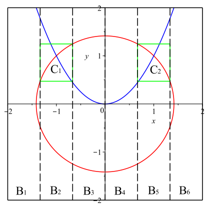

Figure 2: How to find candidates boxes C_1, C_2.

3.1 An orthogonal monotone system in a box

In this subsection, we will give a criterion to determine whether $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$ has at most one real zero in a box in $\mathbb{R}^2$. Our method is based on the geometric properties of the planar curves defined by f_1, f_2.

Let $g \in \mathbb{R}[x, y]$ be a bivariate polynomial and $B \subset \mathbb{R}^2$ a box. We define a sign function first:

$$Sign(g(B)) = \begin{cases} 1, & \text{if } g(B) > 0 \\ -1, & \text{if } g(B) < 0 \\ 0, & \text{if } 0 \in g(B). \end{cases}$$

Next we give the definition of monotonicity of the bivariate polynomial and the properties of the monotone function.

Definition 3.1. Let $g \in \mathbb{R}[x, y]$ and $B \subset \mathbb{R}^2$ a box. We say g is **monotonically increasing** (**decreasing**) in B if $Sign(g_x(B))Sign(g_y(B)) < 0$ ($Sign(g_x(B))Sign(g_y(B)) > 0$).

Remark: A polynomial $g \in \mathbb{R}[x, y]$ is monotonically increasing (decreasing) in $B = I_1 \times I_2$, we may have $\mathbb{V}(g) \cap B = \emptyset$ or $\mathbb{V}(g) \cap B \neq \emptyset$. If $\mathbb{V}(g) \cap B \neq \emptyset$, it means that the function $y = \phi(x)$ satisfying $g(x, \phi(x)) = 0$ is monotonically increasing (decreasing) in B.

Lemma 3.2. *Let $g \in \mathbb{R}[x, y]$, $S = \mathbb{V}(g)$ and $B \subset \mathbb{R}^2$ a box. Denote $g_x = \frac{\partial g}{\partial x}$, $g_y = \frac{\partial g}{\partial y}$. $\forall p_1(x_1, y_1), p_2(x_2, y_2) \in S \cap B$, $p_1 \neq p_2$, we have:*
(a) If $Sign(g_x(B))Sign(g_y(B)) < 0$, then $(x_1 - x_2)(y_1 - y_2) > 0$.
(b) If $Sign(g_x(B))Sign(g_y(B)) > 0$, then $(x_1 - x_2)(y_1 - y_2) < 0$.

Proof. We prove only (a) since (b) can be proved in a similar way. Assume that there exist two different points $(x_1, y_1), (x_2, y_2) \in S \cap B$ s.t. $(x_1 - x_2)(y_1 - y_2) \leq 0$. Then we claim that:

$$g_x(p)(x_1 - x_2) + g_y(p)(y_1 - y_2) \neq 0, \forall p \in B. \quad (1)$$

Since $Sign(g_x(B)) \cdot Sign(g_y(B)) < 0$, we have

$$g_x(p) > 0, g_y(p) < 0 \quad or \quad g_x(p) < 0, g_y(p) > 0, \forall p \in B.$$

If one of $x_1 - x_2$ and $y_1 - y_2$ is 0, then the other one is not equal 0, we can easily get equation (1). If $x_1 - x_2 \neq 0$ and $y_1 - y_2 \neq 0$, then we have $x_1 - x_2 < 0, y_1 - y_2 > 0$ or $x_1 - x_2 > 0, y_1 - y_2 < 0$. Thus, we also have (1).

Using mean value theorem for multivariate function [31], there is a point $p' \in B$, s.t. $g_x(p')(x_1 - x_2) + g_y(p')(y_1 -$

$y_2) = 0$. It is a contradiction with equation (1). Therefore, $(x_1 - x_2)(y_1 - y_2) > 0$. □

Definition 3.3. Let $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$ and $B \subset \mathbb{R}^2$ a box. We say F is an **orthogonal monotone system** in B, if one of f_1 and f_2 is monotonically increasing in B, and the other one is monotonically decreasing in B.

Using Lemma 3.2, we can prove a nice property of the orthogonal monotone system:

THEOREM 3.4. *If F is an orthogonal monotone system in B, then $F = 0$ has at most one root in B.*

PROOF. Assume that $F = 0$ has two real roots in B: (x_1, y_1), (x_2, y_2). Without loss of generality, we assume that f_1 is monotonically increasing and f_2 is monotonically decreasing in B. By Lemma 3.2, for f_1, $(x_1 - x_2)(y_1 - y_2) > 0$, but for f_2, $(x_1 - x_2)(y_1 - y_2) < 0$. It is a contradiction. Therefore, $F = 0$ has at most one root in B. □

The following figure shows an illustration for Theorem 3.4.

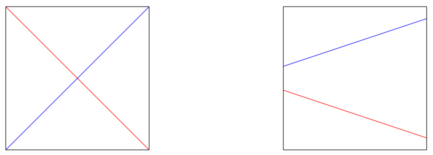

Figure 3: An orthogonal monotone system in a box.

