COMPUTING ROOTS OF SYSTEMS OF POLYNOMIALS BY LINEAR CLIPPING

MICHAEL BARTOŇ AND BERT JÜTTLER

ABSTRACT. We present an algorithm which computes all roots of a given bivariate polynomial system within a given rectangular domain. In each step, we construct the best linear approximants with respect to the L^2 norm and use them to define planar strips enclosing the zero sets of the two polynomials. Since both polynomials are described by their Bernstein-Bézier representations, the computation of these strips is computationally efficient. The two strips lead to a reduced domain, which is obtained by intersecting the current domain with the smallest axis-aligned bounding box enclosing the intersection of both strips. It is shown that the sequence of boxes generated by the algorithm possesses convergence rate 2 at single roots. The method can be adapted to polynomials with interval coefficients.

1. INTRODUCTION

Systems of polynomial equations appear in the context of various applications. They are ubiquitous in the field of geometric computing and Computer Aided Design [3], where free-form curves and surfaces are usually described by piecewise polynomial parametric representations. In particular, intersection algorithms often involve numerical methods for solving systems of polynomial equations [10]. The problem of inverse kinematics for serial manipulators in robotics leads to polynomial systems [9]. Other applications include numerical simulations in computational elastoplasticity [24], where a certain bivariate system has to be solved in each iteration step.

The investigation of numerical algorithms for solving polynomial systems has been an active research area for a long time. Many related references can be found in [16].

For instance, *homotopy techniques* form an important class of algorithms. These techniques (see, e.g., [11, 23]) start with the solutions of a simpler system with the same structure of the set of solutions. This system is then continuously transformed into the original system, and the solutions are found by tracing the solutions. Homotopy techniques are particularly well suited for $\Omega = \mathbb{C}^n$.

Another class of algorithm combines *bisection* steps with Descartes' rule of signs in order to isolate the roots [2, 15, 20]. Recently, this technique has been adapted to the case of univariate spline functions [14].

The rich literature on roots of polynomials also contains various results on *enclosures* of polynomials and their roots, e.g., [7, 12, 21]. In particular, techniques of interval and affine arithmetic have been used to deal with the effects of uncertainties and numerical errors.

We will focus on polynomials given in Bernstein–Bézier (BB) representation. This representation forms an essential part of the technology for free-form curves and surfaces

Key words and phrases. Polynomial systems, root solver, intersection of algebraic curves, linear enclosure.

Both authors were supported by Austrian Science Fund (FWF) through SFB F013 "Numerical and Symbolic Scientific Computing", subproject 15.

in Computer Aided Design [3, 10, 19]. Compared to other representations, it has two main advantages.

First, the BB representation is *numerically stable*, see [4]. This observation also applies to evaluation and to the computation of BB representations with respect to subdomains via de Casteljau's algorithm or other suitable algorithms (cf. [13]).

Second, the BB representation provides the convex hull and variation diminution properties. Consequently, error bounds and bounds on the number of roots can directly be generated from the coefficients.

Robust algorithms for solving systems of polynomial equations, which are based on the BB representation, find all roots in a bounded domain [6, 17, 22]. When combined with a local preconditioning step, they achieve second order convergence for single roots [15].

In this paper we describe a new algorithm for computing all roots of a system of polynomials within a given domain. After formulating the problem and analyzing the best linear approximation of bivariate polynomials, we describe the algorithm and discuss its convergence rate. The theoretical results will then be illustrated by several examples. Finally we conclude this paper.

2. Preliminaries

2.1. **The root–finding problem.** We consider a system of d polynomial equations in d unknowns. In order to simplify the presentation, we restrict ourselves to the case of two polynomial equations in two unknowns ($d = 2$). All results can be extended to any value of d. Let

$$\left.\begin{array}{rcl} p(x,y) &=& 0 \\ q(x,y) &=& 0 \end{array}\right\} \tag{1}$$

be the given system of two polynomial equations. All solutions (roots) within given domain

$$D = [\alpha, \beta] \times [\gamma, \delta] \tag{2}$$

are to be found. More precisely, we want to generate a set of domains of maximum diameter 2ε which contain the roots, where the parameter ε specifies the desired accuracy.

In the remainder of the paper, the notion of domain always refers to the Cartesian product of two intervals, as in (2). We denote with

$$a = \text{area}(D) = (\beta-\alpha)(\delta-\gamma) \quad \text{and} \quad h = \text{diam}(D) = \sqrt{(\beta-\alpha)^2 + (\delta-\gamma)^2} \tag{3}$$

the *area* and the *diameter* of a domain, respectively.

We assume that both p and q have the bidegree (m, n) (or less) with respect to x and y. Consequently, the polynomials belong to the $(m + 1)(n + 1)$ dimensional linear space $\Pi^{m,n}$ of all polynomials of bidegree less than or equal to (m, n). As a basis of this space, we choose the tensor-product of the Bernstein polynomials

$$U_i^m(x) = \binom{m}{i} \frac{(x-\alpha)^i(\beta-x)^{m-i}}{(\beta-\alpha)^m}, \quad V_j^n(y) = \binom{n}{j} \frac{(y-\gamma)^j(\delta-y)^{n-j}}{(\delta-\gamma)^n} \tag{4}$$

with respect to the intervals $[\alpha, \beta], [\gamma, \delta] \subset \mathbb{R}$, respectively. Any polynomial $p \in \Pi^{m,n}$ has a Bernstein-Bézier (BB) representation

$$p(x,y) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{ij} U_i^m(x) V_j^n(y), \quad [x,y] \in [\alpha,\beta] \times [\gamma,\delta], \tag{5}$$

with respect to the domain $[\alpha, \beta] \times [\gamma, \delta]$, with certain coefficients $b_{ij} \in \mathbb{R}$. See [8, 19] for more information.

