Draft of September 1, 2009

KHOVANSKII-ROLLE CONTINUATION FOR REAL SOLUTIONS

DANIEL J. BATES AND FRANK SOTTILE

Abstract. We present a new continuation algorithm to find all nondegenerate real solutions to a system of polynomial equations. Unlike homotopy methods, it is not based on a deformation of the system; instead, it traces real curves connecting the solutions of one system of equations to those of another, eventually leading to the desired real solutions. It also differs from homotopy methods in that it follows only real paths and computes no complex solutions of the original equations. The number of curves traced is bounded by the fewnomial bound for real solutions, and the method takes advantage of any slack in that bound.

Introduction

Numerical continuation gives efficient algorithms for finding all complex solutions to a system of polynomial equations [18]. Current implementations [4, 15, 19] routinely and reliably solve systems with thousands of solutions having dozens of variables. Often it is the real solutions or the solutions with positive coordinates that are sought. In these cases, numerical continuation first finds all complex solutions, which are then sifted to obtain the desired real ones, i.e., those with sufficiently small imaginary part. This is inefficient, particularly for fewnomial systems (systems of polynomials with few monomials [13]) which possess bounds on their numbers of positive [7] and of real [3] solutions that can be far smaller than their number of complex solutions. More fundamentally, homotopy methods do not exploit the real algebraic nature of the problem.

We present a numerical continuation algorithm to find all nondegenerate real solutions (or all positive solutions) to a system of polynomials. This Khovanskii-Rolle continuation algorithm is modeled on the derivation of fewnomial bounds [3, 7] for the numbers of real solutions. It is efficient in that the number of paths followed depends upon the corresponding fewnomial bounds and not on the number of complex solutions. This algorithm also naturally takes advantage of any slack in the fewnomial bound. All path tracing is along real curves and except for a precomputation, the algorithm is completely real.

To the best of our knowledge, this is the first continuation algorithm that finds only real solutions to a system of equations. There are other numerical algorithms that find only real solutions. Exclusion [18, §6.1] recursively subdivides a bounded domain excluding subdomains that cannot harbor solutions. This is an active area with research into

2000 *Mathematics Subject Classification.* 14P99, 65H10, 65H20.

Key words and phrases. fewnomial, Khovanskii–Rolle, Gale dual, homotopy, continuation, polynomial system, numerical algebraic geometry, real algebraic geometry.

Bates and Sottile supported by the Institute for Mathematics and Its Applications.

Bates supported by NSF grant DMS-0914674.

Sottile supported by the NSF CAREER grant DMS-0538734 and NSF grant DMS-0701050.

better exclusion tests [10, 16] and complexity [1]. Lasserre, Laurent, and Rostalski [14] recently proposed another method based on sums-of-square and semidefinite programming. Cylindrical algebraic decomposition [9] is a symbolic method that computes of a detailed stratification of the ambient space in addition to the solutions. Gröbner bases may be used to compute a rational univariate representation [17] from which the solutions are found by substituting the roots of a univariate polynomial into the representation.

Many existing methods have drawbacks. Exclusion suffers from the curse of dimensionality: the growth in the number of cells may be exponential in the number of variables. Cylindrical algebraic decomposition is infeasible in dimensions above 3 due to the complexity of the data structures involved. Gröbner basis algorithms essentially compute all complex solutions and much more in addition to the real solutions.

Our algorithm is based on Khovanskii's generalization of Rolle's Theorem [13] and the proof of the fewnomial bound [3, 7], and we are indebted to Bernd Sturmfels who observed that this proof leads to a numerical algorithm to compute real solutions to systems of polynomials. This algorithm does not directly solve the given polynomial system but rather an equivalent (Gale dual [8]) system consisting of master functions in the complement of an arrangement of hyperplanes. If the original system involves n polynomials in n variables having a total of $l+n+1$ monomials, then the Gale dual system consists of l master functions in the complement of an arrangement of $l+n+1$ hyperplanes in $\mathbb{R}^l$. Gale duality [8] asserts that the two systems are equivalent and produces an algebraic bijection between complex solutions, which restricts to a bijection between their real solutions, and further restricts to a bijection between their (appropriately defined) positive solutions.

Here, we describe the Khovanskii-Rolle algorithm and a proof-of-concept implementation. This implementation uses the system Bertini [4] for precomputation, but it is otherwise implemented in Maple. It is also restricted to the case of finding positive solutions when $l = 2$. Detailed examples are on our webpage [6], which also contains our software and a brief guide. We plan significant further work on this algorithm, including source code and compiled binaries of an implementation that places no restrictions on l and links to numerical homotopy continuation software [4, 15, 19] for the precomputation. We also plan theoretical work on the technical challenges posed by this algorithm.

As the algorithm does not solve the original system, but rather an equivalent system in a different number of variables, we first explain that transformation, together with an example and an important reduction in Section 1. We also discuss curve tracing and homotopy continuation in Section 1. In Section 2, we explain our algorithm, give a proof of correctness, and examine its complexity. We illustrate the Khovanskii-Rolle algorithm on an example in Section 3 and describe the performance of our implementation on this example and on one that is a bit more extreme. Surprisingly, our simple maple implementation for finding positive solutions outperforms both Bertini [4] and PHCpack [19]. In Section 4, we describe our implementation, and we end the paper with a discussion of future work. Please note that we intend "nondegenerate real solutions" when we say "real solutions" throughout, unless otherwise noted.

1. Background

We first explain how Gale duality transforms a system of polynomial equations into an equivalent system of rational master functions. This reduces the problem of finding all positive solutions to the polynomial equations to finding solutions to the master functions in a polyhedron, which we may assume is bounded. Then we discuss general curve-tracing algorithms and finally outline the method of homotopy continuation to highlight how it differs from Khovanskii-Rolle continuation.

1.1. Gale Duality.

The Khovanskii-Rolle algorithm does not in fact directly solve polynomial systems, but rather solves systems of master functions defined in the complement of an arrangement of hyperplanes. Solutions of the system of master functions correspond to solutions of the original system via Gale duality [8]. We begin with an example.

Example 1.1. The system of Laurent polynomials (written in diagonal form)

$$\begin{array}{rclcrcl} cd &=& \frac{1}{2}be^2 + 2a^{-1}b^{-1}e - 1 & & cd^{-1}e^{-1} &=& \frac{1}{2}(1 + \frac{1}{4}be^2 - a^{-1}b^{-1}e) \\ bc^{-1}e^{-2} &=& \frac{1}{4}(6 - \frac{1}{4}be^2 - 3a^{-1}b^{-1}e) & & bc^{-2}e &=& \frac{1}{2}(8 - \frac{3}{4}be^2 - 2a^{-1}b^{-1}e) \\ ab^{-1} &=& 3 - \frac{1}{2}be^2 + a^{-1}b^{-1}e\ , & & & & \end{array} \tag{1.1}$$

has as support (set of exponent vectors) the columns of the matrix

$$\mathcal{A} := \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 0 & 1 & 0 & 1 & -1 & -2 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 2 & -1 & -2 & 1 & 0 \end{pmatrix}.$$

Since

$$\mathcal{A}\begin{pmatrix} -1 & 1 & -2 & 1 & -2 & 2 & -1 \\ 1 & 6 & -3 & 6 & -2 & 7 & 1 \end{pmatrix}^T = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix},$$

we have the following identity on the monomials

$$\begin{array}{rcl} (a^{-1}b^{-1}e)^{-1}(cd)^1(be^2)^{-2}(cd^{-1}e^{-1})^1(bc^{-1}e^{-2})^{-2}(bc^{-2}e)^2(ab^{-1})^{-1} &=& 1\,, \\ (a^{-1}b^{-1}e)^1\ (cd)^6(be^2)^{-3}(cd^{-1}e^{-1})^6(bc^{-1}e^{-2})^{-2}(bc^{-2}e)^7(ab^{-1})^1 &=& 1\,. \end{array} \tag{1.2}$$

A Gale system dual to (1.1) is obtained from the identity (1.2) by first substituting x for the term $\frac{1}{4}be^2$ and y for $a^{-1}b^{-1}e$ in (1.1) to obtain

$$\begin{array}{rclcrcl} cd &=& 2x + 2y - 1 & & cd^{-1}e^{-1} &=& \frac{1}{2}(1 + x - y) \\ bc^{-1}e^{-2} &=& \frac{1}{4}(6 - x - 3y) & & bc^{-2}e &=& \frac{1}{2}(8 - 3x - 2y) \\ ab^{-1} &=& 3 - 2x + y\ , & & & & \end{array}$$

Then, we substitute these linear polynomials for the monomials in (1.2) to obtain the system of rational master functions

$$\begin{aligned} y^{-1}(2x+2y-1)\ (4x)^{-2}\left(\tfrac{1+x-y}{2}\right)\ \left(\tfrac{6-x-3y}{4}\right)^{-2}\left(\tfrac{8-3x-2y}{2}\right)^{2}(3-2x+y)^{-1} &= 1\,,\\ y(2x+2y-1)^{6}(4x)^{-3}\left(\tfrac{1+x-y}{2}\right)^{6}\left(\tfrac{6-x-3y}{4}\right)^{-2}\left(\tfrac{8-3x-2y}{2}\right)^{7}(3-2x+y) &= 1\,. \end{aligned} \tag{1.3}$$

If we solve these for 0, they become $f = g = 0$, where

$$\begin{aligned} f &:= (2x+2y-1)(1+x-y)(8-3x-2y)^2 \ - \ 8yx^2(6-x-3y)^2(3-2x+y)\,,\\ g &:= y(2x+2y-1)^6(1+x-y)^6(8-3x-2y)^7(3-2x+y) \ - \ 32768\,x^3(6-x-3y)^2\,. \end{aligned} \tag{1.4}$$

Figure 1 shows the curves defined by f and g and the lines given by the linear factors in f and g. The curve $f = 0$ has the three branches indicated and the other arcs belong to $g = 0$.