We give the following algorithm to determine if a system F is an orthogonal monotone system in B.

Algorithm 1 $A =$ **IsOMSys**(F, B) :

```
Input: A bivariate polynomial system $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$, a box $B$.
Output: A boolean number to show whether $F$ is an orthogonal monotone system in $B$.
1. Compute $f_{ix} = \frac{\partial f_i}{\partial x}, f_{iy} = \frac{\partial f_i}{\partial y}, i = 1, 2$.
2. Compute $t_1 = Sign(f_{1x}(B))Sign(f_{1y}(B))$, $t_2 = Sign(f_{2x}(B))Sign(f_{2y}(B))$.
3. If $t_1 \cdot t_2 = -1$, return 1; Else return 0.
```

Remark: In the step (2),(3), the signs of $\frac{\partial f_1}{\partial x}(B)$, $\frac{\partial f_1}{\partial y}(B)$ and $\frac{\partial f_2}{\partial x}(B)$, $\frac{\partial f_2}{\partial y}(B)$ can be determined by using upper and lower bounding polynomials.

3.2 How to transform a system to an orthogonal monotone system in a box

In order to make the system F to be an orthogonal monotone system in a box B, we need to transform the system F into an equivalent system MF^T, where M is a 2×2 invertible matrix. The following lemma is well-known:

LEMMA 3.5. *Let $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$ and $M \in \mathbb{R}^{2\times 2}$ be a 2×2 invertible matrix. Then $\mathbb{V}(F) = \mathbb{V}(MF^T)$.*

For a system F, we give the preconditioner which transforms locally the system into a system $UJ_F^{-1}(p) \cdot F^T$, where $U = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ is the rotation matrix (In the following, the notation U always means this matrix) and $p \in B$. In general, we choose p as $m(B)$. The idea is from [14, 18, 27, 28], they transform locally the system F into $J_F^{-1}(p) \cdot F^T = (\tilde{f_1}, \tilde{f_2})^T$, s.t. $\mathbb{V}(\tilde{f_1})$ and $\mathbb{V}(\tilde{f_2})$ are almost orthogonal to each other in the neighborhood of p. Next, we do a rotation by multiplying the matrix U to make a new system $UJ_F^{-1}(p) \cdot F^T$ which is likely to become an orthogonal monotone system in B. See the example below for illustration.

Example 3.6. Let $F = (y - x^2, x - 2y)$, $B = [-0.1, 0.1] \times [-0.1, 0.1]$. We know that $F = 0$ has a unique root $(0, 0)$ in B. However, F is not an orthogonal monotone system in B since $\frac{\partial (y-x^2)}{\partial x}(0, 0) = 0$. Moreover, no matter how much we shrink the box B, we can not get an orthogonal monotone system. Then, let $G' = J_F^{-1}(m(B)) \cdot F^T = (x - 2x^2, y - x^2)^T$ and $G = UJ_F^{-1}(m(B)) \cdot F^T = (-3x^2 + x + y, -x^2 + x - y)^T = (g, g')^T$. It is easy to check that $g_x(B) > 0, g_y(B) > 0$ and $g'_x(B) > 0, g'_y(B) < 0$, thus G is an orthogonal monotone system in B. See Figure 4.

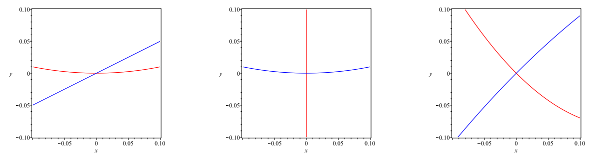

Figure 4: The left figure is of $F = 0$, the middle figure is of $G' = 0$, the right figure is of $G = 0$.

For a point $p \in \mathbb{R}^2$ and a positive number $\delta > 0$, we define a set of box as $B(p, \delta) = \{B | B \text{ is a box and } p \in B, w(B) < \delta\}$. Then, we have the following lemma:

THEOREM 3.7. *Let $F = (f_1, f_2) \in \mathbb{R}[x, y]^2$ and p^* a simple zero of F. Then, $\exists \delta > 0$ s.t. $\forall B \in B(p^*, \delta)$, $UJ_F^{-1}(m(B)) \cdot F^T$ is an orthogonal monotone system in B.*

PROOF. Note that a system F is an orthogonal monotone system in a box B if and only if the product of all the elements of $J_F(B)$ has negative sign. Let $G = UJ_F^{-1}(m(B)) \cdot F^T$, when $B \to p^*$, we have $\lim_{B \to p^*} J_G(B) = \lim_{B \to p^*} UJ_F^{-1}(m(B)) \cdot J_F(B) = UJ_F^{-1}(p^*) \cdot J_F(p^*) = U = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. Therefore we prove the theorem. □

Remark: If p^* is a singular root, then for any box B containing p^*, we can not transform the system F into an orthogonal monotone system in B since $0 = \det(J_F(p^*)) \in \det(J_F(B))$. Hence, our method is invalid for singular roots.