2.2. L^2 **norm and best linear approximation.** The space $\Pi^{m,n}$ will be equipped with the L^2 inner product

$$\langle f, g\rangle^D = \int_\alpha^\beta \int_\gamma^\delta f(x,y)\, g(x,y)\,\mathrm{d}y\,\mathrm{d}x \tag{6}$$

with respect to the domain D and the norm

$$\|f\|_2^D = \frac{1}{a}\sqrt{\langle f, f\rangle^D}, \tag{7}$$

where $a = \text{area}(D)$, induced by it. The factor $1/a$ is introduced in order to obtain a norm which is invariant with respect to affine transformations of the x and y–axes. More precisely, for any affine transformation

$$\tau : \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} v_0 \\ v_1 \end{pmatrix} + \begin{pmatrix} u_{00} & 0 \\ 0 & u_{11} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{8}$$

with $u_{00}u_{11} \neq 0$, the norms of f with respect to the domain D and of $f \circ \tau^{-1}$ with respect to the domain $\tau(D)$ are identical,

$$\|f\|_2^D = \|f \circ \tau^{-1}\|_2^{\tau(D)}. \tag{9}$$

There exists a unique linear polynomial $\overline{p}$ which minimizes $\|p - \overline{p}\|_2^D$. This polynomial $\overline{p}$ will be called the *best linear approximant* of p with respect to the domain D. It has the bilinear BB–representation

$$\overline{p}(x,y) = \sum_{i=0}^{1}\sum_{j=0}^{1} c_{ij} U_i^1(x) V_j^1(y), \quad [x,y] \in D, \tag{10}$$

where the coefficients satisfy the constraint

$$c_{10} - c_{00} = c_{11} - c_{01}. \tag{11}$$

Let

$$\mathbf{B} = (b_{kl})_{k=0,\ldots,m,l=0,\ldots,n} \in \mathbb{R}^{(m+1)(n+1)} \text{ and } \mathbf{C} = (c_{kl})_{k=0,\ldots,1,l=0,\ldots,1} \in \mathbb{R}^4 \tag{12}$$

be the vectors obtained by collecting the BB–coefficients of p and $\overline{p}$, respectively. We define the *approximation operator*

$$\mathcal{A} : \mathbb{R}^{(m+1)(n+1)} \to \mathbb{R}^4 : \quad \mathbf{B} \mapsto \mathcal{A}(\mathbf{B}) \tag{13}$$

which assigns to each coefficient vector $\mathbf{B}$ the coefficients of the best linear approximant, $\mathbf{C} = \mathcal{A}(\mathbf{B})$, which is obtained from Eq. (14). It is a linear operator, as the best approximation is simply an orthogonal projection of $\Pi^{m,n}$ into the subspace of linear polynomials. Consequently, the coefficients c_{ij}, $i,j = 0,1$ can be computed as

$$c_{ij} = \sum_{k=0}^{m}\sum_{l=0}^{n} A_{ij}^{kl} b_{kl}, \quad i,j = 0,1, \tag{14}$$

where b_{kl} are the BB–coefficients of p. The coefficients A_{ij}^{kl} of the approximation operator do not depend on p. They satisfy

$$A_{10}^{kl} - A_{00}^{kl} = A_{11}^{kl} - A_{01}^{kl}, \quad k = 0, \ldots, m, l = 0, \ldots, n, \tag{15}$$

as the constraint (11) is satisfied for any choice of $p \in \Pi^{m,n}$. Moreover we have the following result.

Lemma 2.1. *The approximation operator $\mathcal{A}$ is independent of the domain D. More precisely, the coefficients A_{ij}^{kl} in* (14) *do not depend on α, β, γ and δ.*

Algorithm 1 BLC(p, q, D) $\{$Bivariate Linear Clipping$\}$

1: **if** diam$(D) \geq \varepsilon$ **then**
2: $\quad \overline{p}, \overline{q} \leftarrow$ best linear approximants to p, q with respect to $||.||_2^D$
3: $\quad \delta^p, \delta^q \leftarrow$ bounds on $\|p - \overline{p}\|_\infty^D$ and $\|q - \overline{q}\|_\infty^D$
4: $\quad p^U \leftarrow \overline{p} + \delta^p,\ p^L \leftarrow \overline{p} - \delta^p$ $\{$upper and lower bound on $p\}$
5: $\quad q^U \leftarrow \overline{q} + \delta^q,\ q^L \leftarrow \overline{q} - \delta^q$ $\{$upper and lower bound on $q\}$
6: $\quad L \leftarrow$ parallelogram bounded by the $p^U = 0$, $p^L = 0$, $q^U = 0$, $q^L = 0$
7: $\quad R \leftarrow$ axis-aligned bounding box containing L
8: $\quad D' \leftarrow D \cap R$ $\{D' = \emptyset$ is possible$\}$
9: $\quad$ **if** diam$(D') \geq \frac{1}{2}$ diam(D) **then**
10: $\quad\quad$ subdivide the domain, apply BLC to the subdomains, return the results
11: $\quad$ **else**
12: $\quad\quad$ **return** (BLC(p, q, D'))
13: $\quad$ **end if**
14: **else**
15: $\quad$ **return** (D)
16: **end if**

Proof. Any two domains D, D' are related by a unique orientation-preserving (i.e., satisfying $u_{00} > 0$, $u_{11} > 0$) affine transformation τ, which maps D bijectively onto D'. The BB coefficients of p with respect to D the and of $p' = p \circ \tau^{-1}$ with respect to D' are identical, as τ transforms the Bernstein polynomials with respect to D into the Bernstein polynomials with respect to D'. Since the norm is invariant under the affine transformation τ, the BB coefficients of the best linear approximation $\bar{p}$ of p with respect to $||.||_2^D$ are the same as the BB coefficients of the best linear approximation $\bar{p}'$ of p' with respect to $||.||_2^{D'}$. Consequently, the approximation operator is independent of D. □

Similarly we define the *degree elevation operator*

$$\mathcal{E} : \mathbb{R}^4 \to \mathbb{R}^{(m+1)(n+1)} : \quad \mathbf{C} \mapsto \mathcal{E}(\mathbf{C}) \tag{16}$$

which generates from $\mathbf{C}$ the BB coefficients of the representation of $\bar{p}$ as a polynomial of bidegree (m, n). It can be represented analogously to Eq. (10). Again it is independent of the domain, see [8, 19].

3. Computing roots via bivariate linear clipping

3.1. **The algorithm.** The new method for computing the roots of a bivariate polynomial system is described in Algorithm 1, which will be called BLC. Some steps will be explained in more detail. See also Figures 1 and 2.