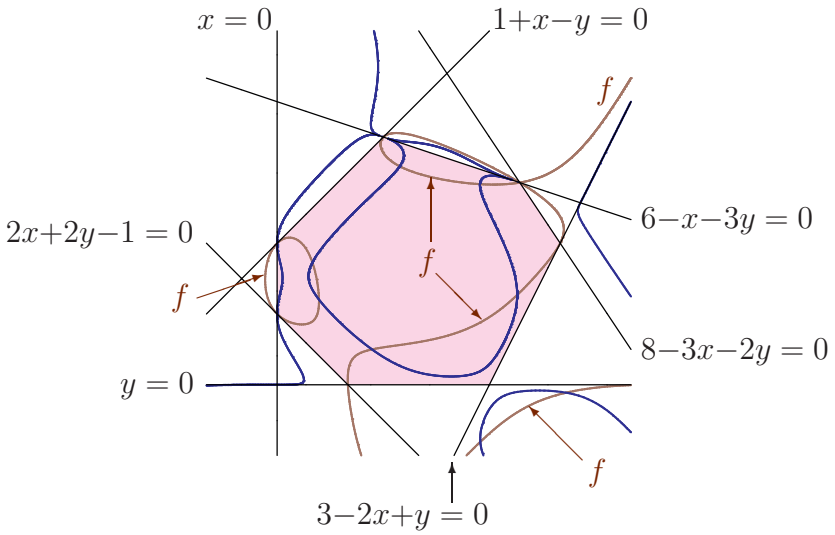


FIGURE 1. Curves and lines

It is clear that the solutions to $f = g = 0$ in the complement of the lines are consequences of solutions to (1.1). More, however, is true. The two systems define isomorphic schemes as complex or as real varieties, with the positive solutions to (1.1) corresponding to the solutions of $f = g = 0$ lying in the central heptagon. Indeed, the polynomial system (1.1) has 102 solutions in $(\mathbb{C}^\times)^5$. This may be verified with either symbolic software such as Singular [11], or with numerical solvers. Ten of these solutions are real, with six positive. We computed them using both Bertini [4] and PHCpack [19] (which produce the same solutions) and display them (in coordinates (a,b,c,d,e)) to 3 significant digits.

$$\begin{aligned} &(-8.92,-1.97,-0.690,3.98,-1.28)\,, && (-0.0311,8.52,-1.26,-14.1,-1.39)\,,\\ &(-0.945,3.41,1.40,0.762,1.30)\,, && (-5.21,4.57,2.48,1.20,1.34)\,,\\ &(2.19,0.652,0.540,2.27,1.24)\,, && (2.45,0.815,0.576,1.38,1.20)\,,\\ &(3.13,1.64,0.874,0.962,1.28)\,, && (2.00,0.713,1.17,3.28,2.20)\,,\\ &(1.61,1.03,2.37,1.98,2.35)\,, && (0.752,3.10,2.36,1.55,1.48)\,. \end{aligned} \tag{1.5}$$

For the system $f = g = 0$ of master functions (1.4) in the complement of the lines shown in Figure 1, symbolic and numerical software finds 102 solutions with ten real solutions and six lying in the heptagon of Figure 1. We give the real solutions to 3 significant digits

$$(-0.800, -0.0726)\,,\ (4.13, 5.26)\,,\ (1.44, -0.402)\,,\ (2.04, -0.0561)\,,$$
$$(0.249, 0.864)\,, (0.296, 0.602)\,,\ (0.670, 0.250)\,,\ (0.866, 1.54)\,,\ (1.43, 1.42)\,,\ (1.70, 0.634)\,.$$

These correspond (in order) to the solutions in the list (1.5) under the map $(a, b, c, d, e) \mapsto (be^2/4, ea^{-1}b^{-1})$. Gale duality generalizes this equivalence.

Let $\mathcal{A} = \{0, \alpha_1, \ldots, \alpha_{l+n}\} \subset \mathbb{Z}^n$ be a collection of $1+l+n$ integer vectors, which we consider to be exponents of (Laurent) monomials: If $\alpha = (a_1, \ldots, a_n)$, then $x^\alpha := x_1^{a_1} \cdots x_n^{a_n}$. A polynomial with *support* $\mathcal{A}$ is a linear combination of monomials from $\mathcal{A}$,

$$f \ := \ \sum_{\alpha\in\mathcal{A}} c_\alpha\, x^\alpha\,,$$

where $c_\alpha \in \mathbb{R}$ – our polynomials and functions are real. Since we allow negative exponents, f is defined on the non-zero real numbers, $(\mathbb{R}^\times)^n$. We consider systems of polynomials

$$f_1(x_1, \ldots, x_n) \ = \ f_2(x_1, \ldots, x_n) \ = \ \cdots \ = \ f_n(x_1, \ldots, x_n) \ = \ 0\,, \tag{1.6}$$

in which each polynomial has the same support $\mathcal{A}$. We also assume that the system is general, by which we mean that it defines a finite set of non-degenerate points in $(\mathbb{R}^\times)^n$, in that the differentials of the polynomials f_i are linearly independent at each solution.

This implies that the polynomials f_i are linearly independent over $\mathbb{R}$, in particular that the coefficient matrix of n of the monomials, say $x^{\alpha_1}, \ldots, x^{\alpha_n}$, is invertible. We may then solve (1.6) for these monomials to obtain a diagonal system

$$x^{\alpha_i} \ = \ p_i(x^{\alpha_{n+1}}, \ldots, x^{\alpha_{l+n}})\,, \qquad \text{for } i = 1, \ldots, n\,, \tag{1.7}$$

where the p_i are degree 1 polynomials in their arguments. This has the same form as (1.1).

Let $\mathcal{B} := \{\beta_1, \ldots, \beta_l\} \subset \mathbb{Z}^{l+n}$ be a basis for the free Abelian group $\mathcal{A}^\perp$ of integer linear relations among the vectors in $\mathcal{A}$,

$$\mathcal{A}^\perp \ := \ \{(b_1, \ldots, b_{l+n}) \in \mathbb{Z}^{l+n} \mid b_1\alpha_1 + \cdots + b_{l+n}\alpha_{l+n} \ = \ 0\}\,.$$

If we write $\beta_j = (\beta_{j,i} \mid i = 1, \ldots, l+n)$, then the monomials x^{α_i} satisfy

$$1 \ = \ (x^{\alpha_1})^{\beta_{j,1}} \cdots (x^{\alpha_{l+n}})^{\beta_{j,l+n}}\,, \qquad \text{for } j = 1, \ldots, l\,.$$

Using the diagonal system (1.7) to substitute $p_i = p_i(x^{\alpha_{n+1}}, \ldots, x^{\alpha_{l+n}})$ for x^{α_i}, we obtain

$$1 \ = \ p_1^{\beta_{j,1}} \cdots p_n^{\beta_{j,n}} \cdot (x^{\alpha_{n+1}})^{\beta_{j,n+1}} \cdots (x^{\alpha_{l+n}})^{\beta_{j,l+n}}\,, \qquad \text{for } j = 1, \ldots, l\,.$$

If we now set $y_i := x^{\alpha_{n+i}}$ and write $p_{n+i}(y) = y_i$ for $i = 1, \ldots, l$, then we get

$$1 \ = \ p_1(y)^{\beta_{j,1}} \cdots p_{l+n}(y)^{\beta_{j,l+n}}\,, \qquad \text{for } j = 1, \ldots, l\,. \tag{1.8}$$

These polynomials $p_i(y)$ define an arrangement $\mathcal{H}$ of hyperplanes in either $\mathbb{R}^l$ or $\mathbb{C}^l$. Because the exponents $\beta_{i,j}$ may be negative, the system (1.8) only makes sense in the complement $M_\mathcal{H}$ of this arrangement. These rational functions, $p_1(y)^{\beta_{j,1}} \cdots p_{l+n}(y)^{\beta_{j,l+n}}$, are called *master functions*, and they arise in the theory of hyperplane arrangements. We give the main result of [8].