Theorem 3.7 tells us that, for each simple root p^* of F, we can always find a small box B containing p^* s.t. $UJ_F^{-1}(p) \cdot F^T$ is an orthogonal monotone system in B (p is chosen as $m(B)$

and not required to be origin point). Here, the next question is how to determine whether F does have a root in B.

3.3 Existence

In this subsection, we give a method to determine whether an orthogonal monotone system F has a real root or not in B. We introduce some lemmas and one can find more related results in [29].

LEMMA 3.8. *Let $g \in \mathbb{R}[x,y]$ and $S = \mathbb{V}(g)$. $B = I_1 \times I_2$ is a box. If $Sign(g_x(B)) \cdot Sign(g_y(B)) \neq 0$, then*

(a) $\forall \hat{x} \in I_1, \hat{y} \in I_2$, the straight lines $x = \hat{x}$ or $y = \hat{y}$ intersects S at most once in B. Moreover, the straight lines and S are not tangent.

(b) S can not be a loop inside B.

PROOF. By Definition 3.1, we know g is monotone in B. Thus (a) (b) holds by Lemma 3.2. □

LEMMA 3.9. *Let $g \in \mathbb{R}[x,y]$ and $S = \mathbb{V}(g)$, B is a box. If $Sign(g_x(B)) \cdot Sign(g_y(B)) \neq 0$, then B contains at most one component of S.*

PROOF. Assume that the box B contains two components of S. By Lemma 3.8, the curve S intersects every sides of B at most once. Then, consider the two parallel sides of B, g decreases along one side and increases along the other. It is a contradiction with $Sign(g_x(B)) \neq 0$ or $Sign(g_y(B)) \neq 0$. See Fig 5 (a) for illustration. g decreases along the left side and increases along the right side. Notice that when the component intersects B at one point only, we also have the same conclusion, see Fig 5 (b) and (c) for illustration. Therefore, we have that B can not contain two components of S, i.e., B contains at most one component of S. □

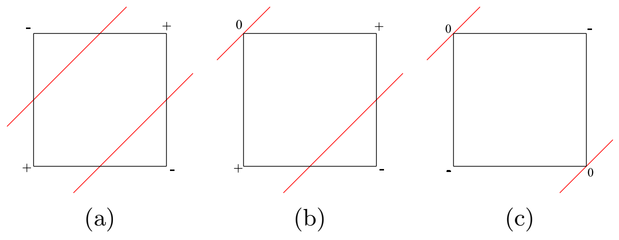

Figure 5: B contains two components of S.

For a box $B = [a_1, b_1] \times [a_2, b_2]$, let $V(B) = \{(a_1, b_1), (a_1, b_2), (a_2, b_1), (a_2, b_2)\}$ be the set of the vertexes of B, $\partial B = \{(x,y) \in B | x = a_1 \text{ or } x = a_2 \text{ or } y = b_1 \text{ or } y = b_2\}$ be the boundaries of B. By Lemma 3.8 and Lemma 3.9, we can directly get that if g is a monotonically function in B, then $\#(S \cap \partial B) \leq 2$, where $\#(A)$ denotes the number of elements of a set A. Next, we determine whether F has a root or not in B based on the three cases: $\#(S \cap \partial B) = 0, 1, 2$.

Let $Sign(g, V(B)) = \{Sign(g(p)) | \forall p \in V(B)\}$. For example, $Sign(g, V(B)) = \{1,1,1,1\}$ means the evaluation of g at the four vertexes are all positive. We can compute $Sign(g, V(B))$ to determine $\#(S \cap \partial B) = 0, 1$ or 2. The case $\#(S \cap \partial B) = 0, 1$ are simple cases, the following lemmas are easy to be proved by Lemma 3.8:

LEMMA 3.10. *Let $g \in \mathbb{R}[x,y]$ and $S = \mathbb{V}(g)$. B is a box. If $Sign(g_x(B))Sign(g_y(B)) \neq 0$, the following are equivalent:*

(1) $S \cap \partial B = \emptyset$.

(2) $S \cap B = \emptyset$.

(3) $Sign(g, V(B)) = \{1,1,1,1\}$ or $\{-1,-1,-1,-1\}$.

LEMMA 3.11. *Let $g \in \mathbb{R}[x,y]$ and $S = \mathbb{V}(g)$. B is a box, p is a vertex of B. If $Sign(g_x(B)) \cdot Sign(g_y(B)) \neq 0$, then the following are equivalent:*

(1) $\#(S \cap \partial B) = 1$.

(2) $S \cap \partial B = \{p\}$.

(3) $S \cap B = \{p\}$.

(4) $Sign(g, V(B)) = \{0,1,1,1\}$ or $\{0,-1,-1,-1\}$.