In line 2 we generate the best linear approximants $\overline{p}$ and $\overline{q}$ of p and q with respect to the L^2 norm on the current domain $D = [\alpha, \beta] \times [\gamma, \delta]$. We use the approximation operator $\mathcal{A}$ and degree elevation operator $\mathcal{E}$ in order to generate the BB representation of the approximants of bidegree (m, n). In order to speed up the computations, the coefficients of the operators $\mathcal{A}$ and $\mathcal{E}$ are precomputed and stored in a lookup table.

In line 3, we need to find a bound δ^p (and similarly for δ^q). We use the convex hull property of BB representations,

$$\delta^p = \max_{i,j} |b_{i,j} - \bar{c}_{i,j}|, \quad i = 0, \ldots, m, j = 0, \ldots, n, \tag{17}$$

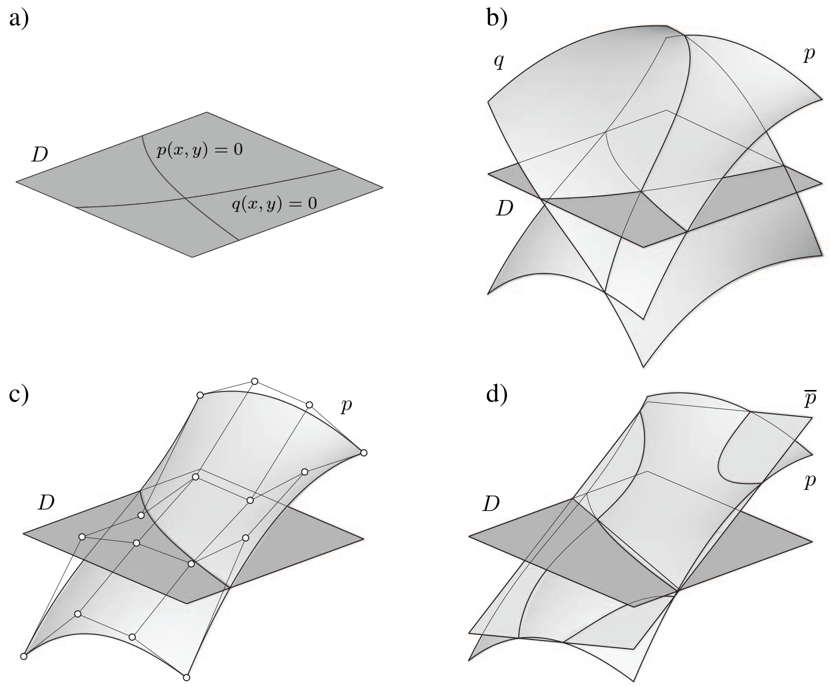


FIGURE 1. One iteration of BLC. a) The polynomial system (1), b) graph surfaces, c) BB representation of p, d) best linear approximant $\overline{p}$.

where $b_{i,j}$ and $\bar{c}_{i,j}$ are the coefficients of the Bernstein–Bézier representations of p and $\overline{p}$ of bidegree (m, n). See Fig. 2e.

In line 6 we compute a parallelogram by intersecting 4 lines. See [18, section 7.2] for a robust technique for intersecting line segments.

If the diameter $\text{diam}(D')$ of this domain D' is sufficiently small, when compared to the length of the diameter of the previous domain D, then BLC is applied to it (line 17). Otherwise we subdivide the original domain D into four pieces and apply BLC to the four subdomains (line 10). Here, we request that the diameter shrinks at least by 50%, but this is just a heuristic setting.

3.2. **Convergence rate.** BLC generates sequences of boxes which converge towards the root(s) of the polynomial system. For each of them, the diameters of the boxes form a monotonically decreasing sequence. We analyze their convergence rates, see [5]. First, in order to make this paper self-contained, the approximation order of the best linear approximant is analyzed.

Lemma 3.1. *For any given polynomial p with domain $D_0 = [\alpha_0, \beta_0] \times [\gamma_0, \delta_0]$, we consider a subdomain $D = [\alpha, \beta] \times [\gamma, \delta] \subseteq D_0$. In line 3 of Algorithm we generate a bound $\delta^p = \delta^p(D)$ as the maximum difference of the BB coefficients with respect to D, see Eq.* (17)*. There exists a constant C_p depending solely on p and D_0, but not on D, such that the bound satisfies*

$$\delta^p \leq C_p\, h^2, \tag{18}$$

where $h = \text{diam}(D)$.

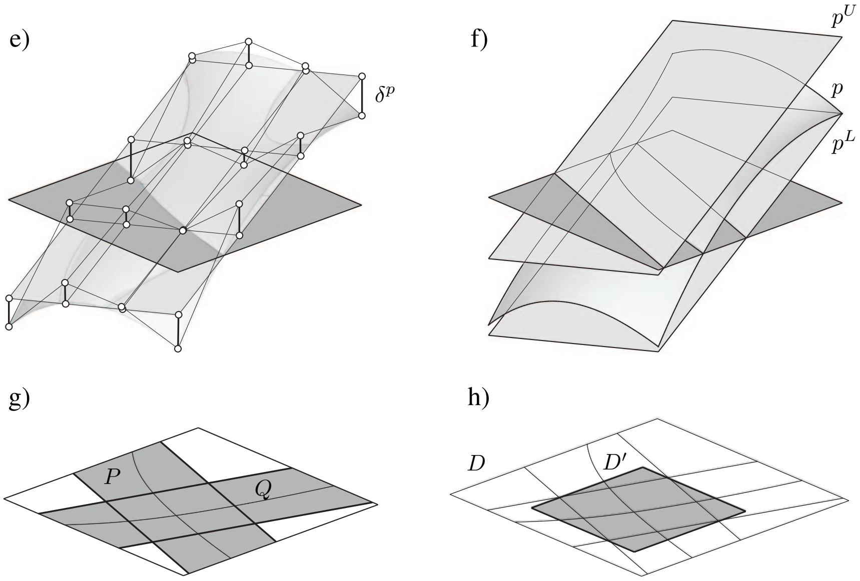


FIGURE 2. One iteration of BLC (continued): e) computation of δ^p, f) lower and upper bounds p^L and p^U, g) bounding strips for the zero sets of p and q, h) the axis-aligned bounding box defines the new domain D'.