Theorem 1.2. *If the integer linear span $\mathbb{Z}\mathcal{A}$ of the exponents $\mathcal{A}$ is equal to $\mathbb{Z}^n$, then the association $x^{\alpha_{n+i}} \leftrightarrow y_i$ defines a bijection between solutions to the system of polynomials* (1.6) *in $(\mathbb{C}^\times)^n$ and solutions to the system of master functions* (1.8) *in $M_{\mathcal{H}}$.*

This bijection restricts to a bijection between real solutions to both systems and further to a bijection between the positive solutions to (1.6) *and the solutions to the system of master functions* (1.8) *which lie in the positive chamber $\Delta := \{y \in \mathbb{R}^l \mid p_i(y) > 0,\ i = 1, \ldots, l+n\}$.*

The bijections of Theorem 1.2 respect the multiplicities of the solutions. Because of this equivalence between the two systems, and because the exponents $\mathcal{A}$ and $\mathcal{B}$ come from Gale dual vector configurations, we call the system (1.8) of master functions Gale dual to the original polynomial system.

Theorem 1.2 may be strengthened as follows. There still is a bijection between real solutions if we only require that $\mathbb{Z}\mathcal{A}$ have odd index in $\mathbb{Z}^n$ and that the span of $\{\beta_1, \ldots, \beta_l\}$ have odd index in $\mathcal{A}^\perp$. The bijection between positive solutions in $\mathbb{R}^n$ and solutions in the positive chamber Δ in $\mathbb{R}^l$ remains valid if we allow $\mathcal{A}$ and β_i to be any real vectors such that $\mathcal{A}$ spans $\mathbb{R}^n$ and $\{\beta_1, \ldots, \beta_l\}$ is a basis of the real vector space $\mathcal{A}^\perp$. The Khovanskii-Rolle continuation algorithm is naturally formulated in terms of real exponents β_i.

1.1.1. *Master functions on bounded polyhedra.* Before we discuss the numerical algorithms of path-following and homotopy continuation, we show that it is no loss to assume that this positive chamber Δ is bounded. Suppose that $p_1(y), \ldots, p_{l+n}(y)$ are degree 1 polynomials in $y \in \mathbb{R}^l$ that span the linear space of degree 1 polynomials, let $(\beta_{i,j}) \in \mathbb{R}^{(l+n)\times l}$ be an array of real numbers, and consider the system of master functions

$$\prod_{i=1}^{l+n} p_i(y)^{\beta_{i,j}} \ = \ 1 \qquad \text{for} \quad j = 1, \ldots, l\,, \tag{1.9}$$

in the polyhedron

$$\Delta \ := \ \{y \in \mathbb{R}^l \mid p_i(y) > 0,\ i = 1, \ldots, l+n\}\,.$$

Suppose that Δ is unbounded. By our assumption on $p_1, \ldots, p_{l+n}$, Δ is strictly convex and therefore has a bounded face, F. Then there is a degree 1 polynomial, $q(y) := a_0 + \alpha \cdot y$ where $\alpha \in \mathbb{R}^l$ and $a_0 > 0$, that is strictly positive on Δ, and such that F is the set of points of Δ where $q(y)$ achieves its minimum value on Δ. Dividing by a_0, we may assume that the constant term of $q(y)$ is 1.

Consider the projective coordinate change $y \Rightarrow \overline{y}$, where

$$\overline{y}_j \ := \ y_j/q(y) \qquad \text{for } j = 1, \ldots, l\,. \tag{1.10}$$

Then we invite the reader to check that

$$y_j \ = \ \overline{y}_j/r(\overline{y}) \qquad \text{for } j = 1, \ldots, l\,,$$

where $r(\overline{y}) := 1 - \alpha \cdot \overline{y}$. Note that $q(y)r(\overline{y}) = 1$.

The coordinate change (1.10) manifests itself on degree 1 polynomials as follows. If $p(y)$ is a degree 1 polynomial, then

$$p(y) \ = \ \overline{p}(\overline{y})/r(\overline{y}) \qquad \text{and} \qquad \overline{p}(\overline{y}) \ = \ p(y)/q(y)\,, \tag{1.11}$$

where $\overline{p}(\overline{y}) := p(\overline{y}) - p(0)\alpha\cdot\overline{y}$. Under the projective transformation (1.10), the polyhedron Δ is transformed into $\overline{\Delta}$, where

$$\overline{\Delta} \;:=\; \{\overline{y}\in\mathbb{R}^{\ell} \mid r(\overline{y})>0 \;\text{ and }\; \overline{p}_i(\overline{y})>0 \quad \text{for } i=1,\dots,l+n\}\,.$$

Proposition 1.3. *Under the coordinate change* (1.10)*, the system of master functions* (1.9) *on* Δ *is transformed into the system*

$$\prod_{i=0}^{l+n}\overline{p}_i(\overline{y})^{\beta_{i,j}} \;=\; 1 \qquad \textit{for} \quad j=1,\dots,l\,,$$

on the polyhedron $\overline{\Delta}$*, where* $\overline{p}_0(\overline{y})=\overline{q}(\overline{y})$ *and* $\beta_{0j}:=-\sum_{i=1}^{l+n}\beta_{i,j}$*. Furthermore,* $\overline{\Delta}$ *is bounded.*

Proof. If $y\in\Delta$ and $\overline{y}\in\overline{\Delta}$ are related by the coordinate transformation (1.10), then $p_i(y)=\overline{p}_i(\overline{y})/r(\overline{y})$. Thus

$$\prod_{i=1}^{l+n}p_i(y)^{\beta_{i,j}} \;=\; \left(\prod_{i=1}^{l+n}\overline{p}_i(\overline{y})^{\beta_{i,j}}\right)\Big/ r(\overline{y})^{\sum_i\beta_{i,j}} \;=\; \prod_{i=0}^{l+n}\overline{p}_i(\overline{y})^{\beta_{i,j}}\;,$$

where $\overline{p}_0(\overline{y})=r(\overline{y})$ and $\beta_{0j}=-\sum_i\beta_{i,j}$. This proves the first statement.

All that remains is to show that $\overline{\Delta}$ is bounded. The polyhedron Δ is the Minkowski sum

$$\Delta \;=\; P \;+\; R\,, \tag{1.12}$$

where P is a bounded polytope and R is a strictly convex polyhedral cone with vertex the origin. The bounded faces of Δ are faces of P. In particular, the bounded face F consisting of points where q achieves its minimum on Δ is also a face of P. Additionally, q achieves it minimum value on R at the origin. In particular, if $0\neq v\in R$, then $\alpha\cdot v=q(v)-q(0)>0$.

Let π be an upper bound for the ratio $\|y\|/q(y)$ for $y\in P$, which bounds $\|\overline{y}\|$ when $\overline{y}$ corresponds to a point in P. Letting v range over a set of generators of the (finitely many) extreme rays of R shows that there is a positive constant ρ such that

$$\alpha\cdot v \;\geq\; \|v\|/\rho \qquad \text{for} \quad v\in R\,.$$

Let $\overline{y}\in\overline{\Delta}$. Then $y:=\overline{y}/r(\overline{y})$ is the corresponding point of Δ. Writing $y=y_P+y_R$ where $y_P\in P$ and $y_R\in R$, then $q(y)=q(y_P)+\alpha\cdot y_R$ with both terms non-negative, and we have

$$\|\overline{y}\| \;=\; \left\|\frac{y}{q(y)}\right\| \;\leq\; \frac{\|y_P\|}{q(y)}+\frac{\|y_R\|}{q(y)} \;\overset{!}{\leq}\; \frac{\|y_P\|}{q(y_P)}+\frac{\|y_R\|}{\alpha\cdot y_R} \;\leq\; \pi+\rho\;.$$

This shows that $\overline{\Delta}$ is bounded. □

We remark that this coordinate transformation $y\mapsto\overline{y}$ is a concrete and explicit projective transformation transforming the unbounded polyhedron Δ into a bounded polyhedron $\overline{\Delta}$.

1.2. **Curve tracing.** Let $g\colon\mathbb{R}^n\to\mathbb{R}^{n-1}$ be a smooth function such that $g^{-1}(0)$ is a smooth curve C. Given a point $c\in\mathbb{R}^n$ on or near C, to move along the arc of C near c is to produce a series of approximations to points along this arc. Predictor-corrector methods [2] are the standard general numerical method for this task. They proceed as follows:

I Prediction. Move some specified distance (the *steplength*) Δt along the tangent line to C at c. (The tangent direction is the kernel of the Jacobian matrix of g at C.) That is, if $\widetilde{T}$ is the unit tangent vector to C pointing in the appropriate direction, set $c_p := c + \Delta t \cdot \widetilde{T}$.

II Correction. Choose a linear equation $L(z) = 0$ vanishing at c_p and transverse to $\widetilde{T}$, for example $L(z) := \widetilde{T} \cdot (z - c_p)$. If Δt is sufficiently small, then c_p is close to a point on C where $L = 0$. Adding L to g gives a system $\widetilde{g}$ with c_p close to a solution, and Newton iterations starting at c_p for the system $\widetilde{g}$ approximate this nearby point on C with $L = 0$.

Since C is smooth and Δt small, the Jacobian matrix of $\widetilde{g}$ has full rank in a cylinder around C near the point c and c_p lies in that cylinder. This procedure will produce a sequence of approximations $c_p, c_1, \ldots, c_k$ to a point on C.

III Adaptive Steplength. If correction fails to converge after a preset number (say $k = 2$) of iterations, then reduce Δt by some factor, say $\frac{1}{2}$, and return to Step I. Otherwise, if several consecutive predictor-corrector steps have succeeded, increase Δt by some factor, say 2. In this way, mild curves are tracked quickly and wild curves are tracked cautiously. The entire predictor-corrector procedure is terminated as a failure if Δt falls below a threshold which is a function of the user's patience and the available numerical precision.