This case is very special since it needs to check whether a function vanishes exactly or not at a point. It means the function should be exactly evaluated. Of course, we can change the length of the sides to avoid this case when we meet this special case. Next, we will analyze the last case: $\#(S_1 \cap \partial B) = \#(S_2 \cap \partial B) = 2$.

LEMMA 3.12. *Let $F = (f_1, f_2)$ be an orthogonal monotone system, $S_1 = \mathbb{V}(f_1), S_2 = \mathbb{V}(f_2)$ and $\#(S_1 \cap \partial B) = \#(S_2 \cap \partial B) = 2$. Assume that $S_1 \cap \partial B = \{p, p'\}$, we have:*

(1) If $f_2(p)f_2(p') \leq 0$, $F = 0$ has a unique root in B.

(2) If $f_2(p)f_2(p') > 0$, $F = 0$ has no root in B.

PROOF. Since B contains a unique component of S_1, we can parameterize $f_1 = 0$ with an analytic function in B by the implicit function theorem. Assume $(x(t), y(t)), t_0 \leq t \leq t_1$ is the parameterization of $f_1 = 0$ in B from p to p', i.e., we have $(x(t_0), y(t_0)) = p, (x(t_1), y(t_1)) = p'$. Consider the univariate function $g(t) = f_2(x(t), y(t))$. If $f_2(p)f_2(p') \leq 0$, i.e. $g(t_0)g(t_1) \leq 0$, then $\exists t' \in [t_0, t_1]$ s.t. $g(t') = 0$ by the intermediate value theorem. Thus $(x(t'), y(t'))$ is a root of $F = 0$. Since F is an orthogonal monotone system in B, we know that $F = 0$ has a unique root in B. If $f_2(p)f_2(p') > 0$, i.e., $g(t_0)g(t_1) > 0$, then there are even number roots of $g(t) = 0$ in $[t_0, t_1]$. Thus, $F = 0$ has even number roots in B. Since $F = 0$ has most one root in B, we can know that $F = 0$ has no root in B. □

3.4 How to check the existence

We give a method to check the existence of a root of a system in a box in this subsection. Let $F = (f_1, f_2)$ be an orthogonal monotone system in B and the other notations be as above. We assume that $S_1 \cap \partial B = \{k_1, k_1'\}, S_2 \cap \partial B = \{k_2, k_2'\}$. If we can exactly compute the points k_1, k_1', we can easily know that $F = 0$ has a unique root or no root in B by Lemma 3.12. However, it is unnecessary. Notice that one of f_1 and f_2 is monotonically increasing and the other is monotonically decreasing in B, we need only to determine the position of these points. By computing the $Sign(f_1, V(B))$ and $Sign(f_2, V(B))$, we immediately know which sides of B these points k_1, k_1', k_2, k_2' lie on. Then we use the following method to compute $sign(f_2(k_1))$:

(1) If k_1 and k_2, k_2' are not on the same side, then we can arbitrarily take a point p on the side which k_1 are on, we know that $sign(f_2(p)) = sign(f_2(k_1))$. See the left one in Figure 6.
(2) If k_1 and k_2 (k_2') are on the same side and $k_1 \neq k_2$ (k_2'). Then we can use the bisection method to get a point p on the side separating k_1, k_2 (k_2'), which means that p divides the side into two parts, k_1 in the one part and k_2 (k_2') on the other part. Then we have $sign(f_2(p)) = sign(f_2(k_1))$. See the right one in Figure 6.

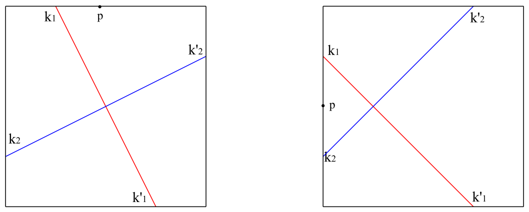

Figure 6: Examples for $\#(S_1 \cap \partial B) = \#(S_1 \cap \partial B) = 2$.

By this method, we can easily know $f_2(k_1)f_2(k_1') < 0$ or > 0. The bad case is that k_1, k_2 are on the same side and we can not separate them. This case happens when $k_1 = k_2$ or $||k_1 - k_2||$ is less than the given bisection precision. Based on above discussion, we give the algorithm **Existence**.

