INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Subdivision methods for solving polynomial equations

Bernard Mourrain — Jean-Pascal Pavone

N° 5658

Août 2005

Thème SYM

ISSN 0249-6399 ISRN INRIA/RR--5658--FR+ENG

Subdivision methods for solving polynomial equations

Bernard Mourrain , Jean-Pascal Pavone

Thème SYM — Systèmes symboliques
Projet Galaad

Rapport de recherche n° 5658 — Août 2005 — 22 pages

Abstract: This paper presents a new algorithm for solving a system of polynomials, in a domain of $\mathbb{R}^n$. It can be seen as an improvement of the *Interval Projected Polyhedron* algorithm proposed by Sherbrooke and Patrikalakis [SP93]. It uses a powerful reduction strategy based on univariate root finder using Bernstein basis representation and *Descarte's rule*. We analyse the behavior of the method, from a theoretical point of view, shows that for simple roots, it has a local quadratic convergence speed and gives new bounds for the complexity of approximating real roots in a box of $\mathbb{R}^n$. The improvement of our approach, compared with classical subdivision methods, is illustrated on geometric modeling applications such as computing intersection points of implicit curves, self-intersection points of rational curves, and on the classical parallel robot benchmark problem.

Key-words: resolution, symbolic-numeric computation, polynomial equation, subdivision, real solution, Bernstein basis, Descartes rule, complexity.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Méthode de subdivision pour la résolution d'équations polynomiales

Résumé : Dans ce rapport nous présentons un nouvel algorithme pour la résolution de systèmes d'équations polynomiales en plusieurs variables, dans un domaine de $\mathbb{R}^n$. Il peut-être vu comme une améllioration de l'algorithme *Interval Projected Polyhedron* proposé par Sherbrooke and Patrikalakis [SP93]. Il utilise une technique de réduction de domaines efficaces, qui s'appuie sur une méthode de résolution de polynômes en une variable, utilisant la représentation dans la base de Bernstein et la *règle de Descartes*. nous analysons le comportement de la méthode d'un point de vue théorique et monrons que pour des racines simples, il a une vitesse de convergence locale quadratique. Ceci nous conduit à de nouvelle bornes de complexité pour lápproximation de racines dans une boite de $\mathbb{R}^n$. L'apport de notre méthode, comparée avec des méthodes classiques de subdivision, est illustré sur des applications en modélisation géométrique, telles que le calcul de points d'intersection de courbes implicites, de points d'auto-intersection de courbes paramétrées et sur un problème classique de robotique.

Mots-clés : résolution, calcul symbolique-numerique, équation polynomiale, subdivision, solution réelle, base de Bernstein, règle de Descartes, complexité.

1 Introduction

Solving polynomial equations is ubiquitous in geometric problems. We can identify two main families of solvers: a first family of solvers which exploit the algebraic properties associated to these polynomials and all their polynomial combinations. It usually leads to methods which gives global informations on the set of solutions [CLO92]; a second family of solvers which treat the polynomials as real value functions and analyse the zero-level of these functions. It usually leads to local methods, such as the famous Newton (-Raphson) method [Rhe98].

The approach that we describe in this paper combines, in some way, these two characteristics. We exploit the properties of polynomial representations in the Bernstein basis, to deduce easily informations on the corresponding real functions in a domain of $\mathbb{R}^n$. Bernstein polynomial representations are ubiquitous in geometric modeling. It is known to be numerically more stable than the monomial basis representation [FG96], [FR87]. This Bernstein representation has a direct geometric meaning, in terms of control points. It provides useful properties such that the convex hull and the variation diminishing properties. These properties in conjunction with subdivision techniques explain the large variety of algorithms proposed until today for solving univariate polynomials, starting with Lane and Riesenfeld [JR81], up to the *Bezier clipping* methods initiated by Nishita and al [TNK90]. They combine a global control on the domain where the roots are searched with local and efficient refinements.

The situation in the multivariate case has not been studied so extensively. Two main subfamilies coexist: A first family which is based on subdivision techniques such as [EK01]; a second family of solvers is based on reduction techniques, as in [SP93].

The subdivision approaches use an exclusion test, based on the convex hull property in [SP93], for checking for the existence of a solution in the search domain. The result of the test is of the form: "no solution" or "maybe one". If the answer is "no solution" then the domain is rejected. Otherwise the domain is subdivided, generally in a way independent of the data, and this process is repeated until the domain satisfies a termination criterion. This termination criterion can be simply based on the size of the domain, but it can be more elaborated [EK01], [SM88] [GS01b] (using for instance Miranda theorem). The subdivision approach provides algorithms that produce a large number of iterations especially in the case of multiple roots, but the iteration cost is significantly smaller than in reduction approaches, that we mention now.

Reduction approaches use a technique to contract the domain where the roots are searched, such as the convex hull property used in [SP93]. Its power resides in its capacity to concentrate on the parts of the domain where the roots are. Reductions cannot replace completely subdivisions, because it is not always possible to reduce the domain, if we have to separate the roots, but they reduce drastically the number of iterations, and thus have a great impact on the performance of the solver.

In this paper, we analysis in detail the subdivision and reduction approaches. We propose a general scheme for comparing and evaluating them. In addition, we consider a new reduction technique and new preconditioning steps, which influence drastically the efficiency of the solvers. This scheme allows us to compare a large variety of algorithms, including the

one previously known. Reporting on extended experimentations, we show the impact of different strategies on the behavior of the solvers. This leads to a new reduction-subdivision solver, which overcome the other methods. It can be seen as an improvement of the well-known *interval projected polyhedron* [SP93]. We use efficient univariate solvers to optimise the reduction steps, and show that the cost of solving such univariate polynomial equations is compensated by the speed of the reduction.

The paper is organised as follows. In the next section, we recall the main properties of Bernstein basis, that we will use for isolating real roots. In section 3, we describe and analyse the behavior of a family of univariate polynomial solvers, using this representation. In section 4, we describe the multivariate subdivision solver, which uses as a main ingredient a univariate solver. The importance of the preconditioning step and the subdivision strategy are discussed. We analyse the behavior of the method, from a theoretical point of view, shows that locally it is converging quadratically to a simple root and gives new bounds for the complexity of approximating real roots in a box $\mathcal{D} \subset \mathbb{R}^n$. Experimentations on geometric problems and classical benchmarks show the performances of the implementation and conclude the papers.

2 Bernstein polynomial representation

Let us first recall the main properties of Bernstein polynomial representation, that we are going to use. For a more detailed list of properties of this representation, we refer for instance to [Far90].

Any univariate polynomial $f(x) \in \mathbb{K}[x]$, of degree d, can be represented as

$$f(x) = \sum_{i=0}^{d} b_i \binom{d}{i} \frac{1}{(b-a)^d} (x-a)^i (b-x)^{d-i}.$$

The polynomials $B_d^i(x;a,b) := \binom{d}{i} \frac{1}{(b-a)^d} (x-a)^i (b-x)^{d-i}$ form the Bernstein basis on $[a,b]$. Hereafter, we are going to consider the sequence of values $\mathbf{b} = [b_0, .., d_d]$ together with the corresponding interval $[a,b]$, as a representing our polynomial f.