Proof. For any domain D we consider the maximum (ℓ_∞) norm $\|.\|^D_{\mathrm{BB},\infty}$ of the BB coefficients, the L^2 norm as defined in (7), and the maximum (L^∞) norm

$$\|r\|^D_\infty = \max_{(x,y)\in D} |r(x,y)|. \tag{19}$$

All norms on the $(m+1)(n+1)$–dimensional real linear space $\Pi^{m,n}$ are equivalent. Hence, there exist constants C_1 and C_2 such that for all polynomials $r \in \Pi^{m,n}$ the inequalities

$$\|r\|^D_{\mathrm{BB},\infty} \le C_1 \|r\|^D_2 \quad \text{and} \quad \|r\|^D_2 \le C_2 \|r\|^D_\infty \tag{20}$$

are satisfied. All norms are invariant with respect to affine transformations of the form (8). Consequently, the constants C_1 and C_2 are independent of the domain D. The bound δ^p generated in line 3 of the algorithm satisfies

$$\begin{aligned} \delta^p = \|p - \overline{p}\|^D_{\mathrm{BB},\infty} &\le C_1 \|p - \overline{p}\|^D_2 \le C_1 \|p - T^p_{(a,b)}\|^D_2 \le C_1 C_2 \|p - T^p_{(a,b)}\|^D_\infty \\ &\le \tfrac{1}{2} C_1 C_2 \max_{(s,t)\in D_0} \left(|(\frac{\partial^2 p}{\partial x^2})(s,t)| + 2|(\frac{\partial^2 p}{\partial x \partial y})(s,t)| + |(\frac{\partial^2 p}{\partial y^2})(s,t)|\right) h^2, \end{aligned} \tag{21}$$

where $T^p_{(a,b)}$ is the linear Taylor polynomial to p at an arbitrary point $(a,b) \in D$. □

As the next step we study the limit of the best linear approximant $\bar{p}$. We present a general result, which does not make any assumptions about the shape of the domain (i.e., the ratio of the lengths of the domain boundaries).

Lemma 3.2. *Consider a contracting sequence of domains $\{D_i\}_{i=1}^\infty$, i.e.,* $\mathrm{diam}(D_i) \to 0$ *as $i \to \infty$. We assume that the sequence is nested, $D_{i+1} \subseteq D_i$. Let (a,b) be the unique limit point of this sequence. For any polynomial p, let $\overline{p}_i$ be the best linear approximant* *with respect to the L^2 norm on D_i. Then*

$$\lim_{i\to\infty}(\nabla\overline{p}_i)(a,b) = (\nabla p)(a,b). \tag{22}$$

Proof. For each domain $D_i = D = [\alpha,\beta]\times[\gamma,\delta]$ we consider the three linear polynomials

$$\Delta_1(x,y) = 7(\beta-\alpha)(\delta-\gamma) - 6(x-\alpha)(\delta-\gamma) - 6(\beta-\alpha)(y-\gamma)$$
$$\Delta_2(x,y) = -6(\delta-\gamma) + 12\frac{x-\alpha}{\beta-\alpha}(\delta-\gamma),\ \Delta_3(x,y) = -6(\beta-\alpha) + 12\frac{y-\gamma}{\delta-\gamma}(\beta-\alpha)$$

which are the unique linear polynomials that satisfy the 9 equations

$$\begin{array}{lll} (1/a^2)\langle\Delta_1,1\rangle^D = 1 & (1/a^2)\langle\Delta_1,x-\alpha\rangle^D = 0 & (1/a^2)\langle\Delta_1,y-\gamma\rangle^D = 0 \\ (1/a^2)\langle\Delta_2,1\rangle^D = 0 & (1/a^2)\langle\Delta_2,x-\alpha\rangle^D = 1 & (1/a^2)\langle\Delta_2,y-\gamma\rangle^D = 0 \\ (1/a^2)\langle\Delta_3,1\rangle^D = 0 & (1/a^2)\langle\Delta_3,x-\alpha\rangle^D = 0 & (1/a^2)\langle\Delta_3,y-\gamma\rangle^D = 1. \end{array} \tag{23}$$

Consequently, they form the *dual basis* of the basis $\{1, x-\alpha, y-\gamma\}$ of the space of linear polynomials with respect to the inner product $(1/a^2)\langle .,.\rangle^D$ on D. Recall that the coefficients of the orthogonal projection of a vector into a subspace of a linear space are the inner products with the dual basis of the basis of that subspace. Therefore, the best linear approximant of any polynomial $p \in \Pi^{m,n}$ with respect to the norm $\|.\|_2^D$ induced by this inner product is

$$\bar{p}(x,y) = (1/a^2)\langle\Delta_1,p\rangle^D + (1/a^2)\langle\Delta_2,p\rangle^D(x-\alpha) + (1/a^2)\langle\Delta_3,p\rangle^D(y-\gamma). \tag{24}$$

A short computation (evaluation of the integrals and taking the limit) confirms that

$$\begin{gathered} \lim_{\substack{\beta\to\alpha\\ \delta\to\gamma}} \frac{1}{a^2}\langle\Delta_1,p\rangle^D = p(\alpha,\gamma), \\ \lim_{\substack{\beta\to\alpha\\ \delta\to\gamma}} \frac{1}{a^2}\langle\Delta_2,p\rangle^D = (\frac{\partial}{\partial x}p)(\alpha,\gamma), \quad \lim_{\substack{\beta\to\alpha\\ \delta\to\gamma}} \frac{1}{a^2}\langle\Delta_3,p\rangle^D = (\frac{\partial}{\partial y}p)(\alpha,\gamma). \end{gathered} \tag{25}$$

As $h = \mathrm{diam}(D) \to 0$ implies $\beta\to\alpha$ and $\delta\to\gamma$, this completes the proof of (22). □

Remark 3.3. Consider a single root (a,b) of the system (1), which is characterized by two linearly independent gradients $(\nabla p)(a,b)$ and $(\nabla q)(a,b)$. If the sequence of domains D_i converges to this root, then the best linear approximants $\bar{p}_i$, $\bar{q}_i$ converge to the tangent planes $T^p_{(a,b)}$, $T^q_{(a,b)}$ of the graphs of p and q at the root. Their intersections with the plane $z = 0$ converge to the tangents at the root. See Fig. 3.