Algorithm 2 $A =$**Existence**(F, B, ϵ_b) :

```
Input: An orthogonal monotone system $F = (f_1, f_2)$ in a box $B$, a given bisection precision $\epsilon_b > 0$.
Output: 1,0,or Unknown.
1. Compute $Sign(f_1, V(B))$ and $Sign(f_2, V(B))$.
2. If $Sign(f_1, V(B))$ or $Sign(f_2, V(B))$ is $\{1,1,1,1\}$ or $\{-1,-1,-1,-1\}$, then return 0. (Lemma 3.10)
3. Else if $Sign(f_1, V(B))$ or $Sign(f_2, V(B))$ is $\{0,1,1,1\}$ or $\{0,-1,-1,-1\}$, then (Lemma 3.11)
   (1) Let $S_1 \cap B = \{p^*\}$ or $S_2 \cap B = \{p^*\}$.
   (2) If $f_1(p^*) = 0$ and $f_2(p^*) = 0$, then return 1.
   (3) Else, return 0.
4. Else, let $\mathbb{V}(f_1) \cap \partial B = \{k_1, k_1'\}$, $\mathbb{V}(f_2) \cap \partial B = \{k_2, k_2'\}$. Let $L_1, L_1'$ denote the two side $k_1, k_1'$ are on.
   (1) For $i = 1$ to 2 do.
       (a) If $i = 1$ let $L = L_1$, $k = k_1$, else, let $L = L_1'$, $k = k_1'$.
       (b) If $k_2$ and $k_2'$ are not on $L$, arbitrarily take a point $p_i \in L$ and compute $t_i = sign(f_2(p_i))$.
       (c) Else if $k_2$ or $k_2'$ are on $L$, we use the bisection method separating $k, k_2$ or $k, k_2'$ on $L$.
           (i) If we separate $k, k_2$ at the point $p_i$, then compute $t_i = sign(f_2(p_i))$.
           (ii) Else we reach a given precision and can not separate $k, k_2$ or $k, k_2'$, then let $t_i = 0$.
   (2) If $t_1 t_2 = 0$, then return Unknown.
   (3) Else if $t_1 t_2 < 0$, then return 1. (Lemma 3.12)
   (4) Else, then return 0. (Lemma 3.12)
```

Remark: $A = 1$ means that the system F has unique root in B, $A = 0$ means that the system F has no root in B. If we can not separate k, k_2 or k, k_2', we will return **Unknown**. We will handle the **Unknown** case in next section.

The correctness of the algorithm is guaranteed by the analysis in Section 3.3. The termination is obvious.

3.5 How to get all real zeros of a system

In this subsection, we will show how to get all the isolating boxes of a given system in $\mathbb{R}^2$.

Considering the coordinate transformation: $x \to \frac{1}{x}, y \to \frac{1}{y}$, we map the interval $(-b, -1)$ to $(-1, -1/b)$ and map the interval $(1, b)$ to $(1/b, 1)$, where $b > 1$. Hence, we need only to consider finding real roots in $[-1, 1] \times [-1, 1]$. Given a system $F = (f_1, f_2)$, let d_1, d_2 be the degrees of f_1, f_2 about x and d_1', d_2' be the degrees of f_1, f_2 about y. Consider the following system:

$$\begin{aligned} F_{(x)} &= (x^{d_1} f_1(\tfrac{1}{x}, y), \quad x^{d_2} f_2(\frac{1}{x}, y)), \\ F_{(y)} &= (y^{d_1'} f_1(x, \tfrac{1}{y}), \quad y^{d_2'} f_2(x, \frac{1}{y})), \\ F_{(xy)} &= (x^{d_1} y^{d_1'} f_1(\tfrac{1}{x}, \tfrac{1}{y}), \quad x^{d_2} y^{d_2'} f_2(\frac{1}{x}, \frac{1}{y})). \end{aligned}$$

We isolate the real roots of these systems $F_{(x)}, F_{(y)}, F_{(xy)}$ in $[-1, 1] \times [-1, 1]$, then we can easily get the isolating boxes of the original system. Notice that when we isolate the real roots of these systems $F_{(x)}, F_{(y)}, F_{(xy)}$ in $B \subset [-1,1] \times [-1,1]$. We need not to consider the zero $(0, 0)$ since it is not related to any zero of the original system. For examples, we can get the isolating boxes of the original system in $[-1, 1] \times (1, \infty)$ by isolating the real roots of $F_{(y)}$ in $[-1, 1] \times (0, 1)$. Doing this way, one advantage is to avoid coefficient expansion since we only compute the univariate interval polynomial in some boxes $B \subset [-1, 1] \times [-1, 1]$.

Example 3.13. Let $F = (x^2 + y^2 - 8, x - y)$, then $F_{(xy)} = (y^2 + x^2 - 8x^2y^2, y - x)$. We can isolate the real roots of $F_{(xy)}$ in $(0, 1) \times (0, 1)$ and get only real zero $(\frac{1}{2}, \frac{1}{2})$. Then we know that $(2, 2)$ is the only one real zero of the original system F in $(1, \infty) \times (1, \infty)$. By isolating real roots of $F, F_{(x)}, F_{(y)}, F_{(xy)}$ in their corresponding intervals, we can obtain all the real roots of the original system.

If F has singular zeros, we can not determine whether a box contains only a singular zero or not. In that case, we will repeat subdividing the candidate boxes until the widths of the obtained boxes are less than a given termination precision $\epsilon > 0$. Finally we get some suspected root boxes s.t. each of them may contain several zeros (counting the multiplicities of the zeros) or no zero.

We give the following example to show that the existence and uniqueness conditions can be applied for the non-polynomial case.