A fundamental algorithm that we will use on such a representation is the de Casteljau algorithm [Far90]: $b_i^0 = b_i,\ i = 0,\dots,d,\ b_i^r = (1-t)\, b_i^{r-1} + t\, b_{i+1}^{r-1}(t),\ i = 0,\dots,d-r$. It allows us to subdivide the representation of f into the two subrepresentations on the intervals $[a,(1-t)a+tb]$ and $[(1-t)a+tb,b]$. It requires at most $2d(d+1)$ arithmetic operations.

A simple but interesting property that we are going to use is the following:

Theorem 1 (Descartes rule). *[Ris91], [BPR03], [MRR04] The number of real roots of $f(x) = \sum b_i B_d^i(x;a,b)$ in $]a,b[$ is bounded by the number $V(\mathbf{b})$ of sign changes of $\mathbf{b} = (b_i)_{i=0..n}$, and is equal modulo 2.*

As a consequence, if $V(\mathbf{b}) = 0$ there is no root in $]a,b[$ and if $V(\mathbf{b}) = 1$, there is one root in $]a,b[$. Another interesting property of this representation is the following:

Theorem 2 (Convex hull). *[Far90], [Ris91] Let* $\mathbf{b} = (b_i)_{i=0,\ldots,d}$ *be the control coefficients of* $f(x)$ *on the interval* $[a,b]$ *and* $\mathbf{c} = [(\frac{(d-i)\,a+i\,b}{d}, b_i)_{i=0,\ldots,d}]$ *the corresponding control points. The graph* $\{(t,f(t)); t \in [a,b]\}$ *is in the convex hull of the control points* $\mathbf{c}$.

By a direct extension to the multivariate case, any polynomial $f(x_1,\ldots,x_n) \in \mathbb{K}[x_1,\ldots,x_n]$ of degree d_i in the variable x_i, can be decomposed as:

$$f(x_1,\ldots,x_n) = \sum_{i_1=0}^{d_1} \cdots \sum_{i_n=0}^{d_n} b_{i_1,\ldots,i_n}\, B_{d_1}^{i_1}(x_1;a_1,b_1)\cdots B_{d_n}^{i_n}x(x_n;a_n,b_n).$$

where $(\,B_{d_1}^{i_1}(x_1;a_1,b_1)\cdots B_{d_n}^{i_n}(x_n;a_n,b_n))_{0\le i_1\le d_1,\ldots,0\le i_n\le d_n}$ is the tensor product Bernstein basis on the domain $\mathcal{D} := [a_1,b_1]\times\cdots\times[a_n,b_n] \subset \mathbb{R}^n$ and $\mathbf{b}(f) = (b_{i_1,\ldots,i_n})_{0\le i_1\le d_1,\ldots,0\le i_n\le d_n}$ are the control coefficients of f on $\mathcal{D}$. The polynomial f is represented in this basis by the n^{th} order tensor of control coefficients $\mathbf{b}(f)$. The control points of f are

$$\mathbf{c}(f) = ((\frac{(d_1-i_1)\,a_1+i_1\,b_1}{d_1}, \ldots, \frac{(d_n-i_n)\,a_n+i_n\,b_n}{d_1}, b_{i_1,\ldots,i_n})_{0\le i_1\le d_1,\ldots,0\le i_n\le d_n}).$$

Let $p_{i_1,\ldots,i_n}(f) = ((\frac{(d_1-i_1)\,a_1+i_1\,b_1}{d_1}, \ldots, \frac{(d_n-i_n)\,a_n+i_n\,b_n}{d_1})$.

De Casteljau algorithm also applies in each of the direction x_i, $i = 1,\ldots,n$ so that we can split this representation accordingly. This can be used either to split the domain or to restrict the representation to a subdomain. For a multivariate polynomial of degree d_i in x_i, we check that this restriction operation costs $2\sum_{i=1}^n d_i \prod_{i=1}^n (d_i+1) = \mathcal{O}(d^{n+1})$ where $d = \max\{d_1,\ldots,d_n\}$. Thus as the dimension and the degree increase, a good method to isolate the roots, should consider carefully when to apply this reduction operation, in order to save the computation time.

Definition 1. *For any* $f \in \mathbb{K}[\mathbf{x}]$ *and* $j = 1,\ldots,n$, *let*

$$m_j(f;x_j) = \textstyle\sum_{i_j=0}^{d_j} \min_{\{0\le i_k\le d_k, k\ne j\}} b_{i_1,\ldots,i_n}\, B_{d_j}^{i_j}(x_j;a_j,b_j),$$
$$M_j(f;x_j) = \textstyle\sum_{i_j=0}^{d_j} \max_{\{0\le i_k\le d_k, k\ne j\}} b_{i_1,\ldots,i_n} x\, B_{d_j}^{i_j}(x_j;a_j,b_j).$$

We have the following property:

Lemma 1 (Projection Lemma). *For any* $\mathbf{u} = (u_1,\ldots,u_n) \in \mathcal{D}$, *and any* $j = 1,\ldots,n$, *we have* $m_j(f;u_j) \le f(\mathbf{u}) \le M_j(f;u_j)$.

Proof. As for $k = 1,\ldots,n$, $\sum_{k=0}^{d_k} B_{d_k}^{i_k}(u_k;a_k,b_k) = 1$, we have

$$\begin{aligned}
&\sum_{i_1=0}^{d_1} \cdots \sum_{i_n=0}^{d_n} b_{i_1,\ldots,i_n}\, B_{d_1}^{i_1}(u_1;a_1,b_1)\cdots B_{d_n}^{i_n}(u_n;a_n,b_n)\\
&\le \left(\sum_{i_j=0}^{d_j} \max_{\{0\le i_k\le d_k, k\ne j\}} b_{i_1,\ldots,i_n} B_{i_j}(u_j;a_j,b_j)\right) \times \sum_{\{0\le i_l\le d_l, l\ne j\}} \prod_{k\ne j} B_{d_k}^{i_k}(u_k;a_k,b_k)\\
&\le M_j(f;u_j).
\end{aligned}$$

A similar proof applies for $m_j(f;u_j)$.

As a direct consequence, we obtain the following corollary:

Corollary 1. *For any root $\zeta = (\zeta_1, \ldots, \zeta_n) \in \mathbb{R}^n$ of the equation $f(\mathbf{x}) = 0$ in the domain $\mathcal{D}$, we have $\underline{\mu}_j \leq \zeta_j \leq \overline{\mu}_j$ where*

- *$\underline{\mu}_j$ (resp. $\overline{\mu}_j$) is either a root of $m_j(f; x_j) = 0$ or $M_j(f; x_j) = 0$ in $[a_j, b_j]$ or a_j (resp. b_j) if $m_j(f; x_j) = 0$ (resp. $M_j(f; x_j) = 0$) has no root on $[a_j, b_j]$,*
- *$m_j(f; u) \leq 0 \leq M_j(f; u)$ on $[\underline{\mu}_j, \overline{\mu}_j]$.*

Definition 2. *For a system of polynomials $\mathbf{f} = (f_1, \ldots, f_s)$, we define*

$$m_j(\mathbf{f}; u_j) = \sup\{m_j(f_k; u_j); k = 1, \ldots, s\},\ M_j(\mathbf{f}; u_j) = \inf\{M_j(f_k; u_j); k = 1, \ldots, s\}.$$

3 Univariate Root Solver

Our approach for solving multivariate systems is based on efficient methods for isolating roots of univariate polynomials. In this section, we describe the different univariate solvers that we use for this purpose.