Theorem 3.4. *Consider a single root* (a,b) *of the system* (1) *within the given domain* $D = [\alpha,\beta]\times[\gamma,\delta]$. BLC *generates a sequence of domains* $(D_i)_{i=1,\dots,\infty}$ *converging towards this root. The sequence of diameters* $(h_i)_{i=1,\dots,\infty}$ *of these domains possesses the convergence rate* 2.

Proof. We denote with $\phi \in (0,\frac{\pi}{2})$ the angle enclosed by the tangent plane $T^p_{(a,b)}$ of the graph of p at (a,b) and the plane $z = 0$. Let $\phi_i \in (0,\frac{\pi}{2})$ be the angle enclosed by the best linear approximant $\bar{p}_i$ of p with respect to D_i and the plane $z = 0$, see Fig. 3. We have $\phi \neq 0$ as (a,b) is a single root. Due to Lemma 3.2,

$$\phi_i \geq \frac{\phi}{2}, \quad \text{hence} \quad \cot\phi_i \leq \cot\frac{\phi}{2}, \tag{26}$$

holds for all but finitely many values of i.

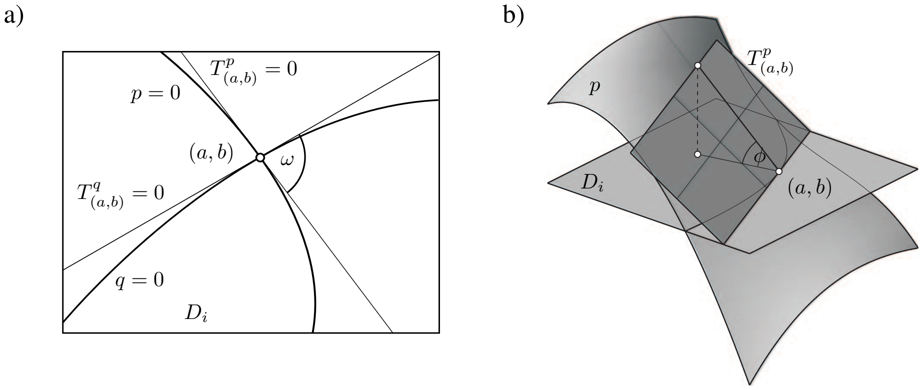


FIGURE 3. a) Polynomial system with a single root, b) graph of p and its tangent plane $T^p_{(a,b)}$ with slope ϕ.

We denote with d_i^p the distance of the two parallel lines $p^U = 0$ and $p^L = 0$, which are constructed in line 6 of the Algorithm BLC, where the domain is $D = D_i$. Due to Lemma 3.1 and to the bound (26) on the slope of p^U and p^L,

$$d_i^p \leq 2\delta^p \cot\frac{\phi}{2} \leq 2C_p h_i^2 \cot\frac{\phi}{2} = \bar{C}_p h_i^2 \tag{27}$$

with $\bar{C}_p = 2C_p \cot\frac{\phi}{2}$, holds for all but finitely many values of i, see Fig. 4. Analogously, we obtain a bound on the distance d_i^q of the two parallel lines $q^U = 0$ and $q^L = 0$,

$$d_i^q \leq \bar{C}_q h_i^2. \tag{28}$$

Let $\omega \in (0, \frac{\pi}{2}]$ be te angle enclosed by $\nabla p(a, b)$ and $\nabla q(a, b)$, see Fig. 3a. According to Remark 3.3, the angle $\omega_i \in (0, \frac{\pi}{2}]$ enclosed by the lines $\bar{p}_i = 0$ and $\bar{q}_i = 0$ satisfies

$$\omega_i \geq \frac{\omega}{2}, \quad \text{hence} \quad \frac{1}{\sin\omega_i} \leq \frac{1}{\sin\frac{\omega}{2}}, \tag{29}$$

for all but finitely many values of i, where $\bar{p}_i$ and $\bar{q}_i$ are the best linear approximants of p and q with respect to the L_2 norm on D_i. Clearly, ω_i is the angle enclosed by the two planar strips whose intersection defines the parallelogram L in line 6 of the algorithm, see Fig. 5. As the edges of the parallelogram L have the lengths

$$\frac{d_i^p}{\sin\omega_i} \quad \text{and} \quad \frac{d_i^q}{\sin\omega_i}, \tag{30}$$

the diameter of L is bounded by

$$\text{diam}(L) \leq (d_i^p + d_i^q)\frac{1}{\sin\omega_i} \leq (d_i^p + d_i^q)\frac{1}{\sin\frac{\omega}{2}} \leq (\bar{C}_p + \bar{C}_q)h_i^2\frac{1}{\sin\frac{\omega}{2}} \tag{31}$$

for all but finitely many values of i, see (27) and (28). Finally, the diameter of the next domain $D_{i+1} = D_i \cap R$, where R is the axis–aligned bounding box of L, which is generated in line 8 of algorithm BLC, satisfies

$$h_{i+1} = \text{diam}(D \cap R) \leq \text{diam}(R) \leq \sqrt{2}\,\text{diam}(L) \leq \sqrt{2}\,(\bar{C}_p + \bar{C}_q)\frac{1}{\sin\frac{\omega}{2}}\; h_i^2. \tag{32}$$

This completes the proof. □

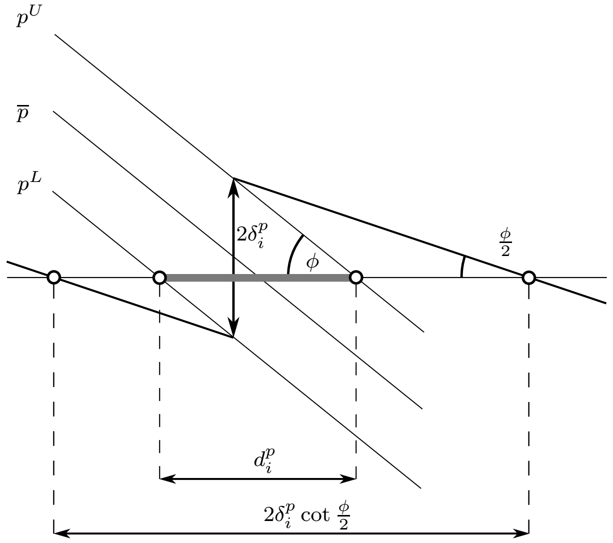


FIGURE 4. The bound on the d_i^p. The figure shows the projection into the plane perpendicular to $z = 0$ and $\overline{p}$. The strip enclosed by $p^U = 0$ and $p^L = 0$ is shown in grey.