IV Resetting or Stopping. If the step was a success, decide whether to stop, based on stopping criteria specific to the problem being solved. If it is not time to stop, set $c = c_k$ and go to Step I. If the step was a failure, leave c as is and go to Step I.

If another branch of the curve C is near the arc we are tracing and if the steplength Δt is large enough, then c_p may fall in the basin of convergence of the other branch. The only general remedy for this *curve-jumping* is to restrict both the maximum allowed steplength Δt and the number of Newton corrections, typically to 2. Even then, curve-jumping may occur, only to be discovered with problem-specific checking procedures as described in Section 4.

There are many possible variations. For example, the tangent predictor could be replaced (after one or more initial steps) with an interpolating predictor. For more details, see [2].

1.3. **Homotopy continuation.** Predictor-corrector methods are the method of choice for homotopy continuation. They involve tracking a curve with a specific structure defined in a product of $\mathbb{C}^n$ (the space of variables, treated as $\mathbb{R}^{2n}$) and $\mathbb{C}$ (a parameter space).

For the purposes of our discussion, let

$$f(z) \;=\; 0\,,\ \text{with } f\;:\;\mathbb{C}^n \to \mathbb{C}^n\,,$$

be the *target polynomial system* for which the isolated complex solutions are sought. Based upon the structure of the system f, a *start system*

$$g(z),\ \text{with } g\;:\;\mathbb{C}^n \;\longrightarrow\; \mathbb{C}^n\,,$$

is chosen and the *homotopy*

$$H(z,t) \;:=\; f(z)\cdot(1-t) + \gamma\cdot t\cdot g(z),\ \text{with } H\;:\;\mathbb{C}^n\times\mathbb{C}\to\mathbb{C}^n\,,$$

is formed. Here, $0 \neq \gamma \in \mathbb{C}$ is a randomly chosen complex number. The start system $g(z)$ is chosen so that it is easily solved and so that its structure is related to that of f.

At $t = 1$, we know the solutions of $H(z,1) = \gamma g(z)$, and we seek the solutions of $H(z,0) = f(z)$. For any given value of t, $H(z,t)$ is a polynomial system with some number N_t of

isolated solutions. This number is constant, say N, for all except finitely many $t \in \mathbb{C}$. For a general γ, restricting $H(z,t)$ to the interval $(0,1]$ gives N real arcs, one for each solution to $g(z) = 0$ when $t = 1$.

Some arcs will extend to $t = 0$ and their endpoints give all isolated solutions of the target system $f(z) = 0$. Some may diverge (wasting computational time), and two or more arcs will converge to each singular solution. Different choices of start system $g(z) = 0$ will have different numbers of the extraneous divergent arcs. This is illustrated in Figure 2.

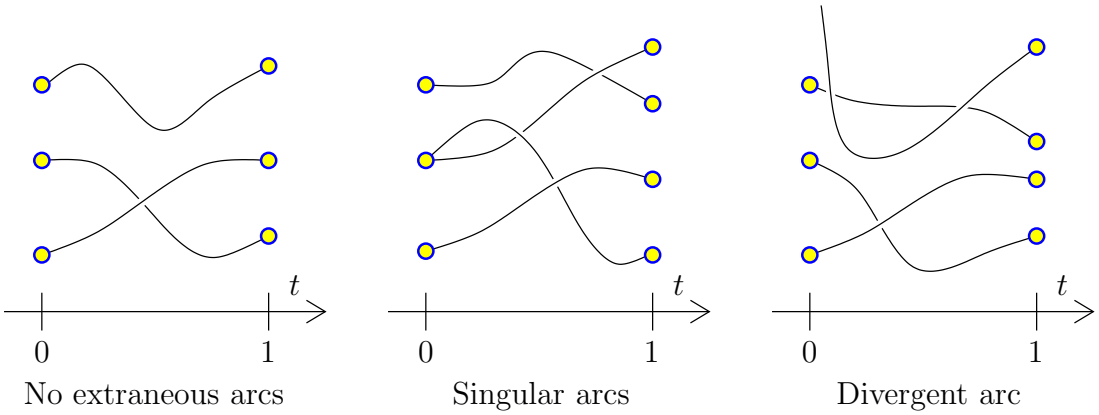


FIGURE 2. Arcs in a homotopy

Here, we *track paths* rather than *trace curves*. The predictor step is the same as in curve-tracing, but rather than correct along a line normal to the tangent, we freeze the value of t and correct back to the curve for that value of t. For more details, see [18].

1.4. **Curve-tracing versus path-tracking.** The primary difference between path-tracking in homotopy continuation and general curve-tracing as described in Section 1.2 is that in curve-tracing there are only application-specific safeguards against curve-jumping. Homotopy continuation has a well-developed theory to avoid path-crossing. For almost all γ (those that do not satisfy a certain algebraic relation), all paths will remain distinct until $t = 0$. As a result, all paths may be tracked to some small value of t and compared to check for path-crossing.

A second difference involves methods to follow ill-conditioned curves. While the curves to be tracked are generally non-singular, they are not necessarily well-conditioned. These algorithms involve repeatedly solving linear systems $Ax = b$, typically when A is some Jacobian, and the efficacy of this step depends upon the condition number $\kappa(A)$ of A. Wilkinson [20] showed that we can expect no more than $P - \log(\kappa(A))$ digits of accuracy when solving $Ax = b$ with P digits of precision. For homotopy continuation, adaptive precision techniques can overcome these potential losses in accuracy [5]. While this could likely be extended to general curve-tracing, the details have not yet been worked out.

Finally, when tracing a curve with large curvature and large initial steplength Δt, it will take some time for Δt to become small enough so that Newton's method can succeed. There are many other situations in which curve-tracing will experience difficulties. However, the paths tracked in homotopy continuation are generally very well-behaved with gentle shifts in curvature, except possibly near $t = 0$. For that, *endgames* [12] have been developed to

handle singularities at $t = 0$. Methods for handling high curvature, perhaps by adapting endgames to the general curve-tracing setting, are a topic for future research.

2. The Khovanskii-Rolle algorithm

We describe the structure of the Khovanskii-Rolle algorithm in the simple case of finding positive solutions in Section 2.2 and in the general case of finding real solutions in Section 2.3. Section 4 will discuss our actual implementation of the algorithm. Some details of various subroutines—most notably a method for following curves beginning at singular points in the boundary of Δ—will be discussed there.

The algorithm has two elementary routines: curve-tracing (as described in Section 1.2), and polynomial system solving for a precomputation. We shall treat both elementary routines as function calls in our description of the Khovanskii-Rolle continuation algorithm. The algorithm proceeds iteratively with the starting points for curve-tracing being either the output of the previous stage of the algorithm or solutions to certain polynomial systems restricted to faces of the domain polytope, Δ.

We first describe the main continuation routine. This is a curve-tracing algorithm that uses solutions to one system of equations as input to find the solutions to another. We then describe our algorithm for finding all solutions to a system of master functions within the positive chamber, followed by the general method for all real solutions.

2.1. **Continuation step.** Suppose that we have smooth functions $f_1, \ldots, f_{l-1}, g$ on a domain $\Delta \subset \mathbb{R}^l$ with finitely many common zeroes in Δ. Also suppose that the zero locus of $f_1, \ldots, f_{l-1}$ is a smooth curve C in Δ. Let J be the Jacobian determinant J-det$(f_1, \ldots, f_{l-1}, g)$ of the functions $f_1, \ldots, f_{l-1}, g$.

The goal of this step is to trace C to find all common zeroes of $f_1, \ldots, f_{l-1}, g$ in Δ, taking as input the zeroes of J on the curve C and the points where C meets the boundary of Δ. This uses an extension of Rolle's theorem due to Khovanskii. We only use a very simple version of this theorem, which may be deduced from the classical Rolle theorem.

Proposition 2.1 (Khovanskii-Rolle Theorem)**.** *Between any two zeroes of g along an arc of C there is at least one zero of J.*

Figure 3 illustrates the Khovanskii-Rolle Theorem when $l = 2$. Note that the Jacobian determinant J here is the exterior product $df \wedge dg$.

Khovanskii-Rolle continuation begins with a solution of $J = 0$ on C and traces C in both directions until either it encounters a solution of $g = 0$ on C or some other stopping criterion is satisfied, as described below.

This clearly yields all solutions to $g = 0$ on compact arcs of C. For non-compact arcs that have two or more solutions, the Khovanskii-Rolle Theorem guarantees that solutions of $J = 0$ will be adjacent to solutions of $g = 0$. To ensure that all zeroes of $g = 0$ on C are found, the algorithm also traces non-compact components of C starting from the points where C meets the boundary of Δ. Here is a more complete pseudocode description.

Algorithm 2.2 (Continuation Algorithm)**.**

Suppose that $f_1, \ldots, f_{l-1}$ are differentiable functions defined on a domain $\Delta \subset \mathbb{R}^l$ that define a smooth curve C, and g is another differentiable function with finitely many simple zeroes on C. Let $J :=$ J-det$(f_1, \ldots, f_{l-1}, g)$ be the Jacobian determinant of these functions.