Descartes rule 1 yields a simple subdivision algorithm, which *isolated* the roots, as described in [MVY02] or [MRR04]. The behavior of the algorithm can be analysed using to the two circles theorem (see [BPR03], [Meh] or [MRR04]), which shows that at if $f(x) = 0$ has only simple roots on $[a, b]$, an upper bound of the number of recursion steps of this algorithm is at most $\lceil \log_2 \left(\frac{2}{\sigma}\right) \rceil$, where σ is the minimal distance between the complex roots of f.

In order to approximate a root within a given precision $\epsilon > 0$, after it has been isolated, a usual approach is by bisection, that is by splitting the interval into two subintervals and by choosing the interval containing the root. This splitting can be performed

- either in the Bernstein basis by de Casteljau algorithm (which requires $\mathcal{O}(d^2)$ arithmetic operations),
- or in the monomial basis using Horner methods (which requires $\mathcal{O}(d)$ arithmetic operations).
- or by a secant-like method, which consists in intersecting the interval $[a, b]$ with the line with the lowest slope which is joining the first control point to another point, and by using this point to split the Bernstein representation,
- by computing iteratively the first intersection of the convex hull of the control polygon, with the x-axis and in subdividing the polynomial representation at this point [Roc90],
- or by a Newton-like method, which consists in splitting the interval at the point where the tangent cuts the interval $[a, b]$ if it exists or at the middle otherwise. These operations are performed in the monomial basis.

The two first schema converge linearly to the root in the interval, whereas the other have a superlinear and quadratic convergence speed [Hen77]. In the third method, each iteration requires $\mathcal{O}(d)$ arithmetic operations. This method can be improved by computing the intersection of the convex hull of the control points with the x-axis, but it requires

$\mathcal{O}(d\log(d))$ arithmetic operations and interval arithmetic for a numerically stable implementation [PM02]. The fourth method requires $\mathcal{O}(d^2)$ arithmetic operations.

By choosing the adequate arithmetic rounding mode during the iterations of these methods, we *guaranty* that the first (resp. last) root of the polynomial is approximated below (resp. above) at the end of the computation.

Experimentations done on univariate polynomials with random roots or coming from ray tracing problems tracing problems shows the superiority of Horner and Newton iterations in terms of speed, compared to the other methods. Such algorithms allows us to solve more than 10^6 equations of degree 9, within a precision $\epsilon = 10^{-12}$, on an Intel Pentium4 2.0Ghz, 512 Mo of RAM workstation. Newton iteration seems to be ahead in "simple" situations where the speed of convergence compensates the arithmetic cost of the iteration.

4 Multivariate root finding

In this section, we consider a system of s polynomial equations in n variables:

$$f_1(x_1,\ldots,x_n)=0,\ldots,f_s(x_1,\ldots,x_n)=0$$

with coefficients in $\mathbb{R}$, that we will also denote by $\mathbf{f}(\mathbf{x})=0$. We are looking for an approximation of the real roots of $\mathbf{f}(\mathbf{x})=0$ in the domain $\mathcal{D}=[a_1,b_1]\times\cdots\times[a_n,b_n]$, within a precision ϵ.

4.1 The algorithm

The general framework of the families of algorithms that we will consider consists

1) in applying a preconditioning step on the equations;
2) in reducing the domain;
3) and if the reduction ratio is too small, in splitting the domain;

until the size of the domain is smaller than a given precision ϵ. We have several options for each of these steps, leading to different algorithms with different behaviors, as we will see in the last section. Indeed the solvers that we will consider are parameterised by the

- **preparation strategy**: a transformation of the initial system followed by a projection on the x_i axes;
- **reduction solver**: a method derived from univariate root finding techniques used to reduce the initial domain, according to the available projections.
- **subdivision rule**: a technique used to subdivide the domain, in order to simplify the forthcoming steps, for searching of the roots.

4.2 Preconditioner

Let us describe here the two preparation steps that we use to improve, at each iteration, the numerical quality of the system. Namely, we transform the system $\mathbf{f} = 0$ into an equivalent one $M\,\mathbf{f} = 0$, where M is an $s \times s$ invertible matrix

As such a transformation may increase the degree of some equations, with respect to some variables, it has a cost, which might not be negligible in some cases. Moreover, if for each polynomials of the system not all the variables are involved, that is if the systems is sparse with respect to the variables, such a preconditioner may transform it into a system which is not sparse anymore. In this case, we would prefer a partial preconditioner on a subsets of the equations sharing a subset of variables.

For the sake of simplicity, we will assume hereafter that the polynomials $f_1, \ldots, f_s$ are expressed in the same Tensor product Bernstein basis, that we denote hereafter by $\mathbf{B}$. We are going to consider two types of transformation.

Global transformation A typical difficult situation for a method which exploits the value of the functions $(f_i)_{i=1,\ldots,s}$, is when two of these functions have closed graphs on the domain $\mathcal{D}$. A way to avoid such a situation is to transform these equations in order to increase the distance between these graphs. As this distance is not straightforward to compute, we replace it by the distance between the control points of the functions on the domain $\mathcal{D}$: for $f, g \in \mathbb{R}[\mathbf{x}]$, let $\mathrm{dist}(f,g)^2 = ||f-g||^2$ with

$$||f||^2 = \sum_{0\le i_1\le d_1,\ldots,0\le i_n\le d_n} |\mathbf{b}_{i_1,\ldots,i_n}(f)|^2, \tag{1}$$

where $\mathbf{b}(f)$ is the vector of control coefficients of the function f in the Bernstein basis $\mathbf{B}$. This norm on the vector space of polynomials generated by the basis $\mathbf{B}$ is associated to a scalar product that we denote by $\langle\,|\,\rangle$. The aim of this preconditioner is to optimize the angles between the vectors, that is to produce a system which is orthogonal for $\langle\,|\,\rangle$. We obtain it by eigenvector computation.

Proposition 1. *Let $Q = (\langle f_i | f_j\rangle)_{1\le i,j\le s}$ and let E be a matrix of unitary eigenvectors of Q. Then $\tilde{\mathbf{f}} = E\,\mathbf{f}$ is a system of polynomials which are orthogonal for the scalar product $\langle\,|\,\rangle$.*

Proof. Let E be the matrix of (real) unitary eigenvectors of Q and let $\tilde{\mathbf{f}} = E\mathbf{f} = (\tilde{f}_1, \ldots, \tilde{f}_n)$. Then the matrix of scalar product $(\tilde{f}_i|\tilde{f}_j)$ is

$$E^t Q E = \mathrm{diag}(\sigma_1, \ldots, \sigma_n),$$

where $\sigma_1 \ge 0, \ldots \sigma_n \ge$ are the positive eigenvalues of Q. This shows that the system $\tilde{\mathbf{f}}$ is orthonormal for the scalar product $\langle\,|\,\rangle$.