Example 3.14. Let $F = (f_1, f_2) = (e^{x-y} - 2y, 2sin(x + y) + 2x^2 - 1)$, $B = [0, \frac{1}{2}] \times [0, \frac{1}{2}]$. We have $\frac{\partial f_1}{\partial x} = e^{x-y}$, $\frac{\partial f_1}{\partial y} = -e^{x-y} - 2$, and $\frac{\partial f_2}{\partial x} = 2cos(x + y) + 4x$, $\frac{\partial f_2}{\partial y} = 2cos(x +$

y). Then we know $\frac{\partial f_1}{\partial x}(B) > 0, \frac{\partial f_1}{\partial y}(B) < 0$ and $\frac{\partial f_2}{\partial x}(B) > 0, \frac{\partial f_2}{\partial y}(B) > 0$. Hence, F is an orthogonal monotone system in B. Next we check the existence condition. Since $f_1(0,0) > 0, f_1(\frac{1}{2},0) > 0, f_1(0,\frac{1}{2}) < 0, f_1(\frac{1}{2},\frac{1}{2}) = 0$, we have that the curve $S_1 = \mathbb{V}(f_1)$ intersect ∂B twice, i.e., $S_1 \cap \partial B = \{p,q\}$ where $p = (\frac{1}{2},\frac{1}{2})$, $q = (0,y^*)$ with $y^* \in (0,\frac{1}{2})$. We can get the evaluation of f_2 at p,q: $f_2(p) > 0, f_2(q) < 0$. Thus F has a unique root in B by Lemma 3.12.

4 ALGORITHM

Based on the discussion above, we have the main algorithm **Birealrootfinding** below:

```
Algorithm 3   R, SR =Birealrootfinding($F, B, \epsilon, n$) :
Input: A bivariate polynomial system $F = (f_1, f_2) \in \mathbb{R}[x,y]^2$, a box $B$, termination precision $\epsilon > 0$, an integer $n > 1$.
Output: An isolating box set R and a suspected root box set SR.
1. Let R = $\emptyset$, SR = $\emptyset$.
2. CS =Candidates($F, B, n$).
3. While CS $\neq \emptyset$:
   (1) Take one element $C = C_x \times C_y$ out of CS.
   (2) $F' = UJ_F^{-1}(m(C)) \cdot F^T$.
   (3) $A$ =IsOMSys($F', C$).
       (a) If $A = 1$, then
           (i) $A'$ =Existence($F', C, \epsilon$).
           (ii) If $A' = 1$ then, put $C$ into R.
           (iii) If $A'$ =Unknown then, put $C$ into SR.
       (b) Else if $w(C_x) > \epsilon$, then
           (i) CS' =Candidates($F, C, 2$).
           (ii) Put CS' into CS.
       (c) Else, put $C$ into SR.
4. Return R,SR.
```

Remark: For a system F with only simple root, we usually obtain **SR** $= \emptyset$, it means we get all the simple root of F. The bad case is that there are some roots on the boundary of some candidates boxes, then we will return a **Unknown** in **Existence** since we can not split k, k_2 in **Existence**. We can avoid this case by changing the length of the box or combining two adjacent boxes with the same output "Unknown" to form a new box. Subdividing the new box and check the conditions, one usually succeed in finding the results.

The correctness of the algorithm **Birealrootfinding** is guaranteed by Theorem 3.4 and Lemmas 3.10, 3.11, 3.12. The termination is guaranteed by Theorem 3.7, the convergence of bounding polynomials [8] and the given termination precision ϵ.

5 EXPERIMENTS

We implement our algorithm in Maple 2016, and we do some experiments in Maple under Windows 7 with a computer of 8 Intel i7-4790 CPU and 8 GB RAM. The codes will be put on the web later.

Let $F = (f_1, f_2)$. Let $d, t, |c|$ denote the maximal total degree, the maximal number of terms, the maximal absolute value of coefficients of f_1, f_2. We isolate the real roots of these system in $B = [-10, 10] \times [-10, 10]$. The termination precision $\epsilon = 10^{-6}$, $n = 11$. The results are in Table 1.

example	d	t	$\lvert c\rvert \leq$	Times
poly1	8	9	21	1.529s
poly2	3	4	2	0.437s
poly3	14	50	198	84.427s
poly4	14	50	198	62.884s
poly5	14	50	198	87.205s
poly6	200	50	5	1601.132s
poly7	200	500	100	8937.609s
poly8	500	50	5	13771.659s
poly9	500	500	100	263009.824s
poly10	1000	50	5	43444.313s
poly11	1000	500	100	736920.972s

Table 1: Comparison for systems with different sizes

The poly 1 is $F = (x^2 + y^2 - 1, 2x^8 + 8x^6y^2 + 12x^4y^4 + 8x^2y^6 + 2y^8 - x^6 - 21x^4y^2 + 9x^2y^4 - 3y^6)$. The poly 2 is $F = (0.3x^3 - 1.72xy^2 - 1.04xy - 0.46, -1.92x^2y + 1.7y^3 + 0.96)$. The poly3,4,5 are generated as below: randomly generate f and g in the variables x, y, z and set $f_1 = discrim(f, z)$ and $f_2 = resultant([f, g], z)$. Let the system poly3 be (f_1, f_2), poly4 be $(f_1, f_2 - 0.0001)$ and poly5 be $(f_1, f_2 - 0.000001)$. We have that the system poly3 has singular roots and the systems poly4,5 which are derived from poly3 with a slight perturbation of f_2 are nearly singular systems. We can find that the slighter the perturbation, the longer the computing time, besides that, the singular system poly3 and the slighter perturbation system poly5 both get some suspected root boxes. The poly 6,7,8,9,10,11 are randomly generated.