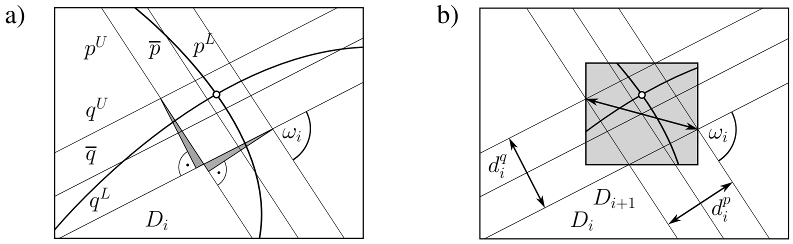
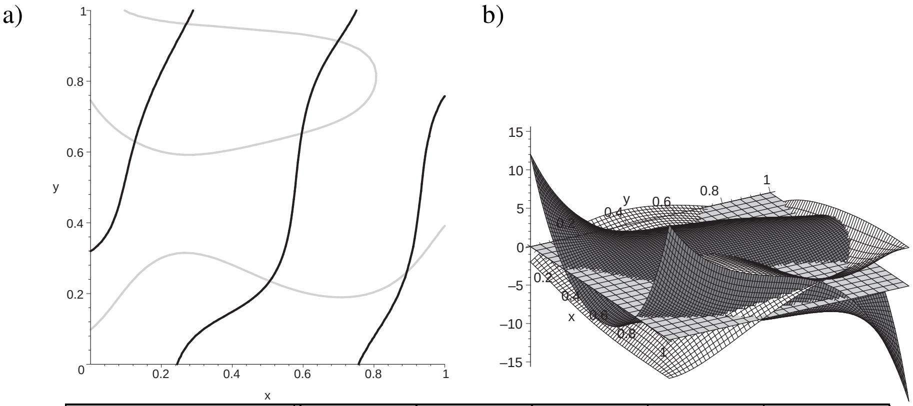


FIGURE 5. a) Two strips bounded by $q_i^U = 0$, $q_i^L = 0$ and $p_i^U = 0$ define the parallelogram L; the length of the edges can be computed with the help of the grey triangles. b) The diameter $\leftrightarrow$ of the parallelogram and the new domain D_{i+1}.

4. NUMERICAL EXAMPLES

The computing times were obtained from an implementation in C, running on a Linux PC with an Intel(R) Xeon(TM) CPU (2.40GHz) with 1.98GB of RAM. In order to obtain a measurable quantity of execution time, a loop of 10^3 repetitions was measured and averaged.



ε	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
clipping steps	22	29	34	36	36
subdivision steps	15	15	15	15	15
no root ($D' = \emptyset$)	37	40	40	40	40
all iterations	74	84	89	91	91
no. of roots	9	6	6	6	6
time (ms)	1.97	2.34	2.83	2.92	3.08

FIGURE 6. Example 4.1 Top: a) The zero sets of polynomials p and q, b) the graphs of p and q. Bottom: Computation time and number of iterations for various values of ε.

Example 4.1. (Single root) We applied algorithm BLC to two polynomials p and q of bidegree $(5, 5)$ with the BB coefficients

$$P^{5,5} = \begin{bmatrix} 12 & -2 & -2 & -2 & -2 & -15 \\ -3 & 3 & 3 & 3 & 3 & 3 \\ -5 & 5 & 5 & 5 & 5 & 3 \\ -5 & 5 & 5 & 5 & 5 & 5 \\ -5 & -15 & -15 & -15 & -15 & 5 \\ 15 & 5 & 5 & 5 & 5 & -15 \end{bmatrix}, \quad Q^{5,5} = \begin{bmatrix} -2 & 3 & 3 & 2 & -1 & -2 \\ -3 & -3 & -1 & 10 & -15 & 3 \\ -5 & -5 & -1 & 10 & -15 & 3 \\ -5 & 5 & 5 & 10 & -15 & 5 \\ -5 & 5 & 5 & 10 & -15 & 5 \\ -5 & -5 & 2 & 3 & 5 & 5 \end{bmatrix},$$

where $D = [0, 1] \times [0, 1]$, see Fig. 6. The table reports the number of subdivision steps (line 10), clipping steps (otherwise), all iterations, detected roots and the computing time for several values of the prescribed accuracy ε.

Example 4.2. (Transition from two single roots to a double root) In order to demonstrate different behaviour of the Algorithm 1 in the double root and near double root cases, we consider the sequence of polynomial systems

$$\begin{aligned} 0 &= x^2 + y^2 - \left(1 + \tfrac{1}{10^k}\right), \\ 0 &= xy - \tfrac{1}{2} \end{aligned} \tag{33}$$

over the domain $[-2, 2] \times [-2, 2]$, where $k = 0, 1, \ldots$, see Fig.7a. The system has four real roots over $[-2, 2] \times [-2, 2]$ for all k and two double roots in the limit case for $k = \infty$. Fig.7b shows the relation between the number of iterations of Algorithm 1 with respect to the prescribed accuracy for different values of k. As the the value of k is increased,