We illustrate the impact of the global preconditioner in figure 1, on two bivariate functions which graphs are very closed to each other before the preconditioning, and which are well separated after this preconditioning step. In this case, one of the new functions is even not intersecting the zero-level (gray plane), so that we can deduce directly that there is no root in the domain $\mathcal{D}$.

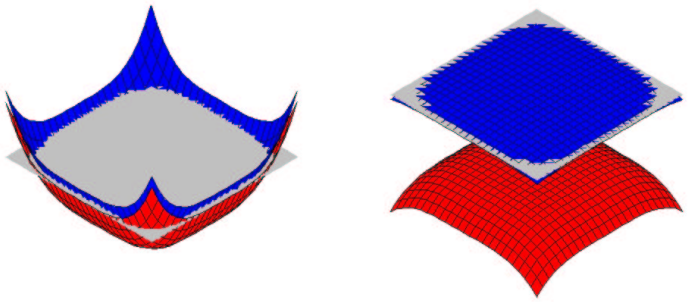


Fig. 1. Global preconditioner

Local straightening In this section we consider square systems, for which $s = n$. Since we are going to use the projection lemma 1, interesting situation for reduction steps, are when the zero-level of the functions f_i are orthogonal to the x_i-directions. We illustrate this remark in dimension 2, by figure 2: In the case (a), the reduction based on corollary 1, will be of no use (because the projection of the graphs cover the intervals), whereas in case (b), a good reduction strategy will yield a good approximation of the roots. This

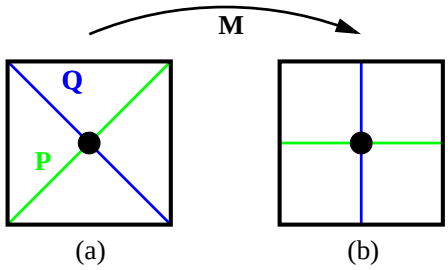


Fig. 2. Local preconditioner

idea of this preconditioner is thus to transform the system $\mathbf{f} = 0$, in order to be closed to the case (b). Namely, we transform locally the system $\mathbf{f}$ into a system $J_{\mathbf{f}}^{-1}(\mathbf{u}_0)\mathbf{f}$, where $J_{\mathbf{f}}(\mathbf{u}_0) = (\partial_{x_i} f_j(\mathbf{u}_0)_{1\leq i,j\leq s}$ is the Jacobian matrix of $\mathbf{f}$ at the point $\mathbf{u}_0 \in \mathcal{D}$. See [GS01a] for a previous application of this idea.

A direct computation shows that locally (in a neighborhood of $\mathbf{u}_0$), the level-set of $\tilde{f}_i$ $(i = 1, \ldots, n)$ are orthogonal to the x_i-axes.

4.3 Reduction strategy

We describe several reduction strategies, which have been considered. See also [Spe94].

In [SP93], a method called Interval Projected Polyhedron (or IPP) is described, in order to reduce the domain of search. It is based on the convex hull property 2.

A direct improvement consists in computing the first (resp. last) root of the polynomial $m_j(f_k; u_j)$, (resp. $M_j(f_k; u_j)$), in the interval $[a_j, b_j]$, and keep the intervals $[\underline{\mu}, \overline{\mu}]$ defined in corollary 1. As we can see in the following example, figure 4.3 the improvement compared with the IPP approach can be substantial. The actual implementation of this reduction steps

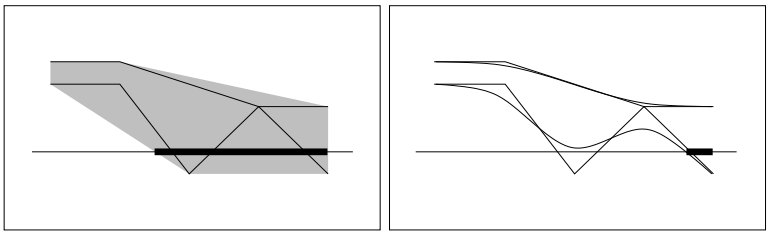


Fig. 3. Convex hull vs. root finding

allows us to consider the convex hull reduction, as one iteration step of this reduction process. By increasing the number of iterations, we improve the interval containing the extreme roots. The precision required in the approximation of the roots is not an important aspect of this step, since we are more interested on the ratio of the size of the new interval by the size of the initial one. This allows us to reduce more efficiently the domain, by computing several subdomains or by rejecting it more quickly, as it is illustrated in figure 4, This method can be even further improved by taking into account simultaneously all the projections of the polynomials f_j of the system. The guarantee that the computed intervals contain the root of f, is achieved by controlling the rounding mode of the operations during the de Casteljau computation.

4.4 Subdivision strategy

Here some simple rules that can be used to subdivide a domain. We will show in the last section their impact on the performance of the solver

The subdivision method simply checks the sign of control coefficients of the polynomials f_j on the domain $\mathcal{D}$. If for one of non-zero polynomial f_k, its control coefficient vectors has

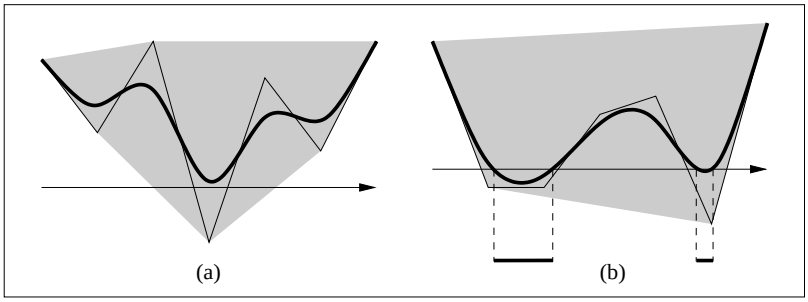


Fig. 4. Convex hull vs. root finding

no sign change, than $\mathcal{D}$ does not contain any root and should be excluded. The domain $\mathcal{D}$ is then split in half in a direction j for which $|b_j - a_j|$ is maximal.

This is approach used in [EK01] who argued that the reduction approaches are not so interesting because they cannot avoid anyway a lot of subdivisions. In section 6, we will analyse through experimentations the effectiveness of this remark.

A variant of this approach consists in subdividing the domain in a direction j if $|b_j - a_j| > \epsilon$ and if the control coefficients of $M_j(f_k; u_j)$ are not all positive, those of $m_j(f_k; u_j)$ not all negative. This allows us to have domains more adapted to the geometry of the roots, but still a postprocessing step for gluing together connected domains may be required.

5 Analysis

In this section, we analyse the behavior of the method in terms of the invariants of $\mathbf{f}$. Hereafter, the $||\cdot||$ or $||\cdot||_\infty$ is the ∞-norm, a box $B \subset \mathbb{R}^n$ is a product of intervals $I_1 \times \cdots \times I_n$, $|B|$ is the size of the box that is the maximum of the size of the intervals I_l $l = 1, \ldots, n$. For any polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ and any box $B \subset \mathbb{R}^n$, we denote by $\mathcal{C}(f; B)$ the piecewise linear function $B \to \mathbb{R}$, defined by the control points $\mathbf{c}(f)$ of f in the Bernstein basis of B.