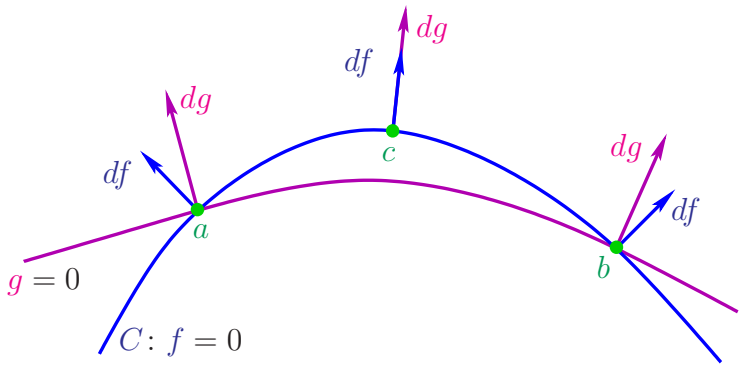


FIGURE 3. $df \wedge dg(a) < 0$, $df \wedge dg(c) = 0$, and $df \wedge dg(b) > 0$.

INPUT: Solutions S of $f_1 = \cdots = f_{l-1} = J = 0$ in the interior of Δ and the set T of points where C meets the boundary of Δ.
OUTPUT: All solutions U to $f_1 = \cdots = f_{l-1} = g = 0$ in the interior of Δ.

For each s in S, follow C in both directions from s until one of the following occurs.

(2.1)
(1) A solution $u \in U$ to $g = 0$ is found,
(2) A solution $s' \in S$ to $J = 0$ is found, or
(3) A point $t \in T$ where C meets the boundary of Δ is found.

For each t in T, follow C in the direction of the interior of Δ until one of the stopping criteria (2.1) occurs. Each point $u \in U$ is found twice.

The computation of the sets S and T of solutions is presented after the full Khovanskii-Rolle algorithm.

Proof of correctness. By the Khovanskii-Rolle Theorem, if we remove all points where $J = 0$ from C to obtain a curve C°, then no two solutions of $g = 0$ will lie in the same connected component of C°. No solution can lie in a compact component (oval) of C°, as this is also an oval of C, and the number of solutions $g = 0$ on an oval is necessarily even, so the Khovanskii-Rolle Theorem would imply that $J = 0$ has a solution on this component.

Thus the solutions of $g = 0$ lie in different connected components of C°, and the boundary of each such component consists of solutions to $J = 0$ on C and/or points where C meets the boundary of Δ. This shows that the algorithm finds all solutions $g = 0$ on C, and that it will find each twice, once for each end of the component of C° on which it lies. □

Each unbounded component of C has two ends that meet the boundary of Δ. Writing $\mathcal{V}(f_1, \ldots, f_{l-1}, g)$ for the common zeroes in Δ to $f_1, \ldots, f_{l-1}, g$, and $\mathrm{ubc}(C)$ for the unbounded components of C, we obtain the inequality

$$2\#\mathcal{V}(f_1, \ldots, f_{l-1}, g) \;\leq\; \#\text{paths followed} \;=\; 2\#\mathrm{ubc}(C) \;+\; 2\#\mathcal{V}(f_1, \ldots, f_{l-1}, J)\,, \tag{2.2}$$

as each solution is obtained twice. This gives a bound on the number of common zeroes of $f_1, \ldots, f_{l-1}, g$ in Δ and it will be used to estimate the complexity of this algorithm.

2.2. **Khovanskii-Rolle algorithm for positive solutions.** The Khovanskii-Rolle algorithm yields all solutions to a system of equations of the form

$$\psi_j(y) \;:=\; \prod_{i=0}^{l+n} p_i(y)^{\beta_{i,j}} \;=\; 1, \qquad \text{for } j = 1,\ldots,l\,, \tag{2.3}$$

in the polyhedron

$$\Delta \;:=\; \{y \in \mathbb{R}^l \mid p_i(y) > 0\}\,, \tag{2.4}$$

which we assume is bounded. Here $p_0(y),\ldots,p_{l+n}(y)$ are degree 1 polynomials that are general in that no $l+1$ of them have a common solution. Also, $\beta_{i,j}$ are real numbers that are sufficiently general in a manner that is explained below.

We first make some observations and definitions. In Δ, we may take logarithms of the master functions in (2.3) to obtain the equivalent system

$$\varphi_j(y) \;:=\; \log\psi_j(y) \;=\; \sum_{i=0}^{l+n} \beta_{i,j}\log p_i(y) \;=\; 0 \qquad \text{for } j = 1,\ldots,l\,.$$

The Khovanskii-Rolle algorithm will find solutions to these equations by iteratively applying Algorithm 2.2. For $j = l, l-1, \ldots, 2, 1$ define the Jacobian determinants

$$\widetilde{J}_j \;:=\; \text{J-det}(\varphi_1,\ldots,\varphi_j;\, \widetilde{J}_{j+1},\ldots,\widetilde{J}_l)\,.$$

Also, write the rational function $\psi_j(y) = \psi_j^+(y)/\psi_j^-(y)$ as a quotient of two polynomials ψ_j^+ and ψ_j^-. Writing $f_j(y)$ for the difference $\psi_j^+(y) - \psi_j^-(y)$, $\psi_i(y) = 1$ is equivalent to $f_j(y) = 0$.

Algorithm 2.3 (Khovanskii-Rolle Continuation)**.**

Suppose that φ_j and $\widetilde{J}_j$ for $j = 1,\ldots,l$ are as above. For each $j = 0,\ldots,l$, let S_j be the solutions in Δ to

$$\varphi_1 \;=\; \cdots \;=\; \varphi_j \;=\; \widetilde{J}_{j+1} \;=\; \cdots \;=\; \widetilde{J}_l \;=\; 0\,,$$

and for each $j = 1,\ldots,l$, let T_j be the set of solutions to

$$f_1 \;=\; \cdots \;=\; f_{j-1} \;=\; 0 \qquad \text{and} \qquad \widetilde{J}_{j+1} \;=\; \cdots \;=\; \widetilde{J}_l \;=\; 0 \tag{2.5}$$

in the boundary of Δ.

INPUT: Solutions S_0 of $\widetilde{J}_1 = \cdots = \widetilde{J}_l = 0$ in Δ, and sets $T_1,\ldots,T_l$.
OUTPUT: Solution sets $S_1,\ldots,S_l$.

For each $j = 1,\ldots,l$, apply the Continuation Algorithm 2.2 with

$$(f_1,\ldots,f_{l-1}) \;=\; (\varphi_1,\ldots,\varphi_{j-1}\,,\, \widetilde{J}_{j+1},\ldots,\widetilde{J}_l)$$

and $g = \varphi_j$. The inputs for this are the sets S_{j-1} and T_j, and the output is the set S_j.

The last set computed, S_l, is the set of solutions to the system of master functions (2.3) in Δ. We describe our implementation for positive solutions when $l = 2$ in Section 4 and illustrate the algorithm in Section 3.

Proof of correctness. Note that for each $j = 1,\ldots,l$, the jth step finds the solution set S_j, and therefore preforms as claimed by the correctness of the Continuation Algorithm 2.2. The correctness for the Khovanskii-Rolle algorithm follows by induction on j. □

The Khovanskii-Rolle algorithm requires the precomputation of the sets S_0 and $T_1, \ldots, T_l$. The feasibility of this task follows from Lemma 3.4 in [7], which we state below. By our assumption on the generality of the polynomials p_i, a face of Δ of codimension k is the intersection of Δ with k of the hyperplanes $p_i(y) = 0$.

Proposition 2.4. *Let $p_i, i = 0, \ldots, l+n$, $\varphi_j, \widetilde{J}_j, j = 1, \ldots, l$, and Δ be as above.*

(1) *The solutions T_j to (2.5) in the boundary of Δ are the solutions to*

$$\widetilde{J}_{j+1} = \cdots = \widetilde{J}_l = 0$$

in the codimension-j faces of Δ.

(2) $\widetilde{J}_j \cdot \prod_{i=0}^{l+n} p_i(y)^{2^{l-j}}$ *is a polynomial of degree $2^{l-j}n$.*

By Proposition 2.4(1), solving the system (2.5) in the boundary of Δ is equivalent to solving (at most) $\binom{l+n+1}{j}$ systems of the form

$$p_{i_1} \;=\; \cdots \;=\; p_{i_j} \;=\; \widetilde{J}_{j+1} \;=\; \cdots \;=\; \widetilde{J}_l \;=\; 0 \tag{2.6}$$

and then discarding the solutions y for which $p_i(y) < 0$ for some i. Replacing each Jacobian $\widetilde{J}_j$ by the polynomial $J_j := \widetilde{J}_j \cdot \prod_{i=0}^{l+n} p_i(y)^{2^{l-j}}$, and using $p_{i_1}(y) = \cdots = p_{i_j} = 0$ to eliminate j variables, we see that (2.6) is a polynomial system with Bézout number

$$n \cdot 2n \cdot 4n \cdots \; 2^{l-j-1}n \;=\; 2^{\binom{l-j}{2}} n^{l-j}$$

in $l - j$ variables. Thus the number of solutions T_j to the system (2.5) in the boundary of Δ is at most $2^{\binom{l-j}{2}} n^{l-j} \binom{l+n+1}{j}$.