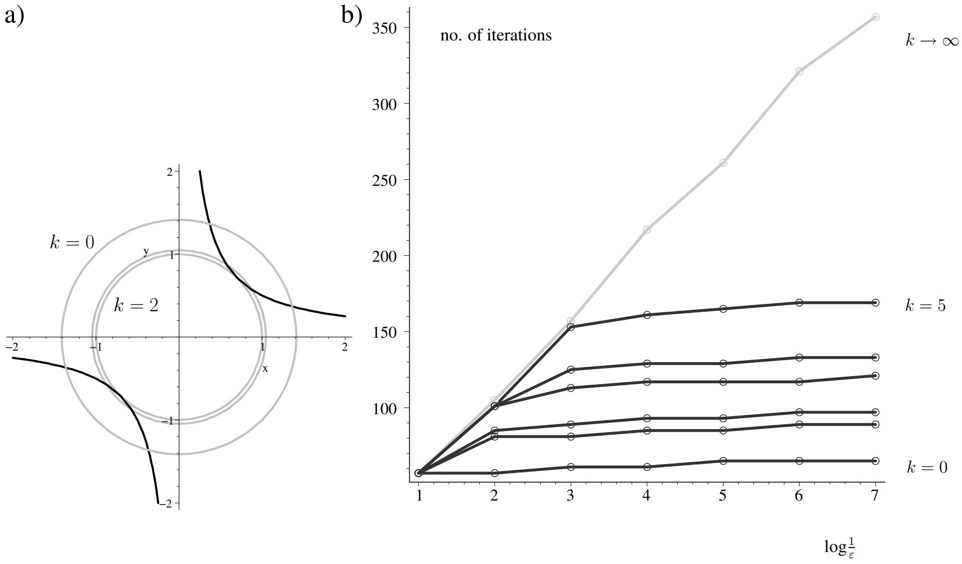


FIGURE 7. Example 4.2. a) The system (33) for $k = 0, 1, 2$, b) relation between the number of iterations and the accuracy ε for $k = 0, \ldots, 5$ and $k = \infty$.

the roots get closer and the roots are separated later. Consequently, a higher number of iterations is needed.

Example 4.3. (Self-intersection) In [15], the self intersections of a rational curve $C(u)$ of degree 19 are found by solving the system

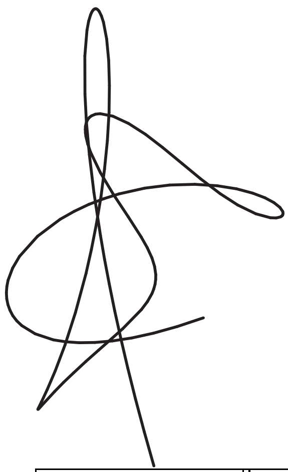
$$\frac{C(u) \times C(v)}{u - v} = O, \quad u < v, \tag{34}$$

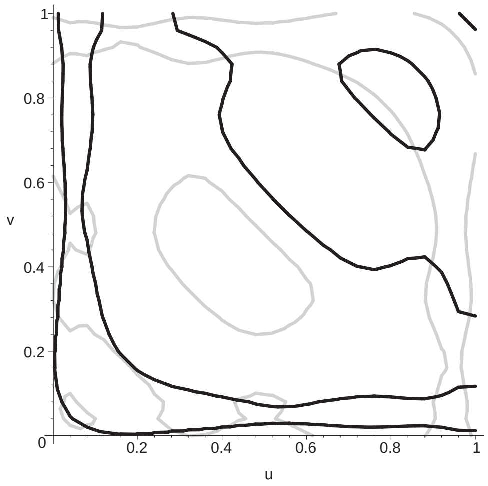
where $\times$ is the cross product and the curve is described by homogeneous coordinates. This leads to a polynomial system of bidegree $(37, 37)$, see Fig. 8. We applied BLC to this example; the number of iterations and computing times are reported in the table.

In [15], several existing methods are compared for $\varepsilon = 10^{-6}$. These methods are SBD (subdivision), RD (reduction with a variant of the the IPP algorithm), SBDS (subdivision with a global preconditioner), RDS (reduction with a variant of the IPP algorithm and a global preconditioner), and RDL (reduction with a variant of the IPP algorithm and a local preconditioner). Among these methods, only RDL has the same convergence rate as BLC for single roots (2). Though it is difficult to compare different implementations running on different hardware, it can be concluded that the performance of BLC compares well with the existing algorithms.

Example 4.4. (Numerical robustness) In order to demonstrate the robustness of the method, we applied it to the system which consists of the two polynomials of degree 12,

$$p(x, y) = \prod_{i=0}^{11} (x + y - \frac{2 + 2i}{13}), \quad q(x, y) = \prod_{i=0}^{11} (3x + y - \frac{19 + 6i}{26}), \tag{35}$$





ε	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
clipping steps	28	54	67	71	56
subdivision steps	75	79	79	79	79
no root ($D' = \emptyset$)	170	225	225	225	225
all iterations	273	358	371	375	383
no. of roots	56	13	13	13	13
time (ms)	139	305	322	330	336

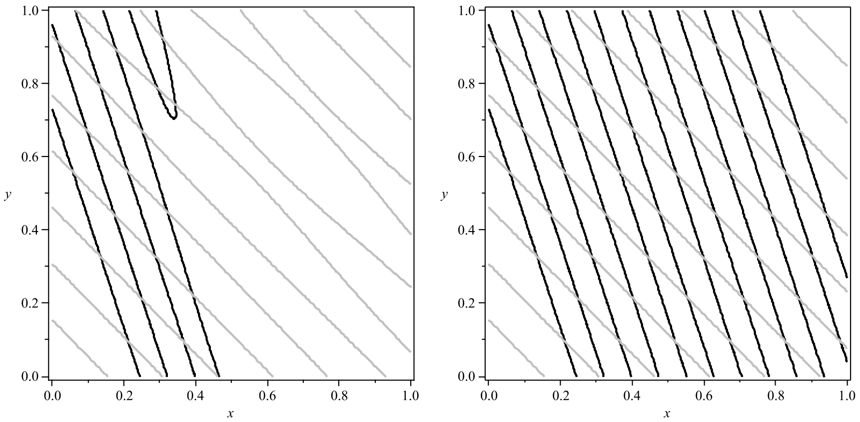
FIGURE 8. Example 4.3. Left: Rational Bézier curve of degree 19 with 13 self-intersections. Right: Polynomial system (34). Below: Computing times and number of iterations for various values of ϵ.

TABLE 1. Results of [15] for example 4.3 with $\varepsilon = 10^{-6}$.

method	no. of iterations	subdivision steps	no. of roots	time (ms)
SBD	3979	3979	39	3540
RD	560	376	15	537
SBDS	1577	1577	16	1589
RDS	282	63	13	345
RDL	126	36	13	134
BLC	282	79	13	322

where the domain is the unit square. Similar to the Wilkinson polynomial in the one-dimensional case, this system is very sensitive with respect to perturbations of the monomial coefficients. The stability is greatly enhanced by using the tensor-product Bernstein basis, see Fig. 4.4.