For any compact domain $\mathcal{D} \subset \mathbb{R}^n$ and any $f \in \mathbb{R}[x_1, \ldots, x_n]$, we denote by $K_1(f, \mathcal{D})$ the Lipschitz constant of f in $\mathcal{D}$, satisfying

$$|f(x) - f(y)| \leq K_1(f, \mathcal{D})\|x - y\|_\infty.$$

Let $K_2(f, \mathcal{D}) = \max_{x \in \mathcal{D}, 1 \leq i,j \leq n} |\partial_i \partial_j f(x)|$ and $K_2(\mathbf{f}, \mathcal{D}) = \max\{K_2(f_i, B); i = 1, \ldots, s\}$.

For any linear map $M : \mathbb{R}^n \to \mathbb{R}^n$, let $m_1(M)$ be the smallest singular value of M such that if $y = M\,x$ and $\det(M) \neq 0$, then $||x||_2 \leq \frac{1}{m_1(M)}||y||_2 \leq \frac{\sqrt{n}}{m_1(M)}||y||_\infty$.

An important property of the Bernstein representation that we will use hereafter is the following:

Proposition 2. *[Dah86] [dB87] [PK94] Let $B \subset \mathbb{R}^n$ be a box and $f \in \mathbb{R}[x_1, \ldots, x_n]$, then $\forall x \in B$,*

$$|f(x) - \mathcal{C}(f; B)(x)| < K_2(f, B)|B|^2.$$

5.1 Local quadratic convergence

We consider here the local preconditioner method described in section 4.2, and shows that locally, it is converging quadratically to a simple root.

Proposition 3. *Let B be a box of $\mathbb{R}^n$ and $\mathbf{u}_0$ its center such that $\det(J_{\mathbf{f}}(\mathbf{u}_0)) \neq 0$. Then*

$$|\tilde{M}_j(\tilde{\mathbf{f}}; u_j) - \tilde{m}_j(\tilde{\mathbf{f}}; u_j)| \leq \frac{2\,(n^2+1)\sqrt{n}\,K_2(f, B)}{m_1(J_{\mathbf{f}}(u_0))}|B|^2,$$

where $\tilde{\mathbf{f}} = J_{\mathbf{f}}^{-1}(\mathbf{u}_0)\mathbf{f}$, $m_1(J_{\mathbf{f}}(u_0))$ is the smallest singular value of $J_{\mathbf{f}}(u_0)$ and $\tilde{m}_j, \tilde{M}_j$ are the corresponding bounding polynomials of the system $\tilde{\mathbf{f}}$.

Proof. By a Taylor expansion at $\mathbf{u}_0$, we have

$$\mathbf{f}(\mathbf{u} + \mathbf{u}_0) = \mathbf{f}(\mathbf{u}_0) + J_{\mathbf{f}}(\mathbf{u}_0)\,\mathbf{u} + R(\mathbf{u})$$

where $R(\mathbf{u})$ only involves monomials of degree ≥ 2. By the mean value theorem, for any $\mathbf{u}$ with $\mathbf{u}_0 + \mathbf{u} \in B$ and any $i = 1, \ldots, n$, we have

$$R_i(\mathbf{u}) = \sum_{1 \leq j,k \leq n} u_j u_k \partial_j \partial_k (f_i)(\nu_i),$$

with $\nu_i \in B$, so that $|R_i(\mathbf{u})| \leq n^2 K_2(f_i, B)|B|^2$. Thus

$$\tilde{\mathbf{f}}(\mathbf{u} + \mathbf{u}_0) = J_{\mathbf{f}}^{-1}(\mathbf{u}_0)\mathbf{f}(\mathbf{u}_0) + \mathbf{u} + J_{\mathbf{f}}^{-1}(\mathbf{u}_0)R(\mathbf{u}) = \mathbf{v} + \mathbf{u} + F(\mathbf{u}).$$

where $\mathbf{v} = J_{\mathbf{f}}^{-1}(\mathbf{u}_0)\mathbf{f}(\mathbf{u}_0) \in \mathbb{R}^n$ and $|F_i(\mathbf{u})| \leq \tilde{k}_2|B|^2$ with $k_2 = K_2(\mathbf{f}, B)$ and $\tilde{k}_2 = \frac{n^{\frac{5}{2}} k_2}{m_1(J_{\mathbf{f}}^{-1}(\mathbf{u}_0))}$.

By proposition 2, we have $|\mathcal{C}(R_i) - R_i(u_i)| < k_2|B|^2$ and

$$|\mathcal{C}(F_i) - F_i(u_i)| < \frac{\sqrt{n}}{m_1(J_{\mathbf{f}}^{-1}(\mathbf{u}_0))}k_2|B|^2 \leq \tilde{l}_2|B|^2.$$

We deduce that for $i = 1, \ldots, n$,

$$m_i(\tilde{f}_i; u_i) \geq \mathbf{v}_i + u_i - (\tilde{k}_2 + \tilde{l}_2)|B|^2,\ M_i(\tilde{f}_i; u_i) \leq \mathbf{v}_i + u_i + (\tilde{k}_2 + \tilde{l}_2)|B|^2.$$

which proves the proposition.

This also proves that the distance between two consecutive roots, one of $m_i(\tilde{f}_i; u_i)$ and one of $M_i(\tilde{f}_i; u_i)$ is less than $2\,(\tilde{k}_2 + \tilde{l}_2)|B|^2$.

As a consequence, we immediately deduce that any reduction strategy using this preconditioner and the projection lemma 1 near a simple root, will converge quadratically.

Corollary 2. *Let B be a box of $\mathbb{R}^n$ and $\mathbf{u}_0$ its center such that $\det(J_{\mathbf{f}}(\mathbf{u}_0)) \neq 0$. Let $\tilde{B}$ be the box obtained after one reduction step, then we have*

$$|\tilde{B}| \leq \frac{2\,(n^2+1)\sqrt{n}\,K_2(f,B)}{m_1(J_{\mathbf{f}}(\mathbf{u}_0))}|B|^2.$$

5.2 Global convergence

We consider here a domain $\mathcal{D} = [a_1, b_1] \times \cdots \times [a_n, b_n] \subset \mathbb{R}^n$ in which we are searching the roots of $\mathbf{f} = 0$. We denote them by $\zeta_1, \ldots, \zeta_t \in \mathcal{D} \subset \mathbb{R}^n$.

For $r \in \mathbb{R}_+$, we denote by $S_i(r) = \{x \in \mathbb{R}^n; |f_i(x)| \leq r\}$ and $S(r) = \cap_{i=1}^n S_i(r)$. Notice that $S(0)$ is the set of real solutions of $f_1(x) = 0, \ldots, f_s(x) = 0$.