Likewise S_0 consists of solutions in Δ to the system (2.6) when $j = 0$ and so has at most the Bézout number $2^{\binom{l}{2}} n^l$ solutions. Thus the inputs to the Khovanskii-Rolle Algorithm are solutions to polynomial systems in l or fewer variables.

Theorem 2.5. *The Khovanskii-Rolle Continuation Algorithm finds all solutions to the system (2.3) in the bounded polyhedron Δ (2.4), when the degree 1 polynomials $p_i(y)$ and exponents $\beta_{i,j}$ are general as described. It accomplishes this by solving auxiliary polynomial systems (2.6) and following implicit curves. For each $j = 0, \ldots, l-1$, it will solve at most $\binom{l+n+1}{j}$ polynomial systems in $l-j$ variables, each having Bézout number $2^{\binom{l-j}{2}} n^{l-j}$. In all, it will trace at most*

$$l\, 2^{\binom{l}{2}+1} n^l \;+\; \sum_{j=1}^{l} (l+1-j) 2^{l-j} n^{l-j} \binom{l+n+1}{j} \;<\; l\, \frac{e^2+3}{2} 2^{\binom{l}{2}} n^l \tag{2.7}$$

implicit curves in Δ.

Proof. The first statement is a restatement of the correctness of the Khovanskii-Rolle Continuation Algorithm. For the second statement, we enumerate the number of paths, following the discussion after Proposition 2.4. Let s_j, t_j be the number of points in S_j and T_j, respectively, and let r_j be the number of paths followed in the jth step of the Khovanskii-Rolle Continuation Algorithm.

By (2.2), $r_j = t_j + 2s_{j-1}$ and $s_j \leq \frac{1}{2}t_j + s_{j-1}$. So $r_j \leq t_j + \cdots + t_1 + 2s_0$, and

$$r_1 + \cdots + r_l \;\leq\; 2s_0 + \sum_{j=1}^{l} (l{+}1{-}j)t_j\,.$$

Using the estimates

$$s_0 \;\leq\; 2^{\binom{l}{2}} n^l \qquad \text{and} \qquad t_j \;\leq\; 2^{\binom{l-j}{2}} n^{l-j} \tbinom{l+n+1}{j}\,, \tag{2.8}$$

we obtain the estimate on the left of (2.7). Since $l{+}1{-}j \leq l$, we bound it by

$$2l\Big(2^{\binom{l}{2}} n^l \;+\; \frac{1}{2}\sum_{j=1}^{l} 2^{l-j} n^{l-j} \tbinom{l+n+1}{j}\Big)\,,$$

which is bounded by $l\,\frac{e^2+3}{2} 2^{\binom{l}{2}} n^l$, by Lemma 3.5 in [7]. □

Remark 2.6. The bound (2.7) on the number of paths to be followed is not sharp. First, not every system of the linear polynomials

$$p_{i_1}(y) \;=\; p_{i_2}(y) \;=\; \cdots \;=\; p_{i_j}(y) \;=\; 0\,,$$

defines a face of Δ. Even when this defines a face F of Δ, only the solutions to (2.6) that lie in F contribute to T_j, and hence to the number of paths followed. Thus any slack in the estimates (2.8) reduces the number of paths to be followed. Since these estimates lead to the fewnomial bound for s_l, we see that the Khovanskii-Rolle Continuation Algorithm naturally takes advantage of any lack of sharpness in the fewnomial bound.

This may further be improved if Δ has $m < l + n + 1$ facets for then the binomial coefficients in (2.7) become $\binom{m}{2}$.

2.3. **Khovanskii-Rolle algorithm for all real solutions.** Section 2.2 describes how to find solutions to a system of master functions (2.3) in the polyhedron Δ (2.4), which is assumed bounded. Through Gale duality and the coordinate transformation (1.10), this gives a method to find all positive real solutions to a system of polynomial equations (1.6).

To find all real solutions to a system of polynomial equations or of master functions, one could simply repeat this process for every chamber in the complement of the hyperplanes $p_i(y) = 0$ for $i = 1, \ldots, l{+}n$. This is however inefficient as our method (homotopy continuation) for computing the sets S_0 and T_i for $i = 1, \ldots, l$ of starting points for one chamber gives the starting points for all chambers. Besides careful bookkeeping, this requires some projective coordinate transformations so that the tracking occurs in bounded chambers.

Each point in T_j is incident upon 2^j chambers, and thus is the starting point for 2^j arcs to be followed in Algorithm 2.3. Surprisingly, this has a mild effect on the complexity, requiring only that we replace the e^2 in (2.7) by e^4. This observation, which was made while developing this Khovanskii-Rolle algorithm, was the genesis of the bound in [3].

3. Examples

We first illustrate the Khovanskii-Rolle algorithm and our implementation on the master function system of Example 1.1. Write the system (1.3) of master functions in logarithmic

form, $\varphi_1(y) = \varphi_2(y) = 0$, for $y \in \Delta$, which is the heptagon depicted in Figure 1. Here, we have $l = 2$ and $n = 5$ with 7 linear polynomials.

$$\begin{aligned}
\varphi_1(y) &= -\log(y) + \log(2x+2y-1) - 2\log(4x) + \log\left(\tfrac{1+x-y}{2}\right) \\
&\qquad -2\log\left(\tfrac{6-x-3y}{4}\right) + 2\log\left(\tfrac{8-3x-2y}{2}\right) - \log(3-2x+y)\,. \\
\varphi_2(y) &= \log(y) + 6\log(2x+2y-1) - 3\log(4x) + 6\log\left(\tfrac{1+x-y}{2}\right) \\
&\qquad -2\log\left(\tfrac{6-x-3y}{4}\right) + 7\log\left(\tfrac{8-3x-2y}{2}\right) + \log(3-2x+y)\,.
\end{aligned}$$

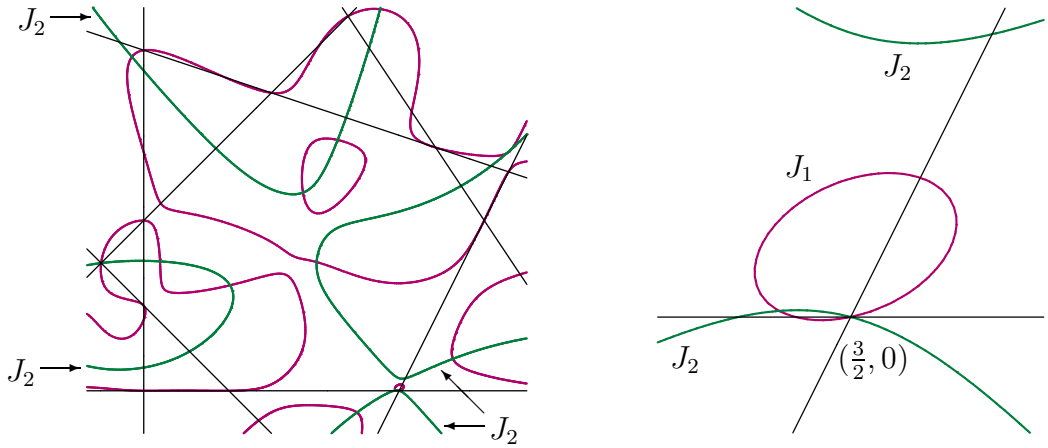
The polynomial forms J_2, J_1 of the Jacobians are (omitting the middle 60 terms from J_1),

$$\begin{aligned}
J_2 &= -168x^5 - 1376x^4y + 480x^3y^2 - 536x^2y^3 - 1096xy^4 + 456y^5 + 1666x^4 + 2826x^3y \\
&\quad +3098x^2y^2 + 6904xy^3 - 1638y^4 - 3485x^3 - 3721x^2y - 15318xy^2 - 1836y^3 \\
&\quad +1854x^2 + 8442xy + 9486y^2 - 192x - 6540y + 720\,. \\
J_1 &= 10080x^{10} - 168192x^9y - 611328x^8y^2 - \quad \cdots \quad + 27648x + 2825280y\,.
\end{aligned}$$

Remark 3.1. Instead of the polynomial form J_2 of the Jacobian of φ_1 and φ_2, we could use the Jacobian $J(f, g)$ of f and g (1.4). This however has degree 25 with 347 terms and coefficients of the order of 10^{17}. The Jacobian of this and f has degree 29 and 459 terms. This control on the degree of the Jacobians is the reason that we use the logarithms of the master function in the formulation of the Khovanskii-Rolle Algorithm.

We now describe the Khovanskii-Rolle algorithm on this example.