We compare the computing times of our method with LUR method [9] and Isolate (the command in Maple in RootFinding package). We consider different polynomial systems with different degrees, number of terms. For each polynomial system of the same type, we randomly generate 3 polynomial systems and calculate average time. We isolate all the real roots of these systems with our method and set the termination precision $\epsilon = 10^{-4}$, $n = 11$. The results are in Table 2.

In Table 2, " \" means it is unable to allocate memory or more than 10 hours. One can find that our method is slower than LUR when we compute the polynomial system with low degree, but LUR can not handle the polynomial systems with high degrees. Our method has great superiority especially for the systems with high degrees and sparse terms. Notice that our method can be parallelized, it will greatly improve the efficiency of our algorithm.

6 CONCLUSION

In this paper, we propose a new method to isolate the real roots of a bivariate system $(f_1, f_2) \in \mathbb{R}[x,y]^2$. By isolating

d	t	Times		
		Isolate	LUR	our method
20	50	15.606s	0.322s	51.111s
20	dense	39.437s	0.639s	472.345s
50	50	6438.026s	10.873s	126.823s
50	500	\	17.852s	5127.660s
100	10	\	148.752s	33.571s
100	50	\	292.127s	616.090s
100	500	\	694.324	8437.350s
150	10	\	701.906s	222.192s
150	50	\	\	2486.276s

Table 2: Computing time comparison.

the upper and lower bounding polynomials of (f_1, f_2), we obtain real root candidate boxes of the system. We check the uniqueness and existence conditions for each candidate box. If it succeeds, we get an isolate box of the given system. If not, we split these candidate boxes until they satisfy the conditions or their widths reach a given precision. We will realize the parallel computing and isolate the real zeros of non-polynomial bivariate systems in the implementation in the full version of this work. In the future, we will extend this method to multivariate systems.

REFERENCES

[1] O. Aberth. 2007. *Introduction to precise numerical methods* (2nd ed.). Elsevier/Academic Press, Amsterdam.

[2] E. Berberich, P. Emeliyanenko, and M. Sagraloff. 2011. An Elimination Method for Solving Bivariate Polynomial Systems: Eliminating the Usual Drawbacks. In *Meeting on Algorithm Engineering and Expermiments.*

[3] Y. Bouzidi, S. Lazard, G. Moroz, M. Pouget, F. Rouillier, and M. Sagraloff. 2016. Solving bivariate systems using rational univariate representations. *Journal of Complexity* 37 (2016), 34–75.

[4] M. Burr, S. W. Choi, B. Galehouse, and C. Yap. 2012. Complete subdivision algorithms, II: Isotopic meshing of singular algebraic curves. *Journal of Symbolic Computation* 47, 2 (2012), 131–152.

[5] M. Burr, S. Gao, and E. Tsigaridas. 2017. The complexity of an adaptive subdivision method for approximating real curves. In *Proceedings of ISSAC'2017.* ACM, 61–68.

[6] L. Busé, H. Khalil, and B. Mourrain. 2005. Resultant-based methods for plane curves intersection problems. In *International Workshop on Computer Algebra in Scientific Computing.* Springer, 75–92.

[7] J. S. Cheng, X. S. Gao, and J. Li. 2009. Root isolation for bivariate polynomial systems with local generic position method. In *Proceedings of ISSAC'2009.* ACM, 103–110.

[8] J. S. Cheng, X. S. Gao, and C. K. Yap. 2009. Complete numerical isolation of real roots in zero-dimensional triangular systems. *Journal of Symbolic Computation* 44, 7 (2009), 768–785.

[9] J. S. Cheng and K. Jin. 2015. A generic position based method for real root isolation of zero-dimensional polynomial systems. *Journal of Symbolic Computation* 68 (2015), 204–224.

[10] J. S. Cheng, J. Wen, and W. Zhang. 2018. Plotting Planar Implicit Curves and Its Applications. In *International Congress on Mathematical Software.* Springer, 113–122.

[11] R. M. Corless, P. M. Gianni, and B. M. Trager. 1997. A reordered Schur factorization method for zero-dimensional polynomial systems with multiple roots. In *ISSAC*, Vol. 97. 133–140.