Proposition 4. *Assume that $0 < \delta < \frac{K_1(f,\mathcal{D})}{K_2(f,\mathcal{D})}$, then for any box B of center $\mathbf{u}_0$ such that $\mathbf{u}_0 \notin S(2\,K_1(f;\mathcal{D})\delta)$ and $|B| < \delta$, the control coefficients of one of the polynomials f_i $(i = 1, \ldots, n)$ on B are of the same sign.*

Proof. Let $r > 0$ and B be a box with center $\mathbf{u}_0 \notin S(r)$ with $|B| \leq \delta$. Assume that $\delta \leq \frac{r}{2\,k_1}$, where $k_1 = K_1(f, \mathcal{D})$. Then $\mathbf{u}_0 \notin S_{i_0}(r)$ for some i_0 in $\{1, \ldots, n\}$. Consequently, $|f_{i_0}(\mathbf{u}_0)| \geq r$. We deduce that $\forall x \in B$,

$$|f_{i_0}(x)| \geq |f_{i_0}(\mathbf{u}_0)| - k_1\,\|x - \mathbf{u}_0\|_\infty \geq r - \frac{r}{2} = \frac{r}{2} > 0.$$

As f_{i_0} cannot vanish on the connected set B, it has a constant sign, say positive. By proposition 2, we have

$$b_{i_1,\ldots,i_n}(f_{i_0}) \geq f_{i_0}(p_{i_1,\ldots,i_n}) - k_2\,\delta^2 \geq \frac{r}{2} - k_2\,\delta^2 \geq \delta(k_1 - k_2\,\delta).$$

where $k_2 = K_2(f_{i_0}, \mathcal{D})$. Thus if $r = 2\,k_1\,\delta$ and $\delta < \frac{k_1}{k_2}$, we have $b_{i_1,\ldots,i_n}(f_{i_0}) > 0$ for all coefficient indexes $i_1, \ldots, i_n$, which proves that the control coefficients of f_{i_0} are of the same sign on B.

We denote by $N(\delta)$ the minimal number of boxes of size δ in a binary subdivision of $\mathcal{D}$, which are covering $S(2\,k_1\,\delta)$.

Corollary 3. *For $0 < \delta < \frac{K_1(f,\mathcal{D})}{K_2(f,\mathcal{D})}$, the number of boxes of size δ kept in the subdivision algorithm is bounded by $N(\delta)$.*

Proof. According to the previous proposition, for the boxes of size δ not covering $S(2\,k_1\,\delta)$, one of the polynomials f_{i_0} has control coefficients in the box of constant sign. Thus such a box is removed by the subdivision algorithm. Consequently the number of boxes of size δ in the subdivision algorithm is bounded by $N(\delta)$.

Proposition 5. *$N(\delta)$, $\delta > 0$ is bounded.*

Proof. $\forall \epsilon > 0$, let $V(\epsilon) = \cup_{i=1}^t V(\zeta_i, \epsilon)$ be a compact union of balls $V(\zeta_i, \epsilon)$ around the roots ζ_i $(i = 1, \ldots, t)$. As $\cap_{r>0} S(r) = \{\zeta_1, \ldots, \zeta_t\}$, we have $V(\epsilon) \subset \cup_{r>0} S(r)^c \cap \mathcal{D}$. As $V(\epsilon)$ is compact, it is covered by a finite union of $S^c(r_1), \ldots, S^c(r_l)$. Thus, there exists $r_0 = \min\{r_1, \ldots, r_n\} > 0$ such that $V^c(\epsilon) \cap \mathcal{D} \subset S^c(r_0) \cap \mathcal{D}$ or $S(r_0) \cap \mathcal{D} \subset V(\epsilon)$.

Let us choose $\sigma > 0$ such that $B(\zeta_i, \sigma) \cap B(\zeta_j, \sigma) = \emptyset$ if $i \neq j$ and $\det(J_{\mathbf{f}}(u)) \neq 0$ for $u \in \cup_{i=1}^s B(\zeta_i, \sigma)$. This is possible since the roots ζ_i are simple ($J_{\mathbf{f}}(\zeta_i) \neq 0$). We denote by $m_1(\sigma) = \min_{x \in V(\sigma)}\{m_1(J_{\mathbf{f}}(x))\}$.

Let δ_0 be such that $S(2\,k_1\,\delta_0) \subset V(\sigma)$ and $\delta_0 < \frac{K_1(f,\mathcal{D})}{K_2(f,\mathcal{D})}$. For $\delta \leq \delta_0$ and $x \in S(2\,k_1\,\delta) \subset V(\sigma)$, we have by the mean value theorem for $j = 1, \ldots, n$ $f_j(x) = (x - \zeta_{i_0}) \cdot \nabla f_j(\nu)$, where $x, \nu \in V(\zeta_{i_0}, \sigma)$. This implies that

$$||x - \zeta_{i_0}||_2 \leq \frac{1}{m_1(\sigma)} |f_j(x)| \leq \frac{2\,k_1\,\delta}{m_1(\sigma)}.$$

Consequently, the center of the boxes of size $\delta < \delta_0$ kept by the algorithm are in $\cup_{i=1}^t V(\zeta_i, \frac{2\,k_1\,\delta}{m_1(\sigma)})$, which implies that $N(\delta) \leq t \times (\frac{4\,k_1}{m_1(\sigma)} + 1)^n$. This proves that for δ small enough and thus for any $\delta > 0$, $N(\delta)$ is bounded.

We consider now the following characteristic quantities of the system $\mathbf{f} = 0$:

- We denote by $N_{\mathcal{D}}(\mathbf{f}) = \max_{\delta > 0} N_\delta$.
- Let $\sigma > 0$ be such that $V(\zeta_i, \sigma) \cap V(\zeta_j, \sigma) = \emptyset$ if $i \neq j$ and $\det(J_{\mathbf{f}}(x)) \neq 0$ for $x \in V(\sigma) = \cup_{i=1}^s V(\zeta_i, \sigma)$.
- Let $\rho = \max\{\delta > 0 \ st.\ S(2\,K_1(f, \mathcal{D})\,\delta) \subset V(\sigma)\}$.

Theorem 3. *Assume that $\mathbf{f}$ is a square system, with simple roots $\zeta_1, \ldots, \zeta_t$ in the domain $\mathcal{D} \subset \mathbb{R}^n$. Then the roots of $\mathbf{f} = 0$ can be approximated within the precision $0 < \epsilon < \frac{1}{2}$, by performing at most*

$$c_0\, N_{\mathcal{D}}(\mathbf{f})\, d^{n+1} (\log |\mathcal{D}| + |\log(\rho^{-1})| + \log(m_1(\sigma)^{-1}) + \log K_2(f, \mathcal{D}) + n^3 \log |\log(\epsilon)|),$$

arithmetic operations, where

- *$c_0 >$ is a constant independent of $\mathbf{f}$, $\mathcal{D}$ and n,*
- *$d = \max_{1 \leq i,j \leq n} \deg_{x_i}(f_{f_j})$,*
- *$K_2(\mathbf{f}, \mathcal{D})$ is a bound on the second derivatives of the polynomials f_i on $\mathcal{D}$ and $K_1(\mathbf{f}, \mathcal{D})$ a bound on their Lipschitz constants on $\mathcal{D}$,*

- $N_{\mathcal{D}}(\mathbf{f})$ *is related to the entropy of* $\mathbf{f} = 0$ *in* $\mathcal{D}$ *(or the number of roots) [YC04] and* $K_1(\mathbf{f}, \mathcal{D})$,
- ρ *is related to the minimal distance between the roots and their distance to the variety* $\det(J_{\mathbf{f}}(x)) = 0$ *and* $K_1(\mathbf{f}, \mathcal{D})$.