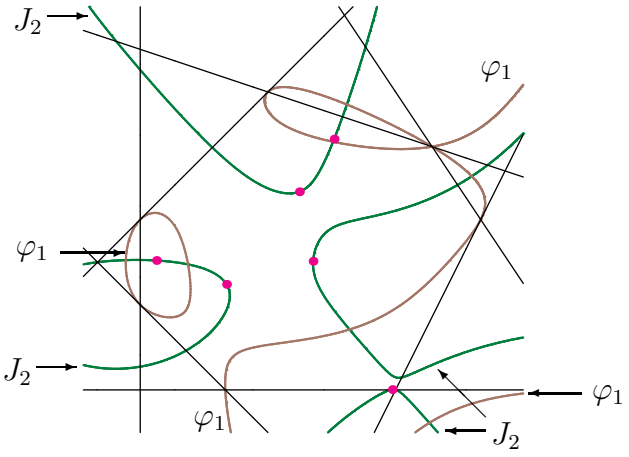
Precomputation. We first find all solutions S_0 to $J_1 = J_2 = 0$ in the heptagon Δ, and all solutions $J_2 = 0$ in the boundary of the heptagon. Below are the curves $J_2 = 0$ and $J_1 = 0$ and the heptagon. The curve $J_2 = 0$ consists of the four arcs indicated. The remaining curves in this picture belong to $J_1 = 0$. On the right is an expanded view in a neighborhood of the lower right vertex, $(\frac{3}{2}, 0)$.



A numerical computation finds 50 common solutions to $J_1 = J_2 = 0$ with 26 real. Only six solutions lie in the interior of the heptagon with one on the boundary at the vertex $(3/2, 0)$. There are 31 points where the curve $J_2 = 0$ meets the lines supporting the boundary of

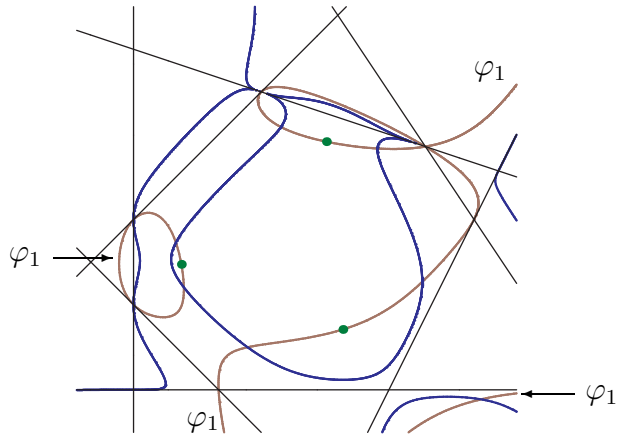
the heptagon, but only eight lie in the boundary of the hexagon. This may be seen in the pictures above.

First continuation step. Beginning at each of the six points in Δ where $J_1 = J_2 = 0$, the algorithm traces the curve in both directions, looking for a solution to $\varphi_1 = J_2 = 0$. Beginning at each of the eight points where the curve $J_2 = 0$ meets the boundary, it follows the curve into the interior of the heptagon, looking for a solution to $\varphi_1 = J_2 = 0$. In tracing each arc, it either finds a solution, a boundary point, or another point where $J_1 = J_2 = 0$. We may see that in the picture below, which shows the curves $\varphi_1 = 0$ and $J_2 = 0$, as well as the points on the curve $J_2 = 0$ where J_1 also vanishes.



In this step, $2 \cdot 6 + 8 = 20$ arcs are traced. The three solutions of $J_2 = \varphi_1 = 0$ will each be found twice, and 14 of the tracings will terminate with a boundary point or a point where $J_1 = 0$.

Second Continuation step. This step begins at each of the three points where $\varphi_1 = J_2 = 0$ that were found in the last step, as well as at each of the six points where $\varphi_1 = 0$ meets the boundary of the heptagon (necessarily in some vertices).



Curve tracing, as described in more detail in the next section, can be carried out even in the presence of singularities, as in the case of the curves initiating at vertices. In this case, this final round of curve-tracing revealed all six solutions within the heptagon, as anticipated. Furthermore, each solution was discovered twice, again, as anticipated.

By Theorem 2.5, the bound on the number of paths followed (using $7 = l + n$ in place of $l + n + 1$ in the binomials as in Remark 2.6) is

$$2 \cdot 2 \cdot 2^{\binom{2}{2}} \cdot 5^2 \;+\; 2 \cdot 2^{\binom{1}{2}} \cdot 5^1 \cdot \tbinom{7}{1} \;+\; 1 \cdot 2^{\binom{0}{2}} \cdot 5^0 \cdot \tbinom{8}{2} \;=\; 298\,.$$

By Theorem 3.10 in [7], the fewnomial bound in this case is

$$2 \cdot 5^2 + \lfloor \tfrac{(5+1)(5+3)}{2} \rfloor \;=\; 74\,.$$

In contrast, we only traced $20 + 12 = 32$ curves to find the six solutions in the heptagon. The reason for this discrepancy is that this bound is pessimistic and the Khovanskii-Rolle Continuation Algorithm exploits any slack in it.

Timings and comparison to existing software. The system of Laurent polynomials (1.1) was converted into a system of polynomials by clearing denominators. It was then run through PHCpack, Bertini, and the proof-of-concept implementation described in the next section. As described in Section 1, this system has 102 regular solutions, 10 of which are real. All runs of this section were performed on a 2.83 GHz running CentOS with Maple 13, Bertini 1.1.1, and PHCpack v2.3.48.

In blackbox mode, PHCpack used polyhedral methods and found 102 regular solutions in around 2.5 seconds, though it only classified eight of them as real. Using all default settings and a 5-homogeneous start system, Bertini found all 102 regular solutions and identified the 10 that are real. However, because of the use of adaptive precision, this took around 23 seconds. Using fixed low precision, this time dropped to around 9 seconds while still identifying the solutions correctly.

The implementation described in the next section found all positive solutions of the Gale dual system of master functions 1.4 in around 15 seconds using safe settings. Almost all of this time was spent in computing the sets S_0 and T_1 in Bertini, using adaptive precision for security. By changing to fixed low precision for the Bertini portions of the computation, the timing for the entire run fell to around 1.4 seconds – the shortest time of all runs described here. Though such efficiency is welcome, security is more valuable. It is expected that more sophisticated software than that described in the next section will be more efficient.

A more extreme example. A polynomial system with high degree, many complex solutions, and few real solutions further illustrates the value of Khovanskii-Rolle continuation. For example, consider the system of Laurent polynomials

$$\begin{aligned} 10500 - tu^{492} - 3500t^{-1}u^{463}v^5w^5 &= 0 \\ 10500 - t - 3500t^{-1}u^{691}v^5w^5 &= 0 \\ 14000 - 2t + tu^{492} - 3500v &= 0 \\ 14000 + 2t - tu^{492} - 3500w &= 0. \end{aligned} \tag{3.1}$$

By solving a set of master functions Gale dual to (3.1) for 0, we obtain

$$\begin{aligned} 3500^{12}x^8y^4(3-y)^{45} - (3-x)^{33}(4-2x+y)^{60}(2x-y+1)^{60} &= 0 \\ 3500^{12}x^{27}(3-x)^8(3-y)^4 - y^{15}(4-2x+y)^{60}(2x-y+1)^{60} &= 0. \end{aligned} \tag{3.2}$$

System (3.1) has 7663 complex solutions but only six positive real solutions. PHC computes these solutions in 39 minutes, 38 seconds while the implementation of the next section takes only 23 seconds to find them, using safe settings.

4. Implementation

We have implemented the Khovanskii-Rolle Algorithm to find all solutions to a system of master functions in a bounded polyhedron Δ, but only when $l = 2$. This proof-of-concept implementation relies on external calls to the Bertini software package [4] as a polynomial system solver. The implementation is in a Maple script that is publicly available at [6]. We plan to implement a general version of the Khovanskii-Rolle algorithm in the future.

We describe some aspects of this implementation, including the precomputation to find the solution sets S_0 and T_1 and the curve-tracing method from these points and from S_1, all of which are smooth points on the traced curves. We also discuss tracing curves from the vertices T_2, which is non-trivial as these curves are typically singular at the vertices. Lastly, we discuss procedures for checking the output.

4.1. **Polynomial system solving.** The precomputation of S_0 and T_1 for Algorithm 2.3 involves finding the solutions of systems of Jacobians (2.6) within Δ and on its boundary. For this, we use the system Bertini [4]. For each system to be solved, the maple script creates a Bertini input file, calls Bertini, and collects the real solutions from an output file. Bertini, as with all homotopy methods, finds all complex solutions. However, as explained previously, this overhead may be much less than would be encountered in the direct use of homotopy methods to solve the original system.

4.2. **Curve-tracing from smooth points.** The points of S_0 and T_1 from which we trace curves in the first step of the algorithm, as well as the points S_2 used in the second step, are smooth points of the curves $J_2 = 0$ and $\varphi_1 = 0$, respectively. This curve-tracing proceeds as in Section 1.2. Indeed, suppose we are tracing a curve C in the polytope Δ, starting from points where C meets the boundary of Δ and from interior starting points where a Jacobian determinant J vanishes, and we seek points of C where some function g vanishes. Then, as described in the Continuation Algorithm 2.2, there are three basic stopping criteria:

(1) The tracer passes a point where $g = 0$, which is a point we are looking for.
(2) The tracer passes a point where $J = 0$, which is another starting point.
(3) The tracer leaves the polytope.