Remark 4.5. (Interval input) In all examples above, we considered the input data to be exact (within the accuracy of the floating point numbers). However, the method can easily be adapted to the case of polynomials with interval coefficients, simply by adding the maximum coefficient tolerance to δ^p and δ^q in line 3 of the algorithm. Note that the



	monomial noise					BB noise				
accuracy ε	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
clipping steps	49	102	118	130	134	126	236	276	304	318
subdivision steps	239	244	244	244	244	497	793	793	793	793
no root ($D' = \emptyset$)	612	715	715	715	715	1076	2330	2334	2334	2334
all iterations	900	1061	1077	1089	1093	1699	3359	3403	3431	3445
no. of roots	106	18	18	18	18	416	50	46	46	46
time (ms)	278	305	317	322	328	513	932	954	965	985

FIGURE 9. Example 4.4. Top: The effect of adding $10^{-}8\%$ relative coefficient error to the monomial (left) and BB (right) coefficients of the two polynomials in (35). In the latter case, the system remains essentially the same, while the modification of the monomial coefficients leads to dramatic changes of the structure of solutions. Bottom: Computation time and number of iterations for various values ofε.

maximum tolerance does not increase during the iterations, as the de Casteljau algorithm propagates the maximum coefficient error. This simple way of dealing with interval input may be seen as an advantage to the preprocessing based RDL algorithm in [15], where more complicated adaptations appear to be needed.

5. CONCLUDING REMARKS

We presented a new method for computing all roots of a bivariate polynomial system within a given domain. In the case of single roots, the algorithm converges quadratically, similar to the RDL technique in [15]. The method can be generalized to systems of n equations with n variables. Future work will focus on methods providing superlinear convergence rates for multiple roots, which extend the results in [1] to the multivariate case.

REFERENCES

[1] M. Bartoň and B. Jüttler, *Computing roots of polynomial by quadratic clipping*, Comput. Aided Geom. Design **24** (2006), 125–141.

[2] G. Elber and M.-S. Kim, *Geometric constraint solver using multivariate rational spline functions*, The Sixth ACM/IEEE Symposium on Solid Modeling and Applications (2001), Ann Arbor, MI, 1–10.

[3] G. Farin, J. Hoschek and M.-S. Kim, eds., *Handbook of computer aided geometric design*, Elsevier, Amsterdam, 2002.

[4] R. T. Farouki and T. N. T. Goodman, *On the optimal stability of the Bernstein basis*, Math. Comp., **65** (1996), 1553–1566.

[5] W. Gautschi, *Numerical Analysis*, Birkhäuser, Boston, MA, 1997.

[6] J. Garloff and A. P. Smith, *Solution of systems of polynomial equations by using Bernstein expansion*, Alefeld, Götz (ed.) et al., Symbolic algebraic methods and verification methods (2001), Springer, Wien, 87–97.

[7] J. Garloff, A. P. Smith, *A comparison of methods for the computation of affine lower bound functions for polynomials*, Computer Science (2005), No. 3478, Springer, Berlin, Heidelberg, 71-85.

[8] J. Hoschek, D. Lasser, *Fundamentals of computer aided geometric design* (1993), AK Peters, Wellesley, MA.

[9] M. L. Husty, M. Pfurner and H.-P. Schröcker, *A new and efficient algorithm for the inverse kinematics of a general serial* $6R$ *manipulator,* Mech. Mach. Theory (2007), **42**, 66–81.

[10] K. Lee, *Principles of CAD/CAM/CAE systems*, Prentice Hall, 1999.

[11] T. Li, *Numerical solution of polynomial systems by homotopy continuation methods*, In F. Cucker (ed.), Special Volume: Foundations of Computational Mathematics, Handbook of Numerical Analysis (2003), **11**, North–Holland, 209–304.

[12] D. Lutterkort and J. Peters, *Optimized refinable enclosures of multivariate polynomial pieces*, Comput. Aided Geom. Design (2001), **18**, 851–863.

[13] E. Mainar and J. M. Peña, *Evaluation algorithms for multivariate polynomials in Bernstein-Bézier form*, J. Approximation Theory (2006), **143**, 44–61.

[14] K. Mørken and M. Reimers, *An unconditionally convergent method for computing zeros of splines and polynomials* (2007), Math. Comp. **76**, 845–865.

[15] B. Mourrain and J.-P. Pavone, *Subdivision methods for solving polynomial equations*, Technical Report no. 5658, INRIA Sophia Antipolis (2005), `http://www.inria.fr/rrrt/rr-5658.html`.

[16] J. M. McNamee. *Bibliographies on roots of polynomials*, J. Comp. Appl. Math., **47** (1993), 391–394; **78** (1997), 1–1; **110** (1999), 305–306; **142** (2002), 433–434.

[17] T. Nishita, T. Sederberg, and M. Kakimoto, *Ray tracing trimmed rational surface patches*, Siggraph Proceedings (1990), ACM, 337–345.

[18] Joseph O'Rourke, *Computational Geometry in C*, Cambridge University Press, second edition, 1998.

[19] H. Prautzsch, W. Boehm and M. Paluszny, *Bézier and B-Spline techniques*, Springer, Berlin, 2002.

[20] F. Rouillier and P. Zimmermann, *Efficient isolation of polynomial's real roots*, J. Comput. Appl. Math. (2004), **162**, 33–50.

[21] S. M. Rump, *Ten methods to bound multiple roots of polynomials*, J. Comput. Appl. Math. (2003), **156**, 403–432.

[22] E. C. Sherbrooke, N. M. Patrikalakis, *Computation of the solutions of nonlinear polynomial systems*, Comput. Aided Geom. Design (1993), **10**, 379–405.

[23] A. J. Sommese, C. W. Wampler, *The numerical solution of systems of polynomials arising in engineering and science*, World Scientific, 2005.

[24] J. Valdman, A. Hofinger, *Numerical solution of the two-yield elastoplastic minimization problem*, submited to *Computing* (2006), available as SFB report at www.sfb013.uni-linz.ac.at.

JOHANNES KEPLER UNIVERSITY, INSTITUTE OF APPLIED GEOMETRY,, ALTENBERGER STR. 69, 4040 LINZ, AUSTRIA; WWW.AG.JKU.AT

E-mail address: {`Bert.Juettler`|`Michael.Barton`}`@jku.at`