Proof. Let $k_1 = K_1(\mathbf{f}, \mathcal{D}), k_2 = K_2(f, \mathcal{D}), \tilde{k}_2 = \frac{2\,(n^2+1)\sqrt{n}\,k_2}{m_1(\sigma)}$. Let $\delta_0 = \min(\rho, \frac{1}{2\tilde{k}_2}, \frac{k_1}{k_2})$. Let $n_0 \in \mathbb{N}$ be such that $2^{-n_0}|\mathcal{D}| < \delta_0$, that is $n_0 \geq \log|\mathcal{D}| + \max(\log(\rho^{-1}), \log(\tilde{k}_2) + 1, \log k_2 - \log k_1)$. We take $n_0 = \log\rho^{-1} + \log\tilde{k}_2 + \log k_2$. By applying the subdivision algorithm n_0 times, we obtain boxes B_0 of size $< \delta_0$ such that the center $\mathbf{u}_0$ is in $S_{2\,k_1\delta_0} \subset V(\sigma)$. Therefore, we have $m_1(J_{\mathbf{f}}(\mathbf{u}_0)) > m_1(\sigma)$.

Let us denote by B_k a box obtained next by application of the reduction algorithm k^{th} times. By corollary 3, we have

$$\tilde{k}_2|B_k| < (\tilde{k}_2|B_{k-1}|)^2 < \cdots < (\tilde{k}_2|B_0|)^{2^k} < (\frac{1}{2})^{2^k}.$$

Thus if $\frac{1}{\tilde{k}_2 2^{2^k}} < \epsilon$, we have $|B_k| < \epsilon$. Since computing the inverse of $J_{\mathbf{f}}(\mathbf{u}_0)$ requires $\mathcal{O}(n^3)$ arithmetic operations and the de Casteljau algorithm $\mathcal{O}(d^{n+1})$, we deduce that the number of arithmetic operations needed to approximate the roots within ϵ is bounded by

$$\mathcal{O}(N_{\mathcal{D}}(\mathbf{f}) \times d^{n+1}(n_0 + \log(\tilde{k}_2) + n^3 \log|\log\epsilon|))$$

which yields the bound of the theorem.

6 Experimentations

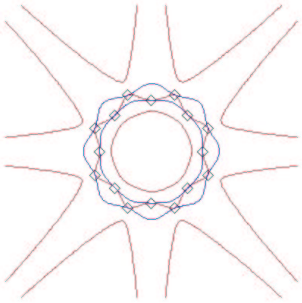
The structure of the algorithm described in the section 4 has been implemented in the C++ library SYNAPS[1]. Our objective in these experimentations is to evaluate the impact of reduction approaches compared with subdivision techniques, with and without preconditioning. The different methods that we compare are the following:

- **sbd** stands for subdivision; it is the simplest approach which looks to the minimum and maximum control coefficients on an equation-per-equation basis. It is similar to [EK01].
- **rd** stands for reduction. Because it uses a more sophisticated reduction step it produces less iterations than the method in [SP93].
- **sbds** stands for subdivision using the global preconditioner (section 4.2) we described. It works in the same way as **sbd** but on a transformed system.
- **rds** stands for reduction using the global preconditioner (section 4.2).
- **rdl** stands for reduction using the local preconditioner (section 4.2).

[1] `http://www-sop.inria.fr/galaad/software/synaps`

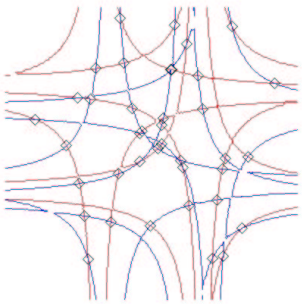
For each method and each example, we output the number of iterations of the solver, the number of subdivisions steps, the number of domains computed at the end and the time in milliseconds on a Intel Pentium 4 2.0Ghz with 512 Mo RAM, workstation. The interesting characteristics are the number of iterations and the size of the output. In these experiments we use a modification of the first reduction solver based on Descartes rule (section 3). It handles all the projections associated to one variable, simultaneously and is able to reject domains earlier. Simple roots are approximated using bisection. In these experimentations, we use one of the simplest univariate root-finder we have described. Notice that changing the root-finder can improve the computation time but not the number of iterations. The subdivision rule consists in a splitting along the largest variable, when the reduction gain is close to 1.

We consider examples of implicit curve intersection problems defined by bi-homogeneous polynomials in examples (a) and (b) . Then we consider the intersection (c) and (d) and the self-intersection (e) and (f) problems for rational curves.



(a) Bidegree: $(8,8),(8,8)$

method	iter.	subd.	result	time (ms)
sbd	84887	84887	28896	3820
rd	82873	51100	20336	4553
sbds	6076	6076	364	333
rds	1486	920	144	163
rdl	1055	305	60	120

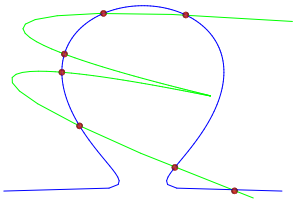


(b) Bidegrees: $(12,12),(12,12)$.

method	iter.	subd.	result	time (ms)
sbd	4826	4826	220	217
rd	2071	1437	128	114
sbds	3286	3286	152	180
rds	1113	748	88	117
rdl	389	116	78	44

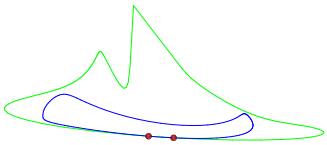
Given two rational curves $C_0 = (x_0(u), y_0(u), z_0(u))$ and $C_1 = (x_1(v), y_1(v), z_1(v))$ we compute their intersection by solving the system: $S = C_0 \times C_1 = 0$, where $\times$ is the cross-product. For computing the self-intersection of a curve, we solve the system $S = \frac{C(u)\times C(v)}{u-v} = 0$ and use a domain control class that clip the domain to $\{u < v\}$. In the following examples of in-

tersection (c) and (d) and self-intersection (e) and (f), the solutions are computed up within the precision of 10^{-6}.



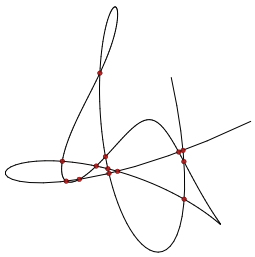
(c) **Bidegree:** $(4, 5)$

method	iter.	subd.	result	time (ms)
sbd	2159	2159	21	40.8
rd	185	100	8	7.76
sbds	681	681	8	26.95
rds	108	38	7	10.45
rdl	75	28	7	4.72



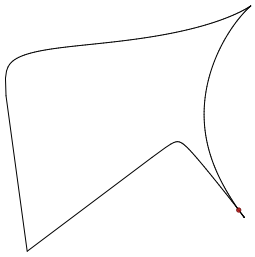
(d) **Bidegrees:** $(14, 16)$

method	iter.	subd.	result	time (ms)
sbd	2193	2193	91	2750.55
rd	584	395	47	286.62
sbds	289	289	2	148.22
rds	70	20	2	51.95
rdl	49	15	2	28.26



(e) **Degree:** 19

method	iter.	subd.	result	time (ms)
sbd	3979	3979	39	3540.41
rd	560	376	15	537.11
sbds	1577	1577	16	1589.27
rds	282	63	13	344.9
rdl	126	36	13	134.21



(f) **Degree:** 12

method	iter.	subd.	result	time (ms)
sbd	4647	4647	47	1102.67
rd	1497	1012	23	409.9
sbds	255	255	2	69.47
rds	85	24	1	32.7
rdl	29	6	1	10.15

On these bivariate systems we conclude on the superiority of the reduction approaches,

namely reduction with local straightening (**rdl**) which have a good convergence (quadratic) in theory as well as in practice. One the other hand in hard case (figure 6) preconditioned reduction and subdivision methods (**sbds, rdl, rds**) outperforms classical subdivision and reduction (**rd, sbd**). Notice that the improvement provided by the combination of reduction and preconditioning is interesting in practice, not only because of the time saved, but also because the size of the result is smaller. Notice that in most of the examples only **rds** and **rdl** provide the good answer.