The computation of the tangent and normal lines is straightforward, as is the linear algebra required for curve-tracing. Each predictor-corrector step gives a point p near C. We compute the values of g, J, and the degree 1 polynomials p_i defining Δ at p. A sign change in any indicates a stopping criteria has been met. When this occurs, bisection in the steplength Δt from the previous approximation is used to refine the point of interest.

It may seem that these (sign-based) criteria will fail when there are clustered (or multiple) solutions to $g = 0$ on the curve between the current and previous approximation. However, the Khovanskii-Rolle Theorem implies there will be zeroes of the Jacobian determinant J interspersed between these solutions (or coinciding with the multiple solutions). Furthermore, the number of solutions to $g = 0$ and to $J = 0$ will have different parities, so that

one of the two functions g and J will change sign, guaranteeing that one of the stopping criteria will be triggered for such a curve segment.

Curve-tracing involves a trade-off between security and speed. Security is enhanced with a small steplength Δt and allowing only one or two Newton correction steps. However, these settings contribute to slow curve-tracing. The examples in Section 3 were computed with secure settings. In Section 4.4 we give methods to detect some errors in this curve-tracing which can allow less secure settings.

4.3. **Curve-tracing from vertices.** The curves to be traced are typically singular at the vertices of Δ. While traditional curve-tracing fails at such points, we employ a simple alternative that takes advantage of the local structure of the curves.

At a vertex v of Δ, we make a linear change of coordinates so that the two incident edges are given by the coordinate polynomials y_1 and y_2. Then $\varphi_1(y) = 0$ may be expressed as

$$\prod_{j=0}^{n+2} p_i(y)^{\beta_{i,1}} \;=\; 1\,.$$

Setting $p_{n+1} = y_1$ and $p_{n+2} = y_2$, we may solve for y_2 to obtain

$$y_2 \;=\; y_1^{-\beta_{n+1,1}/\beta_{n+2,1}} \cdot \prod_{i=0}^{n} p_i(y)^{-\beta_{i,1}/\beta_{n+2,1}}\,.$$

In the neighborhood of v this is approximated by the monomial curve

$$y_2 \;=\; y_1^{-\beta_{n+1,1}/\beta_{n+2,1}} \cdot \prod_{i=0}^{n} p_i(v)^{-\beta_{i,1}/\beta_{n+2,1}}\,,$$

where we have evaluated the terms $p_i(y)$ for $i \leq n$ at the vertex v, i.e., the product is just a constant α. We may write this expression as

$$y_2 \;=\; \alpha \cdot y_1^{\beta}\,; \tag{4.1}$$

note that $\alpha > 0$.

We do not begin tracing the curve $\varphi_1(y) = 0$ at v, but instead begin from a point c_p on the monomial curve (4.1) in Δ near v. If the first predictor-corrector step from c_p succeeds to approximate $\varphi_1(y) = 0$, then we trace this curve as described before. If this predictor-corrector step fails from c_p, then we simply choose a point on the monomial curve a bit further from v and try again. In our experience, this special form of monomial tracking quickly gives way to usual curve-tracing on $\varphi_1(y) = 0$.

4.4. **Procedures for checking the output.** General curve-tracing is not foolproof. If the curves to be traced are very close together, then curve-jumping may occur, leading to missed solutions. However, for curve-tracing in the Khovanskii-Rolle algorithm, there is a simple way to check for such errors.

As described after Algorithm 2.2, each point in the sets S_i for $i = 1, \ldots, l$ should be discovered twice as the end of an arc that is tracked. Thus, if a solution is not found twice, an error occurred in the curve-tracing.

This check will not capture all errors. The development of further verification and certification procedures is a goal for future research. This lack of checks is not uncommon for

new algorithms, including the methods introduced in the early development of numerical algebraic geometry

5. Conclusions

Numerical homotopy continuation is a robust and efficient algorithm for finding all complex solutions to a system of polynomial equations. Real solutions are obtained by selecting those solutions with small imaginary parts. While often practical, this is wasteful and does not exploit the real algebraic nature of the problem.

We presented a numerical continuation algorithm to find all real or positive solutions to a system of polynomials, along with details of our implementation and two examples. This new Khovanskii-Rolle continuation algorithm is efficient in that the number of paths to be followed depends upon the corresponding fewnomial bounds for the numbers of real solutions and not on the number of complex solutions. This is a significant difference between our new algorithm and all other known methods for solving polynomial systems.

This algorithm does not directly solve the given polynomial system but rather an equivalent (Gale dual) system consisting of master functions in the complement of an arrangement of hyperplanes. This appears to be the first curve-tracing algorithm that finds only real solutions to a system of equations.

This paper is the first step in this new line of research. There are clear generalizations to higher dimensions. We plan further research into techniques for tracing the curves that begin at singularities. In contrast to path-following in homotopy continuation, the security of curve-tracing in this algorithm relies on heuristics. Another research direction is to enhance the security and efficacy of curve-tracing.

When our algorithm has been implemented in more than two variables ($l > 2$), we plan to use it to study real solutions to systems of polynomial equations.

References

[1] Eugene Allgower, Melissa Erdmann, and Kurt Georg, *On the complexity of exclusion algorithms for optimization*, J. Complexity **18** (2002), no. 2, 573–588, Algorithms and complexity for continuous problems/Algorithms, computational complexity, and models of computation for nonlinear and multivariate problems (Dagstuhl/South Hadley, MA, 2000).

[2] Eugene Allgower and Kurt Georg, *Introduction to numerical continuation methods*, Classics in Applied Mathematics, 45, SIAM, 2003.

[3] Daniel J. Bates, Frédéric Bihan, and Frank Sottile, *Bounds on the number of real solutions to polynomial equations*, Int. Math. Res. Not. IMRN (2007), no. 23, Art. ID rnm114, 7.

[4] Daniel J. Bates, Jonathan D. Hauenstein, Andrew J. Sommese, and Charles W. Wampler, *Bertini: Software for numerical algebraic geometry*, Available at http://www.nd.edu/∼sommese/bertini.

[5] ———, *Adaptive precision path tracking*, SIAM J. Num. Anal. **46** (2008), no. 2, 722–746.

[6] Daniel J. Bates and Frank Sottile, *Khovanskii-Rolle continuation for real solutions*, `www.math.tamu.edu/~sottile/stories/Rolle/` `www.nd.edu/~dbates1/Rolle/`.

[7] Frédéric Bihan and Frank Sottile, *New fewnomial upper bounds from Gale dual polynomial systems*, Mosc. Math. J. **7** (2007), no. 3, 387–407, 573.

[8] ———, *Gale duality for complete intersections*, Ann. Inst. Fourier (Grenoble) **58** (2008), no. 3, 877–891.

[9] George E. Collins, *Quantifier elimination for real closed fields by cylindrical algebraic decomposition*, Automata theory and formal languages (Second GI Conf., Kaiserslautern, 1975), Springer, Berlin, 1975, pp. 134–183. Lecture Notes in Comput. Sci., Vol. 33.
[10] Kurt Georg, *Improving the efficiency of exclusion algorithms*, Adv. Geom. **1** (2001), no. 2, 193–210.
[11] G.-M. Greuel, G. Pfister, and H. Schönemann, SINGULAR *3.0*, A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2005, `http://www.singular.uni-kl.de`.
[12] Birkett Huber and Jan Verschelde, *Polyhedral end games for polynomial continuation*, Numer. Algorithms **18** (1998), no. 1, 91–108.
[13] A.G. Khovanskii, *Fewnomials*, Trans. of Math. Monographs, 88, AMS, 1991.
[14] Jean Bernard Lasserre, Monique Laurent, and Philipp Rostalski, *Semidefinite characterization and computation of zero-dimensional real radical ideals*, Found. Comput. Math. **8** (2008), no. 5, 607–647.
[15] T. Lee, T.Y. Li, and C. Tsai, *Hom4ps-2.0: A software package for solving polynomial systems by the polyhedral homotopy continuation method*, Computing **83** (2008), 109–133.
[16] B. Mourrain and J.-P. Pavone, *Subdivision methods for solving polynomial systems*, technical report 5658, INRIA, 2005.
[17] Fabrice Rouillier, *Solving zero-dimensional systems through the rational univariate representation*, Appl. Algebra Engrg. Comm. Comput. **9** (1999), no. 5, 433–461.
[18] Andrew J. Sommese and Charles W. Wampler, II, *The numerical solution of systems of polynomials*, World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2005, Arising in engineering and science.
[19] J. Verschelde, *Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation*, ACM Trans. Math. Softw. **25** (1999), no. 2, 251–276, Software available at `http://www.math.uic.edu/~jan`.
[20] J.H. Wilkinson, *Rounding errors in algebraic processes*, Dover Publications, Inc., 1994.

DEPARTMENT OF MATHEMATICS, 101 WEBER BUILDING, COLORADO STATE UNIVERSITY, FORT COLLINS, CO 80523-1874, USA
E-mail address: `bates@math.colostate.edu`
URL: `http://www.math.colostate.edu/~bates/`

DEPARTMENT OF MATHEMATICS, TEXAS A&M UNIVERSITY, COLLEGE STATION, TEXAS 77843, USA
E-mail address: `sottile@math.tamu.edu`
URL: `http://www.math.tamu.edu/~sottile/`