[12] D. I. Diochnos, I. Z. Emiris, and E. P. Tsigaridas. 2009. On the asymptotic and practical complexity of solving bivariate systems over the reals. *Journal of Symbolic Computation* 44, 7 (2009), 818–835.

[13] I. Z. Emiris and E. P. Tsigaridas. 2005. Real solving of bivariate polynomial systems. In *International Workshop on Computer Algebra in Scientific Computing.* Springer, 150–161.

[14] J. Garloff and A.P. Smith. 2000. Investigation of a subdivision based algorithm for solving systems of polynomial equations. (2000).

[15] J. Garloff and A.P. Smith. 2001. Solution of systems of polynomial equations by using Bernstein expansion. In *Symbolic Algebraic Methods and Verification Methods.* Springer, 87–97.

[16] H. Hong, M. Shan, and Z. Zeng. 2008. Hybrid method for solving bivariate polynomial system. *SRATC'2008* (2008).

[17] R. Imbach. 2016. A Subdivision Solver for Systems of Large Dense Polynomials. *arXiv:1603.07916* (2016).

[18] J. B. Kioustelidis. 1978. Algorithmic error estimation for approximate solutions of nonlinear systems of equations. *Computing* 19, 4 (1978), 313–320.

[19] A. Kobel and M. Sagraloff. 2015. On the complexity of computing with planar algebraic curves. *Journal of Complexity* 31, 2 (2015), 206–236.

[20] R. Krawczyk. 1969. Newton-algorithmen zur bestimmung von nullstellen mit fehlerschranken. *Computing* 4, 3 (1969), 187–201.

[21] J. M. Lien, V. Sharma, G. Vegter, and C. Yap. 2014. Isotopic Arrangement of Simple Curves: An Exact Numerical Approach Based on Subdivision. In *ICMS 2014.* Springer, 277–282. LNCS No. 8592. Download from http://cs.nyu.edu/exact/papers/ for version with Appendices and details on MK Test.

[22] Z. Lu, B. He, Y. Luo, and L. Pan. 2005. An algorithm of real root isolation for polynomial systems. *Proceedings of Symbolic Numeric Computation* (2005), 94–107.

[23] A. Mantzaflaris, B. Mourrain, and E. Tsigaridas. 2011. On continued fraction expansion of real roots of polynomial systems, complexity and condition numbers. *Theoretical Computer Science* 412, 22 (2011), 2312–2330.

[24] C. Miranda. 1940. *Un'osservazione su un teorema di Brouwer.* Consiglio Nazionale delle Ricerche.

[25] R. E. Moore. 1966. *Interval analysis.* Vol. 4. Prentice-Hall Englewood Cliffs, NJ.

[26] R. E. Moore. 1977. A test for existence of solutions to nonlinear systems. *SIAM J. Numer. Anal.* 14, 4 (1977), 611–615.

[27] R. E. Moore and J. B. Kioustelidis. 1980. A simple test for accuracy of approximate solutions to nonlinear (or linear) systems. *SIAM J. Numer. Anal.* 17, 4 (1980), 521–529.

[28] B. Mourrain and J. P. Pavone. 2009. Subdivision methods for solving polynomial equations. *Journal of Symbolic Computation* 44, 3 (2009), 292–306.

[29] S. Plantinga and G. Vegter. 2004. Isotopic approximation of implicit curves and surfaces. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing.* ACM, 245–254.

[30] X. Qin, Y. Feng, J. Chen, and J. Zhang. 2013. Parallel computation of real solving bivariate polynomial systems by zero-matching method. *Appl. Math. Comput.* 219, 14 (2013), 7533–7541.

[31] W. Rudin. 1976. Principles of mathematical analysis. (1976).

[32] S. M. Rump. 1983. Solving algebraic problems with high accuracy. In *A new approach to scientific computation.* Elsevier, 51–120.

[33] E. C. Sherbrooke and N. M. Patrikalakis. 1993. Computation of the solutions of nonlinear polynomial systems. *Computer Aided Geometric Design* 10, 5 (1993), 379–405.

[34] S. Smale. 1986. Newtons method estimates from data at one point. In *The merging of disciplines: new directions in pure, applied, and computational mathematics.* Springer, 185–196.

[35] A. Strzebonski. 2008. Real root isolation for exp-log functions. In *Proceedings of ISSAC'2008.* ACM, 303–314.

[36] A. Strzebonski. 2009. Real root isolation for tame elementary functions. In *Proceedings of ISSAC'2009.* ACM, 341–350.

[37] A. Strzebonski. 2012. Real root isolation for exp–log–arctan functions. *Journal of Symbolic Computation* 47, 3 (2012), 282–314.

[38] S. Telen, B. Mourrain, and M. V. Barel. 2019. Truncated normal forms for solving polynomial systems. *ACM Communications in Computer Algebra* 52, 3 (2019), 78–81.

[39] B. Xia and T. Zhang. 2006. Real solution isolation using interval arithmetic. *Computers and Mathematics with Applications* 52, 6-7 (2006), 853–860.