Since our reduction principle is based on projection, difficulties may arise when we increases the number of variables in our systems. Hereafter we show some experiments for a six variable problem of degree ≤ 2 in each variable, coming from the robotics community. In this example we keep the three reduction methods: **rd**, **rdl**, **rds** and add combinations of them that use projections: before and after local straightening (**rd**+**rdl**), before and after global preconditioning (**rd**+**rds**), and the three techniques (**rd**+**rdl**+**rds**).

	domain $= [-3,3]^6$					domain $= [-30,30]^6$			
method	iter.	subd.	result	time (s)					
rd	34125	6059	492	14.5					
rdl	33471	10543	4	23.9	**rd**	249578	54765	532	101.6
rds	9082	2288	8	12.3	**rd**+**rdl**	27833	6623	4	26.9
rds+**rdl**	3555	872	4	6.4	**rd**+**rds**	9908	2051	18	15.3
rd+**rdl**	2814	655	4	2.7	**rd**+**rdl**+**rds**	7196	1483	4	14.3
rd+**rds**	3120	670	15	4.8					
rd+**rdl**+**rds**	1863	409	4	3.7					

As one can see the combination of projections is an improvement, but the important point here is that the global preconditioner tends to be better than local straightening. If we look at the first table, we see that the number of iterations of **rdl** is close to **rd**, but it is not the case for **rds**. The reason is that **rdl** uses a local information (a Jacobian evaluation) while **rds** use a global information computed using all the coefficients in the system.

In conclusion of this section, we first experiment that both subdivisions and reductions methods are bad solutions if they are not preconditioned. But using the same preconditioner reduction will, most of the time, beat subdivision.

7 Conclusion

Our approach can be seen as an enhancement of the previous works of Sherbrooke and Patrikalakis [SP93]. They proposed a reduction method. We show how to generalizes it by univariate root solving methods. Their convergence speed is locally linear, we show that preconditioning can leads to quadratic convergence. We also show that subdivisions methods can be improved by preconditioning. We give an answer to Elber and Kim [EK01], who argued that the reduction approaches are not so interesting because they cannot avoid a lot of subdivisions. Our experimentations shows that reduction can save a lot of subdivisions. In cases of "tangents solutions", a pure subdivision approach has no chance to compete with a reduction approach. However it is better to use a preconditioned subdivision than a pure reduction approach, our experimentations illustrate this phenomenon.

Bibliography

[BPR03] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag, Berlin, 2003. ISBN 3-540-00973-6.

[CLO92] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer Verlag, New York, 1992.

[Dah86] Wolfang Dahmen. Subdivision algorithms converge quadratically. *Journal of Computational and Applied Mathematics*, 2:145–158, 1986.

[dB87] Carl de Boor. B-form basics. *Geometric Modeling*, 2:27–49, 1987.

[EK01] Gershon Elber and Myung-Soo Kim. Geometric constraint solver using multivariate rational spline functions. In *Proceedings of the sixth ACM Symposium on Solid Modelling and Applications*, pages 1–10. ACM Press, 2001.

[Far90] G. Farin. *Curves and surfaces for computer aided geometric design : a practical guide*. Comp. science and sci. computing. Acad. Press, 1990.

[FG96] R.T. Farouki and T.N.T Goodman. On the optimal stability of the bernstein basis. *Mathematics of computation*, 65(216):1553–1566, October 1996.

[FR87] R. Farouki and V. Rajan. On the numerical condition of polynomials in bernstein form. *Computer Aided Geometric Design*, 4(3):191–216, 1987.

[GS01a] Jürgen Garloff and Andrew P. Smith. Investigation of a subdivision based algorithm for solving systems of polynomial equations. *Nonlinear Anal.*, 47(1):167–178, 2001.

[GS01b] Jürgen Garloff and Andrew P. Smith. Solution of systems of polynomial equations by using Bernstein expansion. In *Symbolic algebraic methods and verification methods (Dagstuhl, 1999)*, pages 87–97. Springer, Vienna, 2001.

[Hen77] P. Henrici. *Applied and computational complex analysis*. John Wiley & Sons, New York, NY, 1977.

[JR81] J.Lane and R.Riesenfeld. Bounds on a polynomial. *BIT*, 21:112–117, 1981.

[Meh] K. Mehlhorn. A remark on the sign variation method for real root isolation. Submitted for publication, report ECG-TR-123101-01.

[MRR04] B. Mourrain, F. Rouillier, and M.-F. Roy. Bernstein's basis and real root isolation. Technical Report 5149, INRIA Rocquencourt, 2004. Submitted for journal publication.

[MVY02] B. Mourrain, M. Vrahatis, and J.C. Yakoubsohn. On the complexity of isolating real roots and computing with certainty the topological degree. *J. of Complexity*, 18(2):612–640, 2002.

[PK94] H. Prautzsch and L. Kobbelt. Convergence of subdivision and degree elevation. *Advances in Computational Mathematics*, 1:143–154, 1994.

[PM02] M. P. Patrikalakis and T. Maekawa. *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer Verlag, 2002.

[Rhe98] Werner C. Rheinboldt. *Methods for solving systems of nonlinear equations*, volume 70 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 1998.

[Ris91] J.J. Risler. *Méthodes mathématiques pour la CAO*. Masson, 1991.

[Roc90] Alyn Rockwood. Accurate display of tensor product isosurfaces. In *IEEE Visualization '90 Conf.*, 1990.

[SM88] T. W. Sederberg and R. J. Meyers. Loop detection in surface patch intersections. *Computer Aided Geometric Design*, 5(2):161–171, 1988.

[SP93] E. C. Sherbrooke and N. M. Patrikalakis. Computation of the solutions of nonlinear polynomial systems. *Comput. Aided Geom. Design*, 10(5):379–405, 1993.

[Spe94] M. Spencer. *Polynomial Real Root Finding in Bernstein Form*. PhD thesis, Brigham Young University., 1994.

[TNK90] T.W. Sederberg T. Nishita and M. Kakimoto. Ray tracing trimmed rational surface patches. *Computer Graphics*, 24(4 (Proc. ACM Siggraph 90)):337–345, August 1990.

[YC04] Y. Yomdin and G. Comte. *Tame geometry with applications in smooth analysis*. LNM 1834. Springer-Verlag, 2004